Clustering Novel Intents in a Conversational Interaction System with Semantic Parsing

Dilek Hakkani-Tür, Yun-Cheng Ju, Geoff Zweig, Gokhan Tur

Microsoft, USA

dilek@ieee.org, yuncj@microsoft.com, gzweig@microsoft.com, gokhan.tur@ieee.org

10.21437/Interspeech.2015-70

Abstract

Spoken language understanding (SLU) in today's conversational systems focuses on recognizing a set of domains, intents, and associated arguments, that are determined by application developers. User requests that are not covered by these are usually directed to search engines, and may remain unhandled. We propose a method that aims to find common user intents amongst these uncovered, out-of-domain utterances, with the goal of supporting future phases of dialog system design. Our approach relies on finding common semantic patterns in uncovered user utterances using an Abstract Meaning Representation based semantic parser. We represent the corpus as a graph and find subgraphs that represent clusters, by pruning the corpus graph according to frequency and entropy. We employ crowd-workers to select and label the resulting clusters and compare resulting clusters with two baselines. Experimental analyses show that we obtain higher coverage and accuracy with the semantic parsing based clustering method. Furthermore, since the intents and candidate slots are already induced, these utterances can also be used in unsupervised SLU modeling. In intent classification experiments, we show that the statistical model trained using the clusters formed by this approach results in higher classification F-measure (showing about 25% relative improvement) in comparison to the alternatives.

Index Terms: spoken language understanding, dialog system coverage, discovering user intents, semantic clustering

1. Introduction

Spoken language understanding (SLU) in spoken dialog systems aims at determining the semantic frame that represents user's intent and associated arguments (slots) [1]. For most dialog applications, these frames are manually designed by domain experts and aligned with the back-end knowledge sources. Inferring semantic frames for new domains and user intents is critical for bootstrapping conversational systems and language understanding models from data. It is also an important task for extending the coverage of conversational interaction system to new uncovered domains and intents, that users frequently ask about. In our previous work we have proposed a classification method to detect such *orphan* utterances using syntactic and semantic features [2].

In this paper, we tackle the task of discovering user intents and associated arguments in user utterances addressed to a conversational interaction system, that are not currently covered by a dialog system application. Multi-domain personal agent applications, such as Siri and Cortana, usually direct such utterances to a generic text-based web search engine, such as Bing, which is not necessarily developed with the aim of responding to them, except with limited support on instant answers to mainly factoid questions. We aim at finding intent-wise homogeneous semantic clusters to aid in future dialog system coverage design, and bootstrapping language understanding models for these new domains/intents.

Clustering methods which treat each user utterance as a bag of words could be an obvious choice to group uncovered user utterances. Then, dominant clusters can be presented to dialog system builders or used in unsupervised model training methods, to improve the coverage of the dialog system application. Such approaches treat all words equally, and treat multiple senses of the words similarly. For example, an approach that uses bag of words-based approach may group all of the following user utterances in the same cluster:

play **hotel california**

book *a* **hotel** *in saratoga* **california**

find a **book** *about haunted* **hotels** *in* **california**

as most of the non-stop words are shared between these utterances (as shown in boldface fonts in the examples). However, their semantic intents (i.e., *playing a song, booking a hotel room,* and *finding a book*, respectively), and syntactic and semantic structures are very different. Previous work [3, 4] used web page click information, as well as words in utterances, to group utterances that have similar semantic intents together, with the motivation that web search users that have the same intents would use similar words in their query, and they also would click on the same or similar urls. Web page click information may not be available in all conversational interactions, especially in the context of spoken dialog systems. Hence, we investigate an approach that considers the lexical semantic structure of the utterances when treating them as similar. In this paper, we rely on semantic parses of utterances during clustering them into intent-wise homogeneous groups.

In the rest of this paper, we first present related work based on semantic parsing to infer representations for user utterances. In Section 3, we describe our clustering approach, and then present experimental results that compare our approach with other clustering methods. We ask crowd workers to map the automatically discovered clusters to user intents designed by a domain expert, so that we can evaluate various clustering methods with respect to manually annotated data. We show experimental results both in terms of coverage of the discovered intents and the accuracy of the examples in the resulting clusters.

