INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Efficient MDI Adaptation for n-gram Language Models

Ruizhe Huang[1], *Ke Li*[1], *Ashish Arora*[1], *Daniel Povey*[2], *Sanjeev Khudanpur*[1]

[1]Center for Language and Speech Processing, The Johns Hopkins University, Baltimore, USA
[2]Xiaomi Corporation, Beijing, China

{huangruizhe09, ashish.arora.88888, dpovey}@gmail.com, {kli26, khudanpur}@jhu.edu

Abstract

This paper presents an efficient algorithm for n-gram language model adaptation under the minimum discrimination information (MDI) principle, where an out-of-domain language model is adapted to satisfy the constraints of marginal probabilities of the in-domain data. The challenge for MDI language model adaptation is its computational complexity. By taking advantage of the backoff structure of n-gram model and the idea of hierarchical training method, originally proposed for maximum entropy (ME) language models [1], we show that MDI adaptation can be computed in linear-time complexity to the inputs in each iteration. The complexity remains the same as ME models, although MDI is more general than ME. This makes MDI adaptation practical for large corpus and vocabulary. Experimental results confirm the scalability of our algorithm on large datasets, while MDI adaptation gets slightly worse perplexity but better word error rates compared to simple linear interpolation.

Index Terms: speech recognition, language model adaptation, n-gram, maximum entropy model, MDI

1. Introduction

The n-gram language model (LM) still plays an important role in today's automatic speech recognition (ASR) pipeline. There are several reasons: (i) n-gram LMs can be represented by weighted finite-state transducers (WFST) and integrated into first-pass decoding [2], (ii) training and querying n-gram LMs are cheaper than neural LMs, (iii) in practice, the best performance is achieved by interpolating n-gram and neural LMs [3].

Consider a common scenario when one is developing an ASR system for a new application, while little training data is available and collecting sufficient domain-specific (*in-domain*) data requires a considerable amount of time and efforts. The available data is too limited to estimate a robust LM. Fortunately, we can do better by capitalizing on some large, general-domain background (*out-of-domain*) corpus assuming the out-of-domain data contains information common with the application domain. This motivates LM adaptation [4, 5], which is to estimate a robust LM based on both in- and out-of-domain data.

The question is how to combine information from the two sources in a suitable manner? The commonly used approaches for n-gram LMs fall under two categories: model interpolation and constraint-based methods. The model interpolation methods can be either linear (simple linear [6], history-dependent [7, 8], Bayesian [9] interpolation) or non-linear (log-linear [10] interpolation, fill-up technique [11]). Note that simple linear interpolation is very effective and probably the most popular adaptation method. Recently, [9] found that count-merging, as a special case of maximum a posterior (MAP) adaptation, is theoretically similar to Bayesian interpolation. On the other hand, the constraint-based methods [12], such as ME or MDI models, attempt to choose the adapted LM such that it satisfies some constraints in the adaptation domain, while staying as close as possible to some prior distribution, measured by, *e.g.*, Kullback-Leibler distance. This paper investigates the MDI adaptation.

There has been previous work on MDI adaptation for n-gram LMs [13, 14, 15, 16, 17, 18, 19, 20, 21, 22], with several variants of task definition, *e.g.*, adaptation for cache model, within- or cross-corpus adaptation. Although MDI has appealing theoretical properties, the computation is non-trivial and expensive, which grows almost exponentially (detailed in section 3) with the size of the vocabulary [5] in a naive implementation. To reduce the complexity, [19, 17] proposed approximation algorithms and [23, 24] devised parallelization to speed up the computation. [18] proposed a linear-time algorithm for unigram constraints. On the other hand, there has been work on ME model that utilizes the back-off structure of the LMs to reduce computational complexity to linear time per iteration [1], but it is not clear whether the same trick can carry over to MDI which is more general than ME. Besides, regarding model performance, most previous work has found that MDI adaptation performs slightly worse than simple linear interpolation [21], but we are interested to see if there can be any difference when operating on very large corpus once we have an efficient MDI algorithm for arbitrary marginal distribution constraints. Moreover, in the experiment, we will propose a novel approach of applying MDI adaptation to improve the first-pass LM while keeping the model size unchanged.

