Minimal Training based Semantic Categorization in a Voice Activated Question Answering (VAQA) System

Mithun Balakrishna, Marta Tatu, Dan Moldovan

Lymba Corporation
Richardson, TX 75080

{mithun, marta, moldovan}@lymba.com

Abstract

In this paper, we develop a knowledge based methodology that maps Automatic Speech Recognizer (ASR) transcriptions to predefined semantic categories in a Voice Activated Question Answering (VAQA) system. The proposed semantic categorization methodology, SemCat, uses a novel lexical chains/ontology based algorithm and relies heavily on customized but domain independent Natural Language Processing (NLP) tools and does not require any domain-specific utterance collections or manually annotated text data. SemCat requires minimal manual intervention during training, relying only on the semantics encoded in a brief, manually-created description for each predefined category/slot. SemCat uses these descriptions along with the extended WordNet (XWN-KB) and several domain independent NLP tools including XWN lexical chains to accurately extract information and map user utterances to predefined categories. SemCat also uses the domain ontologies created automatically by the Jaguar knowledge acquisition tool to accurately extract domain/customer specific language/terms.

Index Terms: semantic categorization, speech recognition, question answering

1. Introduction

Currently commercial and research Spoken Dialog (SD) applications are being deployed for a wide variety of domains/applications. The ever increasing demand and the non-availability of domain/application-specific annotated corpora for these new systems has lead to expensive, time-consuming and labor intensive methodologies for creating the SD data-models required by the ASR, Spoken Language Understanding (SLU) modules, etc. One way to create these new domain data-models is to manually/semi-automatically encode Context-Free Grammar (CFG) rules or Statistical Language Model (SLM) probabilities using a qualified SD application designer. Usually, it is possible to adapt the data from a particular application to suit the needs of a new application in a similar/parallel area but this still requires considerable amount of manual labor depending on the amount of domain variations. Another alternative is to collect domain-specific corpora using a prototype or wizard-of-oz application. After manually transcribing, semantically categorizing and statistically analyzing the utterance collections, the required SD language models are trained.

Language understanding methodologies in SD applications can be categorized into two prominent schools of thought. [1, 2] are examples of knowledge based methodologies, which use pattern/template matching or robust parsing techniques. They are fairly accurate but require expert linguistic knowledge and/or fully annotated data to generate domain-dependent grammar rules. [3, 4] are examples of statistical methodologies, which rely on creation of stochastic models for SLU. These methods have better accuracy and coverage but are very data intensive, require fully annotated data for a good performance and are easily effected by data sparseness problems. More recent techniques tried to combine the advantages of both these methodologies [5] or tried variations such as semi-automatic grammar learning [6], automatic grammar tuning [7] or hidden vector state models [8]. These methodologies have good accuracy and coverage but are still labor intensive since they require a fair amount of domain-dependent, manually/semi-automatically annotated knowledge for good performance.

In this paper, we present a knowledge based semantic categorization methodology for our VAQA system which maps the user utterance into predefined semantic categories or redirect the utterance to PowerAnswer [9] (a state-of-the-art, natural-language, question-answering module). SemCat maps the ASR transcriptions into categories/slots with minimal manual intervention. Our definition of minimal manual intervention refers to the minimal usage of expert (human) knowledge, complete abstinence from user utterance collections or manual/semi-automatically annotated data. SemCat relies on the semantics encoded in the brief category descriptions from the speech application designer. SemCat uses XWN-KB and other domain independent NLP tools including XWN lexical chains (extension of the algorithm in [10]) to accurately extract information from user utterances. SemCat uses the ontologies created by the Jaguar knowledge acquisition tool [11] to accurately categorize the utterances containing domain/customer specific terms. Jaguar automatically creates domain/customer specific ontologies by extracting knowledge from largely available (but noisy) text documents (www, internal product documentations, etc).

