Learning to Extract Cross-Session Search Tasks

Hongning Wang
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana IL, 61801 USA
wang296@illinois.edu

Yang Song[1], Ming-Wei Chang[1],
Xiaodong He[1], Ryen W. White[1],
Wei Chu[2]
[1]Microsoft Research, Redmond, WA
[2]Microsoft Bing, Bellevue, WA 98004 USA
{yangsong,minchang,xiaohe,ryenw,wechu}
@microsoft.com

ABSTRACT

Search tasks, comprising a series of search queries serving the same information need, have recently been recognized as an accurate atomic unit for modeling user search intent. Most prior research in this area has focused on short-term search tasks within a single search session, and heavily depend on human annotations for supervised classification model learning. In this work, we target the identification of long-term, or *cross-session*, search tasks (transcending session boundaries) by investigating inter-query dependencies learned from users' searching behaviors. A semi-supervised clustering model is proposed based on the latent structural SVM framework, and a set of effective automatic annotation rules are proposed as weak supervision to release the burden of manual annotation. Experimental results based on a large-scale search log collected from Bing.com confirms the effectiveness of the proposed model in identifying cross-session search tasks and the utility of the introduced weak supervision signals. Our learned model enables a more comprehensive understanding of users' search behaviors via search logs and facilitates the development of dedicated search-engine support for long-term tasks.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Experimentation

Keywords

Cross-session search task, query log mining, semi-supervised clustering, weak supervision

1. INTRODUCTION

Search engine users' information needs span a broad spectrum [11, 15]: simple needs, such as homepage finding, can mostly be satisfied via a single query; but users may also issue a series of queries, collect, filter, and synthesize information from multiple sources to solve a complex task, e.g., planning a vacation. To comprehensively and accurately understand these needs from recorded actions in the user query logs, we must segment and associate chronologically-ordered queries into a semantically-coherent structure.

The primary mechanisms for segmenting the logged query streams are *session*-based, where short inactivity timeouts between user actions are applied as a means of demarcating session boundaries [17, 19]. Recently, there has been significant research on identifying *tasks* within these sessions, e.g., Lucchese et al [15] proposed the concept of a "*task-based session*": where a cluster of queries within the same session serves a particular common search intent. However, those methods rely on the accurate identification of the original session boundaries and the empirically-set timeout threshold may not be a valid criterion for identifying the semantic structure among queries: many tasks have been shown to span multiple search sessions [1, 11]. It suggests that there is value in studying and improving task identification methods spanning session boundaries.

Table 1: An example of cross-session search tasks.

Time	Query	SessionID	TaskID
05/29/2012 14:06:04	bank of america	1	1
05/29/2012 14:11:49	sas	1	2
05/29/2012 14:12:01	sas shoes	1	2
05/30/2012 10:19:34	credit union	2	3
05/30/2012 12:25:19	6pm.com	3	4
05/30/2012 12:49:21	coupon for 6pm	3	4

Motivating Example: Consider a real example of search tasks from a single user shown in Table 1, which is extracted from the logs of Bing.com. We manually annotated the in-session tasks in the last column of the table and segmented the sessions using 30-min inactivity threshold. We can observe that the user performed two tasks in the first search session on May 29, 2012, one for personal banking and another for shopping (for shoe-brand San Antonio Shoes). And on the second day, the user performed two individual search sessions, and each session consists of one single task, i.e., banking and shopping (at the online discount merchant 6pm.com) accordingly. However, humans can easily recognize that those four tasks annotated in three different sessions happen to be only two unique tasks: a shopping task including queries of "sas", "sas shoes", "6pm.com" and "coupon for 6pm", and a personal banking task including queries of "bank of america" and "credit union."

Prior work on identifying cross-session tasks has targeted *pairs* of queries, and made predictions about whether they share the same goal or represent the same task [11, 13]. Unfortunately, pairwise predictions alone cannot generate the partition of tasks, and post-processing is need-

WWW 2013, May 13–17, 2013, Rio de Janeiro, Brazil.
ACM 978-1-4503-2035-1/13/05.

ed to obtain the final task partitions [14]. Besides, such pairwise predictions might not be consistent: e.g., predicting query i and j, query i and k to be in the same task, but query j and k are not. As a result, definite decisions have to be made in post-processing; but such decisions are isolated from the classifier training, and are therefore not guaranteed to be optimal. To understand this limitation, taking the search tasks shown in Table 1 as an example. A lexicon-similarity-based classifier can easily recognize the query "6pm.com" and "coupon for 6pm," and "sas" and "sas shoes" belong the same search tasks, because of query overlap; but it can hardly associate "sas" with "6pm.com." Furthermore, the query "sas" is ambiguous: it has other interpretations such as the business analytic software SAS or special air service in British Army. Hence, even the features leveraging external knowledge bases [15] may be unable to assist. But when we consider the temporal juxtaposition of "sas shoes" and "sas," we can confidently infer that the "sas" here refers to "San Antonio Shoes"; and since we know that the queries "6pm.com" and "sas shoes" are both associated with shoe shopping, we can safely conclude that those four different queries are part of the same shopping task. From this example, we can conclude that the queries belonging to the same search task convey rich dependency relationships, which provide us with valuable information to analyze and exploit the search task structure. In contrast, traditional binary classification methods are only optimized for independent predictions and thus cannot explore such in-depth relationships among queries.

Moreover, existing methods for cross-session search task extraction heavily depend on the manual annotation of tasks [11, 13, 14], which is expensive to acquire at scale. Fortunately, we have the opportunity to leverage problem-specific knowledge to assist with model learning, where various informative signals are available for us to identify such knowledge. For example, identical and reformulated queries, e.g., "sas" and "sas shoes" in Table 1, and queries with identical returned URLs should belong to the same search task with high confidence. Such knowledge can be summarized by a set of annotation rules, i.e., must-link and cannot-link [22], and applied at scale to reduce the burden of manual annotation. We refer to such knowledge as *weak supervision*, because it only provides pairwise supervision over a subset of queries; and the quality of such supervision might vary.

The research described in this paper addresses the above challenges and makes the following research contributions:

- Address the cross-session search task extraction problem in a structural learning framework, where we treat a user's entire query log as a whole and explicitly model the dependency among queries in the same task.

- Explore helpful weak supervision from different perspectives to reduce the burden of manual annotation and guide the supervised model learning for cross-session task extraction.

- Provide a detailed analysis of the proposed method whereby we compare it against state-of-the-art cross-session task extraction baselines and demonstrate significant performance gains on a variety of metrics.

2. RELATED WORK

Various methods have been proposed to segment and organize query logs into semantically coherent structures. The most commonly used unit, the search *session*, was often defined based on a timeout criterion, where different thresholds, ranging from 5 to 120 minutes, have been proposed [4, 9, 19]. In addition, Radlinski and Joachims [17] used a 30-minute timeout together with query similarity measures to define sequences of similar queries that combine to form so-called query chains.

Search tasks within the temporally-demarcated session boundaries have also been studied. Spink et al. [20] demonstrated that multi-tasking behavior, whereby multiple tasks are intertwined within the same time period, occurs frequently. Lucchese et al. [15] referred to such sessions as *task-based* sessions (or in-session tasks). Various methods, based on time splitting [2, 9], lexicon similarity [11, 15], and query reformulation patterns [9, 11], have been proposed to identify in-session tasks.

Recently, researchers have realized the necessity of going beyond the session timeout, and several methods have been proposed to tackle the problem by classifying whether two queries share the same search goal, i.e., same-task prediction. Jones et al. [11] claimed that no particular timeout threshold is necessary a valid constraint for identifying task boundaries. They found over 15% of search tasks are performed across time-out based session boundaries in their search log data set. To extract the cross-session tasks (which were defined as mission and goal), they built classifiers to identify task and sub-task boundaries, as well as pairs of queries belonging to the same task. Kotov et al. [13] and Agichtein et al. [1] studied the problem of cross-session task extraction via binary same-task classification, and found different types of tasks demonstrate different life spans.

