Grammar Fragment Acquisition using Syntactic and Semantic Clustering

Kazuhiro Arai†, *Jeremy H. Wright*‡, *Giuseppe Riccardi*‡, *and Allen L. Gorin*‡

† NTT Human Interface Laboratories
1-1 Hikari-no-oka, Yokosuka, Kanagawa 239-0847, Japan
arai@nttspch.hil.ntt.co.jp
‡ AT&T Laboratories-Research
180 Park Ave., Florham Park, New Jersey 07932, USA
{jwright,dsp3,algor}@research.att.com

ABSTRACT

A new method is proposed for automatically acquiring Fragments to understand fluent speech. The goal of this method is to generate a collection of Fragments, each representing a set of syntactically and semantically similar phrases. First, phrases frequently observed in the training set are selected as candidates. Each candidate phrase has three associated probability distributions of : following contexts, preceding contexts, and associated semantic actions. The similarity between candidate phrases is measured by applying the Kullback-Leibler distance to these three probability distributions. Candidate phrases that are close in all three distances are clustered into a Fragment. Salient sequences of these Fragments are then automatically acquired, and exploited by a spoken language understanding module to classify calls in AT&T's "How May I Help You?" task. The experimental results show that the average and maximum improvements in call-type classification performance of 2.2% and 2.8% are respectively achieved by introducing the Fragments.

1 Introduction

We are interested in providing an automated *call-routing* service via natural spoken dialog systems in a telecommunications environment [1] [2] [3]. In the task of call-routing, services that a user can access are categorized into 14 types and *other* as a complement [1]. By accepting spontaneous speech as input, the spoken dialog system determines which call-type is required by a caller. Once the call-type is determined, the information needed for completing each service is requested using another dialog.

It is notable that the dialog system can classify the call-type adequately if the system can extract phrases strongly associated with particular call-types. For instance, the word sequence *"my credit card"* in the user's utterance is strongly associated with the call-type *CALLING_CARD*. The algorithm for call-type classification is designed to capitalize on the salient association between the phrases in the user's utterance and call-types [1].

This paper describes a new stochastic language model used in the speech understanding process for determining the call-type. Conventional word clustering approaches for generating a stochastic language model focus on subsequent contexts only to minimize the branching factor or the test set perplexity. Since analogous phrases have a similar distribution in not only the following word sequence but also in the preceding word sequence, the similarity of word sequences can be clustered more effectively by referring to both the following and preceding word sequences. Furthermore, an utterance accepted as input of the call-routing dialog system can usually be classified semantically into one of 14 call-types. Some phrases performing an analogous role in this task must have a similar association with the call-type. Thus word sequence similarity can also be computed by using such associations between phrases and call-types [3].

2 Fragment Distance

2.1 Syntactic and Semantic Associations of a Fragment

In this paper an arbitrary word sequence in the training transcriptions is called a *phrase*. Phrases having a higher frequency than a specified threshold are selected as *candidates*. Each *Fragment* is acquired via clustering of candidate phrases based on their similarity. *Fragment grammar* is generated by the Fragments and can be represented by a conventional finite-state machine for speech understanding. A *syntactic association* signifies the relationship between a Fragment and phrases following or preceding the Fragment. A *semantic association* focuses on the association between a Fragment in spoken language and the call-type corresponding to the speech. An example of the syntactic and semantic associations of a Fragment is shown in Figure 1 where f denotes a Fragment, s and c define a preceding or following phrase and call-type, respectively. In Figure 1, f comprises only one phrase, *"calling card"*. Given

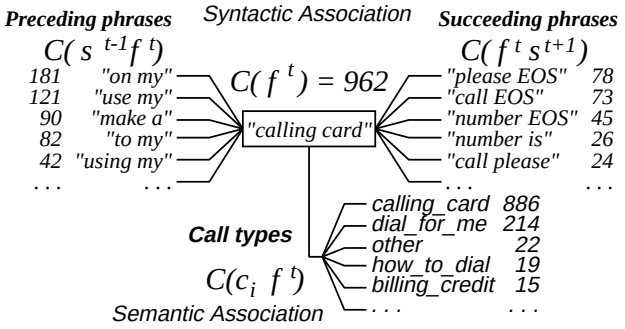

Figure 1: Syntactic and Semantic Associations of a Fragment.

