An Empirical Study on Language Model Adaptation Using a Metric of Domain Similarity

Wei Yuan[1], Jianfeng Gao[2], and Hisami Suzuki[3]

[1] Shanghai Jiao Tong University
1954 Huashan Road, Shanghai 200230
sunnyuanovo@sjtu.edu.cn
[2] Microsoft Research, Asia
49 Zhichun Road, Haidian District, Beijing 100080
jfgao@microsoft.com
[3] Microsoft Research
One Microsoft Way, Redmond WA 98052
hisamis@microsoft.com

Abstract. This paper presents an empirical study on four techniques of language model adaptation, including a maximum *a posteriori* (MAP) method and three discriminative training models, in the application of Japanese Kana-Kanji conversion. We compare the performance of these methods from various angles by adapting the baseline model to four adaptation domains. In particular, we attempt to interpret the results given in terms of the character error rate (CER) by correlating them with the characteristics of the adaptation domain measured using the information-theoretic notion of cross entropy. We show that such a metric correlates well with the CER performance of the adaptation methods, and also show that the discriminative methods are not only superior to a MAP-based method in terms of achieving larger CER reduction, but are also more robust against the similarity of background and adaptation domains.

1 Introduction

Language model (LM) adaptation attempts to adjust the parameters of a LM so that it performs well on a particular domain of data. This paper presents an empirical study of several LM adaptation methods on the task of Japanese text input. In particular, we focus on the so-called *cross-domain* LM adaptation paradigm, i.e. to adapt a LM trained on one domain to a different domain, for which only a small amount of training data is available.

The LM adaptation methods investigated in this paper can be grouped into two categories: maximum *a posterior* (MAP) and discriminative training. Linear interpolation is representative of the MAP methods [1]. The other three methods, including the boosting [2] and perceptron [3] algorithms and minimum sample risk (MSR) method [4], are discriminative methods, each of which uses a different training algorithm.

[1] This research was conducted while the author was visiting Microsoft Research Asia.

We carried out experiments over many training data sizes on four distinct adaptation corpora, the characteristics of which were measured using the information-theoretic notion of cross entropy. We found that discriminative training methods outperformed the LI method in all cases, and were more robust across different training sets of different domains and sizes. However, none of the discriminative training methods was found to outperform the others in our experiments.

The paper is organized as follow. Section 2 introduces the task of IME and the role of LM. In Section 3, we review related work. After a description of the LM adaptation methods in our experiments in Section 4, Sections 5 and 6 present experimental results and their discussions. We conclude our paper in Section 7.

2 Language Model and the Task of IME

Our study falls into the context of Asian language (Japanese in this study) text input. The standard method for doing this is that the users first input the phonetic strings, which are then converted into the appropriate word string by software. The task of automatic conversion is called *IME* in this paper, which stands for *Input Method Editor*, based on the name of the commonly used Windows-based application.

The performance of IME is typically measured in terms of the *character error rate* (CER), which is the number of characters wrongly converted from the phonetic string divided by the number of characters in the correct transcript. Current Japanese IME systems exhibit about 5-15% CER in conversion of real-world data in a wide variety of domains. In the following, we argue that the IME is a similar problem to speech recognition but is a better choice for evaluating language modeling techniques.

Similar to speech recognition, IME can also be viewed as a Bayesian decision problem. Let A be the input phonetic string (which corresponds to the acoustic signal in speech). The task of IME is to choose the most likely word string W^* among those candidates that could have been converted from A:

$$W^* = \arg\max_{W \in GEN(A)} P(W \mid A) = \arg\max_{W \in GEN(A)} \frac{P(W, A)}{P(A)} = \arg\max_{W \in GEN(A)} P(W)P(A \mid W) \quad \textbf{(1)}$$

where **GEN**(A) denotes the candidate set given A.

Unlike speech recognition, there is almost no acoustic ambiguity in IME, because the phonetic string is provided directly by users. Moreover, we can assume a many-to-one mapping from W to A in IME, i.e. $P(A|W) = 1$. So the decision of Equation (1) depends solely upon $P(W)$, making IME a more direct evaluation test-bed for LM than speech recognition. Another advantage is that it is relatively easy to convert W to A, making it possible to obtain a large amount of training data for discriminative learning, as described later.

3 Related Work

Our goal is to quantify the characteristics of different domains of text, and to correlate them with the performance of various techniques for LM adaptation to compare their effectiveness and robustness. This relates our work to the study of domain similarity calculation and to different techniques for LM adaptation.

