Language Modeling for Dialog System

Wei Xu, Alex Rudnicky
School of Compueter Science, Carnegie Mellon University
Pittsburgh, Pennsylvania, 15213, USA
xw@cs.cmu.edu, air@cs.cmu.edu

Abstract

Language modeling for speech recognizer in dialog systems can take two forms. Human input can be constrained through a directed dialog, allowing the decoder to use a state-specific language model to improve recognition accuracy. Mixed-initiative systems allow for human input that while domain-specific might not be state-specific. Nevertheless, for the most part human input to a mixed-initiative system is predictable, particularly when given information about the immediately preceding system prompt. The work reported in this paper addresses the problem of balancing state-specific and general language modeling in a mixed-initiative dialog system. By incorporating dialog state adaptation of the language model, we have reduced the recognition error rate by 11.5%.

1. Introduction

Recent advances in speech recognition technologies and computer hardware have made it possible to build human-computer spoken dialogue systems for a wide variety of application. However, the performance of speech recognition is still a bottleneck of these systems [7]. A lot of research effort has been devoted to detecting and recovering from recognition errors.

In this work, we have tried to improve the recognition performance of Carnegie Mellon Communicator [11], a telephone based automated travel agent system, by incorporating dialogue state adaptation of the language model. Language modeling for speech recognizer in dialog systems can take two forms. Human input can be constrained through a directed dialog, allowing the decoder to use a state-specific language model to improve recognition accuracy [6] [9]. In this way, dialog states have been used to partition the whole set of utterances into subsets and then train standard n-gram language models from each partitioned set. Mixed-initiative dialog systems allow for human input that while domain-specific might not be state-specific. Nevertheless, for the most part, human input to a mixed-initiative system is predictable, particularly when given information about the immediately preceding system prompt. In [10], the state-specific language models were interpolated with a general language model using Viterbi algorithm.

The work reported in this paper addresses the problem of balancing state-specific and general language modeling in a mixed-initiative dialog system. We have shown that by our approach, we can improve the system performance. The performance is reported in terms of perplexity and actual recognition word accuracy.

2. System overview

The dialogue system we do our experiment on is CMU Communicator, a telephone based automated travel planning system. Communicator is a mixed-initiative spoken dialog system. In this system, Sphinx-2 speech recognizer transcribes user' s speech into text and passes to Phoenix parser to generate semantic interpretation. Then dialog manager decides how to interact with user and database. At different state of a dialog, the dialog manager will give different prompt to the user and the user' s response may or may not relate to system' s prompt.

3. Building state dependent language model

Because user's response depends on what is heard by the user, we can define state as preceding system prompt (i.e. the natural language generation frame). User's utterances are classified into 16 states (Table 1) according to its preceding natural language generation frames. We take the following procedures to build the state dependent language model.

- A general language model is built from the whole corpus, using Katz backoff with Good-Turing discount. We use CMU-Cambridge toolkit [1] to build the language model.

- We build a trigram back-off language model for each state. The unigram probabilities are back-off to the unigram probability of the general model:

  $$\hat{P}_S(w) = \begin{cases} P_S(w) & \text{if } w \in V_S \\ \lambda P_G(w) & \text{otherwise} \end{cases}$$

  Where V_S is the vocabulary from the corpus of each state, λ is a normalization factor in order to make $\hat{P}_S(w)$ sum to 1.

- Then each state dependent language model is linearly interpolated with the general language model. The interpolation weights are optimally determined by EM (Expectation-Maximization [2]) algorithm using separate holdout data. The interpolated probability of a word is given by:

  $$P(W) = \alpha P_S(W) + \beta P_G(W)$$

  where the interpolation weights α and β satisfy $\alpha + \beta = 1$

Our corpus is drawn from data collected using the CMU Communicator. The log file of the system contains every system output and user input. Data collected from June 1998 to May 1999 is used as development data, which has 182K words, 42K utterances. Test data is from the recording in June 1999, which has 6289 words, 1750 utterances.

