Towards Unsupervised Spoken Language Understanding: Exploiting Query Click Logs for Slot Filling

Gokhan Tur[1,2], *Dilek Hakkani-Tür*[1,2], *Dustin Hillard*[1], *Asli Celikyilmaz*[1]

[1]Microsoft Speech Labs, [2]Microsoft Research
Mountain View, CA, 94041

{gokhan.tur,dilek,hillard,asli}@ieee.org

Abstract

In this paper, we present a novel approach to exploit user queries mined from search engine query click logs to bootstrap or improve slot filling models for spoken language understanding. We propose extending the earlier gazetteer population techniques to mine unannotated training data for semantic parsing. The automatically annotated mined data can then be used to train slot specific parsing models. We show that this method can be used to bootstrap slot filling models and can be combined with any available annotated data to improve performance. Furthermore, this approach may eliminate the need for populating and maintaining in-domain gazetteers, in addition to providing complementary information if they are already available.

Index Terms: spoken language understanding, slot filling, data mining, named entity extraction, unsupervised learning

1. Introduction and Motivation

Spoken language understanding (SLU) in human/machine spoken dialog systems aims to automatically identify the user's goal-driven intents, as expressed in natural language and extract associated arguments or slots [1]. An example utterance with semantic domain, intent and slot annotations is shown in Table 1. The system can then decide on the most appropriate next action to take, according to the domain specific semantic template.

The state-of-the-art approach for training SLU models relies on supervised machine learning methods. An exhaustive survey on intent determination and slot filling methods can be found in [1]. These models require a large number of in-domain sentences which are semantically annotated by humans, a very expensive and time consuming process. Additionally, SLU models require in-domain gazetteers (such as city, movie, actor, or restaurant names) for better generalization. However, populating and maintaining these gazetteers, which are typically very dynamic and need constant maintenance, requires a significant amount of manual labor and typically semi-automated knowledge acquisition techniques are employed.

Given that building a domain-independent SLU system, or a system which understands anything in any domain, is still an unsolved problem, the approach taken in this paper is to mine web search query click logs in order to quickly bootstrap SLU models into slot filling for target domains. Our approach assumes that a semantic template exists for a given target domain. This bootstrap model can then be improved once annotated training data becomes available.

While similar approaches have successfully been applied to domain detection and intent determination tasks, as well as to automatically populating domain gazetteers, to the best of

Utterance	*show me recent action movies by spielberg*
Domain:	Movie
Intent:	Find_Movie
Genre:	*action*
Date:	*recent*
Director:	*spielberg*

Table 1: An example utterance with semantic annotations.

our knowledge, there is no previous work on exploiting mined data directly for slot filling. For instance, Li et al. used query click logs to determine the domain of the query (typically not in natural language), and then inferred the class memberships of unlabeled queries from those of the labeled queries using the urls the users clicked [2]. For example, the queries of two users who clicked on the same url (such as, www.hotels.com) are assumed to belong to the same domain ("hotels" in this case). In our earlier work, we extended this idea in order to use the noisy supervision obtained from query click information into the semi-supervised label propagation algorithm by sampling high-quality query click data mined from query logs [3]. This resulted in a 20% relative reduction in the domain detection error rate for SLU in a semi-supervised setup.

Most related to slot filling, Singh et al. proposed a semi-supervised learning method to improve the named entity extraction model used for text advertisements [4]. Starting from an existing semantic structure, they trained statistical models to semantically parse short text advertisements by clustering other non-structured elements using topic models. Li et al. exploited query click logs leveraging domain-specific structured information for web query tagging [5]. They have built semi-supervised models using these derived labels. Wang et al. leveraged structured HTML lists to automatically generate gazetteers [6]. These gazetteers were then used to improve the slot filling models. Liu et al., extending this approach, proposed automatically populating gazetteers from the web queries to be used in slot filling [7]. Using a seed gazetteer, they mined the query click logs to expand it using a generative model. They learned target websites where users clicked based on the seed gazetteer entries. For example, www.imdb.com/title is a candidate website for movie names. Then they added other queries hitting the same website with high frequency as new gazetteer candidates, and then used statistical methods to weigh them. The contextual words (such as in *cast of avatar* or *when was the movie as good as it gets released*) were then stripped out using the existing seed gazetteer entries.