In this work, although we focus on cross-session tasks, our solution is actually more general than cross-session only. Our only criterion for extracting search tasks is that queries in the same task should serve for the same high-level information need; tasks can be performed in a single session or can span multiple sessions. The major difference between our work and existing cross-session task extraction work is that instead of making a series of binary same-task predictions, we cast this problem as a structural learning problem, which explicitly models the dependency among queries in a search task. As we have discussed in Section 1, independent binary classification cannot capitalize on dependencies between pairs of predictions. In addition, existing classification-based methods heavily depend on manual annotations for model training. This will greatly limit their generalization capability when there is few or no task annotation available. In this work, we explored a variety of informative signals as weak supervision to release the burden of manual annotation and guide model learning.

3. PROBLEM DEFINITION

In this section, we formally define the problem of cross-session search task extraction.

Query log records the interaction behaviors from a set of different users, $\mathcal{U} = \{u_1, u_2, \ldots, u_N\}$, in a search engine. It stores a sequence of queries $\mathcal{Q}_n = \{q_{n1}, q_{n2}, \ldots, q_{nM}\}$ from user u_n, together with the timestamp t_{ni} when the query is submitted and the corresponding list of returned URLs, $\mathcal{URL}_{ni} = \{url_{ni1}, url_{ni2}, \ldots, url_{niL}\}$. Each query q_{ni} is represented as the original string that users submitted to the search engine, and $\mathcal{Q}_n$ is ordered according to query timestamp t_{ni}. Each URL url_{nil} has two attributes: URL string and click timestamp c_{nil} (c_{nil}=0 if it was not clicked).

Definition (Session) Given user u_n's search history $\mathcal{Q}_n$ and a fixed time-out threshold τ_{cut}, a session $\mathcal{S}_{nt}$ is a set of consecutive queries from $\mathcal{Q}_n$, such that $\forall q_{ni} \in \mathcal{S}_{nt}, q_{nj} \in \mathcal{S}_{nt}, q_{nl} \notin \mathcal{S}_{nt}$, $|t_{ni} - t_{nj}| \leq \tau_{cut}$ and $|t_{ni} - t_{nl}| > \tau_{cut}$.

The definition of session implies that $\{\mathcal{S}_{nt}\}_{t=1}^{T}$ is a set of disjoint partitions of query sequence $\mathcal{Q}_n$, such that $\forall i \neq j$, $\mathcal{S}_{ni} \cap \mathcal{S}_{nj} = \emptyset$ and $\mathcal{Q}_n = \bigcup_i \mathcal{S}_{ni}$. A typical time-out threshold is set to be 30 minutes [13, 15, 17].

Definition (Search Task) Given user u_n's search history $\mathcal{Q}_n$, a search task $\mathcal{T}_{nk}$ is a maximum subset of queries in $\mathcal{Q}_n$, such that all the queries in $\mathcal{T}_{nk}$ correspond to a particular information need.

This definition of search task indicates $\{\mathcal{T}_{nk}\}_{k=1}^{K}$ is also a set of disjoint partitions of query sequence $\mathcal{Q}_n$: $\forall j \neq k$, $\mathcal{T}_{nj} \cap \mathcal{T}_{nk} = \emptyset$ and $\mathcal{Q}_n = \bigcup_k \mathcal{T}_{nk}$. Therefore, each $\mathcal{T}_{nk}$ is not confined to a particular session $\mathcal{S}_{nt}$; instead they can overlap, or one search task can contain multiple sessions. To emphasize such a difference, we will refer to our definition of search task as *Cross-session Search Task* as opposed to the previous definition of *In-session Search Task* [15, 20].

Based on the above notations and definitions, we define the problem of cross-session search task extraction as,

Definition (Cross-Session Search Task Extraction) Given user u_n's search query log $\mathcal{Q}_n$, partition the sequence into disjoint subsets $\{\mathcal{T}_{n1}, \mathcal{T}_{n2}, \ldots, \mathcal{T}_{nk}\}$, such that the partition is consistent with the user's underlying information need; when explicit task annotation is available, the extracted tasks should be consistent with the annotation.

In particular, such task partition can be uniquely determined by a mapping function $y(q_{ni}) \rightarrow \mathcal{T}_{nk}$ from query q_{ni} to its corresponding task partition $\mathcal{T}_{nk}$ for the query sequence $\mathcal{Q}_n$. In addition, we should note that the number of tasks, e.g., K, user u_n can take is not specified in our definition, and therefore the learning method should find the appropriate K for each given $\mathcal{Q}_n$ automatically.

4. SEARCH TASK EXTRACTION WITH LATENT STRUCTURED SVM

We model the cross-session search task extraction as a *supervised clustering problem (SCP)* [6, 8, 22], where given the clustering membership, we need to build up a model which captures the connection between queries.

4.1 Motivation: Best Link vs. All Links

A commonly used assumption in SCP is the *all-link* clustering structure [8, 10], where one needs to associate the queries belonging to the same task together, such that the in-cluster similarity defined by the summation of similarities over all the pairs of instances within a cluster is maximized. However, this objective may not be the most appropriate for our problem: in a task consisting of m queries, many of the $O(m^2)$ pairs are not necessarily similar, or even quite different. Recall the example search tasks shown in Table 1, the query "sas" and "coupon for 6pm" are not directly related under most of similarity metrics, e.g., edit distance or term overlap; putting them into the same task can only hurt the in-cluster similarity. As a result, any algorithm aims at maximizing the *all-link*-based in-cluster similarity can hardly discover this type of task.

A more reasonable way for clustering queries into tasks is to find the *strongest* link between a candidate query and queries in the target cluster, i.e., *bestlink* [10]. For example, after scanning through all the queries listed in Table 1, we can easily infer the relation between "sas" and "coupon for 6pm" based on the decision over the other two queries, "sas shoes" and "6pm.com", which have been recognized as being in the same shoe shopping task.

This motivates us to revise the objective of clustering queries: a query belonging to one particular search task *does not* need to be similar to all the other queries in this task (*all-link*), but there has to be *at least* one query, which is *strongly* associated with this query in that task (*bestlink*). Intuitively, this modeling assumption simulates how a human editor annotates the search tasks in the query log: one might determine if two queries belong to the same task by reasoning transitively over strong connections between queries in the same task.

4.2 Best Link as Latent Structure

Unfortunately, the bestlink structure is *hidden* in the query log, and it is even impossible for the human editors to explicitly annotate, since such structure might not be unique. Therefore, we adopt the structural learning method with latent variables, i.e., latent structural SVMs [5, 24], to realize the bestlink modeling assumption, and utilize the hidden structure to explore the dependency among queries within the same task. We name our method as *bestlink SVM.*

To formalize the idea of bestlink SVM, we denote the hidden best-link structure as h. Before stating clearly the detailed definition of h, it helps to consider h as a graph whose edges connect the "most similar" queries. Given a query sequence $\mathcal{Q} = \{q_1, q_2, \ldots, q_M\}^1$, we define a feature vector for the task partition y specified by the hidden best-link structure h as $\Phi(\mathcal{Q}, y, h)$. And based on $\Phi(\mathcal{Q}, y, h)$, our bestlink SVM is a linear model parameterized by w, and predicts the task partition at testing time by,

$$(\hat{y}, \hat{h}) = \underset{(y,h) \in \mathcal{Y} \times \mathcal{H}}{\arg\max}\ w^{\mathsf{T}} \Phi(\mathcal{Q}, y, h), \qquad (1)$$

where $\mathcal{Y}$ and $\mathcal{H}$ represent the sets of possible structures of y and h respectively. $\hat{y}$ becomes the output for cross-session tasks and $\hat{h}$ is the inferred latent structure. In this paper, we refer to solving Eq (1) as the decoding problem.