3.1 Measuring Domain Similarity

Statistical language modeling (SLM) assumes that language is generated from underlying distributions. When we discuss different domains of text, we assume that the text from each of these domains is generated from a different underlying distribution. We therefore consider the problem of distributional similarity in this paper.

Cross entropy is a widely used measure in evaluating LM. Given a language *L* with its true underlying probability distribution *p* and another distribution *q* (e.g. a SLM) which attempts to model *L*, the cross entropy of *L* with respect to *q* is

$$H(L,q) = -\lim_{n\to\infty}\frac{1}{n}\sum_{w_1\ldots w_n} p(w_1\ldots w_n)\log q(w_1\ldots w_n) \qquad \textbf{(2)}$$

where $w_1\ldots w_n$ is a word string in *L*. However, in reality, the underlying *p* is never known and the corpus size is never infinite. We therefore make the assumption that *L* is an ergodic and stationary process [5], and approximate the cross entropy by calculating it for a sufficiently large *n* instead of calculating it for the limit.

$$H(L,q) \approx -\frac{1}{n}\log q(w_1\ldots w_n) \qquad \textbf{(3)}$$

This measures how well a model approximates the language *L*.

The KL divergence, or relative entropy, is another measure of distributional similarity that has been widely used in NLP and IR [6]. Given the two distributions *p* and *q* above, the KL divergence is defined as

$$D(p(w_1\ldots w_n)\,\|\,q(w_1\ldots w_n)) = \sum_{w_1\ldots w_n} p(w_1\ldots w_n)\log\frac{p(w_1\ldots w_n)}{q(w_1\ldots w_n)} \qquad \textbf{(4)}$$

The cross entropy and the KL divergence are related notions. Given the notations of *L*, *p* and *q* above, [5] shows that

$$H(L,q) = H(L) + D(p\,\|\,q) \qquad \textbf{(5)}$$

In other words, the cross entropy takes into account both the similarity between two distributions (given by KL divergence) and the entropy of the corpus in question, both of which contribute to the complexity of a LM task. In this paper we are interested in measuring the complexity of the LM adaptation task. We therefore define the similarity between two domains using the cross entropy. We will also use the metric that approximates the entropy of the corpus to capture the in-domain diversity of a corpus in Section 5.2.[2]

[2] There are other well-known metrics of similarity within NLP literature, such as the mutual information or cosine similarity [7], which we do not discuss in this paper.

3.2 LM Adaptation Methods

In this paper, two major approaches to cross-domain adaptation have been investigated: maximum *a posteriori* (MAP) estimation and discriminative training methods.

In MAP estimation methods, adaptation data is used to adjust the parameters of the background model so as to maximize the likelihood of the adaptation data [1]. Discriminative training methods to LM adaptation, on the other hand, aim at using the adaptation data to directly minimize the errors on the adaptation data made by the background model. These techniques have been applied successfully to the task of language modeling in non-adaptation [8] as well as adaptation scenarios [9] for speech recognition. But most of them focused on the investigation of performance of certain methods for LM adaptation, without analyzing in detail the underlying reasons of different performance achieved by different methods. In this paper we attempt to investigate the effectiveness of different discriminative methods in an IME adaptation scenario, with a particular emphasis on correlating their performance with the characteristics of adaptation domain.

4 LM Adaptation Methods

We implement four methods in our experiments. The Linear Interpolation (LI) falls into the framework of MAP while the boosting, the perceptron and the MSR methods fall into that of discriminative training.

4.1 Linear Interpolation (MAP)

In MAP estimation methods, adaptation data is used to adjust the parameters of the background model so as to maximize the likelihood of the adaptation data.

The linear interpolation is a special case of MAP according to [10]. At first, we generate trigram models on background data and adaptation data respectively. The two models are then combined into one as:

$$P(w_i \mid h) = \lambda P_B(w_i \mid h) + (1-\lambda) P_A(w_i \mid h) \qquad \textbf{(6)}$$

where P_B is the probability of the background model, P_A is the probability of the adaptation model and the history h corresponds to two preceding words. For simplicity, we chose a single λ for all histories and tuned it on held-out data.

4.2 Discriminative Training

Discriminative training follows the general framework of linear models [2][3]. We use the following notation in the rest of the paper.