2. Related Work

Previous work [5] suggested the use of predicates and associated arguments for determining intents in user utterances using a PropBank [6] style semantic role labeler (SRL), Assert [7]. PropBank style parsing aims to put "*who* did *what* to *whom*" kind of structures to sentences without considering the application that uses this information. More formally, given a predicate

of the sentence, the goal of SRL is to identify all its arguments and their semantic roles. An issue with the PropBank approach is caused by the nominal phrases, especially the prepositional phrases (PP) attached to noun phrases (NP). Consider the example sentence "*I want a flight from Boston to Seattle*". Since SRL only classifies verbal predicate/arguments, it can not extract information from the PPs that are attached to a NP, hence it cannot find important slots for a travel application, namely the origin and destination cities. Another issue with this approach is caused by utterances with no predicates (i.e., verbs), such as "*account balance*". Furthermore, the verbs *be* and *have* are not marked as predicates in the PropBank corpus. This causes utterances, such as *I have a billing question*, to have no predicate. This paper proposes to use AMR style semantic parsing, as explained below, which is free from all these issues. An issue with the previous approach [5] that uses all predicates and arguments is the granularity of the intent. For example, "placing a call" as represented by utterances such as "*call mom*" or "*call the safeway florist*" may all be considered to have the same intent at the semantic predicate level *call*, whereas utterances such as "*schedule a 10am meeting*" and "*schedule a meeting with john tomorrow*" can be represented with the predicate and one of its arguments *schedule meeting*. To obtain such granularity in clustering, our approach relies on forming a semantic parse graph that covers the whole corpus of uncovered examples, and pruning it to get commonly intended, semantically-motivated clusters. The ideas of pruning trees or graphs for clustering [8] or representing a user utterance in a graph are not new, however, we are not aware of other work that uses semantic parse graphs for clustering user utterances.

Note that, this work is related semantic template induction, where the goal is identifying the target semantic template given a set of utterances for the target domain. For example, given a set of similar flight related sentences like "*flight from boston to new york tomorrow*" or "*i need to fly to boston next week*", the goal is identifying the target slots. Few notables studies towards that direction are as follows: Chotimongkol and Rudnicky proposed clustering words together to form concepts (or slots) using Kullback-Leibler divergence based on probability distributions with immediate context (e.g., "previous word is *from*") and using mutual information for phrases (e.g., "New Tork") [9]. This is almost the same as the simultaneous approach taken by Meng and Siu [10]. The latter approach has focused on producing a probabilistic linear context-free grammar (pCFG) via this process for unsupervised SLU modeling.

Recently Chen et al. have proposed using a FrameNet [11] parser, Semafor [12], with spectral clustering for inducing semantic slots [13]. Here is an example utterance with FrameNet semantic tags: "*can_I/capability have a cheap/expensiveness restaurant/locale_by_use*". This approach is the most similar to ours; one big difference is that, instead of AMR style lexical semantic parsing, which is while more shallow, was shown to be more robust and more directly trainable [14]. The underlying parser in the previous work is based on FrameNet templates, which does not provide relational information. Later on Chen et al., have used dependency parsing and random walk algorithm to induce slot relations [15].

3. Approach

3.1. Semantic Parsing

Our approach is based on semantic parsing of user utterances, such as in PropBank [6] or FrameNet [11]. While our work is not limited to a specific parsing approach, in this study, as in our earlier work on detecting the uncovered utterances addressed to a virtual personal assistant [2], we employ the Microsoft NLPwin parser [16]. NLPwin is a generic knowledge-based, high-coverage semantic parser, which can output abstract meaning representation(AMR)-style semantic parses [14]. For example, for the user utterance *tell me a funny joke*, one can get a semantic parse as below, which can also be represented as a graph (see Figure 1):

Input: tell me a funny joke

```
(t / tell
   :ARG0 (y / you)
   :ARG1 (j / joke
      :mod (f/ funny))
   :ARG2 (i / I)
   :mode imperative)
```

`:mode` shows the dialog act of the input utterance, such as imperative, interrogative, or exclamation, if it is not a regular statement. `:ARG0` is usually the subject, `:ARG1`, the direct object, `:ARG2`, the indirect object, consistently with PropBank role sets[6]. `:mod` denotes the modifier. The instance edges in the graph in Figure 1 show the lemma form of the words in the utterance.