The decoding problem of Eq (1) clearly distinguishes the proposed bestlink SVM model from the previous binary-classification-based methods. In bestlink SVM, we model the entire query sequence $\mathcal{Q}$ as a whole, and predict the task membership for all the queries simultaneously; while the previous two-step approaches cannot explore the interactions among queries in the same task, and isolated predictions are made on each pair of queries in those methods.

The definition of h needs to be carefully designed, otherwise the decoding problem (hence the training algorithm as well) can be intractable. We define $h(q_i, q_j) = 1$ if query q_i and q_j are directly connected in h; and otherwise, $h(q_i, q_j) = 0$. To model the first query of a new search task, i.e., the query that does not have a strong connection with any previous queries, we add a dummy query q_0 at the beginning of each user's query log. All the queries connecting to q_0 would be treated as the initial query of a new search

[1] In the following discussion, when no ambiguity is invoked, we drop the index n for user u_n to simplify the notations.

task. Besides, we enforce that a query can only link to another query *in the past*, or formally,

$$\sum_{i=0}^{j-1} h(q_i, q_j) = 1, \forall j \geq 1$$

Taking the search tasks shown in Table 1 as an example, we illustrate the idea of bestlink structure in Figure 1. From the figure, we can clearly notice that the bestlink defines a hierarchical tree structure of "strong" connections among the queries: rooted in the dummy query q_0, each subtree of q_0 corresponds to one specific search task in a user's search history. For a new query, it can only belong to a previous search task or be the first query of a new task. Therefore, the temporal order provides us a helpful signal to explore the dependency between queries.

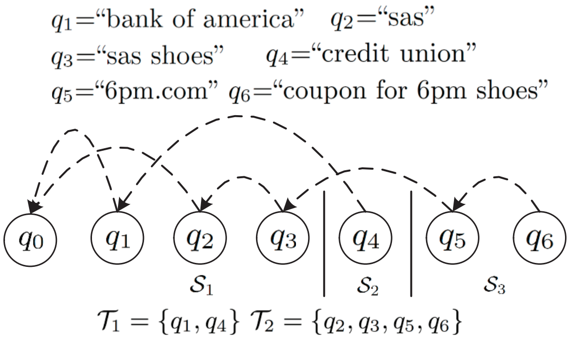


Figure 1: Illustration of hidden search task structure specified in bestlink SVM. $\{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3\}$ are the sessions segmented by the 30-minutes inactivity threshold, $\{\mathcal{T}_1, \mathcal{T}_2\}$ are the search tasks annotated by human editor. The dotted arrows indicate one possible hidden structure identified by bestlink SVM.

We require h to be consistent with y – that is, $h(q_i, q_j) = 1$ implies $y(q_i) = y(q_j)$; in other words, the task partition y is determined by the connected components in h. As a result, the dependency among the queries belonging to the same task is explicitly encoded by the latent bestlink structure h: as shown in Figure 1, predicting "sas" and "sas shoes", "sas shoes" and "6pm.com" belonging to the same task would immediately lead to the conclusion that all these three queries belong to the same task, even though "sas" and "coupon for 6pm.com" are not directly connected to each other.

Accordingly, our feature vector for a particular task partition y is defined over the links in h as,

$$\Phi(\mathcal{Q}, y, h) = \sum_{i,j} h(q_i, q_j) \sum_{s=1}^{S} \phi_s(q_i, q_j), \tag{2}$$

where a set of symmetric pairwise features $\{\phi_s(\cdot,\cdot)\}_{s=0}^{S}$ is given to characterize the similarity between query q_i and q_j. In particular, to accommodate the dummy query q_0, we set $\phi_0(q_0, \cdot) = 1$ and $\forall s > 0, \phi_s(q_0, \cdot) = 0$.

Based on our feature vector design and the directed linkage structure of h, exact inference can be efficiently calculated for the decoding problem in Eq (1). Algorithm 1 described an incremental implementation to solve the exact inference problem, where we only need the queries appearing before the given query to determine its task membership. This makes bestlink SVM feasible to be deployed in the search engine query log system in an online fashion, since the newly arrived queries will not affect the method's prediction on previous queries.

Algorithm 1: Task Partition Prediction

```
Input: Query sequence $\mathcal{Q} = \{q_1, q_2, \ldots, q_M\}$, pairwise
       features $\{\phi_s(\cdot,\cdot)\}_{s=0}^{S}$ and linear weight $w$.
Output: Task partition $\hat{y}$.
//Step 1: Initialize the latent structure $\hat{h}$
$\hat{h}(\cdot,\cdot) = 0$;
//Step 2: Search for the best latent structure $\hat{h}$
for $i = 1 \ldots M$ do
    $j' = \arg\max_{0 \leq j < i} \sum_{s=1}^{S} w_s^\top \phi_s(q_i, q_j)$;
    $\hat{h}(i, j') = 1$;
end
//Step 3: Construct the best task partition $\hat{y}$:
$t = 0$;
for $i = 1 \ldots M$ do
    $j' = \arg\max_{0 \leq j < i} h(i, j)$;
    if $j' = 0$ then
        $\hat{y}(i) = t$;
        $t = t + 1$;
    end
    else
        $\hat{y}(i) = \hat{y}(j')$;
    end
end
return $\hat{y}$
```

4.3 Solving the bestlink SVM

For a given set of query logs with annotated tasks, $\{(\mathcal{Q}_n, y_n)\}_{n=1}^{N}$, we need to retrieve the optimal weight setting w for the proposed bestlink SVM. Empirically, the optimal weight w should minimize the error between the predicted task partition $\hat{y}_n$ and ground-truth y_n. In addition, w should also be optimized for good generalization capability, e.g., maximize the margin between ground-truth partition and wrong partitions [21]. This naturally gives rise to the following optimization problem within the latent structural SVMs framework [5, 24]:

$$\begin{aligned} \min_{w,\xi} \; & \frac{1}{2}||w||^2 + C\sum_{n=1}^{N} \xi_n^2 \\ s.t. \; \forall n, \; & \max_{h \in \mathcal{H}} w^\top \Phi(\mathcal{Q}_n, y_n, h) \geq \\ & \max_{(\hat{y},\hat{h}) \in \mathcal{Y} \times \mathcal{H}} [w^\top \Phi(\mathcal{Q}_n, \hat{y}, \hat{h}) + \Delta(y_n, \hat{y}, \hat{h})] - \xi_n \end{aligned} \tag{3}$$

where $\Delta(y_n, \hat{y}, \hat{h})$ characterizes the distance between the ground-truth partition y_n and predicted partition $\hat{y}$ specified by the latent structure $\hat{h}$, $\{\xi_n\}_{n=1}^{N}$ is a set of slack variables to allow errors in the training set, and C controls the trade-off between empirical loss and model complexity.

Because the ground-truth bestlink structure h_n^* for $\mathcal{Q}_n$ is unobservable in the training data, we cannot measure the distance between (y_n, h_n^*) and $(\hat{y}, \hat{h})$. As a result, we define the margin between the ground-truth task partition y_n and predicted task partition $\hat{y}$ based on the inferred latent structure $\hat{h}$ as,

$$\Delta(y_n, \hat{y}, \hat{h}) = |\mathcal{Q}_n| - |\mathcal{T}_n| - \sum_{i,j} h(i,j)\sigma(y_n, (i,j)) \tag{4}$$

where $|\mathcal{Q}_n|$ is the number of queries in $\mathcal{Q}_n$, $|\mathcal{T}_n|$ is the number of annotated tasks in $\mathcal{Q}_n$, and $\sigma(y, (i,j)) = 1$ if $y(i) =$

Table 2: Pairwise Similarity Features.

