LANGUAGE MODEL ADAPTATION USING RANDOM FORESTS

Anoop Deoras, Frederick Jelinek

Center for Language and Speech Processing,
Johns Hopkins University
{adeoras, jelinek}@jhu.edu

Yi Su

Nuance Communications, Inc.
Yi.Su@nuance.com

ABSTRACT

In this paper we investigate random forest based language model adaptation. Large amounts of out-of-domain data are used to grow the decision trees while very small amounts of in-domain data are used to prune them back, so that the structure of the trees are suitable for the desired domain while the probabilities in the tree nodes are reliably estimated. Extensive experiments are carried out and results are reported on a particular task of adapting Broadcast News language model to the MIT computer science lecture domain. We show 0.80% and 0.60% absolute WER improvement over language model interpolation and count merging techniques, respectively.

Index Terms— Random Forests, Language Modeling, Adaptation

1. INTRODUCTION

Transcribing speech data for a new domain for the purpose of training its acoustic and language models, is a very costly affair. To obtain even an acceptable recognition accuracy, one needs at least a few tens of hours of annotated data. While obtaining such manually annotated transcripts may be labour intensive and costly in some cases, it may not be even possible in others.

To cater to such situations, Language Models (LM) and Acoustic Models (AM) trained on large amounts of out-of-domain data are adapted to make them more representative of the in-domain task. An extensive work has gone into acoustic model adaptation [1, 2, 3]. As pointed out in [4], [5] shows that bootstrapping from as little as 10 minutes of supervised acoustic training data gives a good adapted acoustic model for the in-domain task, which can further be refined using large amounts of unsupervised data.

The task of Language model adaptation, on the other hand, has not received much attention. Also the fact that language model in state-of-the-art speech recognition systems is still restricted to the simple n-gram approach (mainly due to its simplicity and surprisingly much better results, given large amount of data, than other complex sophisticated LMs [6]) further limits the extent of exploration in LM adaptation tasks. Some of the best approaches for supervised adaptation are language model interpolation [7] and count merging [8]. [4] presents an unsupervised language model adaptation using count merging. The authors in this paper investigate the effectiveness of various adaptation strategies, including iterative adaptation and self-adaptation on the test data.

In this paper, we too have focused on the task of language model adaptation. We take a particular case of adapting Broadcast News LM to MIT computer science lecture data. There is a considerable mismatch between the two domains and hence the adaptation task becomes specially interesting and challenging.

Random Forests (RF) based language model [9], has shown superior performance over conventional n-gram language models. The strength of the decision tree based language modeling comes from the fact that the histories are clustered into equivalence classes, determination of which, is done by asking sets of questions about the history. The choice of questions and the determination of the equivalence classification of history is based on the reduction of the perplexity of both, the training data and the small held out data. In this paper, we have extended the idea of decision tree based language modeling to the task of adapting out-of-domain language model to the in-domain data. We have investigated whether clever equivalence classification of the out-of-domain data can help reduce the uncertainty involved in the in-domain data (both, during training and test times).

The rest of the paper is organized as follows. We briefly discuss RFLM in section 2. In section 3, we discuss the language model interpolation and count merging techniques. In section 4, we describe our adaptation idea based on random forests. We then discuss various experimental setups and results in section 5 and finally conclude in section 6.

2. RANDOM FOREST LANGUAGE MODEL

A Random Forest Language Model (RFLM) [9] is a collection of randomized Decision Tree Language Models (DTLMs), which define equivalence classification of histories. The RFLM generalizes the DTLM by averaging multiple DTLMs, which, in turn, generalizes the n-gram LM

by having a sophisticated equivalence classification. The LM probability of a RFLM is defined as follows:

$$\begin{aligned} P_{RF}(w|h) = & \quad \frac{1}{M}\sum_{j=1}^{M} P_{DT_j}(w|h) \\ = & \quad \frac{1}{M}\sum_{j=1}^{M} P(w|\Phi_{DT_j}(h)), \end{aligned} \tag{1}$$

where h is the history and $\Phi_{DT_j}(\cdot)$ is a decision tree. The questions that have been used so far care only about the identity of the words in a history position. If w_i is the word we want to predict, then the general question takes the following form:
Is the word $w_{i-k}, k \geq 1$ in a set of words $\mathcal{S}$?
Because in the normal n-gram situation we know no more than the words in the history, these questions are almost all we can ask.

