Instance-Based On-line Language Model Adaptation

Ali Orkan Bayer, Giuseppe Riccardi

Signals and Interactive Systems Lab - University of Trento, Italy
{bayer, riccardi}@disi.unitn.it

Abstract

Language model (LM) adaptation is needed to improve the performance of language-based interaction systems. There are two important issues regarding LM adaptation; the selection of the target data set and the mathematical adaptation model. In the literature, usually statistics are drawn from the target data set (e.g. cache model) to augment (e.g. linearly) background statistical language models, as in the case of automatic speech recognition (ASR). Such models are relatively inexpensive to train, however they do not provide the necessary high-dimensional language context description needed for language-based interaction. Instance-based learning provides high-dimensional description of the lexical, semantic, or dialog context. In this paper, we present an instance-based approach to LM adaptation. We show that by retrieving similar instances from the training data and adapting the model with these instances, we can improve the performance of LMs. We propose two different similarity metrics for instance retrieval, edit distance and n-gram match score. We have performed instance-based adaptation on feed forward neural network LMs (NNLMs) to re-score n-best lists for ASR on the LUNA corpus, which includes conversational speech. We have achieved significant improvements in word error rate (WER) by using instance-based on-line LM adaptation on feed forward NNLMs.

Index Terms: Language Model Adaptation, Neural Network Language Models, Instance-Based Adaptation, Cache Models

1. Introduction

Language models (LMs) play a crucial role in automatic speech recognition (ASR) systems by constraining the search space. Currently, neural network language models (NNLMs) have gained popularity due to the improvements in computation power. NNLMs are first introduced in [1]. NNLMs project the discrete word space onto a continuous space, therefore they are more robust to the problem of data sparseness [2].

The performance of an LM depends on how closely it is related to the domain which is applied to. Since the performance of an LM is dependent on the domain, LM adaptation techniques can be used to apply an LM to a specific domain. A general review about statistical LM adaptation can be found in [3]. LM adaptation builds a statistical model to compensate the mismatch between the training data and the domain by using task specific data and a background model. There are different approaches like model interpolation, constraint specification, and using topic information. LM adaptation is mostly applied to n-gram models. However, recently there is a couple of applications of LM adaptation for neural networks. One of the approaches adapts the NNLM by cascading an adaptation layer between the projection and hidden layer of the network. During the adaptation process only the weights between the projection layer and adaptation layer are updated [4]. The other approach, which is applied to recurrent NNLMs is to retrain the whole network for a single iteration using the adaptation data [5]. Cache models aim at modeling the local word-frequency fluctuations by using a short-term memory [6]. In [7] an implementation of cache NNLM is presented for spoken language understanding tasks.

Information retrieval approaches have been applied to LM adaptation in [8, 9] which use *tf* and *idf* statistics to retrieve relevant documents for LM adaptation.

Instance-based approaches solve a new problem by remembering similar problems that were encountered before [10]. Therefore, extracting the relevant instances from previous experience plays a crucial role. In addition to retrieval, how to use the retrieved instances for improvement is also important.

In this paper, we present an instance-based on-line LM adaptation approach. We perform LM adaptation for the recognition of conversational speech by retrieving similar utterances from the training data. The next section introduces the instance retrieval process which constitutes the crucial part of the system. Section 3 presents our adaptation models, which are based on NNLMs. Section 4 reports the results of the speech recognition experiments for instance-based on-line LM adaptation.

2. Instance retrieval

The first step in instance-based learning cycle is the retrieval of the most similar instance or instances [10]. The process involves selecting instances from a collection of *previous instances* that are similar to the *new instance* at hand. We have applied this approach to on-line language model adaptation. In our approach the whole training set represents the collection of the previous instances and the ASR hypotheses represent the new instances.

2.1. Retrieval process

The retrieval process consists of scoring each training set utterance with respect to the ASR hypothesis. We have used edit distance and n-gram matching score as similarity metrics, which will be presented in detail. Computing similarity scores between the reference transcription of the training set and the ASR hypotheses has low performance due to ASR errors. To compensate that we have computed the similarity scores between the ASR output of the training set and the ASR hypotheses for the test set. Because of the similar errors that the ASR produces for each set, the similarity scores are more precise. On the other hand, the instances are retrieved from the reference transcriptions of the training set. The retrieval process is depicted in Figure 1.

