Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence

Crowdsourcing-Assisted Query Structure Interpretation*

Jun Han[1] **, Ju Fan**[2] **, Lizhu Zhou**[1]
[1]Tsinghua University, China
[2]National University of Singapore, Singapore
hanj04@gmail.com, fanj@comp.nus.edu.sg, dcszlz@tsinghua.edu.cn

Abstract

Structured Web search incorporating data from structured sources into search engine results has attracted much attention from both academic and industrial communities. To understand user's intent, query structure interpretation is proposed to analyze the structure of queries in a query log and map query terms to the semantically relevant attributes of data sources in a target domain. Existing methods assume all queries should be classified to the target domain, and thus they are limited when interpreting queries from different domains in real query logs. To address the problem, we introduce a human-machine hybrid method by utilizing crowdsourcing platforms. Our method selects a small number of query terms and asks the crowdsourcing workers to interpret them, and then infers the interpretations based on the crowdsourcing results. To improve the performance, we propose an iterative probabilistic inference method based on a similarity graph of query terms, and select the most useful query terms for crowdsourcing by considering their domain-relevance and gained benefit. We evaluate our method on a real query log, and the experimental results show that our method outperforms the state-of-the-art method.

1 Introduction

Structured Web search incorporates the results from structured data sources into Web search, and it has attracted much attention from both academic and industrial communities. Many commercial search engines, such as Google and Yahoo, have begun to offer structured Web search to effectively answer users' queries which target the structured data in various domains-of-interest, e.g., products, movies, etc.

A key problem of structured Web search is how to interpret a keyword query by not only identifying the target domain of the query, but also mapping query terms to their *semantically relevant* attributes in the domain. For example, given a keyword query "`Seattle Microsoft jobs`", we need to first identify the target domain is JOB, and then respectively map "`Seattle`" and "`Microsoft`" to the attributes, i.e., location and company in the JOB domain.

In order to accurately interpret queries, this paper studies the *query structure interpretation* problem. Consider a query log from a search engine and a target domain. For every query in the log, we generate a set of *structure interpretations* for the terms in the query, where each structure interpretation (or interpretation for simplicity) is a mapping from a term to an attribute in the domain (e.g., a mapping from "`Microsoft`" to the company attribute). The generated interpretations can be used to interpret online queries in structured Web search.

Previous works for query structure interpretation have proposed methods to identify the query intent and extract semantic structure of named entities [Li, 2010; Li *et al.*, 2009; Sarkas *et al.*, 2010; Manshadi and Li, 2009; Cheung and Li, 2012]. The methods employ supervised (e.g., conditional random field, CRF) or semi-supervised (e.g., semi-CRF) methods to interpret queries from a small number of correct interpretations (i.e., the "seeds"). However, the existing methods assume all queries have been ideally classified into the target domain, which is impractical to real query logs consisting of queries in various domains. Our experimental result also shows that, even applying an effective query classifier, these methods still achieve low interpretation accuracy.

The limitation of the machine-based methods can be alleviated with the help of crowdsourcing by utilizing well-established platforms such as Amazon Mechanical Turk (AMT), since human are good at interpreting queries from different domains. [Demartini *et al.*, 2013] proposes a framework by publishing all uninterpreted queries for crowdsourcing. This method, however, is costly and time-consuming given the large scale of query logs. To address the problem, we introduce a human-machine hybrid method to select a small number of query terms for crowdsourcing and best utilize the crowdsourcing results for effective interpretation.

This paper studies the research challenges that naturally arise in the proposed method. The first challenge is, given a user-defined budget k (i.e., the maximal number of microtasks), how to select the "most useful" k terms for crowdsourcing. To address this challenge, our method considers two factors to quantify the usefulness of a term: 1) its relevance to the target domain and 2) the benefit, i.e., the reduction of overall error, if the term is correctly interpreted. The

*This work was supported in part by National Natural Science Foundation of China under grant No.60833003 and No.61272090.

second challenge is how to utilize the crowdsourced interpretations to infer other interpretations. We introduce an iterative probabilistic inference model based on a similarity graph of query terms. We prove that the iteration would converge and show the effectiveness of the model in experiments.

To summarize, we make the following contributions.

1) To the best of our knowledge, this paper is the first to study the problem of crowdsourcing-assisted query structure interpretation given budget constraints.

2) We introduce an iterative probabilistic inference model to fully utilize crowdsourcing results, and develop an effective method for selecting query terms for crowdsourcing.

3) We have conducted experiments on real datasets, and the experimental results show that our method outperforms the state-of-the-art method at a low money cost.

This paper is organized as follows. The problem is formulated in Section 2. We introduce our crowdsourcing-assisted method in Section 3, and report the experimental result in Section 4. The related work is reviewed in Section 5 and we conclude the paper in Section 6.