- Training data is a set of example input/output pairs $\{A_i, W_i^R\}$ for $i = 1 \ldots M$, where each A_i is an input phonetic string and each W_i^R is the reference transcript of A_i.
- We assume a set of D+1 features, $f_d(W)$, for d=0…D, where each feature is a function that maps W to a real value. Using vector notation, we have $\mathbf{f}(W)=\{ f_0(W), f_1(W) \ldots f_D(W)\}$ and $\mathbf{f}(W) \in R^{D+1}$. Without loss of generality, $f_0(W)$ is called the base model feature, and is defined in this paper as the log probability that the back-

ground trigram model assigns to W. $f_d(W)$, for d=1…D, are defined as the counts of the word n-gram in W, where n = 1 and 2 in our case.

- Finally, the parameters of the model form a vector of D + 1 dimensions, each for one feature function, $\boldsymbol{\lambda} = \{\lambda_0, \lambda_1 \ldots \lambda_D\}$. The likelihood score of a word string W is

$$Score(W, \boldsymbol{\lambda}) = \boldsymbol{\lambda}\mathbf{f}(W) = \sum_{d=0}^{D} \lambda_d f_d(W) \tag{7}$$

Then the decision rule of Equation (1) can be re-written as

$$W^*(A, \boldsymbol{\lambda}) = \arg\max_{W \in \mathbf{GEN}(\mathrm{A})} Score(W, \boldsymbol{\lambda}) \tag{8}$$

Assume that we can measure the number of conversion errors in W by comparing it with a reference transcript W^R using an error function Er(W^R, W), which is an edit distance in our case. We call the sum of conversion errors over the training data as *sample risk* (SR). Discriminative training methods strive to minimize the SR by optimizing the model parameters, as defined in Equation (9).

$$\boldsymbol{\lambda}^* = \arg\min_{\boldsymbol{\lambda}} SR(\boldsymbol{\lambda}) = \arg\min_{\boldsymbol{\lambda}} \sum_{i=1 \ldots M} Er(W_i^R, W_i(A_i, \boldsymbol{\lambda})) \tag{9}$$

However, SR(.) cannot be optimized easily since Er(.) is a piecewise constant (or step) function of $\boldsymbol{\lambda}$ and its gradient is undefined. Therefore, discriminative methods apply different approaches that optimize it approximately. As we shall describe below, the boosting and perceptron algorithms approximate SR(.) by loss functions that are suitable for optimization, while MSR uses a simple heuristic training procedure to minimize SR(.) directly without applying any approximated loss function. We now describe each of the discriminative methods in turn.

The boosting algorithm [2] uses an exponential function to approximate SR(.). We define a ranking error in a case where an incorrect candidate conversion W gets a higher score than the correct conversion W^R. The margin of the pair (W^R, W) with respect to the model $\boldsymbol{\lambda}$ is estimated as

$$M(W^R, W) = Score(W^R, \boldsymbol{\lambda}) - Score(W, \boldsymbol{\lambda}) \tag{10}$$

Then we define an upper bound to the number of ranking errors as the loss function,

$$\mathrm{ExpLoss}(\boldsymbol{\lambda}) = \sum_{i=1 \ldots M} \sum_{W_i \in \mathbf{GEN}(A_i)} \exp(-M(W_i^R, W_i)) \tag{11}$$

Now, ExpLoss(.) is convex with respect to $\boldsymbol{\lambda}$, so there are no problems with local minima when optimizing it. The boosting algorithm can be viewed as an iterative feature selection method: at each iteration, the algorithm selects from all possible features the one that is estimated to have the largest impact on reducing the ExpLoss function with respect to the current model, and then optimizes the current model by adjusting only the parameter of the selected feature while keeping the parameters of other features fixed.

The perceptron algorithm [3] can be viewed as a form of incremental training procedure that optimizes a *minimum square error* (MSE) loss function, which is an approximation of SR(.). As shown in Figure 1, it starts with an initial parameter setting and adapts it each time a training sample is wrongly converted.

```
1    Initialize all parameters in the model, i.e. $\lambda_0 = 1$ and $\lambda_d = 0$ for $d=1\ldots D$
2    For $t = 1\ldots T$, where $T$ is the total number of iterations
         For each training sample $(A_i, W_i^R)$, $i = 1\ldots M$
             Use current model $\boldsymbol{\lambda}$ to choose some $W_i$ from GEN($A_i$) by Equation (8)
             For $d = 1 \ldots D$
                 $\lambda_d = \lambda_d + \eta(f_d(W_i^R) - f_d(W_i))$, where $\eta$ is the size of the learning step
```