2. Lexical Chains based Semantic Categorization in the VAQA System

In this paper, we present a novel, knowledge based semantic categorization module in our VAQA application architecture. The user utterance transcription (usually an n-best list) from the VAQA ASR is fed to the SemCat module. SemCat maps the user utterance to a predefined category/slot-values or classifies the utterance as an open question and directs the user question to PowerAnswer. The goal of SemCat is to automatically and accurately categorize utterances with minimum manual intervention during training (no annotated text corpora, user utterance collections or manually created CFGs for the SD domain). As shown in Table 1, SemCat requires only a brief, single-line description from the application designer, one for each category and their corresponding optional slots in the SD application.
Table 1: SemCat manual input for AccountPayment example.

<table>
<thead>
<tr>
<th>Semantic Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>arrange a payment</td>
<td>users can arrange automatic future account payments</td>
</tr>
<tr>
<td>report a payment</td>
<td>users can report previously made payments</td>
</tr>
<tr>
<td>make a payment</td>
<td>users can pay their bill using different payment modes</td>
</tr>
<tr>
<td>account</td>
<td>User account can be one of the following types: $PM$</td>
</tr>
<tr>
<td>payment_type</td>
<td>One of the following payment types can be used: $PT$</td>
</tr>
<tr>
<td>payment_mode</td>
<td>Payment modes can be one of the following: $SM$</td>
</tr>
</tbody>
</table>

Figure 1 depicts the architecture of our proposed methodology, SemCat. The ASR n-best list transcriptions for a user utterance is fed to our in-house Case Restorer module to capitalize and restore the cases of the transcriptions. Case Restorer uses the implementation in [12] along with some additional heuristics and a tri-gram language model (built from general LDC English GigaWord corpus and customer specific documents). Availability of customer specific documents is important to restore case for customer specific terms like “Gold Checking Account” or “X’s Value Menu”. Case restoration is important for NLP tools such as the named-entity recognizers, chunk parsers, etc. to work accurately on ASR transcriptions. The case-restored transcriptions are processed by SemCat in two parallel paths. The first path leads to the categorization of the n-best list into the pre-defined semantic task-labels. A categorization confidence score is associated with each possible semantic task-label. The second path identifies all the semantic slots, possible at that dialog state, present in the user utterance. The information from the two paths are then merged by the Category/Slot Resolver to form a decision on whether to redirect the utterance to PowerAnswer or resolve and pass the mapped category/slots to the Dialog Manager.

2.1. Semantic Task Categorization

The semantic task-label categorization path maps the user utterance to a ranked set of semantic category labels. First, the case-restored ASR n-best list transcriptions are tagged with Part of Speech (POS) information using Brill’s tagger. Then, each transcription is fed to the Semantic Categorizer module, which uses eXtended WordNet lexical chains [10] to generate a set of task labels, ranked using categorization confidence scores that indicate mapping distances between categories and an utterance.

In order find relevant task labels, the Semantic Categorizer computes semantic similarities between task label descriptions and an input transcription using lexical chains [10]. According to this methodology, two concepts are topically related if they are connected by paths in the WordNet [13] lexical database. For this paper, we extended this process to use XWN-KB which provides several important enhancements to remedy the present limitations of WordNet: WordNet glosses are syntactically and semantically parsed, disambiguated and organized to form a rich knowledge base that dramatically increases WordNet’s semantic connectivity. Furthermore, we add application specific terms to XWN-KB using domain specific ontologies. Jaguar [11] automatically creates these domain and customer specific ontologies by extracting semantic knowledge from largely available (but noisy) domain/customer documents (www, white-papers, internal documentations, etc).

