International Conference on Acoustics, Speech, and Signal Processing., pp. I565-I568, Toulouse, France, 2006

N-Gram Based Filler Model for Robust Grammar Authoring

Dong Yu, Yun Cheng Ju, Ye-Yi Wang, Alex Acero

Speech Research Group, Microsoft Research
{dongyu, yuncj, yeyiwang, alexac}@microsoft.com

ABSTRACT

We propose a technique for rapid speech application development that generates robust semantic context-free grammars (CFG) given rigid CFGs as input. Users' speech does not always conform to rigid CFGs, so robust grammars improve the caller's experience. Our system takes a simple CFG and then generates a hybrid n-gram/CFG that is written in the W3C SRGS format and thus can run in many standard automatic speech recognition engines. The hybrid network leverages an application-independent word n-gram which can be shared across different applications. In addition, our tool allows developers to provide a few example sentences to adapt the n-gram for improved accuracy. Our experiments show the robust CFG has no loss in accuracy for test utterances that can be covered by the rigid CFG, but offers large improvements for cases where the user's sentence cannot be covered by the rigid CFG. It also has a much better rejection for utterances that contain no slot at all. With a few example sentences for adaptation, our robust CFG can achieve the recognition accuracy close to the class-based n-gram LM customized for the application.

1. INTRODUCTION

While we have seen great progress in speech recognition accuracy over the last decade, building good telephony speech applications is still expensive because of the long development cycle required to get the application to a level accepted by callers. One of the barriers in developing such applications is the development of grammars that recognize the users' input. In an automated movie ticket line, a developer may use the prompt "Welcome to movie line. How many tickets do you want to buy?", and then build a simple digits CFG. But some users may respond "I want to buy two tickets" rather than simply saying "two", which would not be covered by the grammar and thus lead to higher error rates or increased rejection. Sometimes the problem can be ameliorated by careful choice of words in the prompt to instruct the user to stay within the grammar (i.e. "Please say a number between 1 and 5"), but other times the solution is to build grammars with increased coverage.

Eq. (1) is the typical optimization performed by an automatic speech recognition (ASR) system

$$\hat{w} = \arg\max_{w} p(w \mid A) = \arg\max_{w} p(A \mid w) \cdot p(w) \qquad (1)$$

where A is the acoustics, and w is the word sequence hypothesis. $p(w)$ is usually referred to as the language model (LM) probability and in telephony applications is typically specified by a probabilistic context free grammar (PCFG).

Using the ticket booking application as an example, the grammar shown in Figure 1 is typical for the dialog turn starting with the prompt "How many tickets do you want to buy?"

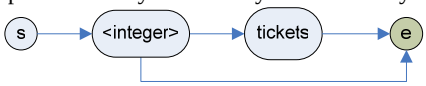

Figure 1. A simple CFG example.

A speech expert who is also familiar with the domain may include additional alternatives in the grammar as shown in Figure 2.

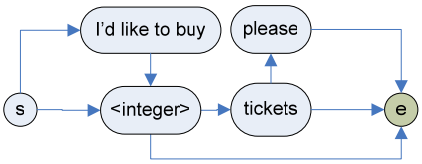

Figure 2. A version of the CFG in Figure 1 with better coverage.

Grammars authored this way are never guaranteed to work well for two reasons. First, the grammar coverage is usually poor since it's impossible for the grammar author to think of all possible callers responses. This is especially true at the early stage of the application development when little real data is available. Second, it's not easy to manually construct the CFG when there are many different ways of asking for the same thing.

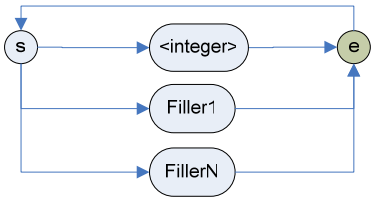

Figure 3. Semantic slot spotting based grammar.

An alternative approach to achieving the same goal is semantic (or keyword) spotting using the grammar structure shown in Fig. 3. where fillers are used to model garbage words. Three categories of filler models (FMs) have been proposed in the literature: acoustic only FM [1-3], acoustic-LM FM [6,7], and LM based FM [4,5]. Acoustic only FMs are multi-state HMMs trained with garbage words. Acoustic-LM filler models use the subword (phoneme or syllable) LM on top of the acoustic fillers to improve the performance. LM based FMs [10] use word n-gram LM to model the words around the keyword. The latter models have been shown to offer better accuracy but require a custom LM trained from domain data. All these existing FMs usually require a likelihood ratio based hypothesis test that is not always supported by commercial recognizers. In this paper, we propose a novel n-gram word LM based FM that can be shared by all applications, and can

achieve the recognition accuracy close to the n-gram LM FMs customized for the application. Our FM does not require hypothesis test built in the decoder, and thus can run in many standard ASR engines.

