Rapidly Deploying Grammar-Based Speech Applications with Active Learning and Back-off Grammars

Tim Paek[1], Sudeep Gandhe[2], David Maxwel Chickering[1]
[1] Microsoft Research, One Microsoft Way, Redmond, WA 98052
[2] USC Institute for Creative Technologies, 13274 Fiji Way, Marina del Rey, CA 90292
{timpaek|dmax}@microsoft.com, gandhe@usc.edu

Abstract

Grammar-based approaches to spoken language understanding are utilized to a great extent in industry, particularly when developers are confronted with data sparsity. In order to ensure wide grammar coverage, developers typically modify their grammars in an iterative process of deploying the application, collecting and transcribing user utterances, and adjusting the grammar. In this paper, we explore enhancing this iterative process by leveraging active learning with back-off grammars. Because the back-off grammars expand coverage of user utterances, developers have a safety net for deploying applications earlier. Furthermore, the statistics related to the back-off can be used for active learning, thus reducing the effort and cost of data transcription. In experiments conducted on a commercially deployed application, the approach achieved levels of semantic accuracy comparable to transcribing all failed utterances with 87% less transcriptions.

1 Introduction

Although research in spoken language understanding is typically pursued from a statistical perspective, grammar-based approaches are utilized to a great extent in industry (Knight et al., 2001). Speech recognition grammars are often manually authored and iteratively modified as follows: Typically, context-free grammars (CFG) are written in a format such as Speech Recognition Grammar Specification (SRGS) (W3C, 2004) and deployed. Once user utterances are collected and transcribed, the grammars are then adjusted to improve their coverage. This process continues until minimal OOG utterances are observed. In this paper, we explore enhancing this iterative process of grammar modification by combining back-off grammars, which expand coverage of user utterances, with active learning, which reduces "the number of training examples to be labeled by automatically processing unlabeled examples, and then selecting the most informative ones with respect to a specified cost function for a human to label" (Hakkani-Tur et al., 2002). This paper comprises three sections. In Section 2, we describe our overall approach to rapid application development (RAD). In Section 3, we explain how data transcription can be reduced by leveraging active learning based on statistics related to the usage of back-off grammars. Finally, in Section 4, we evaluate the active learning approach with simulation experiments conducted on data collected from a commercial grammar-based speech application.

2 RAD Approach & Related Work

Working under the assumption that developers in industry will continue to use CFGs for rapid application development, our approach to grammar modification is as follows:

1. Create a CFG (either manually or automatically).
   1.1 Generate a back-off grammar from the CFG.
2. Deploy the application.
   2.1 Use the back-off grammar for OOG utterances.
3. Gather data from users.
4. Selectively transcribe data by using statistics related to the back-off for active learning; i.e., transcribe only those utterances that satisfy the active learning criterion.
5. Modify CFG either manually or automatically and go to step 1.1.

To begin with, developers start with a CFG in Step 1. If they had access to a grammatical platform

[2] Second author was partly sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). Statements and opinions expressed do not necessarily reflect the position or the policy of the U.S. Government, and no official endorsement should be inferred.

Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue, pages 64–67,
Columbus, June 2008.

such as Regulus (Rayner et al., 2006), they could in principle construct a CFG automatically for any new domain, though most developers will probably manually author the grammar. Two steps are added to the typical iterative process. In Step 1.1, we generate a back-off grammar from the CFG. One way to accomplish this is by constructing a back-off CFG using filler models (Paek et al., 2007), which when applied to the same command-and-control task in Section 4 can result in a 35% relative reduction in semantic error rate for OOG utterances. However, the back-off grammar could also be a SLM trained on artificial data created from the CFG (Galescu et al., 1998). Whatever back-off mechanism is employed, its coverage should be wider than the original CFG so that utterances that fail to be recognized by the CFG, or fall below an acceptable confidence threshold, can be handled by the back-off in a second or simultaneous pass. That is the gist of Step 2.1, the second additional step. It is not only important to generate a back-off grammar, but it must be utilized for handling possible OOG utterances.

Our approach attempts to reduce the usual cost associated with grammar modification after the application has been deployed and data collected in Step 4. The idea is simple: Exploit the fast and accurate CFG recognition of in-grammar (ING) utterances by making OOG utterances handled by the back-off grammar ING. In other words, expand CFG coverage to include whatever gets handled by the back-off grammar. This idea is very complementary with a two-pass recognition approach where the goal is to get utterances correctly recognized by a CFG on the first pass so as to minimize computational expenses (Paek et al., 2007).