While these parsers may not be very robust to erroneous ASR output of naturally spoken utterances, most of the conversational system user utterances are short and simple to parse. Moreover, even if some instances are lost due to errors, a majority of them, or parts of them, are still parsed correctly, resulting in representative examples for clustering.

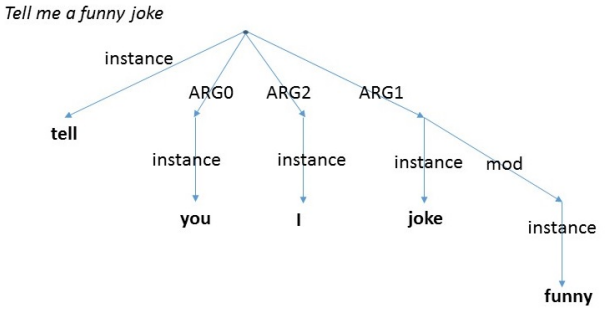

Figure 1: Example AMR parse graph for user utterance *tell me a funny joke*. `:mode` is excluded for simplicity.

3.2. Clustering based on Semantic Parses

AMR represents each sentence as a rooted, directed, acyclic graph, with labels on edges (relations) and leaves (concepts). We follow the AMR approach in representing user utterances in conversational interactions with a machine, and use graphs and the notation in [14].

The root node is common to the semantic parses of all utterances in a corpus. As one traverses the semantic parse graph of a sentence, starting from the root node, the number of utterances that share the subgraph from the root to the current node in their semantic parses gets smaller. For example, all three user utterances: "call mom", "call the florist", and "tell me a joke" include a root node, but only the first two have "call" as a predicate in their semantic parses. As mentioned earlier, user intents as stated by user utterances are highly correlated with the predicates and arguments in their utterances.

Our goal is to find subgraphs in the corpus that can represent intent-wise homogeneous groups of utterances, which are more specific than the root node alone, and more general than the full semantic parses of individual utterances. For example, utterances such as "call mom", "call the florist" have the following semantic parses (we removed the `:mode` for simplicity in these examples):

```
(c / call
   :ARG0 (y / you)
   :ARG1 (m / mom)
```

and

```
(c / call
   :ARG0 (y / you)
   :ARG1 (f / florist)
```

Having a cluster represented by the following parse, where * can match anything:

```
(c / call
   :ARG0 (y / you)
   :ARG1 (*)
```

may group these two utterances into the same semantic cluster, and may be more appropriate than the two less frequent and more specific clusters.

For finding such subgraphs, we represent all the utterances in the corpus in one graph $G = (V, E)$, which we call the corpus graph. $v_j \in V$ represent the vertices, and $e_i \in E$ represent the edges, and there are two types of edges, for arguments and instances, *A* and *I*, respectively ($E = A \cup I$).

One approach to clustering according to semantic parses is to group user utterances according to their predicate and arguments, and use frequent predicates and arguments as user intents (referred as AMR-freq in experiments in the next section) [5]. To obtain these, we prune the corpus graph by cutting infrequent vertices and edges, and use the remaining predicate-argument patterns in the graph as cluster representatives. All utterances that include the pattern in their semantic parse can be considered in the same cluster as the cluster representative. The previous work [5], used PropBank semantic parses, and was limited to verbal predicates as mentioned earlier. In this work, we also consider nominal predicates as extracted by the AMR parses.