2. Background

2.1. n-gram Language Model

A language model (LM) is a probability distribution over word sequences $W = w_1 w_2 \ldots w_L$, usually reduced to a word-by-word probability via the chain rule $p(W) = \prod_{i=1}^{L} p(w_i|w_1^i)$, where $w_i^j \equiv w_i, w_{i+1}, \ldots, w_j$. An n-gram LM assumes that this distribution depends only on the previous $n-1$ words, *i.e.*, $p(w_i|w_1^i) \approx p(w_i|w_{i-n+1}^{i-1})$, where w_{i-n+1}^{i-1} is the history h_i of word w_i. We omit the index i when the context is clear.

Given the vocabulary V, n-gram LM defines a set of conditional probabilities $p(w|h)$ for any $hw \in V^n$. However, the space V^n is very large that not every n-gram hw is seen in the training data, known as the data sparsity problem. Thus, smoothing techniques have been used to estimate the probability $p(w|h)$ for the unseen n-grams. The most popular techniques is backing-off. The idea is to recursively estimate $p(w|h)$ of unseen n-grams based on the lower order $(n-1)$-gram probabilities $p(w|h')$, where $h' = w_{i-n+2}^{i-1}$, which may have been seen in the corpus. More specifically,

$$p(w|h) = \begin{cases} p^*(w|h) & hw \text{ is seen in corpus} \\ bow(h) \cdot p(w|h') & \text{otherwise,} \end{cases} \quad (1)$$

where the discounted probability $p^*(w|h)$ and the back-off weight $bow(h)$ are together to ensure the conditional probabil-

http://dx.doi.org/10.21437/Interspeech.2020-2909

ity sums to one: $\sum_{w \in V} p(w|h) = 1$. We will consider n-gram LMs having back-off structure in the rest of the paper. In practice, such LMs are stored in ARPA format [25]. Note that LMs smoothed by interpolation [25] can also be stored as ARPA. We measure the size of an n-gram LM as the total number of entries of order $1, \ldots, n$ when the LM is represented as ARPA.

2.2. MDI Adaptation

The idea of LM adaptation under the minimum discrimination information (MDI) principle is to compute the adapted distribution such that it satisfies the constraints characterizing in-domain distribution, and also stays closest to the out-of-domain distribution. The constraints are usually expressed as marginal distributions. Formally, given (i) the vocabulary V, (ii) the out-domain LM $p_{out}(w|h)$, (iii) the empirical history distribution $\tilde{p}(h)$ which is commonly approximated by either the in-domain probabilities $p_{in}(h)$ or out-of-domain $p_{out}(h)$, and (iv) K marginal distributions $\tilde{p}(S_i)$ where $S_i \subset V^n, i = 1, \ldots, K$ derived from the in-domain data, the adapted LM $p_{ad}(w|h)$ is defined by minimizing the following conditional Kullback-Liebler (KL) divergence:

$$p_{ad}(w|h) = \arg\min_p D(p||p_{out}|\tilde{p}) \tag{2}$$

$$= \arg\min_p \sum_{h \in V^{n-1}} \tilde{p}(h) \sum_{w \in V} p(w|h) \log \frac{p(w|h)}{p_{out}(w|h)}, \tag{3}$$

while satisfying the constraints:

$$\sum_{h \in V^{n-1}} \tilde{p}(h) \sum_{w \in V} p(w|h) f_i(h, w) = \tilde{p}(S_i), \; i = 1, \ldots, K. \tag{4}$$

where f_i are indicator functions of $(h, w) \in S_i$. Note that Eq. 3 can also be viewed as the KL divergence between the joint distribution $p(h, w)$ and $p_{out}(h, w)$ assuming they have the same history distribution $\tilde{p}(h)$. Also notice in the case that p_{out} is the uniform distribution, p_{ad} is indeed a maximum entropy model.

If the constraints in Equation 4 are consistent, the solution of the above optimization problem exists and is unique [26]. It has the following form, with parameters $\{\lambda_i\}$.

$$p_{ad}(w|h) = \frac{p_{out}(w|h) \cdot \alpha(h, w)}{Z(h, \lambda_1, \ldots, \lambda_K)}, \tag{5}$$

where the scaling factor $\alpha(h, w) = \exp\left(\sum_{i=1}^{K} \lambda_i f_i(h, w)\right)$, and $Z(h, \lambda_1, \ldots, \lambda_K)$ is the normalization term summing up the numerators. This solution can be obtained by generalized iterative scaling (GIS) algorithm [26], sketched in Algorithm 1, or some of its modern fast counterparts [24]. The iterations can be terminated when the results converge or nearly converge.

