

ADAPTIVE LANGUAGE MODELS FOR SPOKEN DIALOGUE SYSTEMS

Roger Argiles Solsona, Eric Fosler-Lussier, Hong-Kwang J. Kuo, Alexandros Potamianos, Imed Zitouni

Bell Labs, Lucent Technologies, 600 Mountain Avenue, Murray Hill, NJ 07974, U.S.A.
{fosler, kuo, potam, zitouni}@research.bell-labs.com

ABSTRACT

In this paper, we investigate both generative and statistical approaches for language modeling in spoken dialogue systems. Semantic class-based finite state and n -gram grammars are used for improving coverage and modeling accuracy when little training data is available. We have implemented dialogue-state specific language model adaptation to reduce perplexity and improve the efficiency of grammars for spoken dialogue systems. A novel algorithm for combining state-independent n -gram and state-dependent finite state grammars using acoustic confidence scores is proposed. Using this combination strategy, a relative word error reduction of 12% is achieved for certain dialogue states within a travel reservation task. Finally, semantic class multigrams are proposed and briefly evaluated for language modeling in dialogue systems.

1. INTRODUCTION

One of the critical information sources for spoken dialogue systems is the language model; this constrains to a large degree what the user can say to the system (or, at least, what the system can *understand*). The typical automatic speech recognition (ASR) component of the dialogue system uses an n -gram grammar for recognition, where the probability of a word is conditioned on the $n - 1$ previous words. An obvious extension of this paradigm for sparse training data is to use class-based n -gram grammars [1], where the probability of a word is conditioned on its history via the word's class. The addition of classes allows system designers to incorporate knowledge about the domain to fill in holes in the training data. For instance, in a travel reservation application, the user can select from over a thousand cities, most of which will not occur in the training set. The system designer can just create a class $\langle city \rangle$ containing all of the relevant city names, and compute a probability distribution over all of the words in the class ($P(w|\langle city \rangle)$).

Another way that the user can incorporate domain knowledge into the ASR engine is to make the language model *state-dependent*. The assumption is that the responses of the user will depend on what the system said to the user. For instance, a different answer is expected if the question is "How may I help you?" versus "What city do you want to fly out of?" Data sparseness is a problem also in this case, since the corpus has to be divided into smaller state-dependent sub-corpora; furthermore, one runs the risk of a user saying a sentence out of context for the dialogue state, which would consequently be poorly modeled by the system. An often cited solution to this problem is to interpolate a dialogue-state specific class n -gram with a class n -gram trained on the entire corpus

This research was partially supported by the DARPA Communicator project. During this work period, Roger Argiles Solsona was a student at Universitat Politècnica de Catalunya, Barcelona, Spain. He is currently at Ecole Nationale Supérieure de l'Aéronautique et de l'Espace, Toulouse, France.

([2, 3], *inter alia*). A further description of this technique can be found in section 3.

One can also integrate knowledge by combining n -grams with stochastic context free grammars (SCFGs) [4, 5]. This technique can incorporate knowledge from the parser of the natural language understanding system. However, for many simple systems, an extensive parser is not available and can be time consuming to construct. Many implementations are also restricted to re-scoring N -best lists provided by the recognizer, although this is not necessarily an inherent limitation of the methodology.¹

In this paper, we propose constructing dialogue-state specific finite state grammars (FSG) from user utterances, and running parallel recognizers with the state-dependent FSGs and context-independent n -grams. These FSGs are very simple to induce; by using the same class-based techniques as in n -gram modeling, one can also generalize the FSGs beyond what is seen in the training corpus. When the user says a sentence that is *in-grammar* (i.e. represented exactly by at least one path in the FSG), the improvement can be substantial over the n -gram. Thus, we have divided the recognition problem into two parts: finding the best hypotheses according to the FSG or n -gram grammar, and then deciding whether the sentence was in-grammar or out-of-grammar.

2. LANGUAGE MODELING FOR SPOKEN DIALOGUE SYSTEMS

The term grammar is used to denote any type of constraint on the legal word sequences. Thus in the strict linguistic sense, it may very well cover not only lexical and syntactic, but also semantic and pragmatic constraints as given by the recognition task. This is especially relevant for spoken dialogue systems where the range of possible user input is constrained by the system prompt and the grammar perplexity is relatively small. Next, we discuss how to expand traditional n -gram and finite state grammars to incorporate semantic information.