The proposed approach in this paper is a natural extension of the work of Liu et al.: Instead of stripping out the context words found in candidate entities, they are used for training slot filling models. The clicked URLs indicate entities, thereby providing us with the context needed for semantic parsing. This is implicitly annotated training data for bootstrapping or improving slot filling models and may eliminate the need for maintaining gazetteers. However, our approach also complements earlier efforts and can be used with any manually crafted or automatically populated gazetteers.

In the next section, we present our slot filling system. Then in Section 3, we present the proposed approach in detail. Then in Section 4, we present the experiments and results.

2. Semantic Parsing

Following the state-of-the-art approaches for slot filling [8, 9, among others], we use discriminative statistical models, namely conditional random fields, (CRFs) [10], for modeling. More formally, slot filling is framed as a sequence classification problem to obtain the most probable slot sequence:

$$\hat{Y} = \underset{Y}{\operatorname{argmax}}\, p(Y|X)$$

where $X = x_1, ..., x_T$ is the input word sequence and $Y = y_1, ..., y_T, y_i \in C$ is the sequence of associated class labels, C.

CRFs are shown to outperform other classification methods for sequence classification [1], since the training can be done discriminatively over a sequence. The baseline model relies on word n-gram based linear chain CRF, imposing the first order Markov constraint on the model topology. Similar to maximum entropy models, in this model, the conditional probability, $p(Y|X)$ is defined as [10]:

$$p(Y|X) = \frac{1}{Z(X)} exp\left(\sum_k \lambda_k f_k(y_{t-1}, y_t, x_t)\right)$$

with the difference that both X and Y are sequences instead of individual local decision points given a set of features f_k (such as n-gram lexical features, state transition features, or others) with associated weights λ_k. $Z(X)$ is the normalization term. After the transition and emission probabilities are optimized, the most probable state sequence, $\hat{Y}$, can be determined using the well known Viterbi algorithm.

The in-domain gazetteer entries are also treated in the same framework as any additional features. While the gazetteer entries are marked without considering context, the slot filling model aims to decide on whether they constitute a slot. More formally, an extra set of features is created for each gazetteer:

$$f_{G,c}(y_{t-1}, y_t, x_t) = \delta(x_t \in G)\delta(y_t = c)$$

where $\delta(x_t \in G)$ is for querying membership (B, I, or O) for that gazetteer, and c is the slot filling class.

In this study, we follow the popular IOB (in-out-begin) format in representing the data as shown below. For the gazetteer features, instead of treating the entry as a single token (such as *gone_with_the_wind*), as done in [9], this is considered to be an additional feature for each word for the linear chain CRF. This is important since gazetteer entries may overlap. For instance, if the city gazetteer is used in a flight information system, the following would be the corresponding data:

flights	from	boston	to	new	york	today
O	O	B-city	O	B-city	I-city	O
↓	↓	↓	↓	↓	↓	↓
O	O	B-dept	O	B-arr	I-arr	B-date

3. Approach

Currently, the state-of-the-art slot filling approaches exploit both context and gazetteer information, as explained above. These approaches need large amounts of semantically annotated natural language data to model the context. Humans do not encounter any problem in understanding that *inception* is a movie name if used in context, such as *find me movies similar to inception*, even if this is the first time they have heard about this movie. However, using context requires sample utterances where slots are marked. Furthermore, constant maintenance of the in-domain gazetteers is required as new entries are introduced everyday, and they will fail in cases where users do not utter the exact name, which is always an issue with longer entries (e.g., *the frog and the princess* for the movie *the princess and the frog* or *beautiful life* for the movie *life is beautiful*). Another big problem is that most gazetteers contain thousands or maybe millions of entries. As an example, in a standard knowledge-base there are more than 100,000 movie names. In most real-life scenarios, the entries in these gazetteers are also highly ambiguous. For example the word *washington* can be a city, state, movie, restaurant, or person name. This problem of name tagging for SLU has usually been overseen in earlier work (such as in ATIS [13]), and simple table lookup-based methods have been used to mark named entities in user utterances.