In Figure 2, we show the mapping of an ASR transcription (i settled bill throw a post that is) to a semantic task label (payment_method_pay). We find the lexical chains between each content word (settle, bill, throw, post) in the ASR transcription and each content word (mail, check, pay, bill) in the semantic task description. The scores of the best lexical chains for each content word in the semantic category description are summed up to form the mapping distance measure for the (transcription, semantic task) pair. In Figure 2, the lexical chain (b) links the fifteenth WordNet sense of the verb settle with the third sense of the verb pay. The lexical chain distance measure for a word pair is determined by their semantic distance in WordNet and the type/length of the semantic-relations linking them. The lengthier the chain, the higher is the lexical chain distance measure. The mapping distance measure for the (transcription, semantic task) pair in Fig. 2 is 1.52 (sum of best lexical chain values) i.e. (bill to bill, 0.00), (settle to pay, 0.14), (post to mail, 0.14), and (settle to check, 1.24).

Figure 3 presents our algorithm to map an n-best list into
For each A in user utterances set
For each B in n-best ASR transcriptions
For each C in semantic category labels set
For each semantic category description D for C
If ValidMapping(B, D)
UpdateBestDistanceMeasure(B, C)
Distance(A, C) = BestDistance(B, C)
For each C in semantic category labels set
NormalizeDistance(A, C)

Figure 3: High-level algorithm to map n-best transcriptions into pre-defined categories in the VAQA application dialog-states.

a ranked semantic tasks set. For each utterance A, the top n-best (n=10) transcriptions are selected for the classification task. By finding the best lexical chains between every (transcription, task description) pair, the utterance is mapped to a ranked list of semantic task labels. A mapping is valid if and only if there exists a lexical chain between every word in description and at least one word in the transcription. The procedure ValidMapping() identifies the mapping between a given pair and returns true if and only if the validity condition for mapping holds. For any valid pair, UpdateBestDistanceMeasure() computes the distance measure and updates its score (a label can have more than one description). BestDistance() finds the best mapping score for the pair and the distance measure is added for the pair. NormalizeDistance() then normalizes the distance measure score based on the number of n-best transcriptions mapping to a particular label and produces a ranked list of categories.

2.2. Semantic Slot Detection

SemCat slot detection module treats the detection and mapping of slot values as a categorization problem due to the presence of a finite number of well-defined slot labels at each dialog state. The case-restored ASR n-best transcriptions are processed by our in-house named entity recognizer, Rose. We augment the lexicon of Rose with domain-specific named entities (e.g. payment_type, payment_mode, etc.) from the example in Table 1 and add the semantic slot values specified by the speech application designer as their possible values. To increase their value coverage, we search the created domain/customer specific ontologies for the designer specified slot values. If the slot value is found in the ontology, we add distinct nodes from the found subtree as possible values for the corresponding named entity. Example: if vehicle is a designer specified slot value then all the ontology sub-tree nodes (like car, truck, etc.) for vehicle are added as possible values for the named entity Insurance_Category.

The case-restored, POS tagged ASR n-best transcriptions are processed by the YamCha chunk parser (trained on TreeBank 2) and together with lexical chains form the input to the Slot Detector module. This module contains a mapping function and a deep semantic parsing algorithm. Let E be the named entities set and V be their corresponding values set, identified in a transcription. For every semantic task label, let S be the semantic slot labels set and EV be their corresponding named entities set. The mapping function assigns each (e1, v1) pair to a particular label (s1, ev1) if and only if e1 is a unambiguous named entity in E and ev1 is a unambiguous named entity in EV and e1 = ev1. The semantic parsing algorithm in Slot Detector is called for a (e1, v1) pair, if and only if the mapping function fails to assign it to any slot in S and if e1 equals some ev in in EV. The semantic parsing module is a more complex procedure to disambiguate the slot-value pairing. We rely on a deep semantic parser [11] to detect semantic roles in the utterance/description, disambiguate the slot-value pairing and map the slot labels accurately to their values in the transcription.