2.1. Semantic Class-based language model

Class-based language models [1] are used when there is insufficient data to generate a word-based language model. We have augmented our trigram grammar with a set of semantic classes: groups of words that share a semantic category relevant to the spoken dialogue task. Examples from the travel reservation domain include city names, states, airport names, months, days, etc.

To develop the class-based grammar, we first select a set of semantic classes appropriate for the domain. We then annotate the language model training corpus with the semantic classes: the training corpus is parsed with our natural language understanding

¹See [4] for an implementation that incorporates the SCFG in the first-pass search.

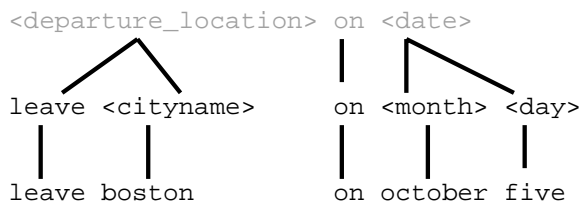


Fig. 1. Semantic parse of “Leave Boston on October five”

grammar and we find the constituents corresponding to the chosen semantic classes [6] (Figure 1). Typically, the relevant classes are found in the lowest non-terminal level of the parse.² A side effect of this process is that phrases will be created with multiple words fall under one concept — for example, NEW_YORK and SAINT_CROIX become phrases of the type $\langle cityname \rangle$.

Let $W = w_1, \dots, w_n$ be a sequence of words, and $c(w_i)$ the class the word w_i belongs to. If the classes are non-overlapping, then $c(w_i)$ is unique and, in the case of the class trigram model, the likelihood of W is as follows [7]:

$$P(W) = \prod_{i=1}^n P(w_i | c(w_i)) P(c(w_i) | c(w_{i-2}) c(w_{i-1})) \quad (1)$$

where $P(w_i | c(w_i))$ is the probability of the word w_i occurring in the semantic class $c(w_i)$. The probability distribution $P(w | c(w))$ depends on the semantic class. For instance, for the ‘month’ class we use the uniform distribution, but for the ‘city-name’ class it is a function of the annual traffic for the airport.

2.2. Semantic Class Finite State Grammars

Finite State Grammars (FSG) have been used successfully when building system-initiated dialogue systems for limited domains, e.g., travel reservation. Compared to the n -gram language model, FSGs have the advantage of very low perplexity and of explicitly modeling long distance dependencies. However, the generative FSG approach suffers from inadequate coverage especially when mixed-initiative dialogue strategies are implemented. To obtain adequate coverage and performance for mixed initiative dialogues a large amount of data must be collected to train the FSGs; FSGs thus become very large, which in turn decreases recognition speed.

We can automatically induce semantic class-based FSGs directly from a parsed training corpus. By re-utilizing the parsed training corpus (see Figure 1), we can increase the coverage by generalizing over the semantic classes. Since collecting all of the sentences in a corpus into one finite state grammar can result in a large automaton, in this framework one can easily introduce thresholding on the number of instances required for inclusion. However, finite state grammars alone, even with semantic class generalizations, will still not have a complete coverage of the utterances in the test corpus. In Section 3.2, we propose combining the FSG and n -gram to improve coverage and performance.

3. LANGUAGE MODEL ADAPTATION

Many adaptation techniques are used to build a language model that performs well in a specific task with a small corpus [8, 9].

²Strictly speaking, full parsing is not necessary, but using the grammar makes it easier to interpret ambiguous words (cf. October five with five p.m.).

In spoken dialogue systems, the language of the user is most assuredly dependent on the state of the dialogue; for each state we can collect a sub-corpus that answers a specific prompt type. Thus, the goal in this work is to use a large amount of mixed or general data and a specific smaller homogeneous state-specific corpus in order to build an adapted language model.

3.1. Combining state independent and dependent n -grams

A well-known method of adaptation to a specific sub-domain (or dialogue state) is to build separate language models on the general and specific training data and then combine the models through linear interpolation [10]. Let LM_G be the language model trained on all the training data collected and LM_{S_i} the model trained on data specific to dialogue state i (i.e. in response to a class of prompts asking for a particular type of information). The combined trigram model LM_C used in state i estimates the likelihood of a word w_i given the two preceding words w_{i-2}, w_{i-1} (in a trigram model) as follows:

$$P_{LM_C}(w_i | w_{i-2}, w_{i-1}) = \lambda P_{LM_{S_i}}(w_i | w_{i-2}, w_{i-1}) + (1 - \lambda) P_{LM_G}(w_i | w_{i-2}, w_{i-1}) \quad (2)$$

where λ designates the interpolation weight, which is trained on a development corpus to optimize word error rate or perplexity.

