UNSUPERVISED SEMANTIC INTENT DISCOVERY FROM CALL LOG ACOUSTICS

Xiao Li

University of Washington
Electrical Engineering Department
Seattle, WA, 98195

Asela Gunawardana, Alex Acero

Microsoft Research
One Microsoft Way
Redmond, WA 98104

ABSTRACT

Unforeseen user intents can account for a significant portion of unsuccessful calls in an automatic voice response system. Discovering these unforeseen semantic intents usually requires expensive manual transcriptions. We propose a method to cluster the acoustics from logged calls by their estimated semantic intents. This is achieved through training a mixture of language models in an unsupervised manner. Each cluster is presented to the application developer with a suggested language model to cover the semantic intent of the data in that cluster. The application developer validates the cluster and its suggested language model, and then updates the application. A quantative evaluation on a corporate voice-dialer application shows that updating the application in this manner yields a relative 13.4% reduction in semantic error rate.

1. INTRODUCTION

Automatic voice response systems have gained increasing popularity in human-machine interaction. Rule based finite state or context free grammar (CFGs) are often used as the language model (LM) for simple, system-initiative dialog applications. Such a restricted strategy often leads to high recognition performance for in-grammar utterances, but completely fail when a user's response is out of grammar. There are two causes that may attribute to an out-of-grammar utterance: (a) The utterance's syntactical structure may not be parsed by the CFG; For example, a user's response "twentieth of July" may cause failure to the grammar "**[month] [day]**." (b) The user may have an unanticipated semantic intent. For example, in a corporate voice dialer application, the grammar for the response to the prompt "Good morning, who would you like to contact?" may be designed expecting the user to provide a name. However, the user may respond "human resources."

At the application design stage, it is difficult for an application developer to anticipate all the ways in which the user may frame their request (leading to problem (a)), and all the semantic intents the user might have (leading to problem (b)). Many efforts have been made to solve problem (a) by building more robust LMs, e.g. by hand-authored combination of CFGs and statistical LMs [1, 2]. Solving problem (b) ideally requires a large amount of transcribed and semantically annotated data from real user calls. As this could be extremely expensive, we investigate approaches to automate this process. To this end, this paper presents a probabilistic framework to discover unforeseen semantic intents in an unsupervised manner, and to tune a deployed application accordingly.

In order to cluster acoustics by their semantic intents and to provide annotations automatically, we use word level hypotheses generated from a large-vocabulary speech recognizer. With these hypotheses available, our problem resembles unsupervised text clustering which has been studied extensively. Among the most popular solutions is the vector space approach [3], where a feature vector is extracted from each sample (e.g. inverse document frequency) and a dissimilarity measure (e.g. cosine distance) is defined in this vector space. Clustering is thus reduced to finding a sample assignment scheme that minimizes the total amount of distortion. While finding the optimal scheme is prohibitive, methods similar to K-means (but the means are restricted to be one of the samples assigned to the cluster) or nearest neighbor search are often practically used to improve efficiency [4, 5, 6].

In this work, we propose a model-based clustering algorithm which has a more principled objective and a lower complexity. We do not need to explicitly define a dissimilarity measure; rather, each cluster is modeled as a probabilistic LM and the clustering is intended to maximize the data likelihood. Furthermore, the final LMs of the clusters can be utilized directly for grammar update, which greatly expedites the grammar tuning process. The rest of the paper is organized as follows: Section 2 presents our algorithm of clustering acoustics by their semantics intents; Section 3 discusses how we present clusters to an application developer. Section 4 describes a quantative evaluation method and the evaluation results on a large-scale voice-dialer application, and Section 5 concludes with some discussion.

2. LANGUAGE MODEL BASED ACOUSTIC CLUSTERING

We use a generative Markov model where the acoustic feature sequence x of an utterance is generated from a word sequence w according to an acoustic model $p(x|w)$, and the word sequence w is generated from a semantic intent (or cluster) c based on a per-cluster n-gram LM $p(w|c)$. The complete likelihood of x, w and c can be factorized as

$$p(x, w, c) = p(x|w)p(w|c)p(c), \tag{1}$$

where x is observed and w and c are hidden.