Fig. 1. The standard perceptron algorithm with delta rule

In our experiments, we used the average perceptron algorithm in [3], a simple refinement to the algorithm in Figure 1, which has been proved to be more robust. Let $\lambda_d^{t,i}$ be the value for the *d*th parameter after the *i*th training sample has been processed in pass *t* over the training data. Then the "average parameters" are defined as in Equation (12).

$$(\lambda_d)_{avg} = \left(\sum_{t=1}^{T}\sum_{i=1}^{M}\lambda_d^{t,i}\right)/(T \cdot M) \tag{12}$$

The minimum sample risk (MSR) method [4] can be viewed as a greedy stage-wise learning algorithm that minimizes the sample risk SR(.) directly as it appears in Equation (9). Similar to the boosting method, it is an iterative procedure. In each iteration, MSR selects a feature that is estimated to be most effective in terms of reducing SR(.), and then optimizes the current model by adjusting the parameters of the selected feature. MSR, however, differs from the boosting method in that MSR tries to optimize the sample risk directly while the boosting optimizes the loss function that is an upper bound of the sample risk.

As mentioned earlier, SR(.) can be optimized using regular gradient-based optimization algorithms. MSR therefore uses a particular implementation of line search, originally proposed in [11], to optimize the current model by adjusting the parameter of a selected feature while keeping other parameters fixed.

Assuming f_d is the selected feature, its parameter λ_d is optimized by line search as follows. Recall that Er(W^R,W) is the function that measures the number of conversion errors in W versus its reference transcript W^R. The value of SR(.) is the sum of Er(.) over all training samples. For each A in training set, let **GEN**(A) be the set of *n*-best candidate word strings that could be converted from A. By adjusting λ_d, we obtain for each training sample an ordered sequence of λ_d intervals. For λ_d in each interval, a particular candidate would be selected according to Equation (8). Then the corresponding Er(.) is associated with the interval. As a result, for each training sample, we obtain a sequence of λ_d intervals and their corresponding Er(.) values. By combining the sequences over all training samples, we obtain a global sequence of λ_d intervals, each of which is associated with a SR(.) value. Therefore we can find the optimal interval of λ_d as well as its corresponding sample risk by traversing the sequence and taking the center of the interval as the optimal value of $\lambda_{d.}$

5 Experimental Results

5.1 Data

The data used in our experiments stem from five distinct sources of text. A 36-million-word *Nikkei* newspaper corpus was used as the background domain. We used four adaptation domains: *Yomiuri* (newspaper corpus), *TuneUp* (balanced corpus containing newspaper and other sources of text), *Encarta* (encyclopedia) and *Shincho* (collection of novels).

For the computation of domain characteristics (Section 5.2), we extracted 1 million words from the training data of each domain respectively (corresponding to 13K to 78K sentences depending on the domain). For this experiment, we also used a lexicon consisting of the words in our baseline lexicon (167,107 words) plus all words in the corpora used for this experiment (that is, 1M words times 5 domains), which included 216,565 entries. The use of such a lexicon was motivated by the need to eliminate the effect of out-of-vocabulary (OOV) items.

For the experiment of LM adaptation (Section 5.3), we created training data consisting of 72K sentences (0.9M~1.7M words) and test data of 5K sentences (65K~120K words) from each adaptation domain. The first 800 and 8,000 sentences of each adaptation training data were also used to show how different sizes of adaptation training data affected the performances of various adaptation methods. Another 5K-sentence subset was used as held-out data for each domain. For domain adaptation experiments, we used our baseline lexicon consisting of 167,107 entries.

5.2 Computation of Domain Characteristics

The first domain characteristic we computed was the similarity between two domains for the task of LM. As discussed in Section 3, we used the cross entropy as the metric: we first trained a word trigram model using the system described in [12] on the 1-million-word corpus of domain B, and used it in the computations of the cross entropy $H(L_A, q_B)$ following equation (3). For simplicity, we denote $H(L_A, q_B)$ as $H(A,B)$.

Table 1 displays the cross entropy between two domains of text. Note that the cross entropy is not symmetric, i.e., $H(A,B)$ is not necessarily the same as $H(B,A)$. In order to have a representative metric of similarity between two domains, we computed the *average cross entropy* between two domains, shown in Table 2, and used this quantity as the metric for domain similarity.