Type	Feature	Description
Query -based	Q-COSINE	cosine similarity between the term sets of q_i and q_j
	Q-EDIT	norm edit dist between query strings of q_i and q_j
	Q-JAC	Jaccard coeff between the term sets of q_i and q_j
	Q-TIME	1.0/(absolute time difference in seconds between q_i and q_j)
	Q-DIST	(# of queries in between of q_i and q_j)/$\lvert Q_n \rvert$
	Q-URL-MATCH-SUM	$\sum_{url \in \mathcal{URL}_i} \big(c(q_j, url)\big) + \sum_{url \in \mathcal{URL}_j} \big(c(q_i, url)\big)$
	Q-URL-MATCH-MAX	$\max_{url \in \mathcal{URL}_i} \big(c(q_j, url)\big) + \max_{url \in \mathcal{URL}_j} \big(c(q_i, url)\big)$
	Q-CLICK-URL-MATCH-AVG	$\sum_{url \in \text{clicked } \mathcal{URL}_i} \big(c(q_j, url)\big) + \sum_{url \in \text{clicked } \mathcal{URL}_j} \big(c(q_i, url)\big)$
	Q-CLICK-URL-MATCH-MAX	$\max_{url \in \text{clicked } \mathcal{URL}_i} \big(c(q_j, url)\big) + \max_{url \in \text{clicked } \mathcal{URL}_j} \big(c(q_i, url)\big)$
URL -based	U-EDIT-DOMAIN-MIN	min norm edit dist between domain of $\mathcal{URL}_i$ and domain of $\mathcal{URL}_j$
	U-EDIT-ALL-MIN	min norm edit dist between $\mathcal{URL}_i$ and $\mathcal{URL}_j$
	U-EDIT-ALL-CLICK-MIN	min norm edit dist between clicked $\mathcal{URL}_i$ and clicked $\mathcal{URL}_j$
	U-EDIT-DOMAIN-AVG	avg norm edit dist between domain of $\mathcal{URL}_i$ and domain $\mathcal{URL}_j$
	U-EDIT-ALL-AVG	avg norm edit dist between $\mathcal{URL}_i$ and $\mathcal{URL}_j$
	U-EDIT-ALL-CLICK-AVG	avg norm edit dist between clicked $\mathcal{URL}_i$ and clicked $\mathcal{URL}_j$
	U-JAC-ALL-CLICK	Jaccard coeff between clicked $\mathcal{URL}_i$ and clicked $\mathcal{URL}_j$
	U-JAC-ALL	Jaccard coeff between $\mathcal{URL}_i$ and $\mathcal{URL}_j$
	U-JAC-DOMAIN-CLICK	Jaccard coeff between domain of clicked $\mathcal{URL}_i$ and domain of clicked $\mathcal{URL}_j$
	U-JAC-DOMAIN	Jaccard coeff between domain of $\mathcal{URL}_i$ and domain of $\mathcal{URL}_j$
	U-SIM-CLICK-MAX	max ODP category similarity of clicked $\mathcal{URL}_i$ and clicked $\mathcal{URL}_j$
	U-SIM-CLICK-AVG	avg ODP category similarity of clicked $\mathcal{URL}_i$ and clicked $\mathcal{URL}_j$
	U-SIM-MAX	max ODP category similarity of $\mathcal{URL}_i$ and $\mathcal{URL}_j$
	U-SIM-AVG	avg ODP category similarity of $\mathcal{URL}_i$ and $\mathcal{URL}_j$
Session -based	S-SAME	if q_i and q_j are in the same session
	S-FIRST	if both q_i and q_j are the first query of session
	S-DIST	# queries in between of q_i and q_j

Note: 1) ***norm edit dist*** is the edit distance between string s and t divided by the maximum length of s and t;
2) $c(q, url)$ is a function counting the number of query terms in q contained in url;
3) ***clicked*** $\mathcal{URL}$ is a subset of URLs, whose click timestamp $c_{il} > 0$.

$y(j)$, otherwise $\sigma(y, (i, j)) = -1$. It is easy to verify that $\Delta(y_n, \hat{y}, \hat{h})$ is non-negative, and equals to zero if and only if the task partition $\hat{y}$ is the same as y_n.

Since we are minimizing the square hinge loss over the predictions in the training set, the optimization problem introduced in Eq (3) can be efficiently solved by the iterative algorithm proposed in [5]: the optimization procedure minimizes Eq (3) by constructing a sequence of convex problems in each iteration, and each iteration guarantees to decrease the objective function. In the employed optimization algorithm, two types of inference are required: loss-augmented inference, i.e., $\max_{(\hat{y}, \hat{h}) \in \mathcal{Y} \times \mathcal{H}} [w^\mathsf{T} \Phi(\mathcal{Q}_n, \hat{y}, \hat{h}) + \Delta(y_n, \hat{y}, \hat{h})]$; and latent variable completion inference, i.e., $\max_{h \in \mathcal{H}} w^\mathsf{T} \Phi(\mathcal{Q}_n, y_n, h)$. Since the calculation of $\Delta(y_n, \hat{y}, \hat{h})$ can be decomposed onto the edges in h, loss-augmented inference can be directly solved via Algorithm 1 by adding an additional cost $\sigma(y_n, (i, j))$ into Step 2 when finding the best link for query q_i. And the latent variable completion inference can also be achieved via Algorithm 1 by restricting Step 2 to only search in the queries with the same task label as q_i. Both inference algorithms are exact, which renders us a more precise optimization result for Eq (3). The detailed algorithm is omitted due to the lack of space.

4.4 Pairwise Similarity Features

Our bestlink SVM requires a set of pairwise similarity features as input to characterize the connection between a pair of queries. In this work, we explored a variety of signals, from lexicon similarity to query semantic category similarity, to measure the similarity between a pair of queries.

Our proposed pairwise similarity features are list in Table 2, and categorized into three types: query-based, URL-based and session-based similarities. To analyze the semantic relationships between queries, we assign each URL to a topic distribution over 385 categories from the second level of "Open Directory Project" (ODP, dmoz.org) with a content-based classifier [18]. The inner product of the predicted topic distribution is used to measure the semantic similarity between queries. Besides, to make the features comparable across each other, we normalize them into the range of [0,1] accordingly, e.g., taking reciprocal of the absolute time difference between two queries.

5. IMPROVING THE MODEL WITH WEAK SUPERVISION SIGNALS

The bestlink SVM proposed in Section 4.2 is a supervised clustering algorithm that requires full annotation of tasks in the query log. As we have discussed in Section 1, various types of signals, which can be automatically derived from the query logs, are helpful for identifying the search tasks. In this section, we discuss how to make use of large quantities of unlabeled data with weak supervision signals in the proposed bestlink SVM.

We explore weak supervision signals for the cross-session search task extraction problem from different perspectives, and formalize them in terms of "must-link" and "cannot-link" [22]. Query matching, e.g., identical or reformulated queries, is a strong indication that two queries belong to the same task. Besides, the returned URLs for the given query are also an important source for determining the task membership: because modern search engines have sophisticated query pre-

Table 3: Partial Annotation Rules.

Type	Description
Must-link ($\tilde{y}(i) = \tilde{y}(j)$)	$q_i = q_j$ $q_i \subset q_j$ or $q_j \subset q_i$ $\mathcal{URL}_i = \mathcal{URL}_j$ clicked $\mathcal{URL}_i$=clicked $\mathcal{URL}_j$
Cannot-link ($\tilde{y}(i) \neq \tilde{y}(j)$)	$q_i \neq q_j$ AND $\mathcal{URL}_i \cap \mathcal{URL}_j = \emptyset$

processing procedures, e.g., spelling correction [7] and query rewriting [12], when it decides to return identical URLs for two different queries, it is a strong signal that the two queries are related. Table 3 lists four types of must-link and one type of cannot-link we have defined in this work. When there is conflict between the automatically generated must-links and cannot-links, e.g., nontransitive, we will drop the cannot-links to make the annotations consistent.