In the frequency based approach, if an intent is more general than the predicate and argument combination, we may miss this intent. For example, "call mom" and "call dad" may both appear frequently in the corpus, and hence each would result in their own clusters. Whereas such arguments that take several values in the corpus are better candidates for slot categories. In this work, we apply entropy-based pruning to group such specific instance edges (referred as AMR-entropy in experiments). Basically, for all instance edges $i_k \in I$, we have a set of vertices that carry the possible values of that instance, we denote the set of values/vertices by $v_k^m \in V_k$, where $m = 1, ..., M_k$. M_k is the number of possible values for that edge in the corpus. For example, if the corpus only had two utterances, "call mom" and "call dad", M_k for the instances of the `:ARG1` edge would be 2, and the `:ARG0` edge would be 1. Our main criterion for pruning the subtree is the entropy of the instances of the current edge i_k, $H(v_k)$:

$$H(v_k) = -\sum_{m=1}^{M_k} p(v_k^m) \log p(v_k^m)$$

We perform a breadth-first traversal of the corpus graph. At each vertex, we prune the instance edge i_k if $H(v_k) > \theta$, where the threshold θ is optimized on the development data.

	Train	Dev	Test
No. examples	16,000	2,000	1,902
No. intent categories	165	88	92

Table 1: Properties of the data sets.

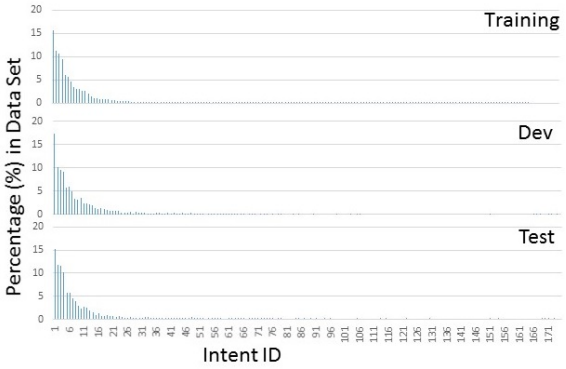

Figure 2: Percentage of user intents in each data subset.

4. Experiments

4.1. Data Sets and Evaluation

We performed controlled experiments, where we use a data set with manually annotated user intents, so that we can compare the clusters from different methods with each other, as well as with domain expert annotations. The data set contains utterances from multiple domains, such as restaurant, hotel, and movie search and information gathering, and calendar scheduling. The properties of the three subsets of our data set, for training, development and testing are presented in Table 1, and the distribution of the intents in each subset is presented in Figure 2.

As a baseline for the newly proposed methods based on semantic parsing, we use two other clustering methods: k-means clustering and affinity propagation [17]. For these methods, we represent utterances as a bag of words, and use cosine similarity as a similarity between pairs of utterances and similarity between utterances and clusters. One advantage of affinity propagation is that it determines the optimum number of clusters, hence we do not need to set it. For k-means clustering, we experimented with k=25,50,75, and 100, and found out the best results on the dev set for unsupervised intent classification were obtained at k=100, hence all reported results with k-means clustering are with 100 clusters. AP resulted in 50 clusters, AMR-freq resulted in 114 clusters, and AMR-entropy resulted in 123 clusters.

After we obtained a set of candidate clusters from each method, and asked a set of crowd users to read the most frequent 10 examples from each cluster. Our goal is to simulate how a dialog system designer would use the outcome of the clustering. We also presented the crowd workers, the most frequent 20 intent categories from the training set (with a very brief description, such as "Scheduling a meeting"), and asked them to mark the intent category that matches the majority of the intents of the users of the given 10 queries. Workers were instructed to mark "Other" if the intent of the cluster was not included in the multiple choice, and write down what they thought was the user intent in a text box in that case. Their description of the clusters provides insight into how a human interprets the grouping. For example, with k-means clustering, we noticed several clusters that were marked with a single keyword (but not an in-

	Intent Coverage			Example Coverage		
	Train	Dev	Test	Train	Dev	Test
Supervised	100%	93.2%	95.7%	100%	99.7%	99.8%
K-means	7.3%	13.6%	13.0%	60.8%	61.3%	59.8%
AP	7.3%	13.6%	13.0%	65.3%	66.4%	64.8%
AMR-freq	11.5%	21.6%	20.7%	77.4%	77.4%	77.4%
AMR-entropy	11.5%	21.6%	20.7%	77.4%	77.4%	77.4%

Table 2: Coverage of crowd annotated clusters, and accuracy of intent assignments.