3. LANGUAGE MODEL ADAPTATION

By far, the two most successful past LM adaptation techniques have been count merging and model interpolation. As observed by [4], both of these methods can be viewed as Maximum A Posteriori adaptation strategies with a different parameterization of the prior distribution. For a word w_i in n-gram history h, let the expected counts from the in-domain and out-of-domain data be denoted $\bar{C}(h, w_i)$ and $\tilde{C}(h, w_i)$ respectively. Let the discounted counts be represented as $\bar{C}_d(h, w_i)$ and $\tilde{C}_d(h, w_i)$ respectively. If we denote the probability of w_i in history h as estimated from the in-domain and out-of-domain samples by $\bar{P}(w_i|h)$ and $\tilde{P}(w_i|h)$, then a count merging approach with mixing parameters α and β estimates the adapted model as:

$$\hat{P}(w_i|h) = \frac{\alpha\tilde{C}_d(h, w_i) + \beta\bar{C}_d(h, w_i)}{\alpha\sum_w \tilde{C}_d(h, w) + \beta\sum_w \bar{C}_d(h, w)} \tag{2}$$

while, model interpolation estimates the adapted model as:

$$\hat{P}(w_i|h) = \lambda\bar{P}(w_i|h) + \bar{\lambda}\tilde{P}(w_i|h) \tag{3}$$

where λ is chosen such that the the model $\hat{P}(\cdot)$ minimizes the perplexity of some in-domain held out data.

4. RANDOM FOREST BASED LM ADAPTATION

Our decision tree training procedure, following [9], consists of a *growing* and a *pruning* stage. In the growing stage, we start from a single node which contains all n-gram histories from the the training text of the out-of domain data. We recursively split every node by an exchange algorithm [10] until further splitting does not decrease the perplexity of this training text. In the pruning stage, we compute the *potential* of each node — the possible increase in in-domain data likelihood from growing the node into a sub-tree. Nodes whose potential fall below a threshold are pruned, bottom up. Thus the structure of the tree (i.e. the depth of the tree, number of nodes, number of leaves) and the parameters of the tree (questions), are tuned to optimize two objective functions: perplexity of the out-of-domain data and perplexity of the in-domain data.

Due to the greediness of the exchange algorithm, restricting the candidate questions at each node-splitting step to a *random subset* of all available questions helps find a better tree. The same argument holds for random initialization of the exchange algorithm. In this paper, we will randomize in only these two ways.

We use interpolated Kneser-Ney smoothing, to compute the LM probabilities

$$\begin{aligned} P(w_i|w_{i-n+1}^{i-1}) = & \quad \frac{\max(C(\Phi(w_{i-n+1}^{i-1}), w_i) - D, 0)}{C(\Phi(w_{i-n+1}^{i-1}))} \\ + & \quad \lambda(\Phi(w_{i-n+1}^{i-1}))P_{KN}(w_i|w_{i-n+2}^{i-1}), \end{aligned} \tag{4}$$

where D is a *constant discount* and $P_{KN}(\cdot)$ is the Kneser-Ney backoff probability distribution.

If we parameterize the distribution of $P(w|h)$ by Θ, where Θ captures the structure of the tree and the questions asked at each node of the tree, then the tree growing phase aims at finding that $\widehat{\Theta}$ which maximizes the likelihood of the out-of-domain data, that is:

$$\widehat{\Theta} = \arg\max_{\Theta} \sum_{w,h} C_B(h, w) \log P(w|h, \Theta), \tag{5}$$

where 'B' represents the background or the out-of-domain data.

Almost all decision tree building algorithms are greedy; ours is no exception. That is to say, despite our best effort, we can only find a locally optimum tree. However, in the process of getting to the local optimum solution, we also discover a family of trees from the most general, consisting only the root node and equivalent to a uni-gram LM, to the most specific, fully grown tree that is equivalent to a regular n-gram LM. Let $\{\Theta\}_B$ be the set of parameters that represent this family. We then try to find the best parameter in that set to maximize the likelihood of the target domain:

$$\Theta^* = \arg\max_{\Theta\in\{\Theta\}_B} \sum_{w,h} C_A(h, w) \log P(w|h, \Theta), \tag{6}$$

where 'A' represents the in-domain data.

5. EXPERIMENTS AND RESULTS

For the purpose of our experiments, we have used IBM's state of the art speech recognizer: Attila. The transcription system was trained on 330 hours of audio from the 1996 English Broadcast News Speech corpus (LDC97S44) and the 1997 English Broadcast News Speech corpus (LDC98S71). The acoustic features used by the system are PLP features. The system has acoustic models with roughly 6000 states and 250000 Gaussians. The size of the Broadcast News corpus

vocabulary is $90K$. More details on various aspects of the system can be found in [11].