Though one may treat such signals as features and manually tune the weights to stress their importance, we want emphasize that this approach is sub-optimal for the following two reasons: 1) features are independent in linear models, the knowledge about one feature cannot help the model learn for other features; instead, if we treat such information as supervision, all the features can be adjusted accordingly; 2) it is difficult to manually set the appropriate weights for all the features; while optimizing the objective function defined on both weak supervision and manual annotations would estimate the weights in a systematic way.

Note that when we apply the proposed must-link and cannot-link to the unlabeled user query logs, we can only get partial annotations on those queries given that the coverage of the weak supervision is not perfect. We denote the partial annotation as $\tilde{y}$, and to accommodate such partial annotations in bestlink SVM, we modify the margin defined in Eq (4) as follows,

$$\tilde{\Delta}(\tilde{y}_n, \hat{y}, \hat{h}) = |\mathcal{Q}_n| - |\mathcal{C}_n| - \sum_{i,j} h(i,j)\tilde{\sigma}(y,(i,j)) \qquad (5)$$

where $|\mathcal{C}_n|$ is the number of connected components (including singletons) defined by must-links in $\mathcal{Q}_n$, and $\tilde{\sigma}(y,(i,j)) = \lambda^+$ if $\tilde{y}(i) = \tilde{y}(j)$, $\tilde{\sigma}(y,(i,j)) = -\lambda^-$ if $\tilde{y}(i) \neq \tilde{y}(j)$, otherwise $\tilde{\sigma}(y,(i,j)) = 0$ when there is no annotation between query i and j. This modificatoin makes our bestlink SVM a semi-supervised clustering algorithm.

We can easily verify that $\tilde{\Delta}(\tilde{y}_n, \hat{y}, \hat{h})$ is a more general definition of the distance between the given (or partial) task partition and the predicted task partition, in which we count how many edges in $\hat{h}$ are consistent with given annotation (or must-links) in $\tilde{y}$, and how many of them are conflicting with the annotation (or must-/cannot-links). In addition, to distinguish the creditability of the rule-based must-link and cannot-link, we assign them different cost factors, i.e., $\lambda^+ > 0$ and $\lambda^- > 0$, which can be set according to model's performance on a manually annotated held-out set.

6. EXPERIMENT RESULTS

In order to evaluate the proposed method, we performed a series of experiments on a large scale search dataset sampled from the query logs from Bing.com. First, we compared the performance of the proposed bestlink SVM to several state-of-the-art methods for the cross-session search task extraction problem. Then, a set of experiments were conducted to study the effectiveness of using weakly supervised data, which is automatically derived from user query logs, for identifying cross-session search tasks.

6.1 Query Log Dataset

We extracted five days' search logs from Bing.com, from May 27 2012 to May 31 2012, for our experiments. During this period, a subset of users are randomly selected and all their search activities are collected, including the anonymized user ID, query string, timestamp, returned URL sets and the corresponding user clicks. The 30-minutes inactivity threshold is used to segment queries into sessions as pre-processing [14, 23]. Since the focus is identifying cross-session search tasks, we further filtered out the users who submitted less than two queries or had less than two sessions during this period. As a result, we collected 7,628 users with 114,723 queries. The basic statistics of this data set are shown in Table 4.

Table 4: Statistics of evaluation query log data set.

# User	# Session	# Query
7628	37547	114723
Query/User	Session/User	Query/Session
15.1±17.2	4.9±3.5	3.1±1.2

Table 5: Statistics of annotated search tasks.

Single-query Task	Multi-query Task
8044	2283
Multi-session Task	Interleaving Task
1307	709
Task/User	Query/Task*
7.2±10.1	6.6±8.2
Session/Task*	Task duration (mins)*
2.8±2.6	491.1±933.5

∗count only in multi-query tasks

In order to evaluate the performance of the proposed method in identifying cross-session search tasks, three editors were recruited to annotate the search tasks. Editors were instructed to group the queries into tasks according to their understanding of users' information needs, and they were encouraged to use external resources, e.g., search for the logged queries and browse the clicked URLs, to infer the relation between queries. The same set of 200 users' query logs are distributed in each editor's annotation assignment to measure their annotation agreement. Cohen's kappa on pairwise annotation of queries showed high inter-annotator agreement, 0.68, 0.73 and 0.77, for the three pairs of editors. After aggregating the three editors' annotations, we got a collection of 10,327 tasks annotated out of 1,436 users' search logs, and the basic statistics of this data set are shown in Table 5.

From Table 5, we observed that in average a user takes 7.2 different tasks during this period, 22.1% of which contain multiple queries, more than 57.2% multi-query tasks span across session boundaries, and 31.1% of them are interleaving. This shows the need of going beyond session boundaries to extract the long-term search tasks. In particular, when we look into those multi-query tasks, they span 6.6 queries, 2.8 sessions and more than 8 hours in average. This indicates that cross-session task extraction is not a trivial problem, and one needs to leverage rich information for identifying the structure of a cross-session search task.

6.2 Search Task Extraction

6.2.1 Baselines

Several methods have been proposed to identify cross-session search tasks based on the idea of same-task classification [11, 13]. However, those methods only provide predictions over pair of queries, and post-processing is needed to obtain the final task partitions. In our experiment, we adapted two best performing clustering methods from Lucchese et al.'s work [15], i.e., QC_wcc and QC_htc, as the post-processing procedure for the baselines. QC_wcc performs clustering by dropping "weak edges" among queries and extracting the connected components as tasks. QC_htc assumes a cluster of queries can be well represented by only the chronologically first and last query in the cluster, and therefore only the similarity among the first and last queries of two clusters is considered in agglomerative clustering. We trained a linear SVM model to classify if two queries are in the same task, treated the predicted positive query pairs as "strong edges," and applied QC_wcc and QC_htc to obtain the final task partition. In this setting, QC_wcc works exactly the same as Liao et al. proposed in [14].

Since our proposed bestlink-SVM can be viewed as a supervised clustering method [6, 8, 22], we also included two state-of-the-art supervised clustering methods, i.e., "adaptive-clustering" [6] and "cluster-svm" [8] as baselines. Adaptive clustering (AdaptClu) performs single-link agglomerative clustering based on binary classification results. To avoid overfitting, it selects a representative subset of all the candidate pairs based on their similarities when training the binary classifier. In our experiment, we used the summation of all the pairwise similarities as defined in Table 2 between two queries (with negative signs for edit-distance-based similarities) for selecting the representative subset of queries. cluster-svm performs correlation clustering by learning a structural SVM model, which simultaneously optimizes the pairwise accuracy and in-cluster similarity defined by *all-link* in one cluster.

To make a fair comparison, all the methods are trained on the same set of pairwise features defined in Table 2.

6.2.2 Performance metrics

A commonly used evaluation metric for search task extraction is pairwise precision/recall [11, 13] defined as,

$$p_{\text{pair}} = \frac{\sum_{i<j} \delta\big(y(q_i), y(q_j)\big)\delta\big(\hat{y}(q_i), \hat{y}(q_j)\big)}{\sum_{i<j} \delta\big(\hat{y}(q_i), \hat{y}(q_j)\big)} \tag{6}$$

$$r_{\text{pair}} = \frac{\sum_{i<j} \delta\big(y(q_i), y(q_j)\big)\delta\big(\hat{y}(q_i), \hat{y}(q_j)\big)}{\sum_{i<j} \delta\big(y(q_i), y(q_j)\big)} \tag{7}$$

where p_{pair} evaluates how many pairs of queries predicted in the same task, i.e., $\delta\big(\hat{y}(q_i), \hat{y}(q_j)\big) = 1$, are actually annotated as in the same task, i.e., $\delta\big(y(q_i), y(q_j)\big) = 1$; and r_{pair} evaluates how many pairs annotated as in the same task are recovered by the algorithm.