Along the main diagonal in the tables below, we also have the cross entropy computed for $H(A,A)$, i.e., when two domains we compare are the same (in boldface). This value, which we call *self entropy* for convenience, is an approximation of the entropy of the corpus A, and measures the amount of information per word, i.e., the *diversity* of the corpus. Note that the self entropy increases in the order of Nikkei → Yomiuri → Encarta → TuneUp → Shincho. This indeed reflects the in-domain variability of text: Nikkei, Yomiuri and Encarta are highly edited text, following style guidelines; they also tend to have repetitious content. In contrast, Shincho is a collection of novels, on which no style or content restriction is imposed. We expect that the LM task to be more difficult as the corpus is more diverse; we will further discuss the effect of diversity in Section 6.[3]

	Nikkei	Yomiuri	TuneUp	Encarta	Shincho
Nikkei	**3.94**	7.46	7.65	9.81	10.10
Yomiuri	7.93	**4.09**	7.62	9.26	9.97
TuneUp	8.25	8.03	**4.41**	9.04	9.06
Encarta	8.79	8.66	8.60	**4.40**	9.30
Shincho	8.70	8.61	8.07	9.10	**4.61**

Table 1. Cross entropy (rows: corpora; column: models)

	Nikkei	Yomiuri	TuneUp	Encarta	Shincho
Nikkei	**3.94**	7.69	7.95	9.30	9.40
Yomiuri		**4.09**	7.82	8.96	9.29
TuneUp			**4.41**	8.82	8.56
Encarta				**4.40**	9.20
Shincho					**4.61**

Table 2. Average cross entropy

5.3 Results of LM Adaptation

We trained our baseline trigram model on our background (Nikkei) corpus using the system described in [12]. The CER (%) of this model on each adaptation domain is in the second column of Table 3. For the LI adaptation method (the third column of Table 3), we trained a word trigram model on the adaptation data, and linearly combined it with the background model, as described in Equation (6).

For the discriminative methods (the last three columns in Table 3), we produced a candidate word lattice for each input phonetic string in the adaptation training set using the background trigram model mentioned above. For efficiency purposes, we kept only the best 20 hypotheses from the lattice as the candidate conversion set for discriminative training. The lowest CER hypothesis in the lattice, rather than the reference transcript, was used as the gold standard.

To compare the performances of different discriminative methods, we fixed the following parameter settings: we set the number of iterations N to be 2,000 for the boosting and MSR methods (i.e., at most 2,000 features in the final models); for the perceptron algorithm, we set T = 40 (in Figure 1). These settings might lead to an

[3] Another derivative notion from Table 1 is the notion of *balanced* corpus. In Table 1, the smallest cross entropy for each text domain (rows) is the self entropy (in boldface), as expected. Note, however, that the second smallest cross entropy (underlined) is always obtained from the TuneUp model (except for Nikkei, for which Yomiuri provides the second smallest cross entropy). This reflects the fact that the TuneUp corpus was created by collecting sentences from various sources of text, in order to create a representative test corpus. Using the notion of cross entropy, such a characteristic of a test corpus can also be quantified.

Domain	Baseline	LI	Boosting	Perceptron	MSR
Yomiuri (800)	3.70	3.70 (0.00%)	3.13 (15.41%)	3.18 (14.05%)	3.17 (14.32%)
Yomiuri (8K)	3.70	3.69 (0.27%)	2.88 (22.16%)	2.85 (22.97%)	2.88 (22.16%)
Yomiuri (72K)	3.70	3.69 (0.27%)	2.78 (24.86%)	2.78 (24.86%)	2.73 (26.22%)
TuneUp (800)	5.81	5.81 (0.00%)	5.69 (2.07%)	5.69 (2.07%)	5.70 (1.89%)
TuneUp (8K)	5.81	5.70 (1.89%)	5.47 (5.85%)	5.47 (5.85%)	5.47 (5.85%)
TuneUp (72K)	5.81	5.47 (5.85%)	5.33 (8.26%)	5.20 (10.50%)	5.15 (11.36%)
Encarta (800)	10.24	9.60 (6.25%)	9.82 (4.10%)	9.43 (7.91%)	9.44 (7.81%)
Encarta (8K)	10.24	8.64 (15.63%)	8.54 (16.60%)	8.34 (18.55%)	8.42 (17.77%)
Encarta (72K)	10.24	7.98 (22.07%)	7.53 (26.46%)	7.44 (27.34%)	7.40 (27.73%)
Shincho (800)	12.18	11.86 (2.63%)	11.91 (2.22%)	11.90 (2.30%)	11.89 (2.38%)
Shincho (8K)	12.18	11.15 (8.46%)	11.09 (8.95%)	11.20 (8.05%)	11.04 (9.36%)
Shincho (72K)	12.18	10.76 (11.66%)	10.25 (15.85%)	10.18 (16.42%)	10.16 (16.58%)