3.2. Combining state-dependent FSGs and state-independent n -grams

As noted above, finite state grammars will often have lower word error rates for in-grammar utterances than n -grams, but will likely have worse performance on out-of-grammar utterances. Thus, combining the two grammars should produce a more robust result. However, linear interpolation is not appropriate here, as the FSG does not carry any probabilities. Therefore, we propose to combine the *results* of the recognition with a state-independent class n -gram language model with the *results* of recognition with a state-dependent FSG. We use a small FSG to recognize the most common utterances and to use the general model for those sentences not included in the FSG. Both recognizers are run in parallel, and a decision is made after both decodings complete. This is similar in spirit to other system combination methodologies (e.g., [11]).

However, we will not know *a priori* if an utterance is included in the FSG; so we need to decide how to combine the results from the two language models in order to minimize the WER. One way to decide whether we should choose the FSG or n -gram recognition is to use an acoustic confidence measure [12]; the measure in question is a phone-based likelihood ratio test, where each phone is compared against its N -best rivals. One advantage of this measure is that it is computed during the Viterbi search, so a second pass is not necessary.

4. EVALUATION

4.1. Database

The database used for our experiments was collected as part of the DARPA Communicator Project (<http://fofoca.mitre.org>), a project aimed at improving natural-language mixed-initiative dialogue systems with a real back-end lookup of live information. The specific task was a travel reservation system allowing users to make travel arrangements for airplane flights, cars, and hotels. Subjects recruited by NIST were instructed to call systems built by several

participating sites. The training corpus used in our experiments contained a total of 19283 sentences (58595 words), composed of data obtained from the June 2000 DARPA Communicator data collection and a database which Colorado University had collected previously.

A first approximation to the state of the dialogue is the prompt or question that was asked by the machine. Thus we can try to improve our language model by building prompt-specific language models. Because the training data were collected across different participating sites with different dialogue systems, however, the system prompts are different across sites. Prompts with essentially the same meaning and function were clustered together in a semi-automatic fashion by first parsing the prompts to generalize the semantic concepts. For example, “When would you like to depart from Detroit?” will be generalized to “When would you like to depart from *<cityname>*?” The different prompts are then clustered semi-automatically into 7 different classes, with an additional “Other” class. Unfortunately, we only had prompts for about half of the LM training data (10573 utterances). The breakdown of these sentences into the different classes are shown in Table 1. A sub-set consisting of 10% of the training data was used as a development set to tune various parameters.

The test set for all the experiments consists of a total of 1257 sentences extracted from the June 2001 DARPA Communicator evaluation period.

State	Request	# Sentences
1	How can I help you?	684
2	Departure-arrival city	1453
3	Departure date	1131
4	Departure time	917
5	Airline preference	218
6	Confirmation	562
7	Next destination	623
8	Others	4596

Table 1. Dialogue-state dependent training corpus

4.2. Experiments

Our initial experiment focused on evaluating the coverage of the finite state grammars that were constructed for the different dialogue states. The finite state grammars were built entirely based on a data-driven approach. Sentences spoken by the user associated with a certain dialogue state were first parsed as described in Section 2.1. From such parsed sentences, only those that appear four or more times were used to construct the finite state grammar. The grammar was then simplified and expanded to include the semantic concepts based on standard finite state functions provided by the Bell Labs Finite State Machine package.

The threshold of four ensured that, except for state 1, more than 70% of the corpus would be included within the FSG. We observed that using an FSG to model state 1 was inappropriate because the variability of the answers given by the user was too large given the open-ended nature of the prompt. With a threshold of four, fewer than 5% of the sentences in state 1 appeared in the resulting FSG. Therefore, the FSG strategy is clearly not appropriate for overly open-ended prompts, and we would fall back to the traditional *n*-gram grammar. As a result, we will not report further results for state 1. Similarly, we will also not report any results for

state 8, the class containing utterances that do not fit in any of the other classes.