This work was partially funded by Portdial FP7 project n. 296170

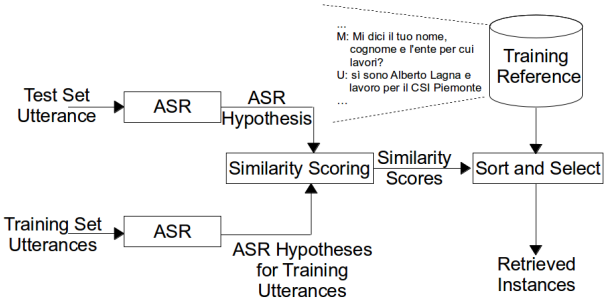

Figure 1: *Instance retrieval process. For each utterance, the similarity score is computed between the ASR hypothesis and the ASR output of the training utterances. The instances are retrieved from the corresponding reference transcriptions with respect to the similarity scores computed before.*

2.2. Similarity metrics

The first similarity metric we have used is the *edit distance*. The computation of the similarity score by using the edit distance metric is composed of several steps. First, each ASR hypothesis is aligned with the ASR output of all the training utterances. Then, edit distance (the total number of insertions, substitutions, and deletions) between them is calculated. Next, the training utterances are sorted in an ascending order with respect to edit distance. Finally, the ASR output of the training utterances is replaced with the corresponding reference transcription of the training set. Thus, at the end we have a list of previous instances that are sorted with respect to edit distance.

The second similarity metric is the *n-gram match score*, which is inspired by BLEU [11] that compares n-grams of the hypothesis with n-grams of the reference for evaluating machine translation systems. As in the previous metric, the similarity computation starts with aligning each ASR hypotheses with the ASR output of the training utterances. Then by using these alignments, the number of matching n-grams is calculated by using Equation 1, where n refers to the number of words in the ASR hypothesis, ug, bg, and ng refer to matching unigram count, matching bigram count and matching n-gram count respectively, and ins refers to the number of insertions.

$$score = (\frac{ug}{n} + \frac{bg}{n-1} + ... + \frac{ng}{1} - \frac{ins}{n})/n \qquad (1)$$

The training utterances are sorted in descending order with respect to the n-gram matching score and they are replaced with the corresponding reference transcription.

We have also applied conventional document retrieval methods that use *tf* and *idf* for computing similarity. However, we could not obtain significant improvements as we have obtained by using proposed similarity metrics.

3. Adaptation models

We have applied instance-based LM adaptation to NNLMs. The NNLM architecture we have used is an implementation of the feed-forward NNLM that is given in [1]. NNLMs output a probability for every word in the lexicon, given a history. The history is fed as input to the neural network using 1-of-n encoding. Every word in the history is mapped to a continuous vector space using the projection layer. The projection layer is connected to the hidden layer and also to the output layer by direct connections. The hidden layer has a non-linear activation function, the sigmoid function in this case. The hidden layer is connected to the output layer, which uses the softmax function to output a probability distribution. The architecture of the NNLM we have used is given in Figure 2.

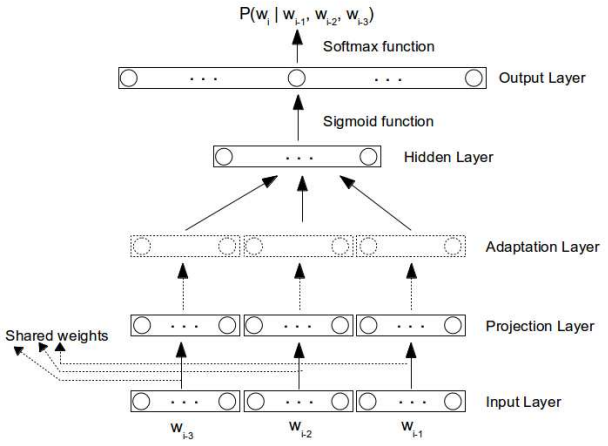

Figure 2: *The feed-forward 4-gram NNLM structure. The history is fed into the input layer using 1-of-n encoding. The projection layer is connected to the hidden layer and to the output layer (not shown in the figure). The hidden layer uses the sigmoid function as the activation function. To output a probability distribution the output layer uses the softmax function. The placement of the optional adaptation layer is also presented, which is only used in the adaptation layer approach.*

The training of NNLMs is done by using the backpropagation algorithm with cross-entropy error function. Learning rate is adjusted and over-fitting is avoided by using validation data.

LM adaptation is applied to a NNLM, the base model, which was built over the whole training set. During the adaptation process, the base model is adapted to the current utterance by using the instances that are retrieved for that utterance.

3.1. Re-training approach

The first approach for NNLM adaptation is to re-train the whole NNLM by using the retrieved instances. In this approach, after the instances are retrieved for an utterance, the whole neural network model is re-trained for 5 iterations over the retrieved instances.