3. Efficient Algorithm: The Hierarchical Training Method

The challenge for implementing the above GIS algorithm is its computational complexity resulting from Line 5 (normalization) and 9 (marginalization). A naive implementation may take $O(K * \#\text{ of seen histories} * |V|)$ time per iteration [5]. An improvement can be made to $O(\#\text{ of seen histories} * |V| + K)$ if we store the constraints in Line 8 in a hash table and accumulate the summation in Line 9, but this complexity is still astronomical when the corpus and vocabulary V is large. Thus, we need to re-organize the summation happening in Line 5 and 9.

Algorithm 1 Generalized Iterative Scaling (GIS) Algorithm

Require: V, $p_{out}(w|h)$, $\tilde{p}(h)$ and $\tilde{p}(S_i)$

1: Set $\lambda_1^{(0)} = \lambda_2^{(0)} = \ldots = \lambda_K^{(0)} = 0, n = 0$
2: **while** stopping criterion not met **do**
3: Compute $\alpha^{(n)}(h, w) = \exp\left(\sum_{i=1}^{K} \lambda_i^{(n)} f_i(h, w)\right)$
4: **for** each seen history h in training data **do**
5: Compute normalization term: $Z(h, \lambda_1^{(n)}, \ldots, \lambda_K^{(n)}) \coloneqq \sum_w p_{out}(w|h) \alpha^{(n)}(h, w)$
6: **end for**
7: Update each entry of back-off LM: $p^{(n)}(w|h) \coloneqq \frac{p_{out}(w|h) * \alpha^{(n)}(h,w)}{Z(h, \lambda_1^{(n)}, \ldots, \lambda_K^{(n)})}$
8: **for** $j = 1, \cdots, K$ **do**
9: Marginalize: $p^{(n)}(S_j) \coloneqq \sum_h \tilde{p}(h) \sum_w p^{(n)}(w|h) f_j(h, w)$
10: Update params: $\lambda_j^{(n+1)} \coloneqq \lambda_j^{(n)} + \log \frac{\tilde{p}(S_j)}{p^{(n)}(S_j)}$
11: **end for**
12: $n \coloneqq n + 1$
13: **end while**
14: **return** $p^{(n)}(w|h)$ as p_{ad}

3.1. The Hierarchical Training Method: MDI v.s. ME

To overcome this challenge, [1] proposed the hierarchical training method for ME models. The algorithm only requires linear time to its inputs per iteration. The trick is based on the back-off structure of probability $p^{(n)}(w|h)$. In this paper, we show that similar algorithmic trick can be applied to MDI with additional cares. This means MDI adaptation incurs no extra computation complexity although it is more general than ME. The key is to handle the non-uniform p_{out} appropriately.

To illustrate the idea, we take trigram LM as an example. Consider a general LM $p(w_3|w_1^2)$ having the back-off structure as in Equation 1. We also define a set of real-valued scaling factors $c(\cdot)$, such that there are only K non-zero $c(hw) - c(h'w)$. Now, we define the *left-aligned* and *right-aligned* summation of the product of $p(w_3|w_1^2)$ and the general scaling factor $c(w_1^3)$:

$$\Sigma_L(w_1^2) = \sum_{w_3 \in V} p(w_3|w_1^2) \cdot c(w_1^3) \tag{6}$$

$$\Sigma_R(w_2^3) = \sum_{w_1 \in V} p(w_3|w_1^2) \cdot c(w_1^3) \tag{7}$$

Two equations only differ in the term summed over. We will show later that Σ_L and Σ_R are related to the normalization term (Line 5) and marginal probability (Line 9) respectively. Notice that in ME, $p(w_3|w_1^2)$ is just a uniform distribution.