2 Problem Formulation

Our work considers a query log $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_{|\mathcal{Q}|}\}$, where each query Q consists of a set of query terms, denoted by $Q = \{q_1, q_2, \ldots, q_{|Q_i|}\}$. Table 1 provides an example log containing 14 queries. We also consider a domain-of-interest (or domain for simplicity) with a set of attributes, denoted by $D = \{A_1, A_2, \ldots, A_{|D|}\}$. For example, the JOB domain containing the employment information has three attributes, i.e., `location`, `company`, and `position`.

Given a query $Q \in \mathcal{Q}$, we aim to interpret the structure of Q by mapping each term $q \in Q$ to its *semantically relevant* attribute in the domain D. Specifically, a term is semantically relevant to an attribute, if and only if the term refers to the values or the name of the attribute. For example, the term "`Microsoft`" is semantically relevant to attribute `company`, since it refers to a value of the attribute. Formally, we define the mapping from a term to its semantically relevant attribute.

Definition 1 (Query Term Interpretation). *An interpretation of term q, denoted by I_q^A, is a pair $\langle q, A\rangle$ where $A \in D$ is a domain attribute which is semantically relevant to q.*

In particular, we also consider the case that q refers to the domain D, e.g., the term "`jobs`" in Table 1 referring to the JOB domain, and formally denote it as I_q^D.

Then, we define the problem of query structure interpretation by considering all the term interpretations.

Definition 2 (Query Structure Interpretation). *Given each query $Q \in \mathcal{Q}$, the problem finds a set of query term interpretations, i.e., $I_Q = \{I_q^A \mid q \in Q, A \in D\}$.*

For instance, given the JOB domain, query Q_1 in Table 1 can be interpreted as $I_{Q_1} = \{$⟨"`Seattle`",`location` ⟩, ⟨"`Microsoft`",`company` ⟩, ⟨"`job`",JOB ⟩ $\}$.

Note that our work does not assume that all queries belong to the domain D, which is different from existing methods [Li, 2010; Li *et al.*, 2009; Sarkas *et al.*, 2010; Manshadi and Li, 2009]. Some queries in Table 1 are irrelevant to the JOB domain, e.g., Q_8 about the restaurant information and Q_{13} about the hotel information.

Table 1: An example query log

ID	Query	ID	Query
Q_1	Seattle Microsoft jobs	Q_8	Google restaurant
Q_2	Seattle google jobs	Q_9	programmer salary
Q_3	Seattle programmer jobs	Q_{10}	HR salary
Q_4	Washington google jobs	Q_{11}	developer salary
Q_5	Google jobs	Q_{12}	Seattle renting
Q_6	Washington jobs	Q_{13}	Seattle hotel
Q_7	Seattle HR	Q_{14}	Seattle restaurant

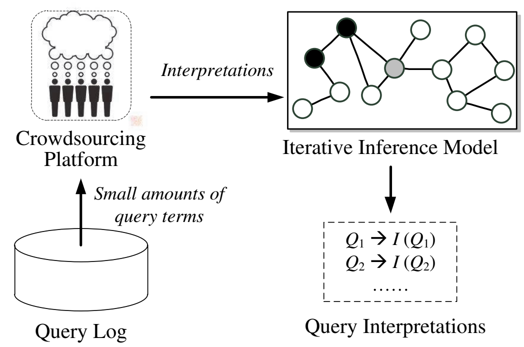


Figure 1: An overview of our crowd-assisted method

3 A Crowdsourcing-Assisted Method

Our method differs from existing query structure interpretation techniques in two key aspects. First, our method leverages the intelligence of crowdsourcing workers to interpret queries from different domains since the workers are good at identifying domain-relevant queries and providing accurate interpretation. Second, the method utilizes a crowd-machine hybrid framework to avoid asking the workers all queries in the log, and thus controls the crowdsourcing cost. Figure 1 provides an overview of our method consisting of two phases.

Phase I: Query Term Selection. This step is to select a small number of terms with *significant usefulness* and generate microtasks for the terms. Specifically, a microtask includes a term as well as the query containing the term, and provides *candidate* domain attributes[1]. When presented with the microtask, the workers are asked to choose the attributes semantically relevant to the corresponding term. Based on workers' choices, we obtain a set of *crowdsourced* interpretations.

Phase II: Interpretation Inference. This step is to infer the structure interpretation for each query in the log. The inference takes two kinds of known interpretations as input: 1) a very small number of *seed* interpretations that maps terms to correct attributes, and 2) the crowdsourced interpretations generated in Phase I. It bootstraps the known term interpretations to predict the unknown ones, and outputs I_Q for each $Q \in \mathcal{Q}$. For effective inference, we propose a graphical model to capture the relationship of query terms, and develop iterative inference based on the graph.

[1] There is also a "none-of-the-above" choice to indicate the term is irrelevant to the domain

Since the usefulness of a term depends on how we use it in the inference, we first introduce the inference model in Section 3.1, while assuming we have obtained the crowdsourced interpretations. Then, we present how to select query terms for crowdsourcing in Section 3.2.