The rest of the paper is organized as follows. In section 2, we introduce the basic ideas behind our grammar authoring paradigm. Specifically, we describe the grammar architecture and the n-gram based FM. In section 3, we illustrate the algorithm that combines the FM with additional context words provided by developers. We evaluate our grammar authoring paradigm in section 4, and conclude the paper in section 5.

2. HYBRID N-GRAM/CFG AUTHORING

In our basic authoring paradigm, a grammar is simply constructed with pre-ambles, post-ambles, and slots as shown in Figure 4, where pre-ambles and post-ambles are fillers modeled with word n-grams, and *p1* and *p2* are the pre-amble and post-amble bypass probability (or weight) respectively. In this construction, <slot> carries semantic information such as numbers, a list of commands, date, time, currency, and credit card number, etc.

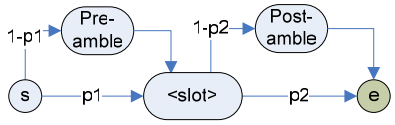

Figure 4. A grammar constructed with pre-ambles, post-ambles, and slots

We use Figure 4 as our basic grammar structure because the semantic spotting problem can be formally described as:

H_0 : <slot> exists

H_a : <slot> does not exist

And the null hypothesis is taken if $p(\text{<slot>} \mid A) > T$ where T is a threshold. Note that

$$p(\text{<slot>} \mid A) = \sum_{w_{pre}, w_{post}} p(w_{pre}\ \text{<slot>}\ w_{post} \mid A) = \frac{\sum_{w_{pre}, w_{post}} p(A \mid w_{pre}\ \text{<slot>}\ w_{post})\, p(w_{pre}\ \text{<slot>}\ w_{post})}{p(A)} \quad (2)$$

where w_{pre} are the pre-amble words (can be empty), w_{post} are the post-amble words (can be empty), and $p(w_{pre}\ \text{<slot>}\ w_{post})$ is the LM probability for an utterance containing <slot>. This posterior probability can be computed from the ASR lattice and that is the approach used in [11]. The approach we chose in this paper considers only the top choice of the recognizer for simplicity.

Our approach does not use an explicit segmental likelihood ratio test although we have knobs (p1 and p2) to balance accuracy and rejections. The ASR engine only needs to pick up the best path based on the overall likelihood score and the transitions are automatically embedded in the path. One special benefit of our approach is the high rejection rate to the out-of-grammar (OOG) utterances since the best path of passing through the whole grammar would have much lower score than the path that loops within the pre-amble node. Note that unlike acoustic FMs, the transitions inside our FM are constrained by the n-gram and so better recognition accuracy can be achieved for the filler part.

Many commercial recognizers allow you to run either an n-gram LM or a CFG but not a hybrid. To solve this problem, we convert the n-gram LM to a determinizable PCFG using the algorithm in [8]. As an example, Figure 5 illustrates a PCFG converted from a two word bi-gram. In this example, the bi-gram $p(w1 \mid w2)$ is not covered by the training data. The probability is smoothed with back-off via the back-off node:

$$p(w1 \mid w2) \approx b(w2)\, p(w1). \quad (3)$$

Similarly, $p(e \mid w1)$ is not in the PCFG and can be estimated as:

$$p(e \mid w1) \approx b(w1)\, p(e). \quad (4)$$

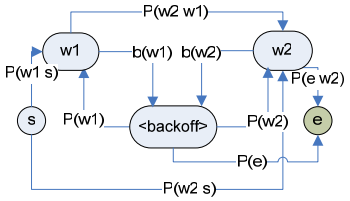

Figure 5. A PCFG converted from a two word bi-gram.