3.2. Computing $\Sigma_L(w_1^2)$ as normalization for history w_1^2

In Equation 6, we compute $\Sigma_L(w_1^2)$ by summing over $w_3 \in V$ given history w_1^2, which costs $|V|$ addition operations. This cost can be greatly reduced by dynamic programming [18, 10]. One can verify that Eq. 6 can be re-written as:

$$\begin{aligned}
\Sigma_L(w_1^2) = & \sum_{w_3 \in V \wedge \text{seen}(w_1^3)} \left(p^*(w_3|w_1^2) - bow(w_1^2) \cdot p(w_3|w_2)\right) \cdot c(w_1^3) \\
& + bow(w_1^2) \sum_{w_3 \in V} p(w_3|w_2) \cdot c(w_2^3) \\
& + bow(w_1^2) \sum_{w_3 \in V \wedge c(w_1^3) \neq c(w_2^3)} p(w_3|w_2) \cdot \left(c(w_1^3) - c(w_2^3)\right).
\end{aligned}$$

For the second term above, we can define $\Sigma_L(w_2) = \sum_{w_3 \in V} p(w_3|w_2) \cdot c(w_2^3)$, which is a sub-problem for $\Sigma_L(w_1^2)$. And the base case is $\Sigma_L(\emptyset) = \sum_{w_3 \in V} p(w_3) \cdot c(w_3)$. Thus, $\Sigma_L(w_1^2)$ can be computed hierarchically and bottom-up from $\Sigma_L(\emptyset)$ along the back-off structure of $p(w_3|w_1^2)$.

As of complexity, computing $\Sigma_L(\emptyset)$ requires $O(|V|)$ time for only once. $\Sigma_L(w_1^2)$ and $\Sigma_L(w_2)$ can be computed from the Σ_L of the lower-order n-grams with the complexity of the number of seen n-grams along the way, plus the number of non-zero difference of scaling factors (K at most). Thus, overall, the total time is proportional to the number of seen entries in $p_{out}(w_3|w_1^2)$ plus $O(K)$.

3.3. Computing Σ_R

Consider Eq. 8, where we compute the right-aligned $\Sigma_R(w_2^3)$ by summing over $w_1 \in V$. Similarly, we define the lower or higher order right-aligned summations as follows:

$$\Sigma_R(w_3) = \sum_{w_1^2 \in V^2} p(w_3|w_1^2) \cdot c(w_1^3) \quad (8)$$

$$\Sigma_R(w_1^3) = p(w_3|w_1^2) \cdot c(w_1^3) \quad (9)$$

One can decompose the computation as in the previous section and verify that dynamic programming does not work here anymore. Instead, we will make use of the idea of *shared computation*, which means we go over the data for only one pass, but we accumulate the values correspondingly to all related constraints.

First, let us compute $\Sigma_R(w_2^3)$ as follows:

$$\begin{aligned}\Sigma_R(w_2^3) = & \sum_{w_1 \in V \wedge \text{seen}(w_1^3)} \big(p^*(w_3|w_1^2) - bow(w_1^2) \cdot p(w_3|w_2)\big) \cdot c(w_1^3) \\ & + \quad p(w_3|w_2) \cdot c(w_2^3) \cdot g(w_2) \\ & + \sum_{w_1 \in V \wedge c(w_1^3) \neq c(w_2^3)} p(w_3|w_2) \cdot bow(w_1^2) \cdot \big(c(w_1^3) - c(w_2^3)\big)\end{aligned}$$

We denote the auxiliary function $g(w_2) = \sum_{w_1 \in V} bow(w_1^2)$ in the second term above and will address its computation later. At the same time, let us decompose $\Sigma_R(w_3)$ in the same way:

$$\begin{aligned}\Sigma_R(w_3) = & \sum_{w_1^2 \in V^2 \wedge \text{seen}(w_1^3)} \big(p^*(w_3|w_1^2) - bow(w_1^2) \cdot p(w_3|w_2)\big) \cdot c(w_1^3) \\ & + \sum_{w_2 \in V \wedge \text{seen}(w_2^3)} (p^*(w_3|w_2) - bow(w_2) \cdot p(w_3)) \cdot c(w_2^3) \cdot g(w_2) \\ & + \quad p(w_3) \cdot c(w_3) \cdot g(\emptyset) \\ & + \sum_{w_2 \in V \wedge c(w_2^3) \neq c(w_3)} p(w_3) \cdot bow(w_2) \cdot g(w_2) \cdot \big(c(w_2^3) - c(w_3)\big) \\ & + \sum_{w_1^2 \in V^2 \wedge c(w_1^3) \neq c(w_2^3)} p(w_3|w_2) \cdot bow(w_1^2) \cdot \big(c(w_1^3) - c(w_2^3)\big)\end{aligned}$$