Table 2: Nodes in the I-graph

ID	Term	ID	Term	ID	Term
q_1	Seattle	q_5	restaurant	q_9	jobs
q_2	Microsoft	q_6	Washington	q_{10}	hotel
q_3	Google	q_7	developer	q_{11}	HR
q_4	programmer	q_8	renting	q_{12}	salary

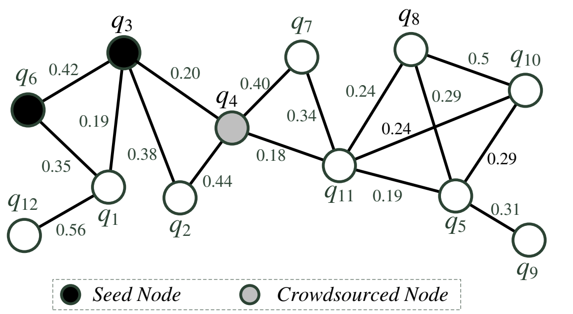


Figure 2: The I-Graph for the queries in Table 1

3.1 Iterative Probabilistic Inference Model

Probabilistic Modeling for Candidate Interpretations. Given a term q, we consider all the attributes that are possibly relevant to it, and generate candidate interpretations. Then, for each candidate I_q^A, we introduce a probability $P(I_q^A)$ to model the *confidence* of the interpretation. The larger the probability $P(I_q^A)$ is, the more likely term q can be interpreted as attribute A. Considering all the candidate interpretations of q, we introduce a candidate vector.

Definition 3 (Candidate Interpretation Vector). *Given a query term q, its candidate interpretation vector consists of the probabilities between q to all attributes in D, denoted by* $\mathbf{P}_q = \big(P(I_q^{A_1}), P(I_q^{A_2}), \ldots, P(I_q^{A_{|D|}})\big)$.

Given the candidate vector of term q, we take the one with the maximal probability as the *best* interpretation of q, i.e.,

$$I_q^A = \arg\max\{P(I_q^{A_i})\}, P(I_q^{A_i}) \in \mathbf{P}_q. \tag{1}$$

In particular, if all probabilities in $\mathbf{P}_q$ are smaller than a pre-defined threshold (e.g., 0.501 in our experiments), we regard term q as an domain-irrelevant term. For example, the term "`renting`" in Q_{12} is irrelevant to domain JOB.

The key challenge in Equation (1) is to estimate the probability $P(I_q^A)$ for each candidate interpretation in the vector. To address the challenge, we develop an inference method based on an interpretation graph.

Interpretation Graph (I-Graph).

The interpretation graph is used to capture the relationship of query terms, which is formally defined as follows.

Definition 4 (Interpretation Graph). *An interpretation graph is an undirected graph, denoted by $\mathcal{G}(\mathcal{T}, \mathcal{E})$, where $\mathcal{T}$ represents all query terms in the log, i.e., $\mathcal{T} = \bigcup_{1 \le i \le |\mathcal{Q}|} Q_i$. Each term $q \in \mathcal{T}$ is associated with its candidate vector $\mathbf{P}_q$. If two terms q_i and q_j are similar to each other, there exists an edge $e \in \mathcal{E}$ with weight $w(q_i, q_j)$.*

Figure 2 and Table 2 provide an example I-graph based on the query terms in Table 1. We can see that each node corresponds to a query term, and nodes are connected by edges weighted with normalized similarities. For example, the two nodes q_2 (i.e., "`Microsoft`") and q_3 (i.e., "`Google`") are connected by an edge with weight 0.38.

We propose a novel method to measure the similarity between two query terms. Since Web queries are usually very short, i.e., the number of terms in a query is small [Cui *et al.*, 2002], traditional string similarity measures, such as Jaccard, are rather limited. To address this problem, we introduce the *context similarity*. The basic idea is that two terms q_i and q_j are similar if the queries containing q_i is similar to those containing q_j. Formally, we introduce the query context.

Definition 5 (Context of Query Term). *The context of a term $q \in Q$ with respect to query Q, denoted by $C(q \mid Q)$, is the set of terms in Q without q, i.e., $C(q \mid Q) = Q - \{q\}$.*

For example, the context of term "`Microsoft`" with respect to Q_1 is $\{Seattle, jobs\}$. Considering all the queries in $\mathcal{Q}$, we obtain a set of contexts for term q, denoted by $\mathcal{C}(q) = \{C(q \mid Q), Q \in \mathcal{Q}\}$. Then, we measure the similarity between q_i and q_j as

$$\mathtt{sim}(q_i, q_j) = \frac{|\mathcal{C}(q_i) \cap \mathcal{C}(q_j)|}{|\mathcal{C}(q_i) \cup \mathcal{C}(q_j)|} \tag{2}$$

Example 1 (Context Similarity). *Consider the queries in Table 1. For the query term "`HR`", we can obtain the set of its contexts, i.e., $C(\text{"HR"}) = \{\{Seattle\}, \{salary\}\}$. Similarly, we can also compute $C(\text{"programmer"}) = \{\{Seattle, jobs\}, \{salary\}\}$. Based on the obtained contexts, we have* $\mathtt{sim}(\text{"HR"}, \text{"programmer"}) = 0.33$.