	% Training	Accuracy Training	F (Dev)	F (Test)
Supervised	100%	100%	74.2%	77.8%
K-means	37.4%	58.1%	34.7%	33.5%
AP	12.2%	73.0%	35.7%	36.3%
AMR-freq	19.4%	54.8%	42.1%	41.7%
AMR-entropy	29.9%	72.1%	46.9%	46.6%

Table 3: Results from unsupervised training experiments.

tent), such as "Boston", consistently by many workers. In many cases, crowd workers were also able to recover the intent in their entry text, such as "find movie reviews", even though it was not presented amongst the multiple choices. We also provided a "No common intent" option, and instructed the workers to use it when they think the top 10 examples do not homogeneously belong to a single intent. We asked 5 crowd workers to annotate each cluster, and used to majority intent as the label of the cluster. If the majority crowd label was "No common intent", or there was no agreement amongst the crowd workers' labels, we did not include that cluster in the experiments.

4.2. Results and Discussion

In the first set of experiments, we checked if the clustering method was able to recover the intent grouping as designed by the domain experts. For that purpose, we measured coverage of intents and examples in the given data sets, by comparing the cluster labels with the manually annotated intents. Intent coverage is the percentage of intents in a given data set that were discovered by clustering, and example coverage is the percentage of utterances with intents discovered by clustering. Many of the intents appear only a few times (as also seen in Figure 2), hence, it is important to look at how frequently the discovered intents appear in the data sets, and example coverage is computed for that purpose. Table 2 shows intent and example coverage in the training, development, and test sets. According to these results, we get lower intent coverage with k-means clustering and AP, than with AMR-freq and AMR-entropy, which seem to result in the same set of intents. The example coverage between k-means and AP is different, showing that, while they resulted in the same number of intents, the ones found by AP appear more frequently in all subsets of the corpus.

While the intent and example coverage of the clusters are useful for comparing approaches, they do not tell us if the resulting clusters homogeneously belong to a single intent. Hence, we compute two other measures: percentage training and accuracy on the training. Percentage training is the percentage of examples covered by the clusters chosen by crowd workers, and shows if a given algorithm results in clusters that cover a majority of the training set. Accuracy training is the percentage of examples in the chosen clusters that are assigned to the correct intent category, as compared to the original manual annotations. The first two result columns of Table 3 show these measures for all methods. K-means resulted in highest percentage training, but the accuracy of these examples is lower, showing that the clusters were not homogeneous intent-wise. AP results in small clusters, with low coverage of the training set, but the accuracy of the resulting clusters is high. AMR-entropy results in clusters that have higher coverage of the training set than AMR-freq, as the further pruning of instances according to entropy result in more general cluster representative subgraphs, and the accuracy of these clusters is higher (72.1%).

To test the quality of the clusters and annotations obtained, we performed one more experiment, and used the selected examples to train multi-domain intent classifiers with icsiboost [18] and computed intent classification F-measure on the development and test sets. The last 2 columns of Table 3 show results of these experiments. Consistently with percentage training and accuracy training measures, the best method according to these experiments is AMR-entropy, which resulted in an F-measure of 46.6%, which is 28% higher than the better baseline with AP, that resulted in an F-measure of 36.3%.

In all the experimental results, we provided the upper bound results with supervised training, that requires a domain expert to examine all the training examples, determine intent categories, and manually annotate all of them. While, we cover intents of a majority of the examples (77.4%) with the proposed methods, further work may be needed to improve the annotations of the clusters, to achieve the F-measure of supervised training.