a phrase, Fragment, call-type or combination thereof, function $C(\ )$ counts the frequency in the training transcriptions. The phrase *"on my"*, for instance, precedes this Fragment 181 times. *EOS* used in the following phrases denotes *End-Of-Sentence*. In semantic association, the *CALLING_CARD* call-type, for instance, is associated with this Fragment 886 times.

In order to generate syntactic probability distributions, a set of phrases that precedes or follows Fragments is generated. In the following discussion, a phrase that precedes or follows a Fragment is called its *context*. Three probability distributions for each Fragment are obtained by using the preceding and following *context* frequencies, and the call-type frequency. The bigram probability distributions focusing on the following and preceding *contexts* are denoted in Equations (1) and (2), respectively.

$$p(s_i^{t+1}|f_j^t) = \frac{C(\ f_j^t s_i^{t+1}\ )}{C(\ f_j\ )} = \frac{C(\ f_j^t\ w_1^{t+1} w_2^{t+2} \cdots w_{N_c}^{t+N_c}\ )}{C(\ f_j\ )} \quad (1)$$

$$p(s_i^{t-1}|f_j^t) = \frac{C(\ s_i^{t-1} f_j^t\ )}{C(\ f_j\ )} = \frac{C(\ w_1^{t-N_c} \cdots w_{N_c-1}^{t-2} w_{N_c}^{t-1}\ f_j^t\ )}{C(\ f_i\ )} \quad (2)$$

where s_i denotes the i-th *context*, f_j is the j-th Fragment in the Fragment grammar, w_k denotes the k-th word in *context* s_i, and N_c is the number of items referred to as the *context*. The probability distribution focusing on semantic associations is obtained from the call-type frequencies. Equation (3) shows the probability distribution based on call-type frequencies where c_i denotes one of the call-types and $C(\ c_i\ f_j\ )$ is the frequency of call-type c_i associated with phrase f_j.

$$p(c_i|f_j) = \frac{C(\ c_i\ f_j\ )}{C(\ f_j\ )} \quad (3)$$

2.2 Kullback-Leibler Distance

The Kullback-Leibler distance is one of the most popular expressions of distance for measuring the similarity between two probability distributions. Back-off smoothing is applied in advance to each probability distribution by using a unigram probability distribution of the *context* and the call-type. Equation (4) shows the definition of the Kullback-Leibler distance between Fragments f_1 and f_2 exploiting the following *context* probability distributions.

$$d_s(f_1\ f_2) = \sum_{\forall s_i \in S} \hat{p}(s_i^{t+1}|f_1^t) \cdot \log \frac{\hat{p}(s_i^{t+1}|f_1^t)}{\hat{p}(s_i^{t+1}|f_2^t)} \quad (4)$$

where S comprises all *context* phrases of length N_c together with their frequencies. Term s_i is one of the *contexts* stored in S, and $\hat{p}(s_i^{t+1}|f_1^t)$ and $\hat{p}(s_i^{t+1}|f_2^t)$ are the smoothed probability distributions for Fragments f_1 and f_2, respectively. Distances based on the preceding *context* and the call-type probability distributions are measured in the same manner. In general, the Kullback-Leibler distance is an asymmetric measure. We therefore symmetrize the Kullback-Leibler measure by defining each type of distance as the average of two distances measured from both Fragments. Thus, the Fragment distance shown in Equation (5) based on the following *context* is used in Fragment clustering. The symmetrized distances based on the preceding *context* and the call-type are defined in the same manner.