We denote $g(\emptyset) = \sum_{w_2 \in V} bow(w_2) \cdot g(w_2)$. Assuming the values of $g(\cdot)$ are known, we can compute $\Sigma_R(w_3)$, $\Sigma_R(w_2^3)$ and $\Sigma_R(w_1^3)$ at the same time: the algorithm enumerates the seen n-grams in LM $p(w_3|w_1^2)$ and all non-zero differences of scaling factors and accumulate values to the corresponding right-aligned sums that are involved. Details can be found in our full paper [27]. The complexity is $O(n *$ # of entries in $p(w|h) + K)$, with n being a small constant.

Now, the remaining problem is how to compute the auxiliary function $g(\cdot)$ as defined previously. It turns out this is a right-aligned sum in the ME case. More specifically, their scaling factors are $c(w_1^2) = bow(w_1^2)$ or $c(w_1^2) = bow(w_1^2) * bow(w_2)$. Thus, computing $g(\cdot)$ can be shown to be also in linear time. In fact, computing the auxiliary function $g(\cdot)$ is what makes the algorithm for MDI different from that of ME.

3.4. The back-off structure of $p_{ad}(w|h)$

Before computing the marginals in Line 9 of Algorithm 1, we still need to show that the probability $p^{(n)}(w|h)$ or $p_{ad}(w|h)$ in Equation 5 has the back-off structure. This is important not only for the computational purpose – so that the tricks for Σ_L and Σ_R can be applied here – but also for being able to represent the final adapted LM in ARPA format.

We claim that, if p_{out} is a back-off model as in Equation 1, then so is the exponential models $p^{(n)}$ and p_{ad}. We prove this by giving the back-off expression of p_{ad}:

$$p_{ad}(w_3|w_1^2) = \begin{cases} p^*_{ad}(w_3|w_1^2) & \text{if } w_1^3 \text{ seen in } p_{out} \\ & \text{or } w_1^3 \text{ is a constraint} \\ bow_{ad}(w_1^2) \cdot p_{ad}(w_3|w_2) & \text{otherwise} \end{cases}$$

where:

$$p^*_{ad}(w_3|w_1^2) = \frac{p_{out}(w_3|w_1^2) \cdot c(w_1^3)}{Z(w_1^2)}$$

$$bow^*_{ad}(w_1^2) = \frac{Z(w_2)}{Z(w_1^2)} \cdot bow_{out}(w_1^2)$$

The lower order n-grams of p_{ad} are defined analogously. There will be at most (# entries in p_{out} + # entries in p_{in}) entries in p_{ad}, same as in linear interpolation.

3.5. Computing marginalization as Σ_R

Lastly, we come to compute the marginals in Line 9 of Algo. 1:

$$p^{(n)}(S_i) \coloneqq \sum_h \tilde{p}(h) \sum_w p^{(n)}(w|h) \cdot f_i(h,w). \quad (10)$$

Since it has been proved that $p^{(n)}$ is a back-off LM, we can view $\sum_w p^{(n)}(w|h) f_i(h,w)$ as a right-aligned sum. However, we need to further consider the multiplication term $\tilde{p}(h)$. Fortunately, the same trick computing Σ_R can be applied here, with some modification of the auxiliary function. For example, let $g(w_2) = \sum_{w_1 \in V} \tilde{p}(w_1^2) bow(w_1^2)$, and then it can be treated in two ways efficiently, either (i) if $\tilde{p}(w_1^2)$ is an unsmoothed maximum likelihood estimation, there will be a lot of zeros for $\tilde{p}(w_1^2)$, or (ii) if $\tilde{p}(w_1^2)$ has a smoothed back-off (joint) distribution, then this amounts to computing the right-aligned sum. It can be shown that the computational complexity of marginalization is linear in both ways. We omit the details here due to space limit. Interested readers can refer to Appendix A at the end of our full paper [27]. In all, we have shown how Algorithm 1 can be implemented efficiently.