Our objective is to train semantic clusters to maximize the likelihood $p(x)$. In practice, recognition is decoupled from cluster training. In other words, we first perform recognition on the acoustics using a task-independent large-vocabulary LM $p(w)$ and an acoustic model $p(x|w)$ trained on a large set of telephony speech.

The decoded word sequence hypotheses w and their scores $p(w|x)$ are then taken as input in training the clusters.

On the other hand, We use per-cluster uni-grams to model $p(w|c)$, where the sentence end probability is set to be equal among all clusters for simplicity. We choose to use uni-grams because telephony applications often anticipate short utterances, and we believe that in such applications a uni-gram LM of a semantic cluster has a perplexity not much higher than that of a bi-gram (or tri-gram), but has much lower computational complexity. The cluster training, therefore, involves estimating the alphabet of c, the prior on semantic clusters $p(c)$ and the LMs $p(w|c)$.

In this section, we first present the estimation algorithm and discuss some implementation issues. Then we describe our initialization method corresponding to a bottom-up clustering approach, and explain how to merge clusters in this bottom-up scheme.

2.1. Estimating the mixture of language models

As is mentioned above, the training process is intended to maximize the likelihood of an acoustic data set $\{x_i\}_{i=1}^{M}$ consisting of M waveforms. Since w and c are hidden, the optimization can be achieved by the standard EM algorithm.

In the E-step, we first compute the posteriors of the word sequence hypotheses w and the cluster hypotheses c given a waveform x_i, according to model (1):

$$p(w, c|x_i) = p(w|x_i)p(c|w) \tag{2}$$

Specifically we use the fixed LM $p(w)$ and the pre-trained acoustic model $p(x|w)$ to compute $p(w|x_i)$, and use the old priors $p(c)$ and the old per-cluster uni-grams $p(w|c)$ to compute $p(c|w)$. Next, we collect sufficient statistics for the maximization step. To estimate the cluster prior $p(c)$, we need to compute the expected class occupancy count Ψ_c which is given by

$$\Psi_c = \sum_{i=1}^{M} \sum_{w} p(w, c|x_i) \tag{3}$$

Similarly, to estimate the mixture of LMs, we need to compute the expected count $\Phi_{c,u}$ of a uni-gram u appearing with the class c,

$$\Phi_{c,u} = \sum_{i=1}^{M} \sum_{w} p(w, c|x_i)\#_u(w) \tag{4}$$

where $\#_u(w)$ is the number of times uni-gram u occurs in the word sequence w. The M-step thus simply consists of normalizing Ψ_c to give the updated cluster prior $p(c)$, and normalizing $\Phi_{c,u}$ to give the updated class-conditional uni-gram probabilities.

2.2. Implementation Issues

For computational feasibility, the word sequence hypotheses w in Equations (3) and (4) are restricted to a lattice, or a N-best list, with $p(w|x_i)$ renormalized accordingly. In the most aggressive case, where an N-best list of length 1 is used, we get one-best word sequences. Our intent in doing recognition is not to find the true word sequence. Rather, the intent is that the utterances with the same semantic intent yield the same decoding results. For example, if all utterances of "tech support" are decoded as "check support", they still can be clustered together. As will be shown in Section 4, the clustering on true transcriptions has only a slight improvement over that on one-best decoding results. From now on, We let w_i^* denote the one-best word sequence of waveform x_i.

Finally, we could also use Viterbi training instead of EM in the optimization process. In other words, $p(c|w)$ is renormalized to 0 or 1, depending on whether c is the best hypothesis given w. Our experiments show that Viterbi training has very similar clustering results as EM training does, while the computation is significantly reduced in the Viterbi case.

2.3. Model initialization

A critical issue in unsupervised clustering is how to initialize the clusters without the knowledge of the alphabet. One natural solution is to start with each utterance being a cluster on its own and apply an agglomerative approach with a certain merging criterion [4]. However, since telephony applications usually see short utterances with a domain-specific vocabulary, we can first assume that each vocabulary item represents a different semantic intent and initialize the clusters into a better shape:

1. The number of clusters is set to the number of vocabulary items that have a count no less than a floor count (one in our case) in the one best decoding results; each cluster corresponds to a vocabulary item.
2. The prior $p(c)$ of the cluster corresponding to the vocabulary item u is set to the normalized number of utterances containing u.
3. Per-cluster n-gram LMs $p(w|c)$ of the cluster corresponding to the vocabulary item u are trained using all word sequences w_i^* containing u.