Then, we define weight of the edge between q_i and q_j as the normalized context similarity, i.e.,

$$w(q_i, q_j) = \frac{\mathtt{sim}(q_i, q_j)}{\sqrt{\sum_{k \ne i} \mathtt{sim}(q_i, q_k) \cdot \sum_{k \ne j} \mathtt{sim}(q_j, q_k)}} \tag{3}$$

Iterative Inference based on I-Graph.

We propose an effective inference method by iteratively updating the candidate vector of each term in the I-Graph. Before iteration, the candidate vectors of different types of terms are initialized as follows.

1) For a term q^s in a seed interpretation that is mapped to A^s, we set $P(I_{q^s}^{A^s})$ to 1 and others in the vector to 0.

2) For a term q in a Crowdsourced interpretation, since we assign a microtask to l workers, we aggregate workers' answers as follows. Suppose the number of workers choosing attribute A_k is l_k. We estimate probability $P(I_q^{A^k})$ in the vector as l_k/l.

3) For a term q with unknown interpretations, we set each probability in the vector as $P(I_q^A) = 0.5$.

We denote the initial candidate vector for each term $q_i \in \mathcal{T}$ as $\mathbf{P}_{q_i}^0$ and the vector after n iterations as $\mathbf{P}_{q_i}^n$. Then, we iteratively update the candidate vector for q_i as follows. If q_i is a seed term, we keep its vector unchanged. On the other hand, if q_i is either a crowdsourced term or a term with unknown interpretations, we compute its vector by considering all the neighbor terms of q_i. Formally, we can compute the vector $\mathbf{P}^n_{q_i}$ as Equation (4), where α is a parameter in $(0, 1)$.

$$\mathbf{P}^n_{q_i} = \begin{cases} \mathbf{P}^0_{q_i} & q_i \text{ is seed} \\ \alpha \sum_{j \neq i} w_{ij} \cdot P^{n-1}_{q_j} + (1-\alpha)\mathbf{P}^0_{q_i} & \text{else} \end{cases} \tag{4}$$

We iteratively update candidate vectors in the I-graph using Equation (4) until convergence. After convergence, we normalize each probability $P(I^A_q)$ by dividing the number of seeds of attribute A, in order to avoid the case that attributes with more seeds dominate those with less seeds.

Example 2. *In our example I-graph in Figure 2, the query terms q_3 and q_6 (black color) are seed interpretations, and q_4 (gray color) is a crowdsourced interpretation. Then, based on our iterative model in Equation (4), we can obtain the converged candidate vector for each term in $\mathcal{T}$.*

Next, we provide some theoretical analysis of our iterative inference method.

Theorem 1. *The iteration of I-graph according to Equation (4) will converge.*

Proof. Let matrix $\mathbf{P}^n = [\mathbf{P}^{n\,T}_{q_1}, \mathbf{P}^{n\,T}_{q_2}, \ldots, \mathbf{P}^{n\,T}_{q_{|\mathcal{T}|}}]^T$.

Construct a matrix $\mathbf{W}$, where $W_{ii} = 0$ and $W_{ij} = w(q_i, q_j)$ if q_i is not seed node, otherwise $W_{ii} = 1$ and $W_{ij} = 0$. Then Equation (4) is equivalent to $\mathbf{P}^n = \alpha\mathbf{W}\mathbf{P}^{n-1} + (1-\alpha)\mathbf{P}^0$. Hence, we can get

$$\mathbf{P}^n = (\alpha\mathbf{W})^{n-1}\mathbf{P}^0 + (1-\alpha)\sum_{i=0}^{n-1}(\alpha\mathbf{W})^i\mathbf{P}^0.$$

Let $\mathbf{D}$ be a diagonal matrix with the i-th element $D_{ii} = \sum_{k \neq i} \mathtt{sim}(q_i, q_k)$. Construct matrix $\mathbf{S}$ satisfying: 1) For q_i which is not a seed, we set $S_{ij} = \mathtt{sim}(q_i, q_j)$ and $S_{ii} = 0$, and 2) For a seed q_i, we set $S_{ij} = 0$ and $S_{ii} = \sum_{k \neq i} \mathtt{sim}(q_i, q_k)$. Thus, we can easily get $\mathbf{W} = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}$.