We trained two language models from the out of domain i.e Broadcast News data. Hereafter we will refer to them as LM-BN-small and LM-BN-big. LM-BN-big is a Kneser Ney smoothed trigram language model trained on $163M$ words from 1996 CSR Hub4 language model data and Hub4 acoustic model training transcripts (collectively this dataset will be hereafter referred to as BN-TRAIN). This LM contains about $44M$ n-grams. LM-BN-small is a pruned version of LM-BN-big and contains about $4M$ n-grams. LM-BN-small is used in the recognition first pass while the LM-BN-big is used for either re-scoring the lattices or acting as a back off language model in the random forests based re-scoring.

From the in-domain i.e. MIT computer science lecture data, we trained a relatively smaller language model, hereafter referred to as LM-MIT. This LM is a Kneser Ney smoothed trigram language model trained on $150K$ words from the MIT computer science lecture transcripts (hereafter referred to as MIT-TRAIN) and contains about $0.2M$ n-grams. We also set aside about $20K$ words development data, hereafter referred to as MIT-DEV. This data is used for tuning the interpolation weights when combining broadcast news LM and MIT LM.

The test setup consists of 500 speech utterances (corresponding roughly to 1.5 hours of speech) from MIT computer science lecture data, hereafter referred to as MIT-TEST. Another set of 400 speech utterances and the corresponding true transcripts (which are not used in MIT-TRAIN or MIT-DEV), from the same genre, were used as held out data, hereafter referred to as MIT-HLD. We used this data to tune the language model scaling factors during the recognition and re-scoring.

IBM's ASR engine compiles a huge static graph by composing the acoustic phonetic tree and language model and is used during the first pass recognition. Due to the limitations posed by the computer memory, we limit the size of the LM used in the first pass decoding to about $4M$ n-grams. Also, due to this limitation, we made use of RF based decoding by re-scoring the word lattices by extracting N best lists from them. We extracted top 5000 hypotheses from the lattices. We have not incorporated word insertion penalty factor in any of our experiments. We only tuned the language model scaling factor on MIT-HLD, separately for each experiment. Finally, we made use of RFLM toolkit [12] to build decision trees and random forests.

We have carried out 3 sets of experiments, details of which are as follows:

5.1. Main Results

In this subsection we present our main results, i.e. the adaptation results using random forests. In it, we compare our method with the model interpolation technique. Table 1 summarizes our findings. Use of LM-BN-small in the first pass recognition without any re-scoring of lattices forms our baseline (first row of Table 1). Use of LM-BN-small in the first pass recognition and later re-scoring of N-best lists by LM-BN-big constitutes our rescoring baseline (second row of Table 1). Use of LM-BN-small in the recognizer and later rescoring the N-best lists with RF constitutes our baseline for random forests (third row of Table 1). The RF is trained using BN-TRAIN data. BN-DEV data is used as the development data for pruning the trees. We used LM-BN-big as the back off language model. Use of interpolated LM-BN-small and LM-MIT (which is hereafter referred to as LM-interp-small) in the first pass without any re-scoring constitutes our baseline for interpolation (fourth row of Table 1). Use of LM-interp-small in the first pass and later re-scoring of N-best lists by the interpolated LM-BN-big and LM-MIT (which is hereafter referred to as LM-interp-big) constitutes our re-scoring baseline for interpolation (fifth row of Table 1). Use of LM-interp-small in the first pass and later re-scoring by RF constitutes our setup for random forest based adaptation (sixth row of Table 1). RF in this setup is trained using BN-TRAIN and MIT-TRAIN[1]. MIT-TRAIN and MIT-DEV data are used as development data. LM-interp-big is used as a back off language model. In all of our RF setups, we use ensembles of 100 randomly grown decision trees.

Setup	WER
LM-BN-small decoding	39.15
+ LM-BN-big re-scoring	38.57
+ BN-TRAIN/BN-DEV built RF re-scoring	38.42
LM-interp-small decoding	30.05
+ LM-interp-big re-scoring	29.87
+ BN-TRAIN/MIT-TRAIN built RF re-scoring	**29.06**

Table 1. Main Results

From the results presented in Table 1, we see that RF based adaptation method outperforms the conventional model interpolation technique by 0.80% absolute WER. These results are also statistically significant, with a p value less than 0.01 (We use SCTK toolkit's mapsswe test). We also computed the perplexities of our LMs and they followed the same trend as their corresponding WERs.