The speech recognition experiments were performed using two different recognizers, one based on *n*-gram statistical language models and another on finite state grammars. The results of the recognition experiments are presented in Table 2. Word error rates for utterances in three states (2, 3, and 4) that contained the most number of training sentences are shown. The fifth column shows the results averaged over these three states; these are the results we will focus our discussion on. Finally, in the sixth column, the average results for states 2 through 7 are given for completeness; we will not focus on these results.

State	2	3	4	Avg (2-4)	Avg (2-7)
# Sentences	308	294	240	842	1257
# Words	710	839	665	2214	3077
3-gram	53.4%	23.7%	46.0%	39.9%	—
class 3-gram	32.7%	18.0%	12.6%	21.1%	19.9
adapted 3-gram	31.7%	16.7%	13.5%	20.6%	19.9
adapted FSG	27.5%	15.9%	12.2%	18.5%	18.4

Table 2. Word error rate (WER) using different language models

4.2.1. Baselines: Trigrams and Semantic class trigrams

A trigram language model was first trained on the general corpus. Table 2 shows that the results for this language model (3-gram) are rather poor, with an average word error rate of 39.9%. The poor performance is probably due to data sparseness in the training data, such as cities, dates, and times used in particular contexts.

We addressed the data sparseness problem by introducing 15 semantic classes into the trigram language model (class 3-gram). Table 2 shows that the word error rate was reduced from an average (over states 2 through 4) of 39.9% to 21.1%. Therefore, using a class-based language model is an effective method to generalize the language model.

The class trigram model is trained with the same corpus as the trigram. The main reasons for state 2 having the worst result are the out-of-vocabulary cities and the high perplexity of members of the semantic classes *<cityname>* and *<citystate>*, as they have more than 600 members. We will use the semantic class-based trigram model as the baseline in further comparisons.

4.2.2. Linear interpolation of *n*-grams

The language models presented above are independent of the dialogue state. In contrast, we next interpolated the state-independent class trigram with a state-dependent class trigram. As in [3], we found that estimating weights chosen to optimize the language model perplexity of a development corpus was worse than simply using $\lambda = 0.5$ in all the states, perhaps due to insufficient development data.

Table 2 shows that the word error rate, averaged over states 2 through 4, was reduced from 21.1% to 20.6%, a relative reduction of 2.4% (adapted 3-gram). The performance of the interpolated model seems to be related to the size of the dialogue-state dependent corpus used for adaptation. For example, states 2 and 3 have more data and better results than state 4.

4.2.3. Combination of state-dependent FSGs and class trigrams

As described in Section 3.2, we can choose between the two recognition results from the FSG or the class trigram by utilizing the phone-based likelihood ratio confidence measure. After several experiments, we found that the best way to use this confidence measure is just to add the scores of all the phones in the sentence and choose the recognition result with the higher score.

The results from the combination of a dialogue-state dependent FSG and the general class trigram language model are shown in Table 2 (adapted FSG). The WER was reduced from 21.1% to 18.5%, representing a relative reduction of 12.3%. To find an upper bound on the potential improvement of combining FSG and n -gram results, we conducted an oracle experiment, which chose between the better of the two hypotheses for each utterance. In this case, the global WER could be reduced to 15.3%, a relative improvement of 27.5%. Therefore there is still much room before reaching this upper bound, and further improvements may be achieved through better confidence measures.

5. FURTHER EXPERIMENTS

One way to improve the baseline state-independent statistical language model is to incorporate long distance constraints. A typical approach is to extract common variable-length word sequences and add them to the vocabulary as new units [13, 14]. This increases the linguistic constraints during decoding and can potentially improve word recognition accuracy. Common sequences are selected from the training data set using a perplexity optimization criterion [15] (a “multigram” approach). Extracted sequences are added to the vocabulary and the language model is re-estimated. An important difference with traditional multigram approaches is that in our case we included both words and semantic classes in the initial vocabulary; we refer to the resulting language model as a *class multigram*. 400 sequences were added in the vocabulary using the perplexity minimization multigram algorithm, the longest sequence containing six words. Surprisingly, the results for the class multigram model were worse than for the baseline class 3-gram model (21.9% vs. 21.1% word error rate for dialogue states 2-4). It is possible that for small training corpora variable-length sequences are not useful. Further work is needed to evaluate if the multigram perplexity optimization criterion is appropriate for *class* multigrams and for small training corpora. Our effort to further smooth the class multigram model by further grouping together word sequences provided marginal improvement over unsmoothed class multigram performance (21.8% vs. 21.9%).