Since $F = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{1/2} = \mathbf{D}^{-1}\mathbf{S}$ is a stochastic matrix and $\mathbf{W}$ is similar to F. Hence the eigenvalues of $\mathbf{W}$ is within the interval $[-1, 1]$. Considering α is in (0,1), we have

$$\lim_{n\to\infty}(\alpha\mathbf{W})^n = 0, \lim_{n\to\infty}\sum_{i=0}^{n}(\alpha\mathbf{W})^i = (I - \alpha\mathbf{W})^{-1}$$

.

Hence,

$$\mathbf{P}^* = \lim_{n\to\infty}\mathbf{P}^n = (1-\alpha)(I - \alpha\mathbf{W})^{-1}\mathbf{P}^0 \tag{5}$$

Thus, we prove the convergence of the iteration. □

3.2 Query Term Selection for Crowdsourcing

This section discusses how to select query terms for crowdsourcing. A straightforward method is to randomly select a subset of query terms in $\mathcal{T}$. However, this method has the following limitations. First, since the query log contains many queries that not belong to the domain, the method may involve many domain-irrelevant terms, which are useless for structure interpretation. Second, the random selection method treats all terms equal and fails to consider the usefulness of different terms in our inference model.

To address the above problems, we propose a novel model to judiciously select query terms by considering the following two factors: 1) the relevance of the terms to the domain, and 2) the benefit of correctly interpreting the terms through crowdsourcing. Our model prefers the terms which are not only more relevant to the domain, but also have more benefit. Formally, given a term $q_i \in \mathcal{T}$, we use $r(q_i)$ and $b(q_i)$ to respectively represent the relevance and the benefit of q_i. Based on the two factors, we assign the following score to q_i.

$$\mathtt{score}(q_i) = r(q_i) \cdot b(q_i). \tag{6}$$

Then, given a budget, i.e., the maximal number of terms for crowdsourcing, we select the terms with the highest scores. The challenge is how to estimate the term relevance and benefit. We discuss the estimation methods as follows.

Estimating term relevance. Given a term q_i, we estimate the relevance $r(q_i)$ as its overall similarity to the seeds, i.e., $r(q_i) = \sum_{q^s} \mathtt{sim}(q_i, q^s)$ where q^s is a seed. Then, we normalize the relevance as $r(q_i)/r^*$ where r^* is the maximum relevance among all the terms, i.e., $r^* = \max_{q_i \in \mathcal{T}} r(q_i)$.

Estimating term benefit. We introduce the *overall error reduction* to model the benefit of correctly interpreting term q_i. Intuitively, if q_i is mapped to an incorrect attribute, the error would "propagate" to other terms through our inference model, which may induce the overall error to all terms. Thus, a correct interpretation of q_i would not only benefit q_i itself, but also reduce the overall error induced by q_i. More formally, let the vector $\triangle\mathbf{P}^0_{q_i}$ denote the error of q_i. We use $\triangle\mathbf{P}^*$ to represent the overall error induced by $\triangle\mathbf{P}^0_{q_i}$ after iterations, which is estimated as follows.

$$\triangle\mathbf{P}^* = \sum_{k=1}^{|\mathcal{T}|}\theta_k\triangle\mathbf{P}^*_{q_k}, \tag{7}$$

where $\triangle\mathbf{P}^*_{q_k}$ is the error of q_k induced by $\triangle\mathbf{P}^0_{q_i}$ after iterations, and θ_k is a weight capturing the importance of $\mathbf{P}^*_{q_k}$ to the overall error. We estimate θ_k by $\sum_{j=1}^{|\mathcal{T}|} w(q_k, q_j)/\ln(1 + |q_k|)$ where $|q_k|$ is number of characters in term q_k.

Next, we discuss how to induce $\triangle\mathbf{P}^*_{q_k}$ from $\triangle\mathbf{P}^0_{q_i}$. For ease of presentation, we use matrix $\mathbf{W}^*$ to denote $(1-\alpha)(I - \alpha\mathbf{W})^{-1}$. Then, based on Equation (5), we have

$$\triangle\mathbf{P}^*_{q_k} = \sum_{i=1}^{|\mathcal{T}|}\mathbf{W}^*_{ki}\triangle\mathbf{P}^0_{q_i} \tag{8}$$

Based on Equations (7) and (8), we have

$$\triangle\mathbf{P}^* = \sum_{k=1}^{|\mathcal{T}|}\theta_k\triangle\mathbf{P}^*_{q_k} = \sum_{i=1}^{|\mathcal{T}|}(\sum_{k=1}^{|\mathcal{T}|}\theta_k\mathbf{W}^*_{ki})\triangle\mathbf{P}^0_{q_i} \tag{9}$$

Using Equation (9), we estimate the benefit of q_i as

$$b(q_i) = \sum_{k=1}^{|\mathcal{T}|}\theta_k\mathbf{W}^*_{ki} \tag{10}$$

Algorithm for Top-k Term Selection. Based on Equation (6), we select top-k query terms as follows. We first compute the best term with highest score $q^* = \arg_{q \in \mathcal{T}} \texttt{score}(q)$. Then, we recalculate the matrix W^* by removing the elements corresponding to q^*, and obtain the next query term. After obtaining k terms, the algorithm terminates.

