Improving Unsupervised Language Model Adaptation with Discriminative Data Filtering

Shuangyu Chang, Michael Levit, Partha Parthasarathy, Benoit Dumoulin

Microsoft Corporation, Sunnyvale, California

shchang@microsoft.com

Abstract

In this paper we propose a method for improving unsupervised language model (LM) adaptation by discriminatively filtering the adaptation training material. Two main issues are addressed in this solution: first, how to automatically identify recognition errors and more correct alternatives without manual transcription; second, how to update the model parameters based on the recognition error cues. Within the adaptation framework, we address the first issue by predicting regression pairs between recognition results from the baseline LM and an initial adapted LM, using features such as language model score difference. For the second issue, we adopted a data filtering approach to penalize potent error attractors introduced by the unsupervised adaptation data, using Ngram set difference statistics computed on the predicted regression pairs. Experimental results on a large real-world application of voice catalog search demonstrated that the proposed solution provides significant recognition error reduction over an initial adapted LM.

Index Terms: unsupervised, discriminative, language model adaptation,

1. Introduction

Consider the following scenario for a real-world speech recognition application. A baseline language model (*BLM*) is first created from training data that was roughly in-domain, based on the best estimate of potential real user inputs. The application is deployed to public and a large amount of real user utterances are serviced and captured; an updated LM is then created by incorporating user utterance data to improve recognition accuracy. Since it can be very expensive to manually transcribe large amount of user utterances, automatic recognition output using *BLM* is used to approximate manual transcriptions. This is thus a typical case of unsupervised LM adaptation [1, 2, 3]. To obtain the updated LM, one basic approach is to train a new LM using unsupervised automatic transcription and interpolate the new LM with *BLM*. The interpolation weights can be trained to optimize the perplexity on a held-out set of transcriptions.

While this process is simple and efficient, the updated LM can only provide limited accuracy improvement over the baseline LM; in fact, for many utterances, the updated LM may even perform worse than the baseline LM. A major cause of this limitation is the recognition errors in the unsupervised transcription data. Some of these errors may not even be due to the quality of the baseline LM itself, but rather attributable to some deficiencies and biases in other parts of the recognition system, such as the acoustic model and the decoder. These errors can get reinforced through the adaptation process to create even stronger error attractors than there were previously in the baseline LM. One common approach to alleviate the error reinforcement issue is to filter the unsupervised transcriptions to retain only those more likely to have been correctly recognized, for example, by setting a threshold on the baseline recognition confidence score. However, such filtering is generally not reliable and is not directly driven by the minimization of the recognition errors from the adapted model. In this work, we propose a method to filter the unsupervised transcription data in a discriminative fashion that aims to reduce the potent error attractors, thus improving overall recognition accuracy for the updated model.

There are two main issues that we have to address in an unsupervised, discriminative solution. First, how to automatically identify recognition errors and their corresponding corrected or improved forms, without manual transcriptions. Previous work in discriminative LM training addressed this issue in various ways. One popular approach to the first issue [4, 5, 6] is to generate likely confusion sets for recognition by modeling phonetic similarities and applying LM constraints through the use of weighted finite state transducers (WFST). In [7] the authors bypassed the phonetic confusion modeling and extended the phrasal cohort approach, which was previously applied only on transcribed data [8], to completely untranscribed scenario and still retains over 40% of gains of supervised approach. In [9] recognition lattice was generated from a weaker acoustic model than the one used for producing unsupervised transcriptions, thus creating contrasting pairs of superior and inferior samples for error gradient computation. A recent work by Ogawa et al. [10] created word alignment network out of recognition word confusion network, to directly estimate recognition errors by type, and further enhanced it with a discriminative classifier. In our proposed solution we leverage the LM adaptation framework and adopt a less ambitious but more focused approach to minimize the negative effect of error attractors introduced by the unsupervised adaptation data, while retaining the benefit from the correct portion of the unsupervised data. This sharper focus allows us to use specialized features to identify recognition regression pairs between the initial adapted LM and *BLM*.