However, it is worth noting that these metrics cannot directly measure the clustering quality, and have some limitations: 1) they ignore singleton tasks, since no pairs can be formed from such tasks; 2) they intrinsically favor methods producing fewer tasks [16]. Inspired by the metrics used in the problem of co-reference resolution in natural language processing, we employed the Constrained Entity-Alignment F-Measure ($f1_{\text{CEAF}}$) as proposed in [16] to evaluate the clustering quality. CEAF defines the clustering precision and recall based on the best alignment between the predicted cluster and ground-truth cluster, where the alignment can be measured by any similarity function defined on two sets:

$$p_{\text{CEAF}} = \frac{\sum_i \pi(\hat{\mathcal{T}}_i, g(\hat{\mathcal{T}}_i))}{\sum_i \pi(\hat{\mathcal{T}}_i, \hat{\mathcal{T}}_i)} \tag{8}$$

$$r_{\text{CEAF}} = \frac{\sum_i \pi(\hat{\mathcal{T}}_i, g(\hat{\mathcal{T}}_i))}{\sum_j \pi(\mathcal{T}_j, \mathcal{T}_j)} \tag{9}$$

where $\pi(A, B)$ is a similarity measure between set A and B, which is chosen to be Jaccard coefficient in our evaluation; and $g(\cdot)$ is the optimal mapping between the predicted task partition $\mathcal{T}$ and ground-truth task partition $\hat{\mathcal{T}}$. Then, $f1_{\text{CEAF}}$ can be calculated as,

$$f1_{\text{CEAF}} = \frac{2 \times p_{\text{CEAF}} \times r_{\text{CEAF}}}{p_{\text{CEAF}} + r_{\text{CEAF}}} \tag{10}$$

Furthermore, we also included Normalized Mutual Information (NMI), a standard metric for evaluating the clustering quality, as one of our evaluation metrics. The detailed definition of NMI can be found in [3]. Basically, the higher the NMI score the better clustering performance an automatic system achieves: NMI= 1 if the prediction is identical to the ground-truth; and NMI= 0 if the prediction is independent from the ground-truth.

6.2.3 Evaluation of search task extraction methods

We randomly split the annotated user query logs into a training set with 712 annotated users, and a testing set with the rest 725 annotated users. The parameters in each model, e.g., C in SVM-based models, are tuned by 5-fold cross-validation on the training set (splitting the annotated users into different folds).

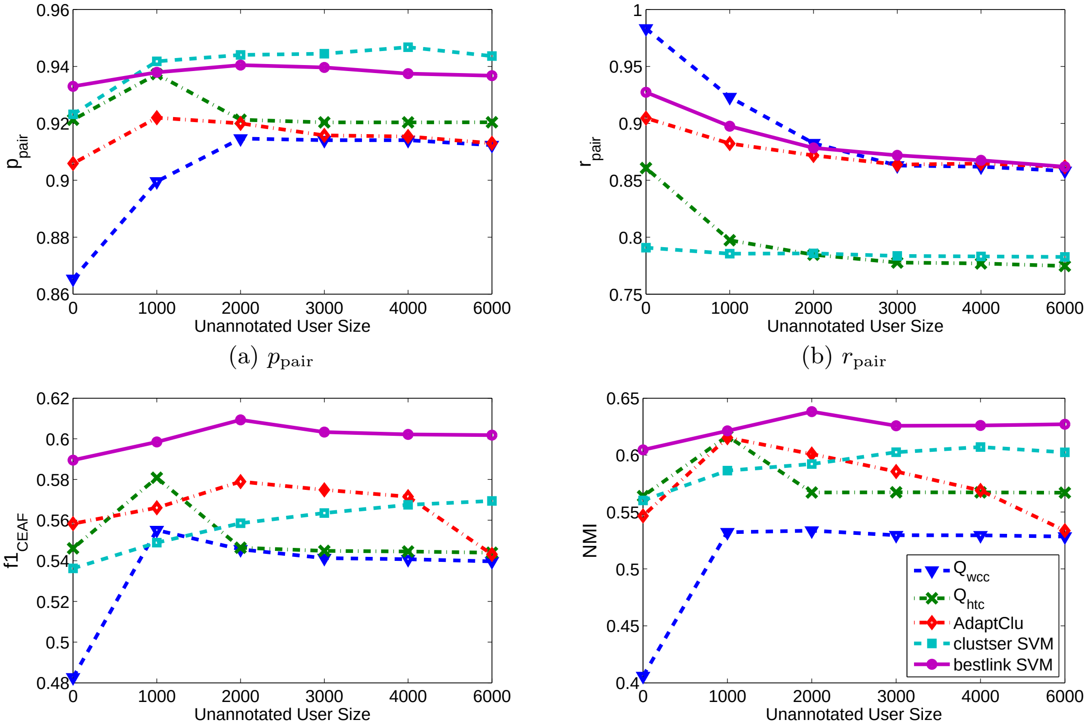
We trained all the methods on the manually annotated training set, and compared their task extraction performance in Table 6, where we averaged the performance under each metric over all the testing cases. A paired two-sample t-test is performed to validate the significance of improvement from the best performing method against the runner-up method under each metric.

Table 6: Search Task Extraction Performance.

	p_{pair}	r_{pair}	$f1_{\text{CEAF}}$	NMI
Q_wcc	0.8653	**0.9833***	0.4826	0.4058
Q_htc	0.9213	0.8607	0.5461	0.5636
AdaptClu	0.9059	0.9046	0.5583	0.5466
cluster-svm	0.9232	0.7908	0.5363	0.5602
bestlink SVM	**0.9330***	0.9273	**0.5895***	**0.6046***
AdaptClu$_{\text{all}}$	0.8681	0.4611	0.2880	0.3236
Rule-based	0.8954	0.5570	-	-

* indicates *p-value*<0.01

In Table 6 we first observed that cluster-svm, which is also a structural learning method, performed much worse than bestlink SVM, especially on r_{pair}. The reason is that cluster-svm optimizes the in-cluster similarity defined by *all-link* among the queries; while in bestlink SVM, the in-cluster similarity is only defined on the *bestlink* among the queries, or more precisely, the edges exist in h (as shown in Eq (2)). To validate this hypothesis, we implemented an additional baseline of *all-link*-based adaptive clustering (AdaptClu$_{\text{all}}$).



(a) p_{pair}

(b) r_{pair}

(c) $f1_{\text{CEAF}}$

(d) NMI

Figure 2: Task extraction performance with increasing volume of weakly supervised data.

In $\text{AdaptClu}_{\text{all}}$, we changed the original single-link agglomerative clustering to all-link agglomerative clustering, where the in-cluster similarity is defined the same as in cluster-svm. As observed in the result, $\text{AdaptClu}_{\text{all}}$ performed significantly worse than AdaptClu, especially on r_{pair}. This result validates our basic modeling assumption in the proposed bestlink SVM, i.e., a query belonging to a particular task should have a strong connection with at least another one query rather than all the other queries in the same task.

Besides, as discussed in Section 2, due to the lack of interaction between the binary classifier training and query clustering in post-processing, the two-step approaches are likely to give suboptimal task extraction performance. Q_wcc and Q_htc are based on the same binary classifier's output, but their performance differs because of distinct strategies used in post-processing. Q_wcc tends to connect all the queries together, and results in a high r_{pair}, but poor performance on other metrics. On the other hand, because Q_htc only compares the first and last queries between two different clusters, it gives a relatively lower r_{pair}, but better clustering performance due to a better p_{pair}, as compared to Q_wcc.