Table 3. CER (%) and CER reduction over Baseline

unfair comparison, as the perceptron algorithm will select far more features than the boosting and MSR algorithm. However, we used these settings as they all converged under these settings. All other parameters were tuned empirically.

In evaluating both MAP and discriminative methods, we used an N-best rescoring approach. That is, we created N best hypotheses using the background trigram model (N=100 in our experiments) for each sentence in test data, and used domain-adapted models to rescore the lattice. The oracle CERs (i.e., the minimal possible CER given the hypotheses in the lattice) ranged from 1.45% to 5.09% depending on the adaptation domain. Table 3 below summarizes the results of various adaptation methods in terms of CER (%) and CER reduction (in parentheses) over the baseline model. In the first column, the numbers in parentheses next to the domain name indicates the number of training sentences used for adaptation.

6 Discussion

6.1 Domain Similarity and CER

The first row of Table 2 shows that the average cross entropy with respect to the background domain (Nikkei) increases in the following order: Yomiuri → TuneUp → Encarta → Shincho. This indicates that among the adaptation domains, Yomiuri is the most similar to Nikkei, closely followed by TuneUp; Shincho and Encarta are the least similar to Nikkei. This is consistent with our intuition, since Nikkei and Yomiuri are both newspaper corpora, and TuneUp, which is a manually constructed corpus from various representative domains of text, contains newspaper articles.

This metric of similarity correlates perfectly with the CER. In Table 3, we see that for all sizes of training data for all adaptation methods, the following order of CER performance is observed, from better to worse: Yomiuri → TuneUp → Encarta →

Shincho. In other words, the more similar the adaptation domain is to the background domain, the better the CER results are.

6.2 Domain Similarity and the Effectiveness of Adaptation Methods

The effectiveness of a LM adaptation method is measured by the relative CER reduction over the baseline model. Figure 3 shows the CER reduction of various methods for each domain when the training data size was 8K.[4]

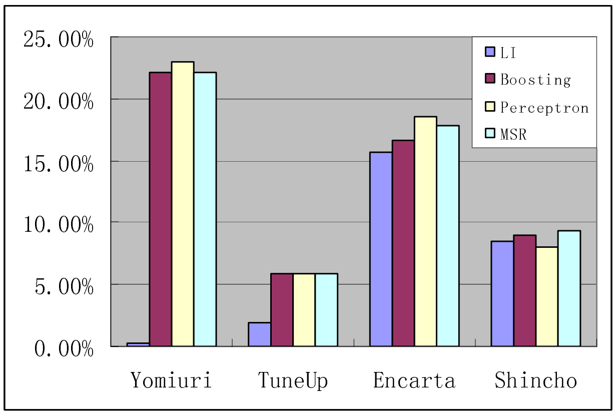

Fig. 2. CER reduction by different adaptation methods

In Figure 3, the X-axis is arranged in the order of domain similarity with the background domain, i.e., Yomiuri $\rightarrow$ TuneUp $\rightarrow$ Encarta $\rightarrow$ Shincho. The first thing we note is that the discriminative methods outperform LI in all cases: in fact, for all rows in Table 3, MSR outperforms LI in a statistically significant manner ($p < 0.01$ using *t*-test);[5] the differences among the three discriminative methods, on the other hand, are not statistically significant in most cases.

We also note that the performance of LI is greatly influenced by domain similarity. More specifically, when the adaptation domain is similar to the background domain (i.e., for Yomiuri and TuneUp corpora), the contribution of the LI model is extremely limited. This can be explained as follows: if the adaptation data is too similar to the background, the difference between the two underlying distributions is so slight that adding adaptation data leads to no or very small improvements.

Such a limitation is not observed with the discriminative methods. For example, all discriminative methods are quite effective on Yomiuri, achieving more than 20% CER reduction. We therefore conclude that discriminative methods, unlike LI, are robust against the similarity between background and adaptations domains.