The second issue that we have to address is how to update the language model based on the identified recognition error cues. Two different approaches to this issue have been prominent in previous work on discriminative LM training. The first approach attempts to modify Ngram-based language models directly, using generalized probablistic or gradient decent [11, 6], or linear programming [12] on some globally defined optimization criteria. One drawback of this approach is the difficulty in dealing with Ngram backoff weight updates and proper normalization of the updated model. In [13] the authors reported moderate improvement when additional steps are taken to update Ngram backoff weights and to constrain parameter updates with probability normalization. The second approach performs re-ranking or re-scoring on an Nbest list or lattice from

a first-pass recognition output using a discriminatively trained LM [14, 9, 15], which can often incorporate additional features beyond Ngrams and be optimized by various machine learning algorithms. In this work, we would like to stay within an existing Ngram modeling and first-pass decoding framework to minimize impact on recognition latency and production system complexity; and instead of directly modifying Ngram parameters in existing model, we update the model through filtering of adaptation training data, by taking advantage of the nature of relatively short utterances in the applications we are targeting.

In the following sections, we first outline the overall framework of our proposed solution. We then describe our approach to identify recognition regression pairs for the LM adaptation scenario, followed by a description of a training data filtering approach to update model with recognition error cues. We present experimental results on a real world application, and finally conclude with a discussion of the advantages and limitations of our approach, and potential future work.

2. Unsupervised, Discriminative LM Adaptation

2.1. Overall framework

To facilitate further description let us first define the proposed adaptation framework. Assume we have a baseline language model BLM that already exists for the target application. Let U denote a large set of real user utterances collected from the application and we have at our disposal a fixed decoder and acoustic model for recognition. We split U into two separate subsets, a larger one U_a, and a smaller one U_b. The overall adaptation process to create a final adapted language model is as follows:

1. Decode set U_a and U_b with BLM to obtain recognition output $R_{BLM}(U_a)$ and $R_{BLM}(U_b)$, respectively.
2. Train a new Ngram model from $R_{BLM}(U_a)$ and interpolate it with BLM to obtain the initial adapted model $ULM_0^{(a)}$.
3. Decode set U_b with $ULM_0^{(a)}$ to obtain recognition output $R_{ULM_0^{(a)}}(U_b)$.
4. Identify a subset C of U_b with regression pairs between $R_{BLM}(U_b)$ and $R_{ULM_0^{(a)}}(U_b)$ such that utterances in C were likely to have worse recognition results using $ULM_0^{(a)}$ than using BLM.
5. Compute the differences of regression pairs in C and apply the statistics to filter set $R_{BLM}(U_a)$, suppressing likely contributors to error attractors.
6. Train a new Ngram model with the filtered set and interpolate it with BLM to obtain the updated adapted model $ULM_1^{(a)}$.

In both step 2 and step 6, it is possible to apply additional data filtering criteria such as thresholding by recognition confidence, to retain only the most confidently recognized results for training new model. It is also possible to run multiple iterations on step 3 through step 6 for further improvements.

In essence, the framework learns some possible error regression patterns that initial adapted model $ULM_0^{(a)}$ may generate over BLM, and makes correction to the unsupervised adaptation data when retraining an updated adapted model $ULM_1^{(a)}$. The following sections will provide a detailed description of our proposed solutions to crucial steps 4 and 5.

2.2. Identifying recognition regressions

As in any discriminative training method, it is critical to have reliable cues about whether a training sample gets a correct or incorrect recognition result so that appropriate rewards and penalties can be assigned to model parameters. With unsupervised training data, these cues are generally difficult to obtain. In our LM adaptation task we can take advantage of the relationship between two language models - an already available baseline model BLM, and an initial adapted $ULM_0^{(a)}$ that can be viewed as a modification of the BLM with statistics learned from the unsupervised adaptation material U_a. For an arbitrary utterance, there are four possible outcomes when it is decoded against each of the two LMs independently. 1) Both can be correct; 2) both can be incorrect; 3) BLM is correct but $ULM_0^{(a)}$ incorrect; 4) BLM is incorrect but $ULM_0^{(a)}$ correct. Cases 1 and 4 are straightforward and we would want to keep the corresponding contribution from U_a. Case 3 is the one we want to penalize since $ULM_0^{(a)}$ degrades the BLM performance. Case 2 is less clear and may depend on the relative degree of incorrectness between the two models. It will thus be very helpful to be able to automatically identify utterances belonging to the different cases. It turns out there exist some automatically computable features that are helpful.