3.6. Implementation Issues

Special care should be taken when dealing with n-grams w_1^3 which containing <s> or <\s>, or whose suffix w_2^3 is not seen. To further speed up the computation, the algorithm can be implemented in a vectorized manner with group-by operation for summing up probabilities of n-grams of the same suffix.

4. Experimental Results

We will show the scalability of our algorithm and the effectiveness of MDI adaptation with two different ways of application.

Table 1: *Comparing the perplexity (PPL), word error rate (WER, in %) of LMs with no adaptation, interpolation and MDI adaptation.*

Corpus	Test set	First-pass LM				Rescoring with large n-gram LM										
		default		MDI		No adapt.	Interpolation		MDI (2-2-2)		MDI (5-3-2)		MDI (6-4-3)	
		PPL	WER	PPL	WER	PPL	PPL	WER	PPL	WER	PPL	WER	PPL	WER
AMI	dev	84.6	20.0	84.3	20.0	384.1	80.5	19.6	86.6	19.4	87.1	19.4	87.9	**19.4**
	eval	79.7	20.2	79.9	20.2	408.8	77.5	20.0	81.8	19.6	82.8	19.6	83.9	**19.6**
SWBD	dev	98.6	12.5	96.9	**12.0**	411.0	92.7	**11.4**	94.5	11.7	95.1	11.7	95.8	11.6
	eval2000	179.2	14.2	117.5	**14.0**	161.6	85.6	13.4	88.9	13.2	89.4	13.2	89.4	**13.2**
	rt03	167.8	17.3	109	**17.2**	149.4	78.6	16.3	82.2	16.2	82.7	16.1	82.7	**16.1**
WSJ	dev93	186.6	7.0	161.3	**6.8**	223.3	134.2	6.3	134.8	**6.2**	135.1	6.3	136.8	6.3
	eval92	164.8	4.7	142.7	4.7	222.2	118.7	4.0	117.4	3.9	118.0	**3.9**	120.0	3.9

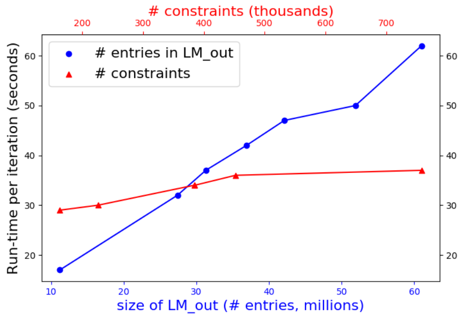

Figure 1: *Run-time of Algorithm 1 with various input sizes.*

We simulate the LM adaptation scenario by taking three speech corpora, Wall Street Journal (WSJ), Switchboard (SWBD) and AMI-IHM (AMI) as in-domain data, and Google One Billion Words [28] and Librispeech [29] as out-of-domain data. We normalize the Google dataset with the similar scripts generating normalized Librispeech LM training texts, resulting 702 million and 803 million words respectively. We compute trigram LMs using the SRILM tool [30] with Kneser-Ney smoothing and default settings, or use default Kaldi's LMs. We find the normalized Google dataset always out-perform Librispeech as the out-of-domain corpus, so we only report the results for Google dataset. We are interested whether the rich LM information in the very large corpus can help the LM and ASR task in the application domains. We use count thresholds to select the in-domain constraints, *i.e.*, the marginals are considered reliable when the counts of the n-grams is above the threshold.

4.1. Scalability

We implemented the proposed algorithm in Python with Numpy and Pandas. In Figure 1, we compare the run-time per iteration in seconds with various size of out-of-domain LM (blue) and various number of constraints (blue). The in- and out-of-domain data are taken to be SWBD and Google. We sample the out-of-domain data at different sizes measured by the total number of seen entries in the ARPA file, and record the run-time per iteration. We control the number of constraints by using different constraint count thresholds. We can see that both lines show linear scalability, and the run-time is denominated by the size of out-of-domain data. Besides, it usually takes $60 \sim 80$ iterations for the algorithm to converge to a near optimal solution, which may be improved by more advanced optimization algorithms.