$$D_s(f_1\ f_2) = \frac{d_s(f_1\ f_2) + d_s(f_2\ f_1)}{2} \quad (5)$$

3 Grammar Fragment Clustering

In Fragment clustering, all phrases are first generated as candidates from the training transcriptions. Then each candidate phrase forms a Fragment as the initial set of a Fragment. The frequency of each Fragment is obtained by summing candidate phrase frequencies. Fragment f_0 having the highest frequency and comprising one phrase is selected as the *reference Fragment*. All Fragments are sorted in the order of Fragment distances measured from f_0. The Fragment distance lists are sorted independently based on preceding *contexts*, following *contexts*, and call-types. Thus three Fragment lists in order of distance are obtained as the result of the sorting. In each Fragment list, the Fragment subset for clustering is determined based on the maximum difference in distance between successive Fragments in that list. For instance, in the Fragment list based on the distance of following *contexts*, the number of candidate Fragments, $N_s(f_0)$, is determined by:

$$N_s(f_0) = \underset{1 \le i \le N_m}{\text{argmax}} \{\ D_s(f_0\ f_{i+1}) - D_s(f_0\ f_i)\ \} \quad (6)$$

where, f_i and f_{i+1} are rank ordered Fragments with respect to the distance of the following *context*. Terms $D_s(f_0\ f_{i+1})$ and $D_s(f_0\ f_i)$ are the distances from reference Fragment f_0 to Fragment f_{i+1} and f_i, respectively. N_m is the maximum number of Fragments to be compared. The number of candidate Fragments based on the distance focusing on preceding *contexts* $N_p(f_0)$ and call-types $N_c(f_0)$ can also be determined by using distance $D_p(f_0\ f_i)$ and $D_c(f_0\ f_i)$. Following these determinations, the maximum number of candidates among the three types of distance is determined by

$$\mathcal{N}(f_0) = \max\{\ N_p(f_0),\ N_s(f_0),\ N_c(f_0)\ \} \quad (7)$$

$< 000 >$	*"hi", "yes", "yes ma'am", "hi operator" "yeah", "yes operator", "yes hi" "yeah please", "yes good morning"*
$< 001 >$	*"a", "a a"*
$< 002 >$	*"make", "place"*
$< 003 >$	*"operator I'd like", "I want", "I like" "I would like", "I'd like"*
$< 004 >$	*"make this", "place a", "make a"*
$< 005 >$	*"have", "need", "want", "would like"*

Table 1: Example of Grammar Fragments

All Fragments, whose ranked order in each Fragment list is less than $\mathcal{N}(f_0)$, are selected as a candidate of similar Fragments. Equation (8) shows the criterion of Fragment classification based on the order of Fragment distance.

$$f_0' = \{\ f_i \mid O_p(f_i) \leq \mathcal{N}(f_0)\ \cap \\ O_s(f_i) \leq \mathcal{N}(f_0)\ \cap \\ O_c(f_i) \leq \mathcal{N}(f_0)\ \} \qquad (8)$$

where f_0' denotes the new Fragment generated by this merging. Terms $O_p(f_i)$, $O_s(f_i)$, and $O_c(f_i)$ represent the ranked order focusing on preceding and following *contexts*, and call-types, respectively. If there is a Fragment similar to the reference Fragment, reference Fragment f_0 is updated by clustering similar Fragments. The clustering algorithm is iterated over the updated Fragment set. When no Fragments are merged into reference Fragment f_0 by Fragment clustering, Fragment f_0 is regarded as one of the Fragments in the next iteration of Fragment clustering in which other Fragments are considered in the selection of the next reference.

Table 1 shows an example of the Fragment grammar generated using the algorithm with the following parameter values. The number of words in a phrase is constrained to three or fewer. Each phrase observed 30 times or more in the training transcription is selected as a *candidate* to participate in the clustering. The maximum number of candidate Fragments N_m is set to 80. The Fragment clustering algorithm yields a total of 288 phrases in 111 Fragments. Sometimes, a given Fragment contains substrings that are themselves Fragments with a higher frequency of occurrence than the given Fragment. When this occurs, the given Fragment can be "parsed" – i.e., the Fragment substrings are replaced by the appropriate non-terminal symbols representing them.