Although the n-gram LM could be trained using in-domain utterances, this is not possible before the application has been built and deployed. So, initially we use a generic n-gram LM built from large amounts of application-independent data. This is a mismatch where we want the n-gram LM to model pre-ambles and post-ambles in real data, while the generic n-gram models complete sentences. Among other problems, this mismatch indicates that the sentence beginning and sentence end probabilities in the generic n-gram are not reliable to predict the segment beginning and segment end in the filler. i.e.,

$$p(\text{<slot>} \mid w) \neq p(\text{<se>} \mid w)$$

$$p(w \mid \text{<slot>}) \neq p(w \mid \text{<ss>})$$

where <ss> and <se> are sentence start and sentence end respectively. To solve this problem, we removed all links from the start node to the internal words and all links from internal words to the end node, updated backoff weights accordingly, and rely on the backoff node to estimate the segment beginning and end probabilities. For example, the PCFG in Figure 5 becomes the PCFG in Figure 6 after the above simplification. The PCFG generated this way can then be used directly as the filler model in constructing grammars shown in Figure 4.

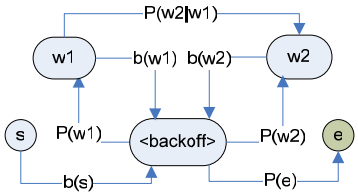

Figure 6. A simplified version of the grammar in Fig 5 after removing links from words to the end node, and from the start node to the words.

3. EXAMPLE-BASED GRAMMAR ADAPTATION

The grammar shown in Figure 4 is usually a good start as a robust grammar using n-gram based FM. To build such a grammar, developers only need to provide a slot grammar (e.g., a name list, cardinal or ordinary number, and date time, etc) and plug it into

the structure shown in Figure 4. The slot grammar can be from a reusable library grammar or created with grammar controls [9].

However, developers' domain knowledge can provide additional context information in the grammar. For example, in the ticket booking case, developers may anticipate that callers are likely to say "tickets" or "please" after the <integer> slot. Integrating these context words into the per-amble and/or post-amble can make the grammar perform more accurately.

We allow developers to provide sample phrases like "…<integer> tickets" and "… buy <integer> tickets". The developer provided phrases, together with the default phrase "… <integer> …", are used to improve the grammar. The eclipses in the phrases are treated as "[filler]" (where [] means it's optional). These phrases are then converted into a training set. For example, the above phrases would give us the following training set: <s> <integer> <e> | <s> <filler> <integer> <e> | <s> <integer> <filler> <e> | <s> <filler> <integer> tickets <e> | <s> <integer> tickets <e> | <s> buy <integer> tickets <e> | <s> <filler> buy <integer> tickets <e>. We used a standard n-gram training algorithm to build the grammar shown in Figure 7.

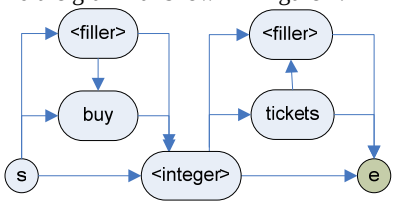

Figure 7. PCFG generated using filler models and additional context cues.

Note that the samples provided by developers are usually not enough to accurately estimate the probabilities in Figure 7. We have noticed, however, that recognition accuracy is not sensitive to such probability estimation errors, and the customized context significantly improves accuracy.

4. EXPERIMENTAL RESULTS

We have conducted a series of experiments to evaluate the effectiveness of the grammars authored using the approaches described above. Our test cases are from SALA II dataset (cellular telephony) and ATIS dataset and are separated into four sets:

- SO (slot only): the utterances only contain the slot part. We have 500 command utterances (SOC) and 500 time utterances (SOT) in this category.
- PP (pre-amble and post-amble): the utterances contain pre-ambles and/or post-ambles around slots. We have 25 command utterances (PPC) and 500 time utterances (PPT) in this category. The preambles in this category are simple words such as "at", "about", and "approximately".
- OOG (out of grammar): the utterances are completely off the topic. We have 500 utterances in this category.
- TS (two slots): the utterances contain two slots plus pre-ambles and post-ambles. TS test cases are from ATIS dataset. We have 669 utterances in this category.

The command slot consists of 24 different commands, many of them are acoustically confusing, such as "start" and "restart", "call" and "all". The time slot allows users to say a time in their favorite ways.

The grammars used in the evaluation are also separated into several categories:

- SO (slot only): the grammar contains only the slot part.
- AFP & AFS (acoustic FM): the grammar uses context independent phone loop based acoustic FM. AFP is the parallel version as shown in Fig. 3; AFS is the sequential version as shown in Fig. 4..
- NFP & NFS (n-gram FM): the grammar uses n-gram FM. NFP is the parallel version as shown in Fig. 3; NFS is the sequential version as shown in Fig. 4 respectively.