Then we add crowdsourced interpretations to the graph by modifying initial vectors of corresponding nodes.

Example 3 (Query Term Selection for Crowdsourcing). *Consider the I-graph in Figure 2. We can see that some nodes have higher relevance but lower benefit (e.g.,* $r(q_2) = 0.48$ *and* $b(q_2) = 0.77$*), while others have lower relevance but higher benefit (e.g.,* $r(q_8) = 0$ *and* $b(q_8) = 1.07$*). By combining both relevance and benefit, we select* q_4 *(*$r(q_4) = 0.36$ *and* $b(q_4) = 1.18$*) as the top-*1 *term for crowdsourcing.*

4 Experiments

4.1 Experiment Setup

Query Log. We evaluated our method on a real Chinese query log from Sogou search engine[2]. The number of queries is $34,000$, and the number of distinct query terms is $43,520$. Note that the query log contains queries belonging to various domains, e.g., JOB, ESTATE, PRODUCT, etc.

Since the log is too large to be manually labeled, we generated three benchmark sets, BENCHMARK1, BENCHMARK2 and BENCHMARK3 by randomly selecting queries from the log. In each benchmark set, we manually identified the queries belonging to the target domain and mapped query terms to domain attributes. The numbers of queries in the three benchmark sets are respectively 161, 154 and 148.

Domains-of-Interest. We examined three domains for query structure interpretation, i.e., ESTATE, JOB and PRODUCT. The domain attributes and number of seeds for each attribute are listed in Table 3. We can see that the proportion of seeds is very small, i.e. 0.8% (ESTATE), 1.9% (JOB) and 0.5% (PRODUCT). Notice that queries in ESTATE and JOB may share some common attributes, e.g., location and company, which increases the challenge of interpreting queries. However, as we will show later, our method still performs well.

Baseline Method. We implemented the state-of-the-art query structure interpretation method [Li, 2010]. Since this method requires all queries to be classified into the target domain (e.g., JOB), we employed a state-of-the-art query classifier [Tang *et al.*, 2009] to obtain the queries classified into the target domain. Then we used the method in [Li, 2010] to interpret the classified queries. Note that both the method in [Li, 2010] and our method used the same set of seeds.

Crowdsourcing. We ran crowdsourcing on AMT platform. In each microtask, we presented a highlight term with its query and candidate attributes and asked worker to select the best attribute. We assigned each microtask to 4 different workers and paid \$0.01 for each assignment. The method of aggregating workers' answers can be referred in Section 3.1.

Evaluation Metrics. We employed the standard precision and recall metrics for evaluation. Specifically, let P denote the number of queries interpreted by a method, and TP denote the number of queries that were correctly interpreted by the method. Specifically, a query was correctly interpreted if *all* its terms had been mapped to correct domain attributes. In addition, let A denote the number of queries that should be interpreted. We compute the metrics, precision, recall and F1 as: $Precision = \frac{\texttt{TP}}{\texttt{P}}$, $Recall = \frac{\texttt{TP}}{\texttt{A}}$, and $F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$.

All the programs were implemented in JAVA and all the experiments were run on a CentOS machine with an Intel(R) Xeon(R) E5607 2.27GHz processor and 32 GB memory.

[2] http://www.sogou.com/

Table 3: the domain information and seeds number

Domain	Attribute	#seeds	Attribute	#seeds
JOB	location	750	time	6
	company	28	position	51
ESTATE	location	258	time	6
	company	2	price	17
	community	52		
PRODUCT	appearance	1	brand	51
	model	137	price	17

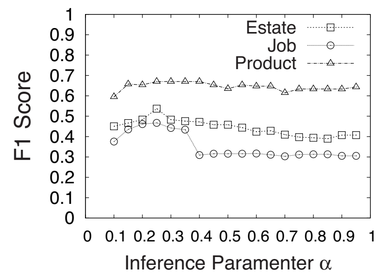


Figure 3: Effect of α in different domains

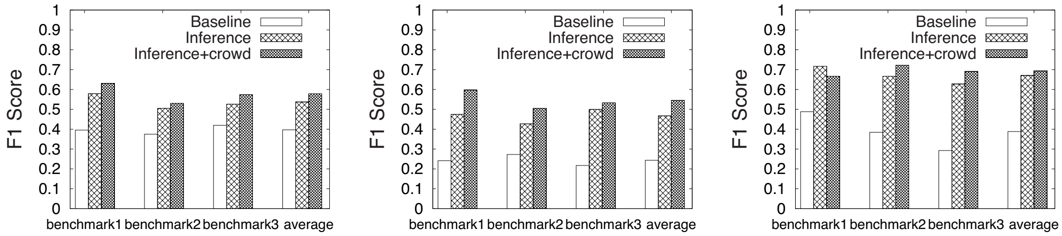
4.2 Effect of Inference Parameter α

In this section, we evaluate the effect of the inference parameter α (Equation (4)) on the performance. For each domain, we ran the experiments with different α on the benchmark sets, then averaged the F1 score. Figure 3 provides the results. With the increase of α, the F1 score first increases and then deceases. It means we can select a best α to achieve good performance. In the remaining experiments, we use $\alpha = 0.25$ by default, which achieves the best performance in all domains as shown in Figure 3.