Figure 2 shows an example of Fragment generalization by parsing Fragments. The phrase *"want to make"* in Fragment $< 015 >$ in Figure 2, for instance, can be decomposed into *"want"*, *"to"*, and *"make"*. The words *"want"* and *"make"* can be replaced by non-terminal symbols $< 005 >$ and $< 002 >$, respectively. Therefore, the phrase *"want to make"* can be represented as "$< 005 >$ *to* $< 002 >$". This parsing allows the Fragment grammar to acquire the ability to represent not only phrases given as input but also as word

a. Several Grammar Fragments

$< 002 >$	*"make", "place"*
$< 005 >$	*"have", "need", "want", "would like"*
$< 015 >$	*"want to make", "like to place" "like to make"*

b. Grammar Fragment created from other Fragments

$< 015 >$	"$< 005 >$ *to* $< 002 >$", *"like to* $< 002 >$"
word sequences matching this Fragment	
"want to place", "want to make", "have to place"	
"need to place", "need to make", "have to make"	
"like to place", "like to make"	
"would like to place", "would like to make"	

Table 2: Grammar Fragment Generalization

sequences not observed in the training transcriptions. Figure 2 shows an example of Fragment generalization by parsing Fragments. In this example, the phrases in Fragment $< 015 >$ are generalized by using Fragments $< 002 >$ and $< 005 >$. The three phrases in Fragment $< 015 >$ can be represented as "$< 005 >$ *to* $< 002 >$" and *"like to* $< 002 >$". These non-terminal symbols in Fragment $< 015 >$ are expanded into phrases such as *"need to place"* and *"would like to place"*. As a consequence of this generalization, Fragment $< 015 >$ acquires an additional seven phrases such as *"want to place"*, *"would like to make"* and *"have to place"*. By applying Fragment generalization, 495 phrases are created in 85 Fragments. This reveals that the average number of phrases in a Fragment, 5.82 (495 / 85), is increased by generalization.

For call-type classification, *Salient Grammar Fragments* are automatically generated from the parsed training transcriptions and associated call-types [1]. *Salient Grammar Fragments* are traditionally generated by using the training transcriptions not parsed with the Fragment grammar. In this case, each *Salient Grammar Fragment* comprises a call-type of the highest association score and a corresponding word sequence. The Fragment grammar enables the *Salient Grammar Fragment* to represent several kinds of word sequences having both syntactic and semantic similarities. In the call-type classification process, the ASR output is first parsed by using the Fragment grammar and the *Salient Grammar Fragments* are extracted from the parsed output. A call-type classifier determines the first and second most likely call-types for each utterance by using the association between *Salient Grammar Fragments* and call-types. The call-type classification performance is evaluated by a scorer using the call-type assigned to each test-set utterance.

4 Experiment

A 10 K database of spoken transcriptions between users and human agents was generated as detailed in [1]. In call-type labeling, one of 15 call-types was assigned to each transcription. In some cases, two or more call-types were labeled to a transcription. The transcriptions were split into three subsets for training