One very effective feature that we have found was the difference between the language model scores of each recognition output string, with respect to the LM that the recognition was run with. For example, for an utterance x, recognition with BLM gives a hypothesis x_{BLM} with LM score $LMS(x_{BLM})$ and recognition with $ULM_0^{(a)}$ gives a hypothesis $x_{ULM_0^{(a)}}$ with LM score $LMS(x_{ULM_0^{(a)}})$; then a large negative value in the difference $LMS(x_{ULM_0^{(a)}}) - LMS(x_{BLM})$ would indicate that it more likely belongs to case 3. Here LM score is the sum of the log probabilities of words in a hypothesis using a particular LM. Figure 1 demonstrates this effect on a data set from a large voice catalog search application (as described in more detail in Section 3). The horizontal axis represents the LM score difference and vertical axis the proportion of different types of outcomes within the group of utterances having the particular LM score difference values. Clearly a large number of case 3 utterances have a large negative LM score difference while case 4 utterances tend to have large positive LM score difference.

One immediate application of this disparity is that we can improve the final recognition accuracy by running separate recognitions using each of BLM and $ULM_0^{(a)}$ and elect to use $ULM_0^{(a)}$ result only when the LM score difference is greater than some threshold, and fall back to using BLM result otherwise. However, since this result selection significantly increases the resource requirement at production time by doubling the amount of decoding required, we will not further pursue this development. Instead, in this work we focus on using the LM score difference to select a set of possible regression pairs where $ULM_0^{(a)}$ performed worse than BLM, to use in the next stage of training updated adapted model.

There are other features that may potentially help distinguish the four outcome cases above, such as the AM score difference and acoustic quality measures such as SNR. However, when combined with LM score difference in a statistical classifier training, these additional features did not offer significant gain and in the rest of this work, we will focus on using only the LM score difference feature. One should also note that the effectiveness of the feature depends heavily on the quality of

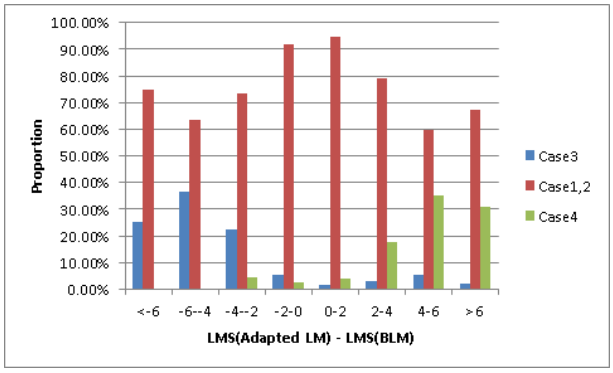

Figure 1: *Distribution of sentence-level recognition dispositions for each range of LM score difference between the initial adapted LM* $ULM_0^{(a)}$ *and BLM, on a large voice catalog search application. Case 3 for BLM is correct but adapted LM incorrect; case 4 for BLM incorrect but adapted LM correct; case 1 and 2 for both models correct or both incorrect, respectively.*

recognition engine and acoustic model, as well as the quality of the *BLM*. Generally a poorer baseline system will result in greater number of regression pairs selected since there are greater amount of deficiencies to exploit.

2.3. Data filtering using Ngram-Diff statistics

Once we have a set of regression pairs containing recognition hypotheses from *BLM* and from initial adapted $ULM_0^{(a)}$, there are various ways to apply this knowledge on improving the unsupervised adaptation. In this work, we chose to modify the unsupervised training data set $R_{BLM}(U_a)$ to suppress possible error attractors introduced or reinforced via the initial adaptation. One advantage of this choice over directly modifying Ngram language model is the simplicity of training process. For example, we do not have to explicitly deal with the decision whether to remove an Ngram or update backoff paths when an Ngram is suppressed.