All of this can be accomplished with reduced transcription effort by keeping track of and leveraging back-off statistics for active learning. If the back-off is a CFG, we keep track of statistics related to which CFG rules were utilized the most, whether they allowed the task to be successfully completed, etc. If the back-off is a SLM, we keep track of similar statistics related to the semantic alignment and mapping in spoken language understanding. Given an active learning criterion, these statistics can be used to selectively transcribe utterances which can then be used to modify the CFG in Step 5 so that OOG utterances become ING. Section 3 covers this in more detail.

Finally, in Step 5, the CFG grammar is modified using the selectively transcribed utterances. Although developers will probably want to do this manually, it is possible to automate much of this step by making grammar changes with minimal edit distance or Levenshtein distance.

Leveraging a wider coverage back-off grammar is of course not new. For grammar-based applications, several researchers have investigated using a CFG along with a back-off grammar either simultaneously via a domain-trained SLM (Gorrell et a1., 2002), or in two-pass recognition using either an SLM trained on CFG data (Gorrell, 2003) or a dictation n-gram (Dusan & Flanagan, 2002). To our knowledge however, no prior research has considered leveraging statistics related to the back-off grammar for active learning, especially as part of a RAD approach.

3 Active Learning

Our overall approach utilizes back-off grammars to provide developers with a safety net for deploying applications earlier, and active learning to reduce transcription effort and cost. We now elaborate on active learning, demonstrate the concept with respect to a CFG back-off.

Active learning aims at reducing transcription of training examples by selecting utterances that are most likely to be informative according to a specified cost function (Hakkani-Tur et al., 2002). In the speech community, active learning has been successfully applied to reducing the transcription effort for ASR (Hakkani-Tur et al., 2002), SLU (Tur et al., 2003b), as well as finding labeling errors (Tur et al., 2003). In our case, the examples are user utterances that need to be transcribed, and the learning involves modifying a CFG to achieve wider coverage of user expressions. Instead of passively transcribing everything and modifying the CFG as such, the grammar can "actively" participate in which utterances are transcribed.

The usual procedure for selecting utterances for grammar modification is to transcribe at least all failed utterances, such as those that fall below a rejection threshold. By leveraging a back-off grammar, developers have more information with which to select utterances for transcription. For a CFG back-off, how frequently a back-off rule fired can serve as an active learning criterion because that is where OOG utterances are handled. Given this active learning criterion, the algorithm would proceed as follows (where i denotes iteration, S_t denotes the set of transcribed utterances, and S_u denotes the set of all utterances):

[1] Modify CFG_i using S_t and generate corresponding back-off_i from the CFG_i.
[2] Recognize utterances in set S_u using CFG_i + back-off_i.
[3] Compute statistics on what back-off rules fired when and how frequently.
[4] Select the k utterances that were handled by the most frequently occurring back-off rule and transcribe them. Call the new transcribed set as S_i.
[5] $S_t = S_t \cup S_i ; S_u = S_u - S_i$
[6] Stop when CFG_i achieves a desired level of semantic accuracy, or alternatively when back-off rules only handle a desired percentage of S_u, otherwise go to Step 1.

Note that the set S_u grows with each iteration and follows as a result of deploying an application with a CFG_i + back-off_i. Step [1] corresponds to Step 5, 1.1, and 2.1 of our approach. Steps [2-4] above constitute the active learning criterion and can be adjusted depending on what developers want to optimize. This algorithm currently assumes that runtime efficiency is the main objective (e.g., on a mobile device); hence, it is critical to move utterances recognized in the second pass to the first pass. If developers are more interested in learning new semantics, in Step [4] above they could transcribe utterances that failed in the back-off. With an active learning criterion in place, Step [6] provides a stopping criterion. This too can be adjusted, and may even target budgetary objectives.