When evaluating the performance, we use the slot accuracy (SA) for the SO, PP, and TS test sets, and rejection rate (RR) for the OOG test set, since word error rate (WER) is not appropriate for the slot recognition tasks. SER and RR are defined as:

$$SA = \frac{\#\text{ of correct slots}}{\text{total }\#\text{ of slots}}, \quad (5)$$

$$RR = \frac{\#\text{ of rejected utterances with default threshold}}{\text{total }\#\text{ of utterances}}. \quad (6)$$

	SO	AFP	AFS	NFP	NFS
SOT	78.2%	76.6%	77.2%	78.0%	79.8%
SOC	92.0%	92.2%	91.2%	91.6%	92.0%

Table 1. Slot accuracy for utterances that only contain slots for time (SOT) and commands (SOC) evaluated with slot-only grammars (SO), acoustic fillers (AFS and AFP) and n-gram fillers (NFS, NFP).

	SO	AFP	AFS	NFP	NFS
PPT	38.8%	36.0%	44.2%	50.0%	60.2%
PPC	52.0%	60.0%	56.0%	60.0%	68.0%

Table 2. Slot accuracy for utterances only contain slots with pre/postambles for time (PPT) and commands (PPC) evaluated with slot-only grammars (SO), acoustic fillers (AFS and AFP) and n-gram fillers (NFS, NFP).

	SO	AFP	AFS	NFP	NFS
OOGT	11.2%	54.0%	16.8%	96.6%	91.2%
OOGC	0.4%	1.0%	0.6%	56.6%	26.2%

Table 3. Rejection rate for out-of-grammar (OOG) utterances in the time and command scenarios evaluated with slot-only grammars (SO), acoustic fillers (AFS and AFP) and n-gram fillers (NFS, NFP).

4.1. Performance Evaluation

Tables 1, 2, 3 and Fig. 8 compare the performance of different grammars using different test sets. In Table 1 we see that NFP and NFS grammars perform as well as SO grammars for SO test sets, and even perform slightly better due to their ability to absorb the noises before and after the utterances. In Table 2 we see that for the test sets with pre or postambles, the NFS grammar consistently performs significantly better than other grammars, with relative SA improvement of 58% for the PP-time set, and 31% for the PP-

command set comparing to the SO grammar. In Table 3 we see that NF grammars reject many more OOG utterances than the SO and AF grammars. Note that, NFP grammar has higher rejection rate on the OOG utterances than the NFS grammar. However, since the number of OOG utterances in a typical application is much less than the PP utterances, NFS grammar performs better overall.

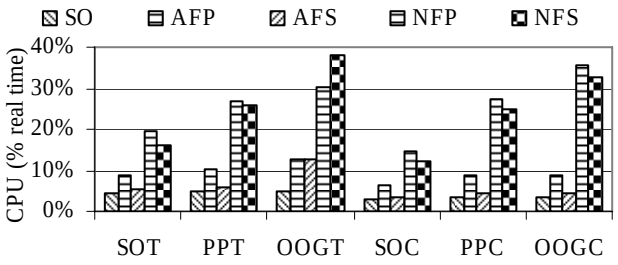

Figure 8. Comparison of different grammars in CPU time.

Fig. 8 indicates that the gain of NF grammars come with costs: the CPU time is about 3-5 times as what needed for the SO or AF grammars. We have also evaluated the sensitivity of the accuracy and CPU cost to the bypass probability p1 (from 0.5 to 0.999) used in Fig. 4, and the bi-gram size (from 0K to 280K). Our experiments show that the accuracy is not sensitive to either the bypass probability or the n-gram size, although increasing the bi-gram size usually boosts the accuracy a little bit. However, more CPU time is needed when the bypass probability becomes smaller or the n-gram size becomes larger. The rejection rate also increases when the n-gram size becomes larger for the OOG test sets.

4.2. N-gram Filler Adaptation

Table 4 compares the performance of the grammars with and without small adaptation. The test cases used in this experiment are from the TS set where the callers can say things containing two slots like "I'd like to book a ticket from Seattle to Boston". An utterance is considered correctly recognized if both slots (cities) are correctly recognized. Two training phrases (with context words) "… flight from <fromCity> to <toCity>" and "… between <fromCity> and <toCity>" are used to generate the adapted NFS grammar, and the domain n-gram filler is trained using a large amount of ATIS training data. Table 4 shows that the NFS grammar has significantly higher slot accuracy than the AF grammar and adding context words in the grammar can further improve the accuracy by 5.23% absolute or reduce the slot error rate by 56%. This accuracy is close to what we can get using the grammar trained with the domain data.