Once the clusters are initialized, we apply a number of EM or Viterbi iterations until the models converge. In each iteration, the clusters with a zero prior are removed. The resulting number of clusters is a crude approximation of the number of semantic intents. Some clusters may contain the same semantics and we therefore apply merging to remove potentially redundant clusters.

2.4. Merging similar clusters

Techniques on merging and splitting have been studied in the field of text clustering [7, 6]. Many of them are based on certain distance measure between two clusters. In our work, we use a low-complexity distance measure based on the K-L divergence [8] between the uni-gram distributions of two clusters. Assuming $\gamma_{c,u}$ is the uni-gram probability of vocabulary item u in cluster c ($\gamma_{c,u}$ is proportional to $\Phi_{c,u}$), the distance is defined as an average of the asymmetrical K-L divergences,

$$D(c_1, c_2) \triangleq \sum_{u} (\gamma_{c_1,u} \log \frac{\gamma_{c_2,u}}{\gamma_{c_1,u}} + \gamma_{c_2,u} \log \frac{\gamma_{c_1,u}}{\gamma_{c_2,u}}), \tag{5}$$

where u is summed over all vocabulary items appearing in cluster c_1 and c_2, and any zero probability $\gamma_{c_1,u}$ or $\gamma_{c_2,u}$ are smoothed by a floor value. We merge c_1 and c_2 if $D(c_1, c_2)$ is smaller than a threshold. Upon merging, $p(w, c_{1,2}|x) = p(w, c_1|x) + p(w, c_2|x)$ and the new model is re-estimated using these new posteriors. A few iterations of EM or Viterbi estimation are applied after all such pairs are merged. Alternatively, we can apply re-estimation after each pair is merged, but this would greatly increase computation and hence was not used in our evaluation.

3. CLUSTER PRESENTATION

The clustering algorithm in Section 2 may result in a significant number of clusters; an important task is to select those of our interest and present them to an application developer. This involves ranking the clusters in order of importance, filtering out "garbage" clusters and representing a cluster in a simple and self-descriptive way. Finally, the application developer needs to select the clusters with uncovered semantic intents and update the application grammar accordingly.

(a) **Ranking** The clusters are ranked by their priors. Since a cluster prior indicates how frequent a semantic intent occurs in the data set, the high-ranking clusters are more important to the system, and are hence inspected with a higher priority.

(b) **Filtering** A cluster with a high prior, however, may not be a relevant one. In real-world applications, there are a good number of calls consisting of silence, noise or extraneous speech. These "garbage" utterances tend to be recognized as some function word sequences ("a", "oh", "the", for example), and they are likely to be clustered together with a high cluster prior. However, different from the utterances in a cluster with meaningful semantics, these garbage word sequences are seldom consistent with each other. We thus use a "consistency" measure to filter out such garbage clusters.

First, we define a similarity measure between two utterances to be the number of word tokens they have in common normalized by the total number of word tokens in both of their one-best decoding results. (The similarity measure can be easily extended to the case of N-bests if necessary.) The consistency is then defined as the normalized pairwise similarity of all utterances in a cluster. The clusters with a consistency lower than a threshold will be considered "garbage" and be discarded.

(c) **Representing** We select a "central" utterance to represent a cluster. This utterance is chosen to have the highest sum of similarities with all other utterances in the same cluster, and is intuitively the most representative one.

The above techniques can effectively help an application developer reduce the number of clusters to study, but he/she still needs to decide, by inspecting the central utterance, whether a cluster has an unforeseen semantic intent or it has one already in the application grammar. Once a cluster is decided to have new semantics, the application developer can make corrections to some one best word sequences of that cluster, if necessary. A new grammar rule can be learned based on these corrected word sequences or simply based on the final LM of that cluster.

4. EVALUATION

A natural evaluation strategy is to see how these clusters help improve speech understanding. Specifically, we update the application grammar according to our discoveries, repeat the corresponding dialog state, compute the semantic error rate (SER) and compare it with the SER obtained using the original application grammar. This section describes in detail our evaluation database, method and results.