4.3 Evaluating Term Selection for Crowdsourcing

In this section, we evaluate our method of query term selection for crowdsourcing in Section 3.2. We compared the method with two baselines: 1) `Non-Crowd` without crowdsourced interpretations and 2) `Random-Crowd` with randomly selected query terms. We selected 150 query terms (0.3% of all query terms) for each domain and ran the experiments on three benchmarks and averaged the F1 score.

Table 4 shows the experimental result. We can see that the `Random-Crowd` method is sometimes even worse than the `Non-Crowd` method. For example, the F1 score of `Non-Crowd` method in the PRODUCT domain is 67.0%,



(a) ESTATE domain (b) JOB domain (c) PRODUCT domain

Figure 4: Comparison with the baseline methods on different domains.

Table 4: Comparison of Crowdsourcing term selection

F1 Score	ESTATE	JOB	PRODUCT
`Non-Crowd`	53.7%	46.7%	67.0%
`Random-Crowd`	49.3%	50.3%	50.2%
Our Method	**57.9%**	**54.5%**	**69.3%**

while that of `Random-Crowd` is 50.2%. The main reason is that the `Random-Crowd` method may select many query terms which are irrelevant to the target domain. Using too many irrelevant interpretations may damage the performance. Our methods achieves the best performance as shown in Table 4. The best performance is attributed to the selection model considering both relevance and benefit of query term.

4.4 Comparison with Existing Method

In this section, we compare our method with the baseline method mentioned in Section 4.1. Specifically, we evaluated both our inference method without crowdsourcing (`Inference`) and the method with crowdsourcing (`Inference+Crowd`). Figure 4 provides the results. We can see that the baseline method cannot effectively interpret queries from different domains in the real query log. For example, the average F1 scores of the baseline method on the three domains are 39%, 24% and 39%. The reason is that the baseline method assumes all queries are well classified. When interpreting noisy queries, the recall is low. For example, in the ESTATE domain, although the precision was acceptable (65%), the recall was very low (30%).

Our inference method significantly outperforms the baseline method. For example, the average F1 score of the inference method is increased by about 20% compared with the baseline in the JOB domain in Figure 4. Moreover, using crowdsourced interpretations can further improve the performance, even though we only selected 0.3% query terms for crowdsourcing. For example, in the PRODUCT domain, the F1 score can be further increased by about 8% when using crowdsourcing. The better performance of our method is mainly attributed to our effective inference model and the method of selecting crowdsourcing query terms.

5 Related Work

Query Structure Interpretation. Query structure interpretation can be applied to various tasks, such as question answering systems [Li *et al.*, 2008], query refinement [Baeza-Yates *et al.*, 2004; Sadikov *et al.*, 2010], deep Web search [Cheng *et al.*, 2010; Pound *et al.*, 2011], etc. This problem has been extensively studied. The methods based on conditional random field (CRF) [Li, 2010; Li *et al.*, 2009; Sarkas *et al.*, 2010] formulate the problem as a query tagging problem, and employ CRF or semi-supervised CRF for tagging. [Manshadi and Li, 2009] employed context-free rules while requiring vocabulary for each domain attribute. [Cheung and Li, 2012] proposed an unsupervised method using knowledge bases (e.g., Freebase), and [Pandey and Punera, 2012] also proposed an unsupervised method based on graphical models. Existing methods have the limitation that they assume the queries have been classified into the target domain using query classifiers [Tang *et al.*, 2009; Ji *et al.*, 2011], and they cannot provide good interpretations for queries from different domains, which is very common in real logs. To this end, we propose a crowdsourcing-assisted method which is different from existing machine-learning based method.

Crowdsourcing. Crowdsourcing has attracted much attention in various communities [Lease and Yilmaz, 2011]. Many crowdsourcing-assisted methods have been proposed to improve the performance of various tasks. [Wang *et al.*, 2012] provided a hybrid method for entity resolution. [Yan *et al.*, 2010] combined automatic search and crowd validation to provide better image search. [Zaidan and Callison-Burch, 2011] introduced crowdsourcing to obtain professional translations from non-professional workers. [Law and Zhang, 2011] decomposed high-level mission into sub-goals to answer queries using crowdsourcing. [Demartini *et al.*, 2013] addressed the interpretation problem by publishing all unknown queries for crowdsourcing and may result in costs larger than budgets. To the best of our knowledge, we are the first to study query interpretation using crowd-machine hybrid techniques given budget constraints.