The technique we introduce for suppressing potential error attractors is based on a concept of *NgramDiff*. For text strings T_1 and T_2, and an integer N, we define *NgramDiff*(T_1, T_2, N) as an asymetric Ngram set difference of order N between the two strings, consisting of all Ngrams of order N in T_1, annotated by the difference in frequency of occurrences of each Ngram in T_1 and T_2. For example, if T_1 = "<s> a c d </s>" and T_2 = "<s> a b c </s>" where "<s>" and "</s>" are begin and end of sentence, respectively, then we will have:

	Ngrams	Diff
NgramDiff$(T_1, T_2, 1)$	d	1
NgramDiff$(T_1, T_2, 2)$	c d	1
	a c	1
	d </s>	1
NgramDiff$(T_1, T_2, 3)$	a c d	1
	c d </s>	1
	<s> a c	1

In each case, the "Diff" count of "1" indicates that the particular Ngram occurred one more time in T_1 than in T_2 (actually all 1 and 0 times in this example, respectively). In this example, we have omitted Ngrams where the difference is less than 1. Consider a regression pair with correct recognition string T_2 from *BLM* and incorrect recognition string T_1 from $ULM_0^{(a)}$. *NgramDiff*(T_1, T_2, N) thus provide statistics on erroneous Ngram patterns that appeared in the incorrect hypothesis more frequently than in the correct hypothesis. Since we know that the recognition system is capable of producing the correct hypothesis T_2 it is likely that the Ngram patterns with positive *NgramDiff* values have contributed to the promotion of the incorrect hypothesis T_1 over the correct hypothesis T_2. Thus *NgramDiff*(T_1, T_2, N) offers a set of potential error attractor Ngrams with their frequencies.

We now compute the *NgramDiff* on all regression pairs that we have identified between $R_{BLM}(U_b)$ and $R_{ULM_0^{(a)}}(U_b)$ and aggregate the statistics into *NgramDiffScore*(i) for each Ngram type i. Then for each recognition result in $R_{ULM_0^{(a)}}(U_a)$, instead of always including it in the unsupervised adaptation set as we did when creating $ULM_0^{(a)}$, we will only include it with a probability that aims to penalize recognition strings that contain a high value of *NgramDiff* statistics, thus more likely error attractors. In our experiments, we have adopted a simple formulation as following:

$$P(accept) = (1 + \sum_i w_n * NgramDiffScore(i))^{-E} \quad (1)$$

Here i ranges over all Ngrams of order n from the current recognition string that have positive *NgramDiffScore*. w_n is a weighting factor for Ngrams of order n and E in the exponent scales and inverts the score, and both are meta parameters that can be optimized on held-out data.

3. Experiments

Our experiments were conducted on data collected from a large catalog search application where users can use voice to query from millions of media items such as movies, music, and video games by their titles, artists, and other relevant attributes. The test set contains 5000 transcribed utterances, with 18,159 words. Another 5000 transcribed utterances were used for validation and tuning. A set of 480K transcribed utterances were available but in our final experiment we will only use the unsupervised recognition output of this set (set U_b) with respect to the *BLM* to learn regression pairs. Finally, a total of about eight million untranscribed utterances were used for unsupervised adaptation (set U_a). All data sets were drawn randomly from the same time period over nine months. For each experiment we measure the relative sentence error rate reduction (SERR) and relative word error rate reduction (WERR) over the baseline model (*BLM*), as well as the perplexity of each model. The baseline *BLM* Ngram model was trained from various data sources collected at a time prior to any of the data sets were captured in production; the major sources included a snapshot of media items in the catalog then and media queries in Bing web query logs. As described in the previous sections, we treat the *BLM* as given and do not use its original training data directly in this work.

To better understand the contribution of different parts of the algorithm, we ran four experiments differing in how the recognition regression pairs were generated:

- Setup 1: Decode test set with initial adapted model $ULM_0^{(a)}$. The set of regression pairs C contains transcription and recognition output of all test utterances that were misrecognized.