State	Test size	$Perp_G$	$Perp_S$	Perp Reduction	$WERR_G$	$WERR_S$	WERR Reduction
need_car	169	7.04	4.74	32.7	12.4	13.0	-4.8
need_hotel	161	9.00	6.71	25.5	19.9	19.3	3.1
hotel_where	202	9.31	6.82	26.8	18.3	15.8	13.5
Epilogue	138	360	205	43.2	73.9	67.4	8.8
arrive_city	1131	6.14	4.93	19.6	24.8	22.6	8.6
Confirm	222	9.38	8.69	7.4	18.0	16.7	7.5
Confirm_flight	1412	9.51	8.25	13.2	22.5	21.3	5.1
depart_date	937	8.83	7.11	19.4	15.7	14.7	6.1
query_go_on	72	9.94	7.06	28.9	16.7	12.5	25.0
query_name	958	8.42	7.11	15.5	30.8	23.7	23.1
query_pay_by_card	29	85.5	55.1	35.6	75.9	69.0	9.1
preferred_airport	69	10.4	7.11	31.9	42.0	39.1	6.9
query_return	706	5.85	4.31	26.3	14.2	11.1	22.0
query_summary	23	5.04	3.55	29.6	17.4	4.4	75.0
Remaining	51	4.87	4.93	-1.2	13.7	13.7	0.0
excuse_me	9	5.37	4.93	8.02	22.2	22.2	0.0
Overall	**6289**	**8.70**	**6.98**	**19.69**	**23.0**	**20.4**	**11.5**

Table 1 The comparison of the perplexity and recognition word error rate between general language model and state dependent language model factorized by dialog state.

The language model for Communicator is a class based language model. The total vocabulary size is about 2500. There are 1354 classes in the language model, among which 20 classes correspond to word classes such as [city] and [airport], etc, while each of the other 1334 classes corresponds to a single word. The perplexities reported in this paper do not take the within-class probability into account.

Perplexity and recognition word error rate of general language model and state dependent language model for each dialog state are shown in Table 1. The result shows a high correlation between dialog states and the responses from user. The word error rate has a significant reduction of 11.5% after using state dependent language model.

4. Clustering of Utterances

The improvement in section 3 is encouraging. However it should be possible to further improve the predictive ability of the language model. The following observations can be made about users' language in a particular state:

- Users tend to talk about a number of different topics and will naturally use different language for each topic. If we know the topic of the utterance (within the state), we can more precisely model the language.
- There are some patterns in user's utterances. Within each utterance cluster, word sequence is more predictable.

Given an utterance, its cluster is not known a priori. So we have a probability distribution of the clusters. The probability of an utterance is the weighted sum of the conditional probability that the utterance is generated from each cluster:

$$P(\mathbf{W}) = \sum_c P(c)P(\mathbf{W} \mid c)$$

$$P(\mathbf{W} \mid c) = \prod_i P(W_i \mid \mathbf{h}_i, c)$$

Where $\mathbf{W}$ is an utterance, c is a cluster, W_i is the i[th] word in utterance $\mathbf{W}$, $\mathbf{h}_i$ is the history words for W_i.

[3][5][8][12] also proposed similar clustering idea. However, our approach is different from previous work in that we directly use trigram instead of unigram to do the clustering. Using unigram for clustering cannot model the local regularities of language, while it is possible to find out some local regularities by using n-gram ($n \geq 2$).

In different dialog state, the distribution of cluster is different. This can be modeled by conditioning the cluster probability on the dialog state:

$$P(\mathbf{W} \mid s) = \sum_c P(c \mid s)P(\mathbf{W} \mid c)$$

We want to classify user's utterances into different clusters such that the utterances within each cluster are similar. This can be done using likelihood as the measure of similarity among the utterances within a cluster. We build $|c|$ trigram language models for each cluster such that the likelihood of the whole data is maximized. EM algorithm can be used to find the optimal parameters that maximize the likelihood. Here, the parameters that need to be optimized are the trigram probabilities $P(w \mid u, v, c)$ in each language model and the cluster probabilities given dialog state $P(c \mid s)$.