6 Conclusion

This paper has studied the structure interpretation problem for queries from different domains in real query logs. We introduced a crowdsourcing-assisted framework to improve the performance. In the framework, we selected a small number of query terms for crowdsourcing, and employed the obtained interpretations for inference. We proposed an iterative probabilistic inference method, and judiciously selected crowdsourcing terms. We evaluated our method on a real query log, and the result shows the superiority of our method.

References

[Baeza-Yates *et al.*, 2004] Ricardo A. Baeza-Yates, Carlos A. Hurtado, and Marcelo Mendoza. Query recommendation using query logs in search engines. In *EDBT Workshops*, pages 588–596, 2004.

[Cheng *et al.*, 2010] Tao Cheng, Hady Wirawan Lauw, and Stelios Paparizos. Fuzzy matching of web queries to structured data. In *ICDE*, pages 713–716, 2010.

[Cheung and Li, 2012] Jackie Chi Kit Cheung and Xiao Li. Sequence clustering and labeling for unsupervised query intent discovery. In *WSDM*, pages 383–392, 2012.

[Cui *et al.*, 2002] Hang Cui, Ji-Rong Wen, Jian-Yun Nie, and Wei-Ying Ma. Probabilistic query expansion using query logs. In *WWW*, pages 325–332, 2002.

[Demartini *et al.*, 2013] Gianluca Demartini, Beth Trushkowsky, Tim Kraska, and Michael J. Franklin. Crowdq: Crowdsourced query understanding. CIDR, 2013.

[Ji *et al.*, 2011] Ming Ji, Jun Yan, Siyu Gu, Jiawei Han, Xiaofei He, Wei Vivian Zhang, and Zheng Chen. Learning search tasks in queries and web pages via graph regularization. In *SIGIR*, pages 55–64, 2011.

[Law and Zhang, 2011] Edith Law and Haoqi Zhang. Towards large-scale collaborative planning: Answering high-level search queries using human computation. In *AAAI*, 2011.

[Lease and Yilmaz, 2011] Matthew Lease and Emine Yilmaz. Crowdsourcing for information retrieval. *SIGIR Forum*, 45(2):66–75, 2011.

[Li *et al.*, 2008] Fangtao Li, Xian Zhang, Jinhui Yuan, and Xiaoyan Zhu. Classifying what-type questions by head noun tagging. In *COLING*, pages 481–488, 2008.

[Li *et al.*, 2009] Xiao Li, Ye-Yi Wang, and Alex Acero. Extracting structured information from user queries with semi-supervised conditional random fields. In *SIGIR*, pages 572–579, 2009.

[Li, 2010] Xiao Li. Understanding the semantic structure of noun phrase queries. In *ACL*, pages 1337–1345, 2010.

[Manshadi and Li, 2009] Mehdi Manshadi and Xiao Li. Semantic tagging of web search queries. In *ACL/AFNLP*, pages 861–869, 2009.

[Pandey and Punera, 2012] Sandeep Pandey and Kunal Punera. Unsupervised extraction of template structure in web search queries. In *WWW*, pages 409–418, 2012.

[Pound *et al.*, 2011] Jeffrey Pound, Stelios Paparizos, and Panayiotis Tsaparas. Facet discovery for structured web search: a query-log mining approach. In *SIGMOD Conference*, pages 169–180, 2011.

[Sadikov *et al.*, 2010] Eldar Sadikov, Jayant Madhavan, Lu Wang, and Alon Y. Halevy. Clustering query refinements by user intent. In *WWW*, pages 841–850, 2010.

[Sarkas *et al.*, 2010] Nikos Sarkas, Stelios Paparizos, and Panayiotis Tsaparas. Structured annotations of web queries. In *SIGMOD Conference*, pages 771–782, 2010.

[Tang *et al.*, 2009] Lei Tang, Suju Rajan, and Vijay K. Narayanan. Large scale multi-label classification via metalabeler. In *WWW*, pages 211–220, 2009.

[Wang *et al.*, 2012] Jiannan Wang, Tim Kraska, Michael J. Franklin, and Jianhua Feng. Crowder: Crowdsourcing entity resolution. *PVLDB*, 5(11):1483–1494, 2012.

[Yan *et al.*, 2010] Tingxin Yan, Vikas Kumar, and Deepak Ganesan. Crowdsearch: exploiting crowds for accurate real-time image search on mobile phones. In *MobiSys*, pages 77–90, 2010.

[Zaidan and Callison-Burch, 2011] Omar Zaidan and Chris Callison-Burch. Crowdsourcing translation: Professional quality from non-professionals. In *ACL*, pages 1220–1229, 2011.