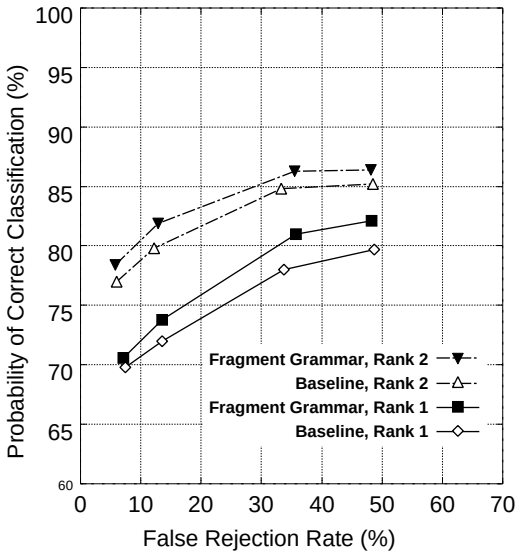

Figure 2: Call-type Classification Performance

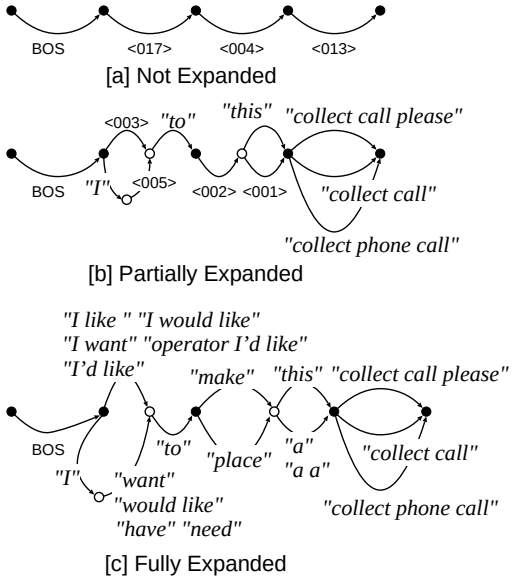

Figure 3: Example of Phrases Accepted by a Salient Grammar Fragment

(8 K), developing (1 K), and testing (1 K) the acoustic and language models for recognition and understanding. The training set had approximately 3.6 K words to define the vocabulary. The speech recognition process is performed by using the *Variable N-gram Stochastic Automaton* [4] as the language model. The training transcription contains 7,844 sentences while the test transcription comprises 1,000 sentences.

In the call-type classification, there are two important performance measures. The first measure is the *false rejection rate*, where a call is falsely rejected or classified as the call-type *other*. The second measure is the *probability of correct classification*. Figure 2 illustrates the probability of correct classification versus the false rejection rate. As a baseline for comparison, the performance without the Fragment grammar is also shown in Figure 2.

The experimental results show that the average and maximum improvements in call-type classification performance of 2.2% and 2.8% respectively were achieved, by introducing the Fragments. These results reveal that the *Salient Grammar Fragments* used in the call-type classifier accept various phrases that are syntactically and semantically similar to the originals providing generality. An example of the variety of phrases accepted by a *Salient Grammar Fragment* is illustrated in Figure 3. Fragment *"BOS* $<$ 017 $>$ $<$ 004 $>$ $<$ 013 $>$*"* shown in Figure 3 has an association with call-type *"COLLECT"*. It is remarkably worthwhile noting that some phrases represented by this *Salient Grammar Fragment* are not observed in the training set. A total of 246 unseen salient phrases have been discovered by clustering and generalizing the Fragment grammar.

5 Conclusion

We described a new method for automatically acquiring Fragments to understand fluently spoken language. The Fragments representing a set of syntactically and semantically similar phrases were generated by using three probability distributions of: following words, preceding words, and associated call-types. The Fragment grammar detected 246 phrases in the test-set that were not present in the training-set. These results revealed that unseen phrases were automatically discovered by our new method. The experimental results showed that the average and maximum improvements in call-type classification performance of 2.2% and 2.8% respectively were achieved by introducing the Fragments.

Acknowledgments The authors thank Dr. Larry Rabiner and Mr. Jay Wilpon of AT&T Laboratories-Research, and Dr. Nobuhiko Kitawaki and Dr. Sadaoki Furui (formerly NTT Human Interface Laboratories) for their support and encouragement of this research.

References

[1] A. L. Gorin, G. Riccardi, and J. H. Wright, "How May I Help You?," *Speech Communication*, Vol. 23, Nos. 1-2, pp. 113-127, 1997.

[2] G. Riccardi, A. L. Gorin, A. Ljolje, and M. D. Riley, "A Spoken Language System for Automated Call Routing," *Proc. of ICASSP*, pp. 1143-1140, 1997.

[3] J. Wright, A. L. Gorin, and G. Riccardi, "Automatic Acquisition of Salient Grammar Fragments for Call-type Classification," *Proc. of Eurospeech*, pp. 1419-1422, 1997.

[4] G. Riccardi, R. Pieraccini, and E. Bocchieri, "Stochastic Automata for Language Modeling," *Computer Speech and Language*, 10 (4), pp. 265-293, 1996.