3.1. Query Click Logs

To alleviate these problems, the proposed approach relies on exploiting an abundant set of web query click logs, which pair web search queries with their click information. Some example queries with resulting clicks are as follows:

Query:	*who played the count of monte cristo*
URL:	www.imdb.com/title/tt0047723/fullcredits
Query:	*avatar soundtrack*
URL:	en.wikipedia.org/wiki/Avatar_soundtrack

While this is very valuable data waiting to be mined for language understanding, it is not generally this straightforward, since most queries are just keywords (instead of natural language sentences) and include typos, and because the implicit supervision via click information is very noisy. However, since there are millions of such pairs, high precision is more important than missing some of the useful queries.

3.2. Mining Data

The proposed approach for mining the clicked URL data consists of two stages: (i) Detection of target URLs and (ii) Mining corresponding data. In the first stage, the target phrases are determined for each slot type. As an example, for the movies domain, this may correspond to detecting websites for each of the movie names. For this step, our approach is similar to Lin et al.'s approach [7]; however, instead of relying on a seed training set, we rely on query frequencies. We start with a target site (such as imdb.com/title for the movie names). This can be manually determined or easily retrieved using a few known examples. Then, it is reasonable to assume that the most frequent phrase hitting a given target URL with frequency above a given threshold will be the main entity. The output of this first stage is a list of entities and their target URLs. More formally,

$$\hat{m} = \underset{m}{\operatorname{argmax}}\, freq_{url}(m)$$

where m is a candidate phrase for each entity and $freq_{url}(m)$ is the frequency of query m resulting in click to the website

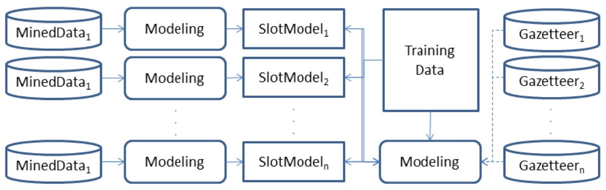

Figure 1: Conceptual figure for semi-supervised slot filling models exploiting mined data optionally using available gazetteers.

url. For example, www.imdb.com/title/tt047723 can be the target URL for the movie *the count of monte cristo*.

Once the entities and associated URLs are determined, in the second stage, all queries hitting those links are extracted if they include the *exact* name of the entity. This naturally provides automated annotation of queries since the entity name is known. For example, at the end of this process, the data we have for the movie names looks like the following:

review of **the hand**
mad men *season one synopsis*
who directed the original **step up** *movie*
four lions *miami showtimes*

3.3. Use for Slot Filling

The mined data for each slot type, s, is then used to train statistical models. Similar to semantic parsing, CRF is employed in this stage, and linear models are trained using the mined data to output $p_s(Y^s|X)$, where $Y^s = y_1^s, ..., y_T^s, y_i \in C^s$ is the tag sequence for that slot type:

$$\hat{Y^s} = \underset{Y^s}{\operatorname{argmax}}\, p(Y^s|X)$$

One advantage of using web query data is that it is always up-to-date (for example, the movies domain includes even the movies which are not yet released) and naturally incorporates the entities, eliminating the need for maintaining a gazetteer. However, these samples are in *query* language, not in *natural* language.

Depending on whether in-domain annotated data is available or not, the bootstrap models trained using this data for each slot can be used in two ways:

- Case 1 (Unsupervised): In cases where there is absolutely no in-domain annotated data, the mined data can be used to build bootstrap slot filling CRF models. For example, the Viterbi algorithm can be used to detect the most probable slot sequence given the model outputs.

$$\hat{Y} = \underset{Y}{\operatorname{argmax}}\, p(Y|Y^s) \times p(Y^s|X)$$

- Case 2 (Semi-supervised): When there is some in-domain annotated data available, the slot specific model can provide extra features to improve performance. In such a case, we build cascaded models, first CRF entity models trained on query logs and a following CRF trained on its output and available annotated natural language data, as depicted in Figure 1. More formally, an extra set of features is created from each unsupervised entity model:

$$f_{y^s,c}(y_{t-1}, y_t, x_t, \hat{y_t^s}) = \delta(\hat{y_t^s} = y^s)\delta(y_t = c)$$

where $\delta(\hat{y_t^s} = y^s)$ is determined by the result of the Viterbi algorithm for the word x_t using the unsupervised model on the training set for slot s, as explained above, and c is the slot filling class.

	Training Data	Test Data	Mined Data
# sentences	2,407	289	2,020,429
# words	20,934	2,542	15,211,944
# slots	3,358	393	2,020,429
# movie names	1,254	156	2,020,429

Table 2: Characteristics of the data used in experiments.

In both cases, the in-domain gazetteers can be exploited as additional features when available, since they provide complementary information.

4. Experiments and Results

In order to demonstrate how query click logs are exploited for slot filling, a proof-of-concept study is performed. The understanding domain *movies* is chosen, where the users present queries about various movies, such as *who is the director of avatar*, *show me the movies with academy awards*, or *when is the next harry potter gonna be released*. The natural language data set from this domain consists of about 2,700 sentences, with about 300 sentences reserved for testing. This includes about 3,750 slots (about 1,400 movie names) from 21 slot types (such as mpaa rating or release date) as shown in Table 2.

We only focus on bootstrapping the slot model for movie names. To compile a gazetteer for the movie name slot, we dump the list of all movies from freebase.com, a total of 148,738 movies. Since using such a gazetteer results in very low precision, we filter out the short entries with low inverse document frequency (IDF) (such as *up*), resulting in 91,442 movies names[1]. This improves the precision from 11% to 85%.

The website detected as the target base URL is imdb.com/title. The number of candidate Bing queries resulting in clicks to this site is about 6 million. Out of these, the number of mined queries which has the exact movie name as the target website is about 2 million. We ended up compiling about 1,000 times more data than the small amount of manually annotated data. However, note that these are mostly keywords intended for web search. Hence, the model trained on these examples[2] resulted in only slightly better performance than one obtained using the filtered gazetteer as shown in Table 3.

It should be noted that, using only the word n-grams in the small amount of training data as features for training the CRF-based slot filling model beats both using only gazetteers or using only the mined slot model. As expected, using filtered gazetteers as additional features improves the performance significantly from 69.72% to 71.52%, while using the full gazetteer results in a degradation in performance. Instead of the gazetteer, using mined model for movie names results in similar overall slot filling performance.

Our goal in this work is to investigate the effectiveness of the mined slot model, compared to these baseline performance figures. If its output is used as an additional feature for CRF-based slot filling, as explained above, this results in 2% better absolute performance than using a filtered gazetteer (79.14% compared to 77.20%[3]) for parsing movie names, however it did

[1]A companion submission focuses on our approach to this process.
[2]We use the CRF++ toolkit in this study.
[3]This is a statistically significant improvement according to z-test with 0.95 confidence interval

	Movie Name F-Measure	All Slots F-Measure
(1) Only full gazetteer	18.07%	14.02%
(2) Only filtered gazetteer	60.91%	30.83%
(3) Only mined model	51.65%	31.28%
(4) Manually annotated data	72.51%	69.72%
(1) + (4)	71.52%	68.56%
(2) + (4)	77.20%	71.52%
(3) + (4)	79.14%	71.72%
(2) + (3) + (4)	78.88%	71.86%

Table 3: Slot filling performances under various conditions

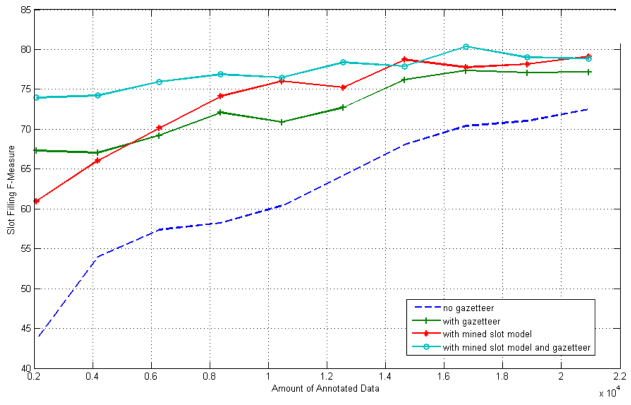

Figure 2: Performance of semi-supervised models for the movie name slot with and without any gazetteer information exploiting mined data.

not result in better performance (78.88% compared to 79.14%) when combined with the filtered gazetteer.

To provide an additional perspective on these results, the learning curves with different sizes of manually annotated data are provided in figures 3 and 2 for the overall performance of movie name slot filling and parsing . Throughout the curves, using the mined slot model results in comparable or better performance than using the filtered gazetteer, except when there is very little data. Furthermore, combination of gazetteer and slot model features usually results in even better performance. For example, when there are only 1456 sentences, the movie name parsing performance increases from 72.73% to 78.37%. However, less improvement is observed as the amount of annotated data increases, as expected.

5. Conclusions and Future Work

In this paper, we have presented a data mining based approach to bootstrap or improve spoken language understanding models for slot filling, and demonstrated its use. Our approach relies on exploiting query click logs for finding queries hitting the webpages strongly associated with slot entities and using contextual information to train slot specific models. These models may then be used as additional features during slot filling. We have demonstrated that this approach may not only eliminate the need to maintain slot specific gazetteers, but can also further improve system performance when combined. Our work can be considered as a promising first step towards developing fully unsupervised (or minimally supervised) SLU models for target domains.

While this approach has been proposed in the context of

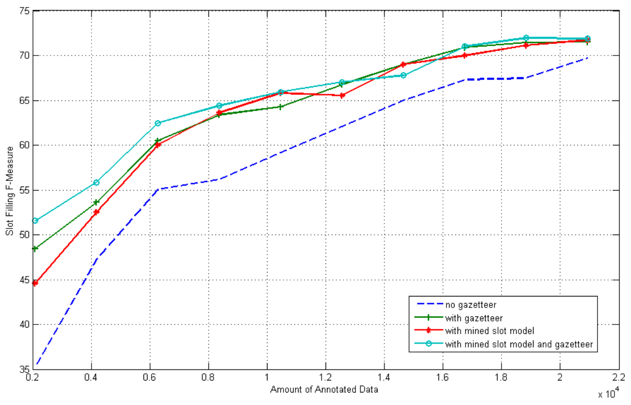

Figure 3: Performance of semi-supervised slot filling models with and without any gazetteer information exploiting mined data.

SLU, the findings can easily be generalized to other domains such as query tagging or information extraction.

Our future work aims to pursue this study further towards extending to other slots where the URL pattern is not that obvious and combining with intent determination in a conversational understanding framework.

6. References

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

[2] X. Li, Y.-Y. Wang, and A. Acero, "Learning query intent from regularized click graphs," in *Proceedings of the ACM SIGIR*, Singapore, 2008.

[3] D. Hakkani-Tür, L. Heck, and G. Tur, "Exploiting query click logs for utterance domain detection in spoken language understanding," in *Proceedings of the ICASSP*, Prague, Czech Republic, 2011.

[4] S. Singh, D. Hillard, and C. Leggetter, "Minimally-supervised extraction of entities from text advertisements," in *Proceedings of the HLT-NAACL*, Los Angeles, CA, June 2010.

[5] X. Li, Y.-Y. Wang, and A. Acero, "Extracting structured information from user queries with semi-supervised conditional random fields," in *Proceedings of the ACM SIGIR*, Boston, MA, 2009.

[6] Y.-Y. Wang, R. Hoffmann, X. Li, and J. Szymanski, "Semi-supervised learning of semantic classes for query understanding: From the web and for the web," in *Proceedings of the CIKM*, Hong Kong, 2009.

[7] J. Liu, X. Li, A. Acero, and Y.-Y. Wang, "Lexicon modeling for query understanding," in *Proceedings of the ICASSP*, Prague, Czech Republic, 2011.

[8] Y.-Y. Wang and A. Acero, "Discriminative models for spoken language understanding," in *Proceedings of the ICSLP*, Pittsburgh, PA, September 2006.

[9] C. Raymond and G. Riccardi, "Generative and discriminative algorithms for spoken language understanding," in *Proceedings of the Interspeech*, Antwerp, Belgium, 2007.

[10] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the ICML*, Williamstown, MA, 2001.

[11] M. Jeong and G. G. Lee, "Exploiting non-local features for spoken language understanding," in *Proceedings of the ACL/COLING*, Sydney, Australia, July 2006.

[12] S. Sarawagi and W. W. Cohen, "Semi-markov conditionalrandom fields," in *Proceedings of the NIPS*, 2004.

[13] P. J. Price, "Evaluation of spoken language systems: The ATIS domain," in *Proceedings of the DARPA Workshop on Speech and Natural Language*, Hidden Valley, PA, June 1990.