Let us consider a dialog state with the following two semantic categories and their corresponding slots: Confirm(Origin:LOCATION, Date:DATE, DepartureTime:TIME), Fly(Origin:LOCATION, Destination:LOCATION, Date:DATE, DepartureTime:TIME). For the transcription ...flight from (LOCATION, Austin) at (TIME, 2:30), (TIME, 2:30) can be assigned to DepartureTime:TIME in both Confirm and Fly using the simple mapping function. But Fly also has two slots which map to the same named entity LOCATION, and hence when we find one or more LOCATION named entities in the transcription, they need to be disambiguated and mapped to the right slots. In this case, the semantic parser uses the preposition based classifier/model (due to the presence of the preposition from in the transcription) to correctly assign (LOCATION, Austin) to Origin:LOCATION based on the input slot label description.

2.3. Semantic Category and Slot Resolution

The goal of the Resolver module is to use the ranked categories list and their confidence scores, slot coverage information, and the ASR confidence scores to resolve and pass the mapped category/slots to the Dialog Manager or redirect the utterance to PowerAnswer. The Resolver uses a filtering algorithm involving several threshold measures. First, the mapping distance threshold is used to filter out labels from the ranked task labels list. An utterance is assigned to a task label if the distance measure is less than an absolute threshold value. Second, to allow an utterance to map to more than one task label, we define a difference threshold value. Hence, any utterance is first mapped to the best task label (with the lowest distance which is lesser than the absolute threshold value) and then to any other task label (with a distance < Max(distance of the best label + difference threshold value), absolute threshold value)). We also define a difference decaying factor, which is the factor used to reduce the difference threshold value as the number of labels assigned to an utterance grows. Finally, we pick the slot values that need to be filled for each of the selected task labels using the ASR confidence score to break any ties. For the example in the previous section, let us assume that Fly is only task label selected for an user utterance using the above task label filters and, for the slot Origin:LOCATION we have two possible values (Origin:LOCATION, Austin) from the second-best transcription and (Origin:LOCATION, Boston) from the fourth-best transcription. We then simply associate the slot originating from the transcription with a better ASR confidence score (i.e. (Origin:LOCATION, Austin)) with the task label Fly. If the number of task labels/slots assigned to an utterance is zero or greater than a threshold value then the utterance is mapped as a Question category and redirected to PowerAnswer.

3. Experimental Settings and Results

To test our proposed semantic categorization methodology, we developed a VAQA system for the telecommunication-provider application domain. We processed a collection of 15300 live-user utterances through 35 different VAQA dialog states using the CMU Sphinx 3.3x as the ASR. We trained the acoustic model for the telephone transcription task using 160 CallHome and 4826 Switchboard-1 conversation sides. The ASR WER for this model along with the SRI HUBS5 2000 tri-gram SLM is 33.1% on HUBS5 (CallHome + SwitchBoard test sets). For tran-
Table 2: Results comparison between a manual CFG-rules based system and SemCat (4 evaluations using 4 sets of brief category/slot descriptions created by application designers).

<table>
<thead>
<tr>
<th>Test Utterance Set (15300 ASR Transcriptions)</th>
<th>Error (%)</th>
<th>MisCat</th>
<th>InCFG</th>
<th>DelCFG</th>
<th>In Del</th>
<th>Total Error</th>
<th>Total Correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual CFG-rules</td>
<td>1.45</td>
<td>3.07</td>
<td>2.95</td>
<td>1.53</td>
<td>0.19</td>
<td>9.17</td>
<td>92.96</td>
</tr>
<tr>
<td>SemCat-DescSet1</td>
<td>2.58</td>
<td>3.59</td>
<td>3.48</td>
<td>0.88</td>
<td>0.67</td>
<td>10.90</td>
<td>89.10</td>
</tr>
<tr>
<td>SemCat-DescSet2</td>
<td>2.74</td>
<td>3.85</td>
<td>3.28</td>
<td>0.63</td>
<td>0.47</td>
<td>10.99</td>
<td>89.01</td>
</tr>
<tr>
<td>SemCat-DescSet3</td>
<td>2.8</td>
<td>3.6</td>
<td>3.8</td>
<td>0.63</td>
<td>0.46</td>
<td>11.32</td>
<td>89.33</td>
</tr>
<tr>
<td>SemCat-DescSet4</td>
<td>3.59</td>
<td>3.54</td>
<td>3.66</td>
<td>1.08</td>
<td>0.46</td>
<td>12.24</td>
<td>87.75</td>
</tr>
</tbody>
</table>