4 Evaluation

For evaluation, we used utterances collected from 204 users of Microsoft *Voice Command*, a grammar-based command-and-control (C&C) application for high-end mobile devices (see Paek et al., 2007 for details). We partitioned 5061 transcribed utterances into five sets, one of which was used exclusively for testing. The remaining four were used for iterative CFG modification. For the first iteration, we started with a CFG which was a degraded version of the grammar currently shipped with the *Voice Command* product. It was obtained by using the mode, or the most frequent user utterance, for each CFG rule. We compared two approaches: *CFG_Full*, where each iterative CFG was modified using the full set of transcribed utterances that resulted in a *failure state* (i.e., when a false recognition event occurred or the phrase confidence score fell below 45%, which was set by a proprietary tuning procedure for optimizing word-error rate), and *CFG_Active*, where each iterative CFG was modified using only those transcribed utterances corresponding to the most frequently occurring CFG back-off rules. For both *CFG_Full* and *CFG_Active*, CFG_i was modified using the same set of heuristics akin to minimal edit distance. In order to assess the value of using the back-off grammar as a safety net, we also compared *CFG_Full+Back-off*, where a derived CFG back-off was utilized whenever a failure state occurred with *CFG_Full*, and *CFG_Active+Back-off*, where again a CFG back-off was utilized, this time with the back-off derived from the CFG trained on selective utterances.

As our metric, we evaluated semantic accuracy since that is what matters most in C&C settings. Furthermore, because recognition of part of an utterance can increase the odds of ultimately achieving task completion (Paek et al., 2007), we carried out separate evaluations for the functional constituents of a C&C utterance (i.e., keyword and slot) as well as the complete phrase (keyword + slot). We computed accuracy as follows: For any single utterance, the recognizer can either accept or reject it. If it is accepted, then the semantics of the utterance can either be correct (i.e., it matches what the user intended) or incorrect, hence:

$$accuracy = CA / (CA + IA + R) \quad (1)$$

where CA denotes accepted commands that are correct, IA denotes accepted commands that are incorrect, and R denotes the number of rejections.

Table 2 displays semantic accuracies for both *CFG_Full* and *CFG_Active*. Standard errors about the mean were computed using the jacknife procedure with 10 re-samples. Notice that both *CFG_Full* and *CFG_Active* initially have the same accuracy levels because they start off with the same degraded CFG. The highest accuracies obtained almost always occurred in the second iteration after modifying the CFG with the first batch of transcriptions. Thereafter, all accuracies seem to decrease. In order to understand why this would be case, we computed the coverage of the i^{th} CFG on the holdout set. This is reported in the 'OOG%' column. Comparing *CFG_Full* to *CFG_Active* on

Approach	*i*	Utterances Transcribed	Keyword Accuracy	Slot Accuracy	Keyword + Slot Accuracy	Processing Time (ms)	OOG%
CFG_Full	1	0	50.25% (0.13%)	46.84% (0.22%)	46.84% (0.22%)	387 (3.9005)	61.10%
	2	590	**66.20% (0.12%)**	**71.02% (0.23%)**	**70.59% (0.23%)**	401 (4.0586)	31.92%
	3	1000	65.80% (0.15%)	69.72% (0.19%)	69.06% (0.19%)	422 (4.5804)	31.30%
	4	1393	66.10% (0.13%)	67.54% (0.22%)	66.88% (0.21%)	433 (4.7061)	30.95%
CFG_Full + Back-off	1	0	66.70% (0.10%)	66.23% (0.22%)	66.01% (0.22%)	631 (11.1320)	61.10%
	2	590	**73.32% (0.11%)**	**72.11% (0.22%)**	**71.68% (0.23%)**	562 (10.4696)	31.92%
	3	1000	72.52% (0.12%)	72.11% (0.21%)	71.46% (0.22%)	584 (10.4985)	31.30%
	4	1393	73.02% (0.10%)	71.02% (0.23%)	70.37% (0.23%)	592 (10.6805)	30.95%
CFG_Active	1	0	50.25% (0.13%)	46.84% (0.22%)	46.84% (0.22%)	387 (3.9005)	61.10%
	2	87	64.09% (0.13%)	**74.29% (0.21%)**	**74.07% (0.22%)**	395 (4.1469)	42.09%
	3	138	**64.29% (0.15%)**	70.15% (0.22%)	69.50% (0.24%)	409 (4.3375)	38.02%
	4	193	64.09% (0.15%)	69.72% (0.23%)	69.06% (0.24%)	413 (4.4015)	37.93%
CFG_Active + Back-off	1	0	66.70% (0.10%)	66.23% (0.22%)	66.01% (0.22%)	631 (11.1320)	61.10%
	2	87	**72.52% (0.10%)**	**76.91% (0.19%)**	**76.47% (0.21%)**	568 (10.3494)	42.09%
	3	138	71.72% (0.14%)	71.90% (0.24%)	71.24% (0.27%)	581 (10.6330)	38.02%
	4	193	71.21% (0.15%)	71.90% (0.25%)	71.24% (0.26%)	580 (10.5266)	37.93%