3.2. Adaptation layer approach

The second approach is to use an adaptation layer. We have cascaded an adaptation layer between the projection layer and hidden layer of the NNLM as given in [4]. In the adaptation phase only the weights between the projection layer and the adaptation layer are updated. We have updated the weights for 5 iterations by using the instances that are retrieved.

3.3. Cache NNLMs

We have used an implementation of cache NNLMs to compare our instance-based on-line LM adaptation approach to. We have used the same architecture that is given in [7] and modified the network for ASR. Therefore, in addition to the basic architec-

ture we have given in Figure 2 we have connected a cache layer to the hidden layer. The cache layer is activated with the words in the previous user turns for each dialog as given in [7].

4. Experiments

We have used the Human-Machine (HM) part of the Italian LUNA conversational corpus [12] in the experiments. The LUNA corpus is collected by a customer care and technical support center for software and hardware. The HM part is collected with a Wizard of Oz approach. The corpus is split into training, development, and test sets, which include 3171, 387, and 634 utterances respectively. The training set has a vocabulary size of 2399, the out-of-vocabulary rate is 3.68% for the test set.

4.1. Baseline system

The baseline system we compare our results to is an ASR system that uses hybrid HMM/ANN acoustic models that were adapted to the corpus. It uses a conventional word based trigram LM with Kneser-Ney smoothing. It performs finite state transducer (FST) decoding and outputs lattices. We have used these lattices to generate a 100-best list for testing the performance of our instance-based on-line LM adaptation approach by performing re-scoring experiments. The performance of the ASR and the oracle word error rate of the 100-best list are given in Table 1.

The base NNLM is the background model that is used for instance-based LM adaptation. It is trained over the whole training data. The NNLM is a 4-gram LM, which uses the architecture that is given in Figure 2. The performance of this model without any adaptation is also given in Table 1.

Table 1: *Baseline performance. ASR uses a conventional trigram language model. Oracle error rates are given for the 100-best list. Base NNLM is a 4-gram NNLM which is trained over the whole training data. Base NNLM is the background model that is used in LM adaptation.*

	WER
1-best	22.3%
Oracle on 100-best list	15.6%
Base NNLM on 100-best list	21.6%

4.2. Instance-based on-line LM adaptation flow

The flow we have used for instance-based on-line LM adaptation is given in Figure 3. The procedure has the following steps. Each utterance that is to be recognized is passed through the baseline ASR system which outputs the best hypothesis and a 100-best list for that utterance. The relevant instances are retrieved with respect to the ASR hypothesis by using the retrieval process that is described in Section 2. On-line LM adaptation is applied to the base NNLM by using the retrieved instances. Finally, the 100-best list is re-scored by using the adapted NNLM.

4.3. Upper bounds for on-line LM adaptation

In this section we present how much performance gain we could get from our NNLM adaptation models. Therefore, reference transcription of the test set and the oracle hypotheses that are obtained from the 100-best list are used to reach the upper bound of the performance gain.

For this purpose, we have used the same adaptation flow

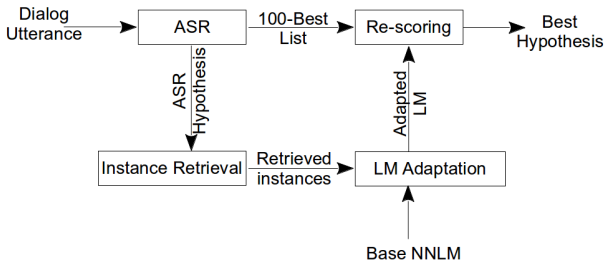

Figure 3: *The flow of on-line LM adaptation. The ASR hypothesis for each dialog utterance is used to retrieve the relevant instances from the training set. The base NNLM is adapted to the current utterance by using these instances. The 100-best list for that utterance is re-scored by using this adapted NNLM.*

for each utterance. However rather than using the retrieved instances, we have directly used the reference transcription, oracle hypothesis from the 100-best list, and the ASR hypothesis for that utterance. The results are given in Table 2. As can be seen from the results, both the re-training approach and the adaptation layer approach give similar results. In addition, it can be seen that both by using the reference transcription and the oracle hypothesis as the relevant instance, the performance improves significantly. On the other hand, using the ASR hypothesis as the relevant instance drops performance of the base model to the same performance of the ASR baseline.