4.1. Application database

We used MSConnect, a large-scale voice-dialer application developed and used daily in Microsoft Corporation, for our evaluation. When connected to the system, a user is prompted "Good morning, who would you like to contact". The baseline application grammar (G_0) for this dialog state has only one rule: "**[first name] [last name]**" (R_0), consisting of 57875 entries. The user is supposed to speak exactly the full name of the person he/she wants to call. After confirmation, the system will transfer the user to the destination or provide other information. Interestingly, this dialog state has encountered the most diversified answers because the user does not know the underlying grammar and may speak things other than a person's name. The acoustics for this dialog state, therefore, is a good data set to evaluate our technique.

We extracted 5464 waveforms from the call logs corresponding to this dialog state. They were split into four subsets to perform four-fold cross-validation. To obtain the ground truth semantic intents, these waveforms were transcribed manually and annotated with semantic IDs; noise, silence and extraneous speech were annotated as "garbage".

4.2. Evaluation method

First we used a recognizer built upon Microsoft English Telephony Server (i.e. the acoustic model $p(x|w)$) and a 20K uni-gram background LM (i.e. the LM $p(w)$) to obtain one-best word sequences. Since our application grammar G_0 was not necessarily covered by this background LM, most name utterances were recognized as acoustically similar word sequences. Our clustering algorithm described in Section 2 was applied to these sequences using Viterbi training, resulting in a large number clusters. We ranked these clusters by their priors, filtered out 'garbage clusters, and selected those only with unforeseen semantic intents, ending with L clusters. For each cluster c_i, $i = 1..L$, we manually designed a rule R_i to cover all semantic intents of the data in this cluster. Note that a rule R_i may contain multiple semantic IDs. For example, if a cluster contains "building one reception", "building two reception" and so on, we can design a general rule "**building [number] receptionist**" which contain multiple semantic IDs. The weight of R_i is set to be the cluster prior $p(c_i)$, whereas that of R_0 is approximated by $1-\sum_{i=1}^{L} p(c_i)$. The semantic IDs in R_i were assigned as the corresponding ones in the ground truth. These rules were added to the baseline grammar G_0 one by one to form a set of new grammars $\{G_i\}_{i=1}^{L}$ used in evaluation, where $G_j = (R_0, R_1, ..., R_j)$.

The next step was to run recognition on the test sets using G_0 and $\{G_i\}_{i=1}^{L}$ sequentially. If the recognizer determines, by using a confidence threshold, that an utterance contains speech, it outputs its corresponding semantic ID in the grammar; otherwise, it outputs "garbage" as its semantic ID. Given the reference semantic IDs and the recognized ones, we were able to compute the SERs, reflecting the impact of our clustering discoveries on speech understanding.

4.3. Evaluation results

As described in 4.2, we applied Viterbi training to our database, followed by ranking, filtering out garbage and selecting clusters with uncovered semantic intents. This process quickly discovered $L = 9$ clusters from our test sets. In fact, the four test sets each discovered all these 9 clusters, but with different priors. The central utterances of these clusters in one of the test sets are "build-

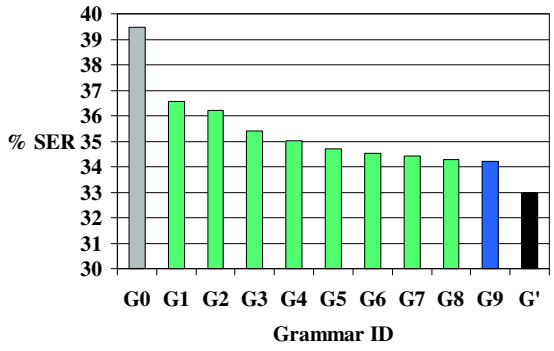

Fig. 1. SERs obtained using baseline grammar (G_0), using updated grammars by clustering on one-best hypotheses (G_1-G_9, where G_9 is the fully updated grammar), and using the fully updated grammar by clustering on the transcriptions (G')

ing thirty reception", "operator", "security", "help desk", "company store", "support", "customer service", "benefits" and "shuttle". EM training gave very similar clustering results.