Specifically, first we calculated the expected counts of the trigram-class pair and state-class pair using current parameters:

$$count(c,s) = \sum_{n:S(\mathbf{W}^n)=s} \frac{P(c \mid s)P(\mathbf{W}^n \mid c)}{P(\mathbf{W}^n)}$$

$$count(u,v,w,c) = \sum_{n:S(\mathbf{W}^n)=s} \frac{P(c \mid s)P(\mathbf{W}^n \mid c)}{P(\mathbf{W}^n)} count(u,v,w,\mathbf{W}^n)$$

Where $s(\mathbf{w}^n)$ is the state of the utterance $\mathbf{W}^n$

Then update parameters by normalizing the counts to probabilities.

$$P(c \mid s) = \frac{count(c,s)}{\sum_s count(c,s)}$$

$$P(w \mid u,v,c) = \frac{count(u,v,w,c)}{\sum_w count(u,v,w,c)}$$

Iterate above two steps until convergence.

Figure 1 shows the distribution of $P(c \mid s)$ obtained through above algorithm. Each block in the picture represents a particular value of $P(c \mid s)$, where darker block indicates higher probability. The figure clearly shows that different dialog state has very different cluster distribution. In some states, most utterances belong to only one cluster; while in some other state, user' s language tend to distribute over many clusters.

After above algorithm, we can get the probability of an utterance belonging to a cluster given its dialog state:

$$P(c \mid \mathbf{W}, s) = \frac{P(c \mid s)P(\mathbf{W} \mid c)}{P(\mathbf{W} \mid s)}$$

Thus each utterance can be labeled with the most probable cluster that it belongs to. Then all the utterances are partitioned into different clusters and a Katz back-off trigram language model is built for each cluster using CMU-Cambridge Toolkit. These trigram language models need to be smoothed since they do not have enough training data. We can do the smoothing by interpolating the cluster model with a general language model. There can be two ways of interpolation.

One way is to interpolate at the utterance level. The probability of an utterance is the weighted sum of the probability calculated using cluster model and the general model.

$$P(\mathbf{W} \mid s) = \alpha_s P_{cluster\_lm}(\mathbf{W}) + \beta_s P_{general}(\mathbf{W})$$

$$P_{cluster\_lm}(\mathbf{W} \mid s) = \sum_c P(c \mid s)P(\mathbf{W} \mid c)$$

where $\alpha_s + \beta_s = 1$

The other way is to interpolate at word level. The probability of a word is the weighted sum of the probability calculated using cluster model and the general model.

$$\tilde{P}(\mathbf{W} \mid c) = \prod_i \left(\alpha_c P(W_i \mid \mathbf{h}_i, c) + \beta_c P_{general}(W_i \mid \mathbf{h}_i)\right)$$

$$P(\mathbf{W}) = \sum_c P(c \mid s)\tilde{P}(\mathbf{W} \mid c)$$

where $\alpha_c + \beta_c = 1$

Again, EM algorithm is used to estimate the optimal interpolation weights on holdout data.

Table 2 compares the perplexity of cluster language model with the language model without clustering. Using word level clustering, the perplexity reduces to 6.85 from 6.98. This improvement is not so significant as reported in [3][5][8] [12], since dialog state has already cluster user utterances in a very good way.

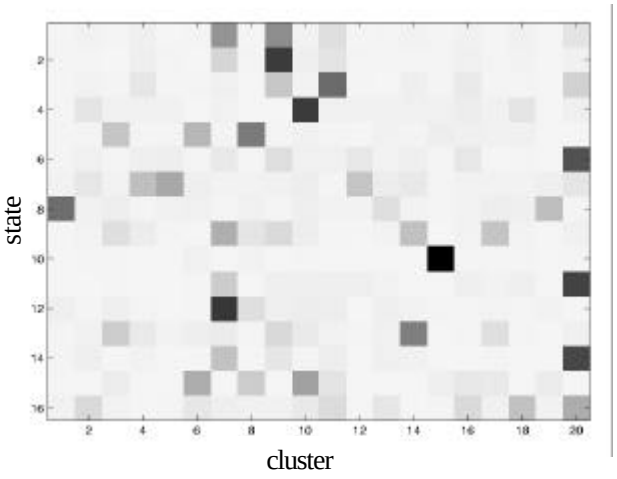

Figure 1 Distribution of cluster for each dialog state

	Perp.	WER
General	8.70	27.0
State	6.98	24.7
Cluster no inter	7.51	
Cluster utt. inter.	7.04	
Cluster word inter.	6.85	24.8