6. CONCLUSION

In this paper, algorithms for training and adapting language models for spoken dialogue systems were proposed. Experiments on the DARPA Communicator travel reservation task showed that adding semantic classes to stochastic n -gram language models or finite state grammars are important for generalization, particularly with very little training data. Language models that are adapted to specific dialogue states can outperform state-independent language models, but must also allow for unexpected utterances by the user in a mixed initiative dialogue system. The traditional method was to combine state dependent and independent n -gram language models through linear interpolation. In this paper, we proposed a novel strategy of combining results from a compact state-dependent FSG and a state-independent n -gram language model using an acous-

tic confidence metric. Experimental results showed that the interpolated n -gram model can provide marginal reduction in error rate for most dialogue states. In contrast, combining results from state-dependent FSGs and the general n -gram language model was shown to consistently outperform the baseline non-adapted model and achieve up to 12% relative error rate reduction.

More work is needed to investigate how to combine n -gram and state-dependent FSGs; utilizing acoustic confidence scores is a promising way for performing language model adaptation. Another area of research is investigating the use of semantic class-based multigrams for language modeling in spoken dialogue systems.

7. REFERENCES

- [1] F. Jelinek, “Self-organized language modeling for speech recognition,” *Readings in Speech Recognition*, A. Waibel and K-F. Lee editors, pp. 450–506, Morgan Kaufmann, San Mateo, Calif., 1990.
- [2] G. Riccardi, A. Potamianos, and S. Narayanan, “Language model adaptation for spoken dialog systems,” in *Proc. ICSLP*, Sydney, Australia, Oct. 1998.
- [3] A. Aaron et al., “Speech recognition for DARPA communicator,” in *Proc. ICASSP*, Salt Lake City, Utah, 2001.
- [4] D. Jurafsky, C. Wooters, J. Segal, A. Stolcke, E. Fosler, G. Tajchman, and N. Morgan, “Using a stochastic context-free grammar as a language model for speech recognition,” in *Proc. ICASSP*, Detroit, MI, 1995.
- [5] K. Hacioglu and W. Ward, “Dialog-context dependent language modeling combining n -grams and stochastic context-free grammars,” in *Proc. ICASSP*, Salt Lake City, Utah, 2001.
- [6] E. Fosler-Lussier and H.-K. J. Kuo, “Using semantic class information for rapid development of language models within ASR dialogue systems,” in *Proc. ICASSP*, Salt Lake City, Utah, 2001.
- [7] P.F. Brown, V.J. DellaPietra, P.V. DeSouza, J.C. Lai, and R.L. Mercer, “Class-based n -gram models of natural language,” *Computational Linguistics*, vol. 18, no. 4, pp. 467–479, 1992.
- [8] K. Sasaki, H. Jiang, and K. Hirose, “Rapid adaptation of n -gram language model using inter-word correlation for speech recognition,” in *Proc. ICSLP*, Beijing, China, 2000.
- [9] P.S. Rao, M.D. Monkowski, and S. Roukos, “Language model adaptation via minimum discrimination information,” in *Proc. ICASSP*, Detroit, Michigan, USA, 1995, pp. 161–164.
- [10] F. Jelinek, “Up from trigrams! the struggle for improved language models,” in *Proc. EUROSPEECH*, 1991, pp. 1037–1040.
- [11] J. Fiscus, “A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER),” in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 1997.
- [12] H. Jiang, F. Soong, and C.-H. Lee, “A data selection strategy for utterance verification in continuous speech recognition,” in *Proc. EUROSPEECH*, Aalborg, Denmark, September 2001.
- [13] I. Zitouni, J.F. Mari, K. Smaili, and J.P. Haton, “Variable-length sequence language model for large vocabulary continuous dictation machine: The n -seqgram approach,” in *Proc. EUROSPEECH*, Budapest, Hungary, 1999, pp. 1811–1814.
- [14] H.-K. J. Kuo and W. Reichl, “Phrase-based language models for speech recognition,” in *Proc. EUROSPEECH*, Budapest, Hungary, 1999, pp. 1595–1598.
- [15] S. Deligne and F. Bimbot, “Language modeling by variable length sequences: Theoretical formulation and evaluation of multigrams,” in *Proc. ICASSP*, Detroit, Michigan USA, 1995, pp. 169–172.