In Section 5, we proposed a method for automatically generating weak supervision from search logs in the form of must-link and cannot-link. In Table 6, we also evaluated the quality of such weak supervision. Since the rule-based supervision merely provides pairwise annotations, we only evaluated its p_{pair} and r_{pair}. In general, p_{pair} of these auto-generated annotations is reasonably good, while r_{pair} is relatively poor. This result is expected: the method described in Table 3 uses strong signals for annotation; but the coverage of such signals is limited, since some relations between two distinct queries can only be inferred by reasoning over the whole query sequence by human judges.

6.2.4 *Effectiveness of weakly supervised data*

To investigate the effectiveness of the weak supervision in helping to train the supervised model, we gradually added the weakly supervised data into our training set. We first obtained the pairwise annotations, as defined in Table 3, for those users who have not been manually annotated; then we gradually added such partially labeled user query logs into the manually-annotated training set. For binary-classification-based baselines, i.e., Q_wcc, Q_htc and AdaptClu, the newly added pairwise annotations are used as regular training supervision; for cluster-svm, the loss function is modified to adopt the partial annotations (similar as Eq (5)). The experimental results are summarized in Figure 2.

From Figure 2 we can study the utility of weakly supervised data on cross-session task extraction. As shown in Figure 2 (c) and (d), the supervised learning methods benefit from a medium volume of weakly supervised data; but when the volume reaches certain limit, the performance stops improving, and even degrades. Figure 2 (a) and (b) help to explain why this happens: all methods' r_{pair} performance drops when adding the weakly supervised data for training, but their p_{pair} performance improves. With the improved p_{pair}, all methods' clustering performance, in terms of $f1_{\text{CEAF}}$ and NMI, gets improved. As shown in Table 6, the weakly supervised data has high precision but low recall, adding more such training signals would bias the models toward recognizing the pairs similar to those high-precision must-links. When the volume of weakly supervised data passes a limit, it will overwhelm the signals from human annotations; and therefore hinders further improvement. Figure 2 also shows that, compared to the two-step methods, the structural learning based method, i.e., cluster-svm and bestlink SVM, can utilize more weakly supervised data before the performance saturates. The reason is that structural learning method directly optimizes (or approximates) the clustering metrics during training. The two-step methods perform classification and clustering independently, and there is inconsistency between training objective and evaluation in these two-step methods. As a result, errors in the learned binary classifier cannot be recovered easily in the clustering stage in those methods.

6.2.5 *Weakly supervised search task extraction*

We are also interested in investigating how well the models could perform when there is only weakly supervised data generated by the proposed must-link and cannot-link. In other words, we want to test if the learning methods' task extraction capability can go beyond the simple annotation rules. In this experiment, we only trained the models on the 6,192 unannotated users with weak supervision, and tested them on the same manually annotated testing set as before. In order to analyze how well the methods generalize from the weakly supervised data, we included a naive baseline Rule-Q_wcc: we adopted Q_wcc by treating the queries connected by the must-links as a task.

Table 7: Task extraction performance when trained only on the weakly supervised data.

	p_{pair}	r_{pair}	$f1_{\text{CEAF}}$	NMI
Rule-Q_wcc	0.9084	0.5136	0.5492	0.5602
Q_wcc	0.9123	0.8582	0.5397	0.5285
Q_htc	0.9204	0.7747	0.5440	0.5669
AdaptClu	0.9131	**0.8613***	0.5426	0.5325
cluster-svm	0.9155	0.7565	0.5197	0.4805
bestlinkSVM	**0.9334***	0.8161	**0.5676***	**0.5893***

* indicates *p-value*<0.01

As shown in Table 7, all the methods improved p_{pair} and r_{pair} against Rule-Q_wcc, and especially for r_{pair}. However, not all of them can improve the clustering quality metric: besides bestlink SVM, only Q_htc improves NMI metric. We looked into the detailed output of those methods and found that: Rule-Q_wcc generated many singleton tasks because of the low coverage of must-links; the baseline models merged some of the small clusters into larger ones, but they still created too many smaller clusters than ground-truth. bestlink SVM further merged the small clusters correctly, making the number of predicted tasks closest to the ground-truth, and therefore it achieved better clustering performance.

We wanted to further investigate how many "complex tasks," which are not covered by the must-links defined in Table 3, can be extracted by learning from the weak supervision. Specifically, we define the complex task as: $\mathcal{T}^*_{\text{strict}}$, in which no must-link can be applied on any pair of queries in it (strict criterion); or $\mathcal{T}^*_{\text{loose}}$, there exists at least one pair of queries cannot be connected via must-links in it (loose criterion). Based on this notation, we define the coverage of complex task as the proportion of complex tasks which can be perfectly recovered by the automatic methods,

$$c_{\text{loose}} = \frac{\sum_{\mathcal{T}_i \in \hat{\mathcal{T}}} \sum_{\mathcal{T}_j \in \mathcal{T}^*_{\text{loose}}} \delta(\mathcal{T}_i, \mathcal{T}_j)}{|\mathcal{T}^*_{\text{loose}}|} \tag{11}$$

$$c_{\text{strict}} = \frac{\sum_{\mathcal{T}_i \in \hat{\mathcal{T}}} \sum_{\mathcal{T}_j \in \mathcal{T}^*_{\text{strict}}} \delta(\mathcal{T}_i, \mathcal{T}_j)}{|\mathcal{T}^*_{\text{strict}}|} \tag{12}$$

where $\delta(\mathcal{X}, \mathcal{Y}) = 1$ when the set $\mathcal{X}$ and $\mathcal{Y}$ are the same, and otherwise $\delta(\mathcal{X}, \mathcal{Y}) = 0$.

In this experiment, we used all the 1436 annotated users as testing set, where we collected 357 strict complex tasks and 1540 loose complex tasks out of the total 2283 multi-query tasks. All the models are trained on the rest 6192 unannotated users with weak supervision, and the experimental results are list in Table 8, where we used sign-test for validating the improvement over the baselines.

We should note that all those complex tasks cannot be identified by the straight-forward Rule-Q_wcc baseline, so that the newly defined task coverage metric measures how well the learning methods can generalize from the weak supervision. From the results we can notice that bestlink SVM, which achieved the best performance against all the other baselines, can successfully recover about 30% of complex tasks by leveraging the knowledge from weak supervision, which validates the effectiveness of using such signals as supervision for model training.

Table 8: Coverage of complex tasks when trained only on the weakly supervised data.

	c_{loose}	c_{strict}
Q_wcc	0.2914	0.2745
Q_htc	0.2617	0.2761
AdaptClu$_{\text{single}}$	0.2837	0.2717
cluster-svm	0.2883	0.2997
bestlinkSVM	**0.3207***	**0.3501***

* indicates *p-value*<0.01

6.3 Feature Weights in bestlink SVM

In order to understand which similarity features are important for the problem of cross-session task extraction, we list the top two positive and top two negative features learned by the proposed bestlink SVM under each category of pairwise similarity features defined in Table 9. To avoid bias introduced by weak supervision, we only demonstrated the weights learned from the manually annotated training set.

Table 9: Top 2 positive and top 2 negative features under each type of pairwise similarity features in bestlink SVM model.

Feature	Weight
Q-COSINE	5.30
Q-JAC	1.51
U-JAC-ALL	4.53
U-SIM-AVG	3.05
S-SAME	1.00
S-DIST	0.60
Q-DIST	-3.38
Q-EDIT	-2.73
U-EDIT-DOMAIN-AVG	-1.39
U-EDIT-ALL-CLICK-AVG	-0.83
S-FIRST	-0.28

As can be noticed in Table 9, Q-COSINE has the largest importance weight for identifying queries belonging to the same task; and U-JAC-ALL is also very informative for recognizing the similar queries. Besides, we found that bestlink SVM assigns relatively low positive weight to S-SAME, and Q-TIME is not the most important feature in the model. The reason is that we already knew 12.7% tasks span cross session boundaries (as shown in Table 5), and placing too large a weight on S-SAME and Q-TIME will forbid the method from identifying those cross-session tasks.