Table 2. Comparison of cluster model with state dependent model[1]

	Perplexity
Base state model	6.975
Clustering using unigram	6.972
Clustering using bigram	6.937
Clustering using trigram	6.853

Table 3 Difference when clustering using different n-gram

To compare the recognition result, first we generate word lattice using the state dependent language model. Then we use state dependent language model and cluster language model to rescore the lattice respectively. It turns out that that the cluster language model slightly increases the word error rate. [3][4][12] also reported that using clustering technique either deteriorates

[1] The word error rate is greater than that reported in Table 1. In Table 1, we do not use lattice rescoring. For some unknown reason, lattice rescoring increases the error rate. But for the purpose of evaluating the recognition performance of cluster language model we need use lattice rescoring.

or slightly improves the recognition performance. [12] suggested that better smoothing method (e.g. Kneser-Ney smoothing) need to be applied to cluster language models in order to get good performance.

We have also compared the effect of different length of gram on clustering. We cluster the utterances using unigram, bigram, and trigram respectively and build different cluster language models. Table 3 shows the different performance when clustering utterances using different n-gram. Using trigram for clustering gives better performance for the cluster language model.

5. Conclustions

Using state dependent language models, both perplexity and word error rate of speech recognition can be improved significantly and the dynamic switching between different state dependent language models has been implemented in the Communicator system to benefit from the reduced recognition error rate.

Utterance cluster language model does not improve the recognition performance. However, better smoothing techniques are expected to improve the performance of cluster model. Using trigram to cluster utterances is better than using unigram for the purpose of language modeling.

Acknowledgement

This research was sponsored by the Space and Naval Warfare Systems Center, San Diego, under Grant No. N66001-99-1-8905. The content of the information in this publication does not necessarily reflect the position or the policy of the US Government, and no official endorsement should be inferred.

References

[1] P.R. Clarkson and R. Rosenfeld. "*Statistical Language Modeling Using the CMU-Cambridge Toolkit*" Proceedings ESCA Eurospeech 1997

[2] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm", Journal of the Royal Statistical Society B, 39:1-38, 1977

[3] D. Gildea and T. Hofmann, "*Topic-based Language Models Using EM*" in Proc. Eurospeech-99, 1999

[4] J. Goodman, "*Putting It All Together: Language Model Combination*", in Proc. IEEE ICASSP-2000, Istanbul, June 2000.

[5] R. Kneser and J. Peters, "*Semantic Clustering for Adaptive Language Modeling*", in Proc. IEEE ICASSP-95, pp. 191-184, 1995

[6] A. Kurematsu and A. Sukenori, " *Language Model Selection based on the Analysis of Japanese Spontaneous Speech on Travel Arrangement*" in Proc. Eurospeech-99, 1999

[7] D.J. Litman and S. Pan, "*Empirically evaluating an adaptable spoken dialogue system*" In Proceedings of the 7th International conference on User Modeling (UM), June 1999.

[8] M. Mahajan, D Beeferman and X. D. Huang, "*Improved Topic-dependent Language Modeling Using Information Retrieval Techniques"* in Proc. IEEE ICASSP-99, 1999

[9] C. Popovici and P. Baggia, "*Specialized Language Models Using Dialogue Predictions*", in Proc ICASSP-97, pp. 815-818, Munich, 1997

[10] G. Riccardi, A. Potamianos and S. Narayanan, "*Language Model Adaptation for Spoken Language Systems*" Proc. of ICSLP-98, 1998

[11] A. Rudnicky, E. Thayer, P. Constantinides, C. Tchou, R. Shern, K. Lenzo, W. Xu, A. Oh, "*Creating Natural Dialogs in the Carnegie Mellon Communicator System*". Proceedings of Eurospeech' 99

[12] K. Seymore and R. Rosenfield, "*Using Story Topics for Language Model Adaptation*" in Proc. Eurospeech-97, 1997