scribing the live user collection, we created 8 different SLMs to cover all the dialog states. All the SLMs were created using the unannotated domain/customer documents and then interpolated with the SRI HUBS 2000 tri-gram SLM. This gave us a reasonably decent ASR WER of 18.6% on the live user collection.

We use the Semantic Error Rate (SemER) evaluation metric presented in [14] to measure the categorization accuracy of SemCat. *MisCat* errors are due to mismatches between the task labels or slots proposed by SemCat and the actual utterance task labels or slots. *InCFG* errors are due to SemCat proposing a label or slot while the utterance’s actual label is a NULL. *OutCFG* errors are due to SemCat proposing a NULL while the utterance actually has a valid label or slot. *Ins* errors are due to the insertion of a label or slot by SemCat while the utterance’s actual label or slot list does not contain such an entity. *Del* errors are due to the deletion of a label or slot present in the utterance’s actual label or slot list while the entity is missing in SemCat’s label or slot list. Total Error (%) is the sum of all the 5 different error counts divided by the total number of reference labels and slots. Total Correct (%) is 100 − MisCat(%) − InCFG(%) − OutCFG(%) − Del(%) − Ins(%) − OutCFG(%) − Del(%).

To our best knowledge, there exists no other parallel semantic categorization methodology that also does not depend on any annotated text corpora, user utterance collection or manually created grammar rules in the SD domain. Therefore, we compare SemCat’s results against the performance of a system which benefits from using hand built language understanding/parsing grammar rules for semantic categorization. These rules were created manually by an application designer using the same category/slot descriptions used for SemCat.

We obtain four different SemCat results based on four different category/slot description sets written by different speech application designers. Table 2 presents the results obtained in the decreasing order of categorization accuracy. The *Manual CFG-rules* system has the highest accuracy of 92.36%. As mentioned before, this system requires a lot of manual effort to create the rules for parsing and categorizing the utterance transcriptions. The best result using our proposed SemCat methodology is obtained for SemCat-DescSet1 system and this is 2.49% worse than the manual grammar based system. The accuracy results for the SemCat-DescSet1, SemCat-DescSet2 and SemCat-DescSet3 systems are very similar though their input category/slot description sets are very different. We believe that this is due to the generalization power of our methodology. We believe that as long as the descriptions relate to the tasks in a good manner, the system overall results will not change drastically. For our proposed methodology, the SemCat-DescSet4 system has the worst result, 3.61% lower than the *Manual CFG-rules* system. Overall, SemCat performance was respectable given the self-imposed minimal manual labor constraint.

4. Conclusions and Future Work

In this paper, we present a knowledge-based semantic categorization methodology for our VAQA system required to map the user utterance to predefined semantic categories or redirect the utterance to PowerAnswer (state-of-the-art, natural-language, question-answering module). Our definition of minimal manual intervention refers to the minimal usage of expert (human) knowledge, more reliance on domain independent NLP tools and data sources, complete abstinence from utterance collections or domain-dependent annotations. SemCat is designed to rely minimally on annotated data and use brief, manually-created, category/slot description along with the extended WordNet Knowledge Base and several domain independent NLP tools including xTended WordNet lexical chains. SemCat also uses the domain ontologies automatically created by the Jaguar knowledge acquisition tool to accurately extract domain/customer specific language/terms for the mapping process. As presented in Section 3, the overall performance of SemCat in VAQA was more than satisfactory given the minimal manual labor constraint imposed on it. Future work in this area will include development of a better slot value matching mechanism and a robust algorithm to resolve the slot value conflicts among the ASR n-best transcriptions.

5. References