4.2. Effectiveness of Adaptation

We compare the LMs with and without adaptation, and with different adaptation methods, *i.e.*, simple linear interpolation and MDI. We evaluate the LMs in perplexity (PPL) and word error rate (WER) when used in an hybrid ASR system. We use the latest recipes in the open-source speech recognition toolkit Kaldi [31] to run the ASR experiments. The acoustic model uses factorized TDNN architecture [32] and is trained with LF-MMI criteria [33]. The features are 40-dimensional MFCC with 100-dimensional i-vectors appended to the MFCC. The training data is augmented with speed and volume perturbation.

4.2.1. Performance in the Rescoring

As in a common adaptation scenario, we adapt the large, out-of-domain LM to satisfy the marginal distribution constraints derived from the in-domain data. As the Google corpus is large, the resulting LM can have $61 \sim 76$ million entries depending on the vocabulary (which is AMI 50k, SWBD 30k, WSJ 20k). So the LMs are used for rescoring. From the right half of Table 1, we can see that LM adaptation is effective for both interpolation and MDI. The interpolated LMs have better perplexity most of the time, which is consistent with previous work [15, 21], but we also find that MDI adapted LMs have better WER. Also note the constraints we use for MDI, where 5-3-2 means count thresholds of $5, 3, 2$ are used for selecting unigrams, bigrams and trigrams as constraints. Thus, in fact, MDI sees less information of the in-domain data than interpolation, *e.g.*, the MDI (6-4-3) above sees the least in-domain data. However, MDI requires additional information about the history distribution $\tilde{p}(h)$, for which we take the maximum-likelihood in-domain $p_{in}(h)$ as an approximation.

4.2.2. Performance in the First-Pass Decoding

We propose a novel approach of applying MDI adaptation: instead of adapting the out-of-domain LM, we adapt the small, in-domain LM, which is used in the first pass decoding of ASR, so that it preserves the marginals of the larger and better interpolated LM. The constraints are selected to be all seen entries in the in-domain LM, such that the model size remains unchanged after adaptation. The results are a bit surprising. As we can see in the left half of Table 1, the perplexity of the first-pass LM gets improved significantly and lies between the default and interpolated LMs. The first-pass WER also gets improved, not so much though. It seems this is the advantage of MDI over interpolation: we have better control of the resulting model size. We will investigate into this as the future work.

5. Conclusion and Future Work

In this paper, we propose an efficient MDI adaptation algorithm for n-gram LMs. The algorithm relies on the back-off structure of the LMs, and takes linear time per iteration. We show empirically our algorithm is truly scalable to very large corpus. We also find that MDI adaptation gets close perplexity to linear interpolation, but better WER. Future work may explore more advanced optimization algorithms, better feature selection and history distribution estimation techniques for MDI. Lastly, as observed in the experiments, we will study using MDI adaptation to improve small LMs for the first-pass decoding of ASR.

6. References

[1] J. Wu and S. Khudanpur, “Efficient training methods for maximum entropy language modeling,” in *Sixth International Conference on Spoken Language Processing*, 2000.

[2] M. Mohri, F. Pereira, and M. Riley, “Weighted finite-state transducers in speech recognition,” *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[3] E. Shareghi, D. Gerz, I. Vulić, and A. Korhonen, “Show some love to your n-grams: A bit of progress and stronger n-gram language modeling baselines,” in *Proceedings of NAACL-HLT*, 2019, pp. 4113–4118.

[4] J. R. Bellegarda, “Statistical language model adaptation: review and perspectives,” *Speech communication*, vol. 42, no. 1, pp. 93–108, 2004.

[5] R. DeMori and M. Federico, “Language model adaptation,” in *Computational models of speech pattern processing*. Springer, 1999, pp. 280–303.

[6] F. Jelinek and R. L. Mercer, “Interpolated estimation of Markov source parameters from sparse data,” *Proceedings, Workshop on Pattern Recognition in Practice*, pp. 381–397, 1980.

[7] X. Liu, M. J. F. Gales, and P. C. Woodland, “Use of contexts in language model interpolation and adaptation,” *Computer Speech & Language*, vol. 27, no. 1, pp. 301–321, 2013.

[8] B.-J. Hsu, “Generalized linear interpolation of language models,” in *ASRU*. IEEE, 2007, pp. 136–140.

[9] E. Pusateri, C. Van Gysel, R. Botros, S. Badaskar, M. Hannemann, Y. Oualil, and I. Oparin, “Connecting and comparing language model interpolation techniques,” *Proceedings of INTERSPEECH*, pp. 3500–3504, 2019.