Filler type	AF	NFS	Adapted NFS	Domain n-gram
SA	2.4%	90.7%	96.0%	96.0%

Table 4. Slot accuracy for the acoustic filler and n-gram fillers without and with adaptation. The slots were cities in the ATIS data. Adapting the n-gram filler with just two sentences cuts the error rate by a factor of 2.

4.3. Discussion

When compared to other filler models, our approach has six advantages. First, our FM can be directly integrated into a PCFG and thus can be applied to any PCFG based ASR engine (many commercial engines support W3C's SRGS for specifying CFGs). Second, the transitions between FMs and the semantic slots in our approach are automatically determined by the ASR's search algorithm (i.e. there is no need to use segmental likelihood ratio based hypothesis test in the decoding process). Third, our FM is more robust and accurate than other FMs due to our novel grammar structure. Fourth, our FM provides higher rejection rate for out-of-grammar (OOG) utterances than other FMs. Fifth, our FM can be shared across all applications and dialog turns and so can be efficient memory wise. Sixth, our FM can benefit from example sentences provided by developers to further improve accuracy.

In addition, since the ASR engine outputs actual words (instead of just a generic symbol such as <garbage> in the acoustic FM case), systems that rely on natural language input, such as a call-routing system, can rely on the text to make decision.

5. SUMMARY

In this paper, we proposed a robust grammar authoring paradigm, in which n-gram based FM is used to model the garbage words between slots. We discussed issues related to making the FM sharable across the applications, and showed our approach to simplifying the FM, and to integrating it with example sentences provided by developers. We demonstrated that the grammars constructed with the proposed paradigm are superior in many aspects. Preliminary experiments confirmed that the paradigm has great potential, especially in rejecting OOG utterances and recognizing utterances with pre-ambles or post-ambles.

At the current stage, our word n-gram is trained using written text mainly from Wall Street Journal. We perceive that the performance of our model can be further improved if we can train the n-gram using real data collected from a variety of spoken dialog applications.

6. REFERENCES

[1] A.S. Manos, V. W. Zue, "A segment-based wordspotter using phonetic filler models", in Proc. ICASSP-1997, pp. 899-902.

[2] R. Rose, and D. Paul, "A hidden Markov model based keyword recognition system", in Proc. ICASSP-1990, pp129-132.

[3] J. Wilpon, L. Rabiner, and C.-H., Lee, "Automatic recognition of keywords in unconstrained speech using hidden Markov models", IEEE Trans. ASSP 38, pp. 1870-1990.

[4] Q. Lin, D. Lubensky, M. Picheny, and P. S. Rao, "Key-phrase spotting using an integrated language model of n-grams and finite-state grammar", In Proc. of Eurospeech -1997, pp. 255-258.

[5] M. E. Hennecke, "and G. Hanrieder, "Easy Configuration of Natural Language Understanding Systems", in Proc. VOTS, COST 249, 2000.

[6] R. Meliani, and D. O'Shaugnessy, "Accurate keyword spotting using strictly lexical fillers", in Proc. of ICASSP-1997, pp. 907-910.

[7] R. C. Rose, "Keyword Detection in Conversational Speech Utterances Using Hidden Markov Model Based Continuous Speech Recognition", J. CSL, vol. 9 (1995), pp. 309-333.

[8] G. Riccardi, R. Pieraccini, and E. Bocchieri, "Stochastic Automata for Language Modeling", J. CSL, vol. 10(4), pp. 265-293, 1996.

[9] Y. Wang and A. Acero, "SGStudio: Rapid Semantic Grammar Development for Spoken Language Understanding." In Proc. Of Eurospeech 2005. September 2005.

[10] M. Weintraub, "Keyword-spotting using SRI'S DECIPHER large-vocabulary speech recognition system,''in *Proc. ICASSP-1993*, pp. 463-466, 1993.

[11] C. Chelba and A. Acero. "SPEECH OGLE: Indexing Uncertainty for Spoken Document Search", in Proc. of ACL. Ann Arbor, June, 2005.