5.2. Count Merging vs Random Forests

In this subsection, we compare the performance of count merging technique and also explore the use of merged counts for building our random forests. We use LM-interp-small in the first pass recognition and later do re-scoring of N-best lists with the count merged LM. The count merged LM is obtained by merging tri-gram counts from BN-TRAIN and MIT-TRAIN (this merged dataset will be hereafter referred to as MGD-TRAIN). The parameters were tuned to have value

[1]Counts from the totality of the in-domain and out-of-domain data gives a little bit better performance. Setup of the fourth row in Table 3 uses just the BN-TRAIN data to grow trees.

$\alpha = 1$ and $\beta = 5$ (2). We denote this LM by LM-CntMrgd. This setup forms the baseline for count merging (second row of Table 2). For the purpose of creating our decision trees, we used the LM-CntMrgd training data. We used MIT-TRAIN and MIT-DEV as the development data. LM-CntMrgd is used as a back off language model. This forms our setup for random forests based adaptation using count merging (third row of Table 2).

Setup	WER
LM-interp-small decoding	30.05
+ LM-CntMrgd re-scoring	29.60
+ MGD-TRAIN/MIT-TRAIN built RF re-scoring	**29.00**

Table 2. Count merging vs random forests

From Table (2), we can see that the random forest based adaptation using merged counts, outperforms the count merged language model technique by 0.60% absolute WER.

5.3. Model strength vs Adaptation effect

In this subsection, we have investigated the potential improvement obtained using random forests when the training and development data used for tree construction is used from in-domain data instead of the out-of-domain. We used LM-interp-small in the recognition first pass and later re-scored the N best lists by RF. RF is trained using BN-TRAIN data alone and BN-DEV data is used as the development data for pruning the trees. We however used LM-interp-big as the back off language model. This constitutes our setup for random forest based adaptation using out of domain training and dev data (third row of Table 3). Our other RF setup consists of BN-TRAIN as the training data and MIT-TRAIN and MIT-DEV as the development data. LM-interp-big is used as the back off language model (fourth row of Table 3). Following table summarizes the improvements obtained using the in-domain data during tree pruning.

Setup	WER
LM-interp-small decoding	30.05
+ LM-interp-big re-scoring	29.87
+ BN-TRAIN/BN-DEV built RF re-scoring	29.25
+ BN-TRAIN/MIT-TRAIN built RF re-scoring	**29.10**

Table 3. Model strength vs adaptation effect

From Table (3), we can see that random forest alone improves over the model interpolation technique. Moreover, the use of small amounts of in-domain data, if used to prune the trees, results in even better performance.

6. CONCLUSION

In this paper, we have presented a novel language model adaptation method using ensembles of randomly grown decision trees. We show that use of small amounts of in-domain data suffices to adapt the out-of-domain model to the in-domain task. We have shown that our method outperforms the model interpolation and count merging techniques.

7. REFERENCES

[1] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech and Language*, pp. 75–98, 1998.

[2] J. L. Gauvain and C. H. Less, "Maximum a Posteriori estimation for multirate gaussian mixture observations of markov chains," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 291–298, 1994.

[3] C. J. Legetter and P. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models," *Computer Speech and Language*, 1995.

[4] M. Bacchiani and B. Roark, "Unsupervised language model adaptation," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2003.

[5] L. Lamel, J. L. Gauvain, and G. Adda, "Unsupervised acoustic model training," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 171–185, 2002.

[6] Joshua T. Goodman, "A bit of progress in language modeling," Tech. Rep., Microsoft Research, 2001.

[7] P. Woodland, T. Hain, G. Moore, T. Niesler, D. Povey, A. Tuerk, and E. Whittaker, "The 1998 HTK Broadcast News Transcription system," *DARPA Broadcast News Transcription and Understanding Workshop*, 1998.

[8] A. Ljolje, D. Hindle, M. Riley, and R. Sporat, "The AT&T LVCSR-2000 system," *NIST LVCSR Workshop*, 2000.

[9] Peng Xu and Frederick Jelinek, "Random forests in language modeling," *EMNLP*, 2004.

[10] Sven Martin, Jörg Liermann, and Hermann Ney, "Algorithms for bigram and trigram word clustering," *Speech Communication*, vol. 24, no. 1, pp. 19–37, 1998.

[11] S. Chen, B. Kingsbury, L. Mangu, D. Povey, G. Saon, H. Soltau, and G. Zweig, "Advances in speech transcription at IBM under the DARPA EARS program," *IEEE Transactions on Audio, Speech and Language Processing*, pp. 1596–1608, 2006.

[12] Yi Su, "Random forest language model toolkit," http://www.clsp.jhu.edu/∼yisu/rflm.html, 2009.