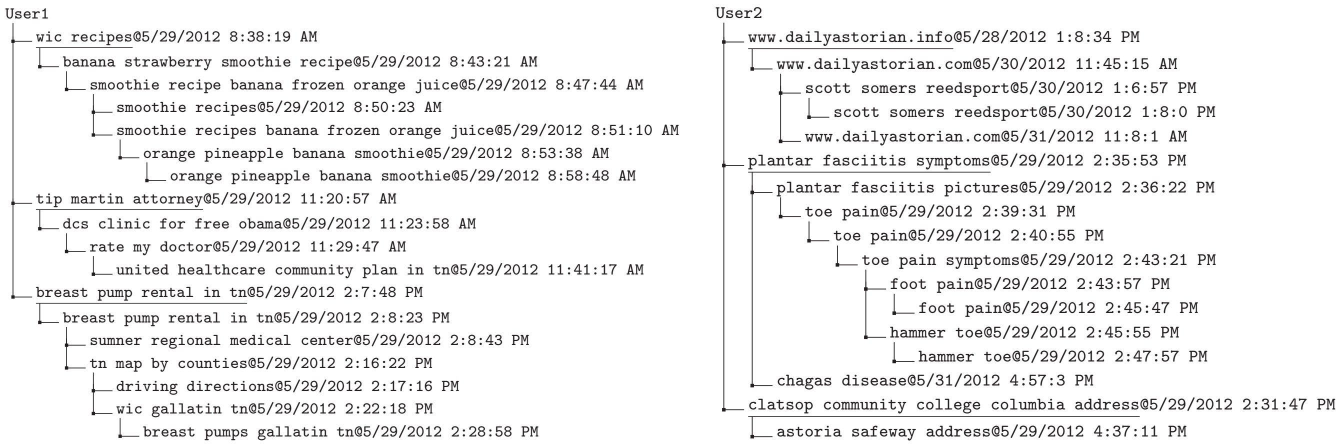

User1
wic recipes@5/29/2012 8:38:19 AM
banana strawberry smoothie recipe@5/29/2012 8:43:21 AM
smoothie recipe banana frozen orange juice@5/29/2012 8:47:44 AM
smoothie recipes@5/29/2012 8:50:23 AM
smoothie recipes banana frozen orange juice@5/29/2012 8:51:10 AM
orange pineapple banana smoothie@5/29/2012 8:53:38 AM
orange pineapple banana smoothie@5/29/2012 8:58:48 AM
tip martin attorney@5/29/2012 11:20:57 AM
dcs clinic for free obama@5/29/2012 11:23:58 AM
rate my doctor@5/29/2012 11:29:47 AM
united healthcare community plan in tn@5/29/2012 11:41:17 AM
breast pump rental in tn@5/29/2012 2:7:48 PM
breast pump rental in tn@5/29/2012 2:8:23 PM
sumner regional medical center@5/29/2012 2:8:43 PM
tn map by counties@5/29/2012 2:16:22 PM
driving directions@5/29/2012 2:17:16 PM
wic gallatin tn@5/29/2012 2:22:18 PM
breast pumps gallatin tn@5/29/2012 2:28:58 PM
User2
www.dailyastorian.info@5/28/2012 1:8:34 PM
www.dailyastorian.com@5/30/2012 11:45:15 AM
scott somers reedsport@5/30/2012 1:6:57 PM
scott somers reedsport@5/30/2012 1:8:0 PM
www.dailyastorian.com@5/31/2012 11:8:1 AM
plantar fasciitis symptoms@5/29/2012 2:35:53 PM
plantar fasciitis pictures@5/29/2012 2:36:22 PM
toe pain@5/29/2012 2:39:31 PM
toe pain@5/29/2012 2:40:55 PM
toe pain symptoms@5/29/2012 2:43:21 PM
foot pain@5/29/2012 2:43:57 PM
foot pain@5/29/2012 2:45:47 PM
hammer toe@5/29/2012 2:45:55 PM
hammer toe@5/29/2012 2:47:57 PM
chagas disease@5/31/2012 4:57:3 PM
clatsop community college columbia address@5/29/2012 2:31:47 PM
astoria safeway address@5/29/2012 4:37:11 PM

Figure 3: Identified latent search task structure.

6.4 Analysis of Identified Tasks

As we have discussed in Section 4.2, the latent structure h defined in bestlink SVM is a tree formed by strong connections between queries, where each subtree of the dummy query q_0 corresponds to a search task. In Figure 3, we illustrated the latent task structure inferred by our bestlink SVM from two different users' query logs.

Comparing to the flat clustering structure given by the traditional search task extraction methods [13, 15], the hierarchical structure inferred by bestlink SVM provides us with more in-depth details to understand users' search behaviors and their information needs. For example, in Figure 3 we can clearly notice that the identified task structure for User2 is more complex than that for User1: User1 attempted three consecutive tasks on May 29; while User2's two major search tasks, i.e., checking daily news and looking for solutions of her health issue, spanned from May 28 to May 31, and were performed in an interleaved manner. And the subtrees in an identified search task represent finer grained subtasks. For instance, as shown in Figure 3, in User2's second identified task of "plantar fasciitis symptoms," there are two subtasks, one starts with "plantar fasciitis pictures" and another starts with "chagas disease."

At the beginning of Section 6, we listed a brief overview of basic properties of search tasks based on a limited number of human annotations. Now we can get a more comprehensive understanding of user's search behaviors based on the automatically extracted search tasks in our whole query log data set. We listed a set of statistics in Table 10, where we applied a proprietary multi-class classifier to categorize a query into 80 different categories, e.g., navigational, commerce, celebrity and etc., in order to annotate the search intent of queries.

As shown in Table 10, user's search intent in each extracted task is quite concentrated: despite the fact that there are in average 4.41 queries in a task, there are only 1.47 different intents. Particularly, when the user's intent is purely navigational, the task will get mostly simplified: only 1.38 unique queries per task. And more than 25% identified tasks only contain navigational queries. Another interesting phenomenon we found is the transition probability between the navigational and non-navigational queries, which is estimated within the identified tasks, is quite different: the chance a user issues a non-navigational query after a navigational query is much lower than the opposite direction. One possible explanation for this is that when user issues a non-navigational query, they usually do not have a clear sense of where to find the information yet, so they are more likely to keep submitting the questions to the search engine; but when they have specific destination in mind, they would start to issue questions to explore more perspectives of the information they are interested in.

Table 10: Statistics of extracted search tasks.

Query/Task	UniQuery/Task
4.41±7.48	2.80±4.04
Intent/Task	% of NavTask
1.47±1.20	25.37
Query/NavTask	UniQuery/NavTask
2.45±2.67	1.38±0.80
P(non-nav|nav)	P(nav|non-nav)
0.288	0.124

7. CONCLUSIONS

Search tasks frequently span multiple sessions, and thus developing methods to extract these tasks from historic data is central to understanding longitudinal search behaviors and in developing search systems to support users' long-running tasks. In this paper, we have presented a novel method for learning to accurately extract cross-session search tasks from users' historic search activities. We developed a semi-supervised clustering model based on the latent structural SVM framework, which is capable of learning inter-query dependencies from users' searching behaviors. A set of effective automatic annotation rules are proposed as weak supervision to release the burden of manual annotation. Comprehensive experimentation using large-scale search logs from a commercial search engine demonstrated the superior performance of our method in identifying cross-session search tasks versus a number of state-of-the-art algorithms. Importantly, we were able to obtain performance gains while reducing the reliance on costly human annotations via the automatically generated weak supervision. The results are promising and pave the way for a range of future work in this area, including user modeling and long-term task based personalization.

8. REFERENCES

[1] E. Agichtein, R. W. White, S. T. Dumais, and P. N. Bennet. Search, interrupted: understanding and predicