EXPLOITING QUERY CLICK LOGS FOR UTTERANCE DOMAIN DETECTION IN SPOKEN LANGUAGE UNDERSTANDING

Dilek Hakkani-Tür *Larry Heck* *Gokhan Tur*

Speech Labs | Microsoft Research
1310 Villa Street, Mountain View, CA
dilek@ieee.org, larry.heck@microsoft.com, gokhan.tur@ieee.org

ABSTRACT

In this paper, we describe methods to exploit search queries mined from search engine query logs to improve domain detection in spoken language understanding. We propose extending the label propagation algorithm, a graph-based semi-supervised learning approach, to incorporate noisy domain information estimated from search engine links the users click following their queries. The main contributions of our work are the use of search query logs for domain classification, integration of noisy supervision into the semi-supervised label propagation algorithm, and sampling of high-quality query click data by mining query logs and using classification confidence scores. We show that most semi-supervised learning methods we experimented with improve the performance of the supervised training, and the biggest improvement is achieved by label propagation that uses noisy supervision. We reduce the to error rate of domain detection by 20% relative, from 6.2% to 5.0%.

***Index Terms*—** Spoken language understanding, semi-supervised learning, web search queries, label propagation, domain detection.

1. INTRODUCTION

In the last decade, a variety of practical goal-oriented spoken language understanding (SLU) systems have been built for limited domains. Three key tasks in such targeted understanding applications are domain classification, intent determination and slot filling [1]. Domain classification is often completed first in SLU systems, serving as a top-level triage for subsequent processing. This modular design approach has the advantage of flexibility; specific modifications (e.g., insertions, deletions) to one domain class can be implemented without requiring changes to other domains classes. Also, such an approach often yields more focused understanding in each domain since the intent determination only needs to consider a relatively small set of intent classes over a single (or limited set) of domains. Such an approach can also be extended to hierarchical SLU models with multiple levels of domains and subdomains. For example, a SLU system in the travel assistance domain may hierarchically represent related subdomains such as flight reservations, hotel booking, and car rental domains.

Similar to intent determination, triage domain detection systems are often framed as a classification problem. More formally, given a user utterance or sentence x_i, the problem is to associate a set $y_i \subset C$ of semantic domain labels with x_i, where C is the finite set of domains covered. To perform this classification task, the class with the maximum conditional probability, $p(y_i|x_i)$ is selected. Usually, supervised classification methods are used to estimate these conditional probabilities. Each domain class is trained from a set of labeled utterances.

Collecting and annotating naturally spoken utterances to train these domain classes if often costly, representing a significant barrier to deployment both in terms of effort and finances. However, it may be possible to overcome this hurdle by leveraging the abundance of *implicitly labeled* web search queries in search engines. Large-scale engines such as Bing or Google log more then 100M search queries per day. Each query in the log has an associated set of URLs that were clicked after the users entered the query. This user click information could be used to infer domain class labels and, therefore, provide (noisy) supervision in training domain classifiers. For example, the queries of two users who click on the same URL (such as, `http://www.hotels.com`) are probably from the same domain ("hotels" in this case).

Previous work on web search query intent classification benefited from the use of query click logs for improving query intent classification. For example, [2] used query click logs for determining the intent of the query (typically not in natural language) and inferred class memberships of unlabeled queries from those of the labeled queries. They formed a bipartite graph of the queries and URLs the users clicked, then transferred labels from queries to URLs and other queries using the label propagation algorithm [3, 4]. Note that these approaches focused on transferring the labels without using the lexical content of the query. The aim in this work is to extend previous work by (a) directly using lexical features of the query in the similarity measure (b) leveraging noisy labels inferred from URL clicks. Furthermore, we show how web search queries can be used to improve the domain detection of more complex queries, specifically *naturally* spoken utterances.

We assume that the clicked URL category can be assigned as the domain label of the user query. For example, we assign the label "hotels" to the user query "Holiday Inn and Suites" when the user has clicked on `http://www.hotels.com`. However most click data is noisy and has low frequency. Hence, we also try to estimate successful clicks by mining query click logs to gather the set of URLs the people who searched by using the same exact query. Recently, [5] studied several features, such as query entropy, dwell times and session length for mining high-quality clicks, and showed that query entropy is the best single indicator feature for a successful click. In [6], Hassan et al. studied user action patterns and dwell time to estimate successful search sessions. We follow a similar approach, and use query entropy and frequency, and integrate other features from domain detection, such as the probabilities assigned by a domain detection model trained on labeled data to sample high quality clicks both for adding as examples to the training set, and to pre-sample the data for use in the label propagation.

Furthermore, we use the label propagation algorithm to transfer domain annotations from labeled natural language (NL) utterances

978-1-4577-0539-7/11/$26.00 ©2011 IEEE

to unlabeled web search queries. We also consider the click information as noisy supervision, and incorporate the domain label we extract from the clicked URL category into the label propagation algorithm.

In the next section, we first describe the criteria we use for mining query click logs, the semi-supervised algorithms for estimating domain labels for the search engine queries, and extension of the label propagation algorithm. Then, in Section 3, we present experiments on a set of natural language utterances from a spoken dialog system application using these methods.

2. APPROACH

Query click data includes logs of search engine users' queries and the links they click from a list of sites returned by the search engine. Previous work has shown that click data can be used to improve search decisions [7]. However, most click data is very noisy, and includes links that were clicked on almost randomly. We first propose a set of measures, some of which were also used in improving search, to sample queries and domain labels from the clicked URLs for use in domain detection. Then we include supervision from the noisy user clicks into the label propagation algorithm that aims to transfer domain labels from labeled examples to the sampled search queries.

2.1. Mining Query Click Logs

We extract a set of queries, whose users clicked on the URLs that are related to our target domain categories. We then mine the query click logs to download all instances of these search queries and the set of links that were clicked on by search engine users who entered the same query. We use the following criteria to sample a subset of these queries:

- **Query Frequency:** refers to the number of times a query has been searched by different users in a given time frame. The motivation for using this feature is that in spoken dialog systems, users may ask the same things as web search users, hence adding frequent search queries to the domain detection training set may help to improve its accuracy.
- **Query (Click) Entropy:** aims to measure the diversity of the URLs clicked on by the users of a query q, and is computed as
$$E(q) = -\sum_{i=1}^{n} P(U_i) ln P(U_i)$$
where $U_i, i = 1, ..., n$ are the set of URLs clicked by the users of query q, and $P(U_i)$ is the normalized frequency of the URL U_i,
$$P(U_i) = \frac{F(U_i)}{\sum_{i=1}^{n} F(U_i)}$$
where $F(U_i)$ is the number of times the URL U_i is clicked. Low click entropy may be a good indicator of the correctness of the domain category estimated from the query click label.
- **Query Length:** refers to the number of words in the query. The number of words in a query is usually a good indicator of such NL utterances, and search queries that include natural language utterances instead of simply a sequence of keywords may be more useful for training data in SLU domain classification.

We add the sampled queries with the domain labels estimated from the clicked URLs to the labeled training set, or use these sampled examples for semi-supervised learning approaches described below.

2.2. Semi-Supervised Learning Approaches

In this study, we compared two established semi-supervised learning methods, namely self-training and label propagation. Furthermore we propose an extension of the label propagation algorithm that also exploits the domain information obtained from the URLs users have clicked on.

2.2.1. Self-Training

Self-training involves training an initial classifier from the existing manually labeled examples, and using it to automatically assign labels for a larger set of unlabeled examples. Then the examples which were assigned classes with high posterior probabilities are added to the training data. This approach has been widely used in many language and speech processing tasks [8, 9, among others]. It has been shown that, when the training set is very limited, self-training improves the performance however, typically, further iterations are not effective.

2.2.2. Label Propagation

Label propagation (LP) is a graph-based, iterative algorithm commonly used for semi-supervised learning [3, 4, among others]. The main idea of this algorithm is to propagate the labels through the dataset along the high density areas defined by the unlabeled examples. This is conceptually similar to the well-known k-Nearest-Neighbor (kNN) classification algorithm.

LP is intuitively a better semi-supervised learning method for our task, since, in theory, it enables the classifier to see samples which have no common phrases to the training set. For example, if the training set has the phrase "hotel" but not "suites", the example query above "holiday inn and suites" may propagate the label to another query, say "ocean-view suites", which will propagate it to others. In self-training, multiple iterations would have the same effect, but since it results in overfitting, LP is usually a better choice.

The LP algorithm has been proven to converge and has a closed form solution for easier implementation. More formally, let $\{(x_1, y_1),$
$..., (x_l, y_l)\}$ be the labeled data set, where $Y_L = y_1, ..., y_l \in 1, ..., |C|$ for $|C|$ classes. Let $\{(x_{l+1}, y_{l+1}), ..., (x_{l+u}, y_{l+u})\}$ be the unlabeled data set, where $Y_U = \{y_{l+1}, ..., y_{l+u}\}$ is unknown. The samples $X = \{x_1, ..., x_{l+u}\} \in R^D$ are from a D-dimensional feature space.

The goal of label propagation is then to estimate Y_U from X and Y_L. As the first step, a fully connected graph is created using the samples as nodes. The edges between the nodes, w_{ij} represent the Euclidean distance with a control parameter σ:

$$w_{ij} = exp\left(- \frac{d_{ij}^2}{\sigma^2} \right) = exp\left(- \frac{\sum_{d=1}^{D} (x_i^d - x_j^d)^2}{\sigma^2} \right)$$

where x_i^d is the value of the d^{th} feature of sample x_i. This graph is then represented using a $(l+u) \times (l+u)$ probabilistic transition matrix T:

$$T_{ij} = P(j \to i) = \frac{w_{ij}}{\sum_{k=1}^{l+u} w_{kj}}$$

A corresponding $(l+u) \times |C|$ matrix is also defined for the labels. The labels for the unlabeled samples are initially randomly set (since they are not important, as shown in the convergence solution below).

The algorithm then iterates as follows:

1. Propagate labels 1 step: $Y \leftarrow TY$

2. Normalize the rows of Y to maintain a probability distribution
3. *Clamp*, i.e., restore the labels of the labeled data

It has been shown that this algorithm converges to a fixed solution:

$$Y_U = (I - \bar{T}_{uu})^{-1}\bar{T}_{ul}Y_L$$

where $(\bar{T})$ is the row normalized matrix of T, such that $\bar{T}_{ij} = T_{ij}/\sum_k T_{ik}$ and $\bar{T}_{ul}$ and $\bar{T}_{uu}$ are the bottom left and right parts of $\bar{T}$, obtained by splitting $\bar{T}$ after the l^{th} row and column into four sub-matrices. See [3] for more details.

In this work, we use the natural language utterances labeled with domain categories as the labeled example set, and the sampled set of search queries as the unlabeled examples, and estimate their labels using the LP algorithm. We then added the unlabeled examples that are assigned a domain category with a high posterior probability by LP and add them to the labeled training data set (*Method 1*).

2.3. Label Propagation with Noisy Supervision

Since the URLs that the users clicked on also provide a noisy label for each query, we check the agreement between the domain category assigned to each example by LP and the domain category of the clicked URL. We added only those examples with high probability labels from LP, that agree with the click label to the training data set (*Method 2*).

As an alternative, the category of the clicked URL can also be used as a feature in the representation of a query. This would allow for propagation of labels between queries that have the same click labels with a higher weight in LP. This is inspired from ideas on extending established feature transformation approaches, namely the supervised latent Dirichlet allocation (sLDA) [10] incorporating the correct labels and factored latent semantic analysis (fLSA) [11] supporting the use of additional features.

In this work, more specifically, we include $|C|$ binary features for each domain, resulting in a $D + |C|$-dimensional feature space and assigned a value of 1 to the feature corresponding to the click label of the query, and 0 to all the others. This results in a straightforward extension of the computation of the Euclidean distance with noisy supervision:

$$w_{ij} = exp\left(- \frac{\sum_{d=1}^{D+|C|}(x_i^d - x_j^d)^2}{\sigma^2} \right)$$

where x_i^{D+k} is the binary feature indicating the click of the URL for the k^{th} domain. We then ran the LP and select the top scoring examples for each domain to add to the classification training data (*Method 3*).

3. EXPERIMENTS AND RESULTS

Similar to prior work on other utterance classification tasks, such as dialog act tagging [12] and intent determination [13], our approach relies on using icsiboost[1], an implementation of the AdaBoost.MH algorithm, a member of the boosting family of classifiers [14]. As features, we use word unigrams, bigrams and trigrams as extracted from the training set. No feature normalization is performed to tag named entities (such as hotel or airline names in a travel system) as the system must learn the domain from the content words instead of entity types since their annotation is typically non-trivial and noisy.

Data Set	Number of examples
Labeled training utterances	3,701
Labeled test utterances	1,014
Web search queries	2,024,550
Queries with all instances	5,000

Table 1. Data sets used in the experiments.

Training Set	Avg. ER
Labeled Examples (LE)	6.2%
LE + 500 random examples	6.1%
LE + 1000 random examples	5.7%
LE + 5000 random examples	6.1%
LE + 500,000 random examples (accd. to prior)	9.9%
LE + all examples	16.6%

Table 2. Error rate, averaged over 3 random orderings, when using query and click label pairs (QCL) in addition to the labeled examples.

3.1. Data Sets and Experiment Set-up

We used a set of over 4,000 natural language utterances from a spoken dialog system application. These utterances belonged to five different domain categories. Furthermore, we downloaded over 2 million queries from Bing web search logs, and the URLs clicked on by users. For a 5,000 query subset of these, we mined the web logs to extract all instances of the same exact query and the URLs clicked on by all users. Table 1 summarizes the data sets used in our experiments.

In each experiment, we split the test set into two and performed two experiments where in each experiment, we used only one half of the test for tuning the optimum number of boosting iterations and used the other half as the previously unseen test set. We averaged the performance figures on the unseen test sets from these two experiments.

For evaluating each method, we compute the error rate (ER), which is the number of examples for which the most probable domain category disagrees with manual annotation, divided by the total number of examples.

3.2. Baselines

As the baseline experiment, we randomly selected the queries which resulted in clicks to target domains. These queries are then added to the training set with their click labels. Table 2 shows results with this baseline approach. Note that, the experiments with 500, 1,000, and 5,000 unlabeled examples are repeated three times, each time with a different random subset, and the average of the error rates from these experiments is reported. Adding a small set of random examples to the set of labeled examples for training improves domain detection ER slightly, however when all the queries are added to the training set, the error rate increases significantly. We also downsampled the query set according to the prior probability of each domain label in the labeled training set. This results in around 500,000 search query examples, however the error rate is still significantly higher when these examples are included in training.

[1] http://code.google.com/p/icsiboost/

Criterion	Avg. ER
Labeled Examples (LE)	6.2%
Query Frequency	5.2%
Query Entropy	5.2%
Query Length	6.0%

Table 3. Domain detection error rates when 1,000 examples are sampled with various methods and are added to the training set.

Approach	Avg. ER
Labeled examples (LE)	6.2%
LE + 1000 random examples	5.7%
Self-Training	5.8%
LE + top LP (*Method 1*)	5.4%
LE + top LP, agreeing (*Method 2*)	5.7%
LE + top noisy LP (*Method 3*)	**5.0%**

Table 4. Domain detection error rates with the proposed label propagation methods.

3.3. Mining Query Click Logs

In order to select 1,000 examples from the search queries, we then employ each of the criteria described in Section 2.1. The results of experiments using the sampled examples as additional training and as unlabeled examples for LP are in Table 3.

For, all methods discussed above, for sampling queries for domain detection help in decreasing the error rate, with query frequency and query click entropy resulting in the most improvement. The examples sampled according to query length reduce the error rate only slightly. A combination of various similar criteria for mining reliable and high-quality query and click data seems to be a promising research direction for our purpose.

3.4. Semi-Supervised Learning

Results with various semi-supervised learning methods are presented in Table 4. In each experiment, 1,000 examples were sampled randomly for semi-supervised learning. Each experiment is repeated twice, with a different random set of initial 1,000 examples, and the average ER from these experiments are reported.

The self-training learning approach decreases the error rate compared to the baseline approach (from 6.2% to 5.8%), however, note that randomly adding 1,000 examples actually helps more, decreasing the error rate to 5.7%.

For LP experiments, the top 500 examples from the output of LP are added to the training set. According to these results, the agreement method (*Method 2*) is not very useful, decreasing the error rate to 5.7%. Applying LP without requiring agreement resulted in better performance of 5.4% error rate.

Incorporating noisy labels into LP results in the best performance for all approaches, resulting in an error rate of 5.0%, a reduction of 20% relative to the initial error rate of 6.2% which was obtained using only manually labeled examples in the training set.

4. CONCLUSIONS

We presented several methods for mining queries from search engine query logs and used them in domain detection for spoken language understanding. We demonstrated that search engine queries can be labeled with categories estimated from the URLs clicked on by their users and other user behavior measures extracted from query logs can be used for sampling queries for use in domain detection. Furthermore, we use raw queries with and without their noisy labels in semi-supervised learning and reduce domain detection error rate by 20% relative to supervised learning using only manually labeled examples.

Future work in this area includes investigating new methods for high quality query and click selection, incorporating these sampling methods with label propagation. Another future direction is extending this work towards other spoken language processing tasks, such as voice search.

5. REFERENCES

[1] R. De Mori, F. Bechet, D. Hakkani-Tür, M. McTear, G. Riccardi, and G. Tur, “Spoken language understanding for conversational systems,” *Signal Processing Magazine Special Issue on Spoken Language Technologies*, vol. 24, no. 3, pp. 50–58, May 2008.

[2] X. Li, Y.-Y. Wang, and A. Acero, “Learning query intent from regularized click graphs,” in *Proceedings of SIGIR’08: the 31st Annual ACM SIGIR conference on Research and Development in Information Retrieval, Association for Computing Machinery, Inc.*, Singapore, July 2008.

[3] X. Zhu and Z. Ghahramani, “Learning from labeled and unlabeled data with label propagation,” Tech. Rep. CMU-CALD-02-107, CMU CALD technical report, 2002.

[4] X. Zhu, *Semi-Supervised Learning with Graphs*, PhD dissertation, Carnegie Melon University, 2005.

[5] A. Singla and Ryen W. White, “Sampling high-quality clicks from noisy click data,” in *Proceedings of WWW2010*, Raleigh, North Carolina, USA, 2010.

[6] A. Hassan, R. Jones, and K. Klinkner, “Beyond DCG: User behavior as a predictor of a successful search,” in *In the Proceedings of the ACM Conference on Web Search and Data Mining (WSDM 2010)*, New York City, USA, February 2010.

[7] E. Agichtein, E. Brill, and S. Dumais, “Improving web search ranking by incorporating user behavior information,” in *Proceedings of SIGIR*, Seattle, WA, USA, 2006, pp. 19–26.

[8] R. Mihalcea, “Co-training and self-training for word sense disambiguation,” in *Proceedings of the CoNLL*, Boston, MA, May 2004.

[9] D. McClosky, E. Charniak, and M. Johnson, “Effective self-training for parsing,” in *Proceedings of the HLT-NAACL*, New York, NY, July 2006.

[10] D. Blei and J. McAuliffe, “Supervised topic models,” in *Proceedings of the NIPS*, 2008.

[11] R. Serafin and B. Di Eugenio, “FLSA: extending latent semantic analysis with features for dialogue act classification,” in *Proceedings of the ACL*, Barcelona, Spain, July 2004.

[12] G. Tur, U. Guz, and D. Hakkani-Tür, “Model adaptation for dialog act tagging,” in *Proceedings of the IEEE SLT Workshop*, 2006.

[13] P. Haffner, G. Tur, and J. Wright, “Optimizing SVMs for complex call classification,” in *Proceedings of the ICASSP*, Hong Kong, April 2003.

[14] R. E. Schapire and Y. Singer, “Boostexter: A boosting-based system for text categorization,” *Machine Learning*, vol. 39, no. 2/3, pp. 135–168, 2000.