5. Conclusions

We propose a set of methods that aim to find common user intents amongst uncovered utterances addressed to a conversational interaction system, with the goal supporting future phases of dialog system design. Our analysis relies on semantic parsing and finding common semantic patterns in semantic parses of uncovered user utterances. In experiments that compare the proposed method with k-means clustering and affinity propagation, we show that we obtain higher coverage and accuracy with the semantic parsing based method. Moreover, the new method results in higher classification F-measure (showing over 25% relative improvement) in comparison to alternatives, in unsupervised intent classification experiments.

6. Acknowledgments

Authors would like to thank Lucy Vanderwende, Arul Menezes, and Chris Quirk for their support with NLPwin.

7. References

[1] G. Tur and R. D. Mori, Eds., *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*. New York, NY: John Wiley and Sons, 2011.

[2] G.Tur, A. Deoras, and D. Hakkani-Tür, "Detecting out-of-domain utterances addressed to a virtual personal assistant," in *Proceedings of Interspeech*, 2014.

[3] T. Lin, P. Pantel, M. Gamon, A. Kannan, and A. Fuxman, "Active objects: Actions for entity-centric search," in *Proceedings of World Wide Web Conference (WWW)*, Lyon, France, 2012, pp. 589–598.

[4] D. Hakkani-Tür, A. Celikyilmaz, L. Heck, and G. Tur, "A weakly-supervised approach for discovering new user intents from search query click logs," in *Proceedings of Interspeech*, 2013.

[5] G. Tur, D. Hakkani-Tür, and A. Chotimongkol, "Semi-supervised learning for spoken language understanding semantic role labeling," in *Automatic Speech Recognition and Understanding, 2005 IEEE Workshop on*. IEEE, 2005, pp. 232–237.

[6] P. Kingsbury, M. Marcus, and M. Palmer, "Adding semantic annotation to the Penn TreeBank," in *Proceedings of the HLT*, San Diego, CA, March 2002.

[7] S. Pradhan, W. Ward, K. Hacioglu, J. H. Martin, and D. Jurafsky, "Semantic role labeling using different syntactic views," in *Proceedings of the Association for Computational Linguistics 43rd annual meeting (ACL-2005)*, 2005.

[8] B. Liu, Y. Xia, and P. S. Yu, "Clustering through decision tree construction," in *Proceedings of the ninth international conference on Information and knowledge management (CIKM)*. ACM, 2000, pp. 20–29.

[9] A. Chotimongkol and A. I. Rudnicky, "Automatic concept identification in goal-oriented conversations," in *Proceedings of the ICSLP*, Denver, CO, September 2002.

[10] H. M. Meng and K.-C. Siu, "Semiautomatic acquisition of semantic structures for understanding domain-specific natural language queries," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 1, pp. 172–181, 2002.

[11] C. F. Baker, C. J.Fillmore, and J. B. Lowe, "The berkeley framenet project," in *Proceedings of the 17th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 1998, pp. 86–90.

[12] D. Das, N. Schneider, D. Chen, and N. A. Smith, "Probabilistic frame-semantic parsing," in *Proceedings of the HLT-NAACL*, Los Angeles, CA, June 2010.

[13] Y.-N. Chen, W. Y. Wang, and A. I. Rudnicky, "Unsupervised induction and filling of semantic slots for spoken dialogue systems using frame-semantic parsing."

[14] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider, "Abstract meaning representation for sembanking," in *Proceedings of the Linguistic Annotation Workshop*, 2014.

[15] Y.-N. Chen, W. Y. Wang, and A. I. Rudnicky, "Jointly modeling inter-slot relations by random walk on knowledge graphs for unsupervised spoken language understanding."

[16] L. Vanderwende, "NLPwin – an introduction," Microsoft Research, Tech. Rep. MSR-TR-2015-23, 2015.

[17] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[18] B. Favre, D. Hakkani-Tür, and S. Cuendet, "Icsiboost," http://code.google.come/p/icsiboost, 2007.