- Setup 2: Decode test set with both baseline BLM and initial adapted model $ULM_0^{(a)}$. The set C contains transcription and recognition output of all test utterances that were correctly recognized with BLM but misrecognized with $ULM_0^{(a)}$.
- Setup 3: Same as Setup 2 but use set U_b instead of test set to generate set C.
- Setup 4: Decode set U_b with both baseline BLM and initial adapted model $ULM_0^{(a)}$. The set C was generated following the algorithm in Section 2.2, thus contains recognition output pairs from BLM and from $ULM_0^{(a)}$ where the former were likely to have performed better than the latter. The threshold for LM score difference was set to -2.

The first two setups were thus cheating experiments involving test data. Setup 3 did not use test data for training but was, of course, a very inefficient use of transcription of U_b, which, if directly used in a supervised adaptation, would have improved the overall performance significantly. Setup 4 represents a true unsupervised solution. In all cases we have only used *NgramDiff* statistics of order 2 and the weighting factor w_n and exponent E in equation 1 were tuned on the tuning set.

Table 1 compares the performance of the four setups to using only the baseline BLM or the initial adapted model $ULM_0^{(a)}$, which already provided a significant error reduction over BLM. Both Setup 1 and 2 further improved over $ULM_0^{(a)}$, suggesting that the *NgramDiff*-based data filtering method was effective in suppressing error attractors. In particular, Setup 2 outperformed Setup 1 since the former focuses exclusively on utterances that did worse with $ULM_0^{(a)}$ than with BLM, which are more likely to be corrected when we filter the unsupervised adaptation data. On the other hand, Setup 1 included all misrecognized utterances with $ULM_0^{(a)}$ but some of them, though erroneous at sentence level, may still be more "correct" (containing fewer word errors) than the corresponding results from BLM, thus more likely to mis-adapt. Setup 3 improved over $ULM_0^{(a)}$ even though the regression pairs were learned from a set disjoint from the test data, suggesting some knowledge in the regression pairs were indeed transferable. Also as expected, it performed worse than cheating Setup 2.

Finally, the entirely unsupervised Setup 4 performed at least as well as Setup 3 although in Setup 4 we identified the regression pairs automatically using the LM score difference metric. Setup 4 provided an additional 3.1% SERR over $ULM_0^{(a)}$, thus extending the gains from unsupervised adaptation by almost two thirds; WERR was also improved by an additional 2.0%. A detailed look at the selected regression pairs shows that 39% of those selected had correct result with BLM and incorrect with $ULM_0^{(a)}$, precisely the kind of regressions we aimed to extract. Furthermore, virtually no utterance in the selected pairs had incorrect BLM result but correct $ULM_0^{(a)}$ result, the kind of patterns that would have led to mis-adaptation. The remaining 61% were incorrect with both BLM and $ULM_0^{(a)}$. Since we selected the regression pairs with the LM score difference metric using a relatively strict threshold, overall the selected ones are more likely to have "more correct" BLM results than $ULM_0^{(a)}$ results, thus minimizing potential harms from including such pairs in generating *NgramDiff* statistics. Interestingly, perplexity goes up between $ULM_0^{(a)}$ and each of the updated adapted models, even though recognition error rate is being reduced. This is due to the discriminative nature of the adaptation where optimizing perplexity no longer correlates with minimizing recognition errors, and a similar phenomenon was reported in previous discriminative training work such as [11].

Table 1: *Performance comparison of different setups on the test set. SERR% and WERR% are relative sentence error and word error rate reductions over the BLM, respectively.*

model	SERR%	WERR%	Perplexity
BLM	0	0	118.1
$ULM_0^{(a)}$	4.79	9.04	65.9
Setup 1	6.88	10.17	70.0
Setup 2	9.34	12.15	68.3
Setup 3	7.50	11.02	75.1
Setup 4	7.92	11.02	76.6

4. Discussion and Future Work

As explained in previous sections, the proposed solution leverages the unique setup of the LM adaptation framework to allow us rely on features such as LM score difference to effectively identify possible regression pairs between the BLM and the initially adapted LM. However, such features are less likely to work as well in other scenarios such as between two more independent LMs. The data filtering strategy presented in Section 2.3 took advantage of the nature of short utterances in our voice search applications. The technique is less likely to be successful for applications with longer utterances since more Ngrams would be inadvertently affected even though they may not be the true target of penalization during data filtering. In those cases direct modification of Ngram counts or parameters may be more effective.