[10] K. Heafield, C. Geigle, S. Massung, and L. Schwartz, “Normalized log-linear interpolation of backoff language models is efficient,” in *Proceedings of ACL (Volume 1: Long Papers)*, 2016, pp. 876–886.

[11] S. Besling and H.-G. Meier, “Language model speaker adaptation,” in *Proceedings of EUROSPEECH*, 1995.

[12] R. Rosenfeld, “A maximum entropy approach to adaptive statistical language modelling,” *Computer Speech and Language*, vol. 10, pp. 187–228, 1996.

[13] S. Della Pietra, V. Della Pietra, R. L. Mercer, and S. Roukos, “Adaptive language modeling using minimum discriminant estimation,” in *Proceedings of ICASSP*, 1992, pp. 633–636.

[14] P. S. Rao, M. D. Monkowski, and S. Roukos, “Language model adaptation via minimum discrimination information,” in *Proceedings of ICASSP*, 1995, pp. 161–164.

[15] P. S. Rao, S. Dharanipragada, and S. Roukos, “MDI adaptation of language models across corpora,” in *Proceedings of EUROSPEECH*, 1997.

[16] M. Weintraub, Y. Aksu, S. Dharanipragada, S. Khudanpur, H. Ney, J. Prange, A. Stolcke, F. Jelinek, and E. Shriberg, “LM95 project report: Fast training and portability,” 1996.

[17] W. Reichl, “Language model adaptation using minimum discrimination information,” in *Proceedings of EUROSPEECH*, 1999.

[18] M. Federico, “Efficient language model adaptation through mdi estimation,” in *Proceedings of EUROSPEECH*, 1999.

[19] R. Kneser, J. Peters, and D. Klakow, “Language model adaptation using dynamic marginals,” in *Proceedings of EUROSPEECH*, 1997.

[20] C.-H. Chueh and J.-T. Chien, “Reliable feature selection for language model adaptation,” in *Proceedings of ICASSP*, 2008, pp. 5089–5092.

[21] S. F. Chen, “Shrinking exponential language models,” in *Proceedings of HLT-NAACL*, 2009, pp. 468–476.

[22] N. Ruiz and M. Federico, “Mdi adaptation for the lazy: Avoiding normalization in lm adaptation for lecture translation,” in *Proceedings of International Workshop on Spoken Language Translation (IWSLT)*, 2012.

[23] R. Rosenfeld, “Adaptive statistical language modeling; a maximum entropy approach,” Carnegie Mellon University, Pittsburgh, PA, Tech. Rep., 1994.

[24] T. Alumäe and M. Kurimo, “Efficient estimation of maximum entropy language models with n-gram features: an SRILM extension,” in *INTERSPEECH*, 2010.

[25] D. Jurafsky and J. H. Martin, *Speech and Language Processing (2nd Edition)*. USA: Prentice-Hall, Inc., 2009.

[26] J. N. Darroch and D. Ratcliff, “Generalized iterative scaling for log-linear models,” *The annals of mathematical statistics*, pp. 1470–1480, 1972.

[27] R. Huang, K. Li, A. Arora, D. Povey, and S. Khudanpur, “Efficient MDI Adaptation for n-gram Language Models,” *arXiv e-prints*, p. arXiv:2008.02385, Aug. 2020.

[28] C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, and T. Robinson, “One billion word benchmark for measuring progress in statistical language modeling,” *Proceedings of INTERSPEECH*, 2014.

[29] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, “Librispeech: an asr corpus based on public domain audio books,” in *Proceedings of ICASSP*. IEEE, 2015.

[30] A. Stolcke, “SRILM - an extensible language modeling toolkit,” in *INTERSPEECH*, 2002.

[31] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, “The kaldi speech recognition toolkit,” in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. CONF. IEEE Signal Processing Society, 2011.

[32] D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohammadi, and S. Khudanpur, “Semi-orthogonal low-rank matrix factorization for deep neural networks.” in *INTERSPEECH*, 2018, pp. 3743–3747.

[33] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, “Purely sequence-trained neural networks for ASR based on lattice-free MMI.” in *INTERSPEECH*, 2016, pp. 2751–2755.