[4] Essentially the same trend is observed with other training data sizes.
[5] The only exception to this was Shincho (800).

6.3 Adaptation Data Size and CER Reduction

We have seen in Table 3 that in all cases, discriminative methods outperform LI. Among the discriminative methods, an interesting characteristic regarding the CER reduction and the data size is observed. Figure 3 displays the self entropy of four adaptation corpora along the X-axis, and the improvement in CER reduction when 72K-sentence adaptation data is used over when 800 sentences are used along the Y-axis. In other words, for each adaptation method, each point in the figure corresponds to the CER reduction ratio on a domain (corresponding to Yomiuri, Encarta, TuneUp, Shincho from left to right) when 90 times more adaptation data was available.

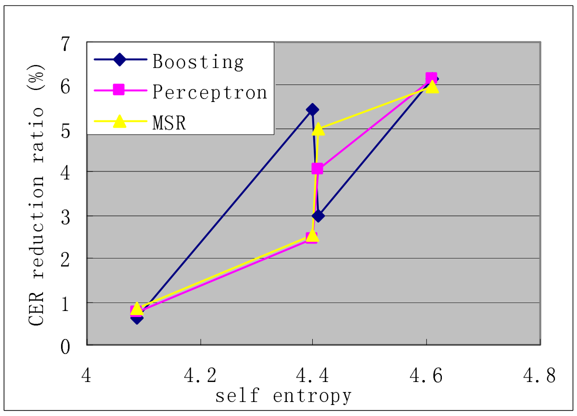

Fig. 3. Improvement in CER reduction for discriminative methods by increasing the adaptation data size from 800 to 72K sentences

From this figure, we can see that there is a positive correlation between the diversity of the adaptation corpus and the benefit of having more training data available. This has an intuitive explanation: the less diverse the adaptation data is, the less distinct training examples it will include for discriminative training. This result is useful in guiding the process of adaptation data collection.

7 Conclusion and Future Work

In this paper, we have examined the performance of various LM adaptation methods in terms of domain similarity and diversity. We have found that (1) the notion of cross-domain similarity, measured by the cross entropy, correlates with the CER of all models (Section 6.1), and (2) the notion of in-domain diversity, measured by the self entropy, correlates with the utility of more adaptation training data for discriminative training methods (Section 6.3). In comparing discriminative methods with a MAP-based method, we have also found that (1) the former uniformly achieve better CER performance than the latter, and (2) are more robust against the similarity of background and adaptation data (Section 6.2).

Though we believe these results are useful in designing the future experiments in domain adaptation, some results and correlations indicated in the paper are still inconclusive. We hope to run additional experiments to confirm these findings. We also did not fully investigate into characterizing the differences among the three discriminative methods; such an investigation is also left for future research.

References

1. Bellagarda, J. An Overview of Statistical Language Model Adaptation. In ITRW on Adaptation Methods for Speech Recognition (2001): 165-174.
2. Collins, M. Ranking Algorithms for Name-Entity Extraction: Boosting and the Voted Perceptron. ACL (2002).
3. Collins, M. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. EMNLP (2002).
4. Gao. J., H. Yu, P. Xu, and W. Yuan. Minimum Sample Risk Methods for Language Modeling. To Appear (2005).
5. Manning, C.D., and H. Schütze. Foundations of Statistical Natural Language Processing. The MIT Press (1999).
6. Dagan, I., L. Lee, and F. Pereira. Similarity-based models of cooccurrence probabilities. Machine Learning, 34(1-3): 43-69 (1999).
7. Lee, L. Measures of distributional similarity. ACL (1999): 25-32.
8. Roark, B, M. Saraclar and M. Collins. Corrective Language Modeling for Large Vocabulary ASR with the Perceptron Algorithm. ICASSP (2004): 749-752.
9. Bacchiani, M., B. Roark and M. Saraclar. Language Model Adaptation with MAP Estimation and the Perceptron Algorithm. HLT-NAACL (2004): 21-24.
10. Bacchiani, M. and B. Roark. Unsupervised language model adaptation. ICASSP (2003): 224-227
11. Och, F.J. Minimum error rate training in statistical machine translation. ACL (2003): 160-167.
12. Gao, J., J. Goodman, M. Li, and K.F. Lee. Toward a unified approach to statistical language modeling for Chinese. ACM Transactions on Asian Language Information Processing l-1 (2002): 3-33.