Table 2. Semantic accuracies for partial (keyword or slot) and full phrase recognitions (keyword + slot) using a CFG trained on either "Full" or "Active" transcriptions (i.e., selective transcriptions based on active learning). Parentheses indicate standard error about the mean. The 'i' column represents iteration. The 'Utterances Transcribed' column is cumulative. The 'OOG%' column represents coverage of the *ith* CFG on the hold-out set. Rows containing "Back-off" evaluate 2-pass recognition using both the CFG and a derived CFG back-off.

keyword + slot accuracy, *CFG_Full* decreases in accuracy after the second iteration as does *CFG_Active*. However, the OOG% of *CFG_Full* is much lower than *CFG_Active*. In fact, it seems to level off after the second iteration, suggesting that perhaps the decrease in accuracies reflects the increase in grammar perplexity; that is, as the grammar covers more of the utterances, it has more hypotheses to consider, and as a result, performs slightly worse. Interestingly, after the last iteration, *CFG_Active* for keyword + slot and slot accuracies was slightly higher (69.06%) than *CFG_Full* (66.88%) (p = .05). Furthermore, this was done with 193 utterances as opposed to 1393, or 87% less transcriptions. For keyword accuracy, *CFG_Active* (64.09%) was slightly worse than *CFG_Full* (66.10%) (p < .05).

With respect to the value of having a back-off grammar as a safety net, we found that both *CFG_Full* and *CFG_Active* achieved much higher accuracies with the back-off for keyword, slot, and keyword + slot accuracies. Notice also that the differences between *CFG_Full* and *CFG_Active* after the last iteration were much closer to each other than without the back-off, suggesting applications should always be deployed with a back-off.

5 Conclusion

In this paper, we explored enhancing the usual iterative process of grammar modification by leveraging active learning with back-off grammars. Because the back-off grammars expand coverage of user utterances to handle OOG occurrences, developers have a safety net for deploying applications earlier. Furthermore, because statistics related to the back-off can be used for active learning, developers can reduce the effort and cost of data transcription. In our simulation experiments, leveraging active learning achieved levels of semantic accuracy comparable to transcribing all failed utterances with 87% less transcriptions.

References

S. Dusan & J. Flanagan. 2002. Adaptive dialog based upon multimodal language acquisition. In *Proc. of ICMI.*

L. Galescu, E. Ringger, & J. Allen. 1998. Rapid language model development for new task domains. In *Proc. of LREC.*

G. Gorrell, I. Lewin, & M. Rayner. 2002. Adding intelligent help to mixed initiative spoken dialogue systems. In *Proc. of ICSLP*.

G.. Gorrell. 2003. Using statistical language modeling to identify new vocabulary in a grammar-based speech recognition system. In *Proc. of Eurospeech*.

D. Hakkani-Tur, G. Riccardi & A. Gorin. 2002. Active learning for automatic speech recognition. In *Proc. of ICASSP*.

S. Knight, G. Gorrell, M. Rayner, D. Milward, R. Koel-ing, & I. Lewin. 2001. Comparing grammar-based and robust approaches to speech understanding: A case study. In *Proc. of Eurospeech*.

T. Paek, S. Gandhe, D. Chickering & Y. Ju. 2007. Handling out-of-grammar commands in mobile speech interaction using back-off filler models. In *Proc. of ACL Workshop on Grammar-Based Approaches to Spoken Language Processing (SPEECHGRAM)*.

M. Rayner, B.A. Hockey, & P. Bouillon. 2006. *Putting Linguistics into Speech Recognition: The Regulus Grammar Compiler*. CSLI.

G. Tur, M. Rahim & D. Hakkani-Tur. 2003. Active labeling for spoken language understanding. In *Proc. of Eurospeech*.

G. Tur, R. Schapire, & D. Hakkani-Tur. 2003b. Active learning for spoken language understanding. In *Proc. of ICASSP*.

W3C. 2004. Speech Recognition Grammar Specification Version 1.0. http://www.w3.org/TR/speech-grammar