The current experiments only made use of 1-best recognition results for identifying regression pairs and for unsupervised adaptation. It is possible to extend the method to more items on N-best list for additional gains but also with greater risk of penalizing insignificant error patterns unnecessarily. There are also a number of other ways to extend the proposed solution, such as running multiple iterations of regression-pair identification and data filtering and model update steps, as well as making explicit use of the positive adaptation pairs where the initial adapted LM indeed performed better than the BLM (case 4 in Section 2.2). Furthermore, it should be possible to take into account the acoustic model scores of the identified regression pairs to compute more precise adjustments needed in the adapted model to minimize recognition errors [11, 6, 12].

5. Conclusions

We have described a discriminative data filtering solution to improve unsupervised LM adaptation. We leveraged the unique setup of the adaptation framework to automatically predict recognition regression pairs between a baseline LM and an initial adapted LM, using features such as LM score difference. *NgramDiff* statistics were computed from the identified regression pairs and applied to the unsupervised adaptation data filtering, to penalize potent error attractors introduced through the initial adaptation. An updated adapted LM can be trained from the filtered adaptation data. We demonstrated the effectiveness of the proposed solution on a large, real-world voice catalog search application and achieved significant additional recognition error reduction over the initial adapted LM.

6. References

[1] Bellegarda, J. "Statistical language model adaptation:review and perspectives," Speech Communication, vol. 42, no. 1, pp. 93-108, 2004.

[2] Bacchiani, M. and Roark, M. "Unsupervised language model adaptation," in Proc. of ICASSP, 2003.

[3] Wang, W. and Stolcke, A. "Integrating MAP, marginals, and unsupervised language model adaptation ", in Proc. of Eurospeech 2007.

[4] Kurata, G., Itoh, N. and Nishimura, M."Acoustically discriminative training for language models," in Proc. of ICASSP, 2009.

[5] Jyothi, P. and Fosler-Lussier, E. "Discriminative language modeling using simulated ASR errors," in Proc. of Interspeech, 2010.

[6] Huang, J., Li, X. and Acero, A., "Discriminative Training Methods for Language Models Using Conditional Entropy Criteria," in Proc. of ICASSP, 2010.

[7] Xu, P., Roark, B. and Khudanpur, S. "Phrasal Cohort Based Unsupervised Discriminative Language Modeling," in Proc. of Interpspeech, 2012.

[8] Sagae, K., Lehr, M., Prudhommeaux, E., Xu, P., Glenn, N., Karakos, D., Khudanpur, S., Roark, B., Saraclar, M., Shafran, I., Bikel, D., Callison-Burch, C., Cao, Y., Hall, K., Hasler, E., Koehn, P., Lopez, A., Post, M. and Riley, D., "Hallucinated n-best lists for discriminative language modeling", in Proc. of ICASSP, 2012.

[9] Jyothi, P., Johnson, L., Chelba, C. and Strope, B. "Distributed Discriminative Language Models for Google Voice Search", in Proc. of ICASSP 2012.

[10] Ogawa, A., Hori, T. and Nakamura, A.: "Error type classification and word accuracy estimation using alignment features from word confusion network", in Proc. of ICASSP 2012.

[11] Kuo, J., Fosler-Lussier, E., Jiang, H. and Lee, C., "Discriminative training of language models for speech recognition", in Proc. of ICASSP, 2002.

[12] Magdin, V. and Jiang, H. "Discriminative Training of n-gram Language Models for Speech Recognition via Linear Programming", in Proc. of ASRU, 2009.

[13] Rastrow, A., Sethy, A. and Ramabhadran B. "Constrained Discriminative Training of N-gram Language Models," in Proc. of ASRU, 2009.

[14] Roark, B., Saraclar, M., Collins, M. and Johnson, M., "Discriminative language modeling with conditional random fields and the perceptron algorithm", in Proc. of ACL, 2004.

[15] Zhou, Z., Gao, J., Soong, F.K. and Meng, H. "A comparative study of discriminative methods for reranking LVCSR N-best hypotheses in domain adaptation and generalization," in Proc. of ICASSP, 2006.