Table 2: *Performance of LM adaptation when the test utterance itself is used as the relevant instance. Reference transcription, oracle hypothesis, and ASR hypothesis are used as the relevant instance.*

Relevant Instance	Re-training	Adaptation Layer
Reference	17.6%	17.9%
Oracle	16.8%	16.8%
ASR	22.3%	22.3%

The cache NNLM, on the other hand, performs almost the same when the reference transcription, the oracle hypotheses, and ASR hypotheses are used in the cache. The performance of cache NNLM is given in Table 3.

Table 3: *Performance of the cache NNLM. Reference transcription, oracle hypotheses, and ASR hypotheses are used in the cache.*

Cache Content	WER
Reference	21.7%
Oracle	21.6%
ASR	21.5%

4.4. Upper bounds for instance-based on-line LM adaptation

In this section we present the results on instance-based on-line LM adaptation where the reference transcription of the test set and the oracle hypotheses are used for retrieving the relevant instances from the training set. The purpose of presenting these results is to show the upper bounds of the instance-based on-line LM adaptation. Only for this section, when computing the similarity scores between the utterance and the instances, we have directly used the reference transcription of the training set,

since the utterances do not contain any ASR errors (in the case of reference) or they are minimum (in the case of oracle).

As can be seen from Table 4 and 5 re-training the whole network performs better than the adaptation layer approach. Retrieving the relevant instances both by using the reference transcription and by using the oracle hypotheses have similar performance.

Table 4: *Upper bounds for instance-based on-line LM adaptation. The instances are retrieved by using the reference transcription. "Ins. Ret." refers to the number of instances that are retrieved; 3, 9, 16, 31, and 158 corresponds to 0.1%, 0.3%, 0.5%, 1.0%, and 5.0% of the number of training utterances. "ng match" refers to the n-gram match score.*

Ins. Ret.	Re-training		Adaptation Layer	
	Edit dist.	ng match	Edit dist.	ng match
1	20.9%	20.9%	**21.7%**	**21.4%**
3	**20.6%**	**20.6%**	22.1%	22.0%
9	20.7%	**20.6%**	22.3%	22.4%
16	20.9%	20.8%	23.2%	22.8%
31	21.5%	21.5%	23.8%	23.5%
158	22.4%	22.4%	24.5%	24.0%

Table 5: *Upper bounds for instance-based on-line LM adaptation. The instances are retrieved by using the oracle hypotheses.*

Ins. Ret.	Re-training		Adaptation Layer	
	Edit dist.	ng match	Edit dist.	ng match
1	20.9%	21.0%	**21.5%**	**21.4%**
3	**20.6%**	20.9%	22.1%	21.9%
9	**20.6%**	**20.8%**	22.5%	22.5%
16	20.9%	20.9%	22.9%	22.8%
31	21.5%	21.5%	24.1%	23.3%
158	22.4%	22.5%	24.5%	24.0%

4.5. Instance-based on-line LM adaptation

In this section, we present the performance of the instance-based on-line LM adaptation experiments. In this setting the ASR hypotheses are used to retrieve the similar instances from the training set. The retrieval of relevant instances has key importance in this approach. To be able to compensate the ASR errors, we compute the similarity scores for each ASR hypothesis by using the ASR output of the training set and corresponding instances are retrieved from the reference transcription of the training data.

Table 6: *Performance of the instance-based on-line LM adaptation. ASR hypotheses are used to retrieve the relevant instances. Therefore, the similarity scores are computed on the ASR output of the training set. Then the corresponding instances are retrieved from the reference transcription of the training set.*

Ins. Ret.	Re-training		Adaptation Layer	
	Edit dist.	ng match	Edit dist.	ng match
1	21.3%	21.3%	**21.8%**	**21.8%**
3	20.9%	**20.7%**	22.5%	22.1%
9	**20.8%**	20.8%	22.6%	22.6%
16	20.9%	20.9%	22.7%	22.8%
31	21.6%	21.4%	23.2%	23.0%
158	22.3%	22.4%	24.1%	24.0%

The performance of instance-based on-line LM adaptation is given in Table 6. The results show that re-training the whole network by using the retrieved instances outperforms the adaptation layer approach. The performance of the both metrics is very close to their upper bounds (compared to Table 4 and 5). By using instance-based on-line LM adaptation, we have achieved 7% relative (1.6% absolute) improvement with respect to the ASR baseline and 4% relative (0.8% absolute) improvement with respect to the cache NNLM. In addition, when compared to the results given in Table 2 it can be seen that there is still a large margin for improvement.

4.6. Statistical significance of the results

In this section, we show that the improvements we have achieved by using instance-based on-line LM adaptation are statistically significant. We have used the bootstrap method that is given in [13] to calculate the confidence intervals. We have calculated *bootstrap-t* confidence intervals using 10^4 bootstrap replications. The p-value is calculated using the randomization method given in [14].