We followed the evaluation method described in 4.2 to obtain SERs using the updated grammars G_1-G_9. As is shown in Figure 1, the SER decreased as more rules were added to the grammar. The decrease slowed down because the semantic clusters added later (with smaller weights) were less likely to appear in the data set. When all new semantics were added (G_9), the SER was reduced from 39.49% to 34.22%, i.e. a relatively 13.4% reduction.

To show that one-best decoding is sufficient for clustering in this application, we also obtained a grammar G' by clustering on the ground truth transcriptions of the train sets, and re-ran the recognition using G'. As is shown in the figure, using G' in recognition gave a SER of 33.0%. This improvement is only trivial considering the cost of manually transcribing thousands of waveforms.

It is also important that adding these new semantic items does not bring a significant amount of confusion between the names and the newly added semantic items. Table 1 summarizes how the semantic errors changed in each category when the grammar was updated from G_0 (baseline) to G_9 (the one including all discoveries). The categories are full names, uncovered semantics and garbage. Obviously the new grammar did not substantially affect the name recognition, while having over half of the utterances with unforeseen semantics correctly recognized. It is worth noting that some originally mis-recognized names were recognized correctly using the updated grammar. This is because the pruning parameters in the Microsoft speech recognizer are automatically adjusted to the LM.

Additionally, we hand-crafted a grammar G^* based on the transcriptions for comparison. This grammar has rules covering all semantic intents of the data, but each rule has a weight equal to that of the name rule R_0 for simplicity. The SER obtained using G^* is 37.0%, lower than that of G_0 but higher than that of G_9. Though a grammar designed in this fashion is by no means an optimized one, it more or less indicates what is achievable by tuning an application grammar in a completely manual manner.

	Names	Uncovered	Garbage.	All
incor→cor	31	318	1	350
cor→incor	55	0	7	62
cor→cor	3218	0	26	3244
incor→incor	1194	225	389	1808
Total	4498	543	423	5464

Table 1. Number of instances changed when $G_0 \rightarrow G_9$; "cor" and "incor" stand for correct and incorrect semantic recognition respectively

5. CONCLUSION AND DISCUSSION

This paper presented a LM based clustering algorithm to discover unforeseen semantics from call log acoustics. Compared to conventional clustering methods, our approach does not require to explicitly define a feature vector or a dissimilarity measure. Rather, it models the dissimilarity implicitly within a probabilistic framework, and trains the clusters in the maximum likelihood sense with a fairly low complexity.

This approach brings a great improvement for a voice-dialer system (probably as good as that brought by a completely manual tuning), while eliminating the needs of manually transcribing a large number of waveforms. Though this work used an in-house application for evaluation, the framework is general enough to be applied to other applications. For more complicated applications with longer utterances, bi-gram LMs can be considered as per-cluster LMs.

The authors would like to thank Ye-Yi Wang, Dong Yu, Peter Mau and Milind Mahajan at Microsoft Research for providing useful tools and advice to this work.

6. REFERENCES

[1] Y.Wang, M. Mahajan, and X. Huang, "A unified context-free grammar and n-gram model for spoken language processing," in *IEEE ICASSP*, 2000.

[2] K.Hacioglu and W.Ward, "Dialog-context dependent language modeling using n-grams and stochastic context-free grammars," in *IEEE ICASSP*, 2001.

[3] W. Wu, H. Xiong, and S.Shekhar, *Clustering and information retrieval*, Kluwer Academic Publishers, 2004.

[4] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning*, Springer-Verlag, 2001.

[5] L.Begeja, D.Gibbon, Z.Liu, and B.Shahraray, "Interative machine learning techniques for improving SLU models," in *NAACL*, 2004.

[6] C.C.Aggarwal, S.c.Gates, and P.S. Yu, "On using partial supervision for text categorization," *IEEE Trans. on Knowledge and Data Engineering*, vol. 16, 2004.

[7] D.R.Cutting, D.R.Karger, J.O.Pedersen, and J.W.Tukey, "Scatter/gather: A cluster-based approach to browsing large document collections," in *Proc. ACM SIGIR*, 1992.

[8] T.M.Cover and J.A.Thomas, *Elements of Information Theory*, Wiley, 1991.