Comparisons are made by considering the ASR baseline system and the cache NNLM as baselines. Thus, the performance of instance-based on-line LM adaptation that is based on re-training and that uses both similarity metrics are compared against these baselines.

Table 7: *The performance of the ASR baseline and the instance-based on-line LM adaptation is given. 90% confidence intervals using 10^4 bootstrap replications are given in brackets. Also p-values for the comparison between the ASR baseline and the two approaches are given. The results show that the improvements are significant.*

	WER	p-value
ASR	22.3% [20.9 - 23.6]	NA
Edit dist. best	20.8% [19.4 - 22.1]	9.9×10^{-5}
n-gram match best	20.7% [19.3 - 22.0]	9.9×10^{-5}

Table 8: *The performance of the Cache NNLM and the instance-based on-line LM adaptation is given. The results show that the improvements are significant.*

	WER	p-value
Cache NNLM	21.5% [20.2 - 22.8]	NA
Edit dist. best	20.8% [19.4 - 22.1]	0.01
n-gram match best	20.7% [19.3 - 22.0]	1.1×10^{-3}

As can be seen from Table 7 and 8 the improvements we have achieved both against the ASR baseline and the cache NNLM are statistically significant.

5. Conclusion

In this paper, we have presented an instance-based on-line LM adaptation method for NNLMs. Instance-based LM adaptation retrieves relevant utterances from the training set and adapts the LM by using these instances. We have presented two different metrics for instance retrieval, edit distance and n-gram match score. In addition, we have given two different adaptation approaches; re-training the whole network and the adaptation layer approach. We have observed that n-gram match score with the re-training approach performs the best. By using re-scoring experiments over 100-best lists, we have obtained 7% relative improvement with respect to the ASR baseline and 4% relative improvement with respect to the cache NNLM. We have also showed that these improvements are statistically significant.

6. References

[1] Bengio, Y. and Ducharme, R. and Vincent, P. and Jauvin, C., "A neural probabilistic language model", Journal of Machine Learning Research, 3:1137–1155, 2003.

[2] Schwenk, H., "Continuous space language models", Computer Speech Language, 21(3):492–518, 2007.

[3] Bellegarda, J., R., "Statistical language model adaptation: review and perspectives", Speech Communication, 42:93–108, 2004.

[4] Park, J. and Liu, X. and Gales, M.J.F. and Woodland, P.C., "Improved neural network based language modelling and adaptation", In Proceedings of Interspeech 2010, 1041–1044, 2010.

[5] Kombrink, S. and Mikolov, T. and Karafiat, M. and Burget, L., "Recurrent neural network based language modeling in meeting recognition", In Proceedings of Interspeech 2011, 2877–2880, 2011.

[6] Kuhn, R. and De-Mori, R., "A cache-based natural language model for speech recognition", IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(6):570–583, 1990.

[7] Zamora-Martinez, F. and Espana-Boquera, S. and Castro-Bleda, M.J. and De-Mori, R., "Cache neural network language models based on long-distance dependencies for a spoken dialog system" In proceedings of ICASSP 2012, 4993–4996, 2012.

[8] Chen, L. and Gauvain, J.L. and Lamel, L. and Adda, G. and Adda, M., "Using information retrieval methods for language model adaptation", In proceedings of Eurospeech 2001, 255–258, 2001.

[9] Eck, M. and Vogel, S. and Waibel, A., "Language model adaptation for statistical machine translation based on information retrieval", In proceedings of LREC 2004, 327–330, 2004.

[10] Aamodt, A. and Plaza, E., "Case-based reasoning: foundational issues, methodological variations, and system approaches", AI Communications, 7(1):39–59, 1994.

[11] Papineni, K. and Roukos, S. and Ward, T. and Zhu W.J., "BLEU: a method for automatic evaluation of machine translation" In proceedings of the 40th Annual Meeting of ACL, 311–318, 2002.

[12] Dinarelli, M. and Quarteroni, S. and Tonelli, S. and Moschitti, A. and Riccardi, G., "Annotating spoken dialogs: from speech segments to dialog acts and frame semantics", in Proceedings of SRSL Workshop of EACL, 2009.

[13] Bisani, M., and Ney, H., "Bootstrap estimates for confidence intervals in ASR performance evaluation", In proceedings of ICASSP 2004, 409–412, 2004.

[14] Yeh, A., "More accurate tests for the statistical significance of result differences", In proceedings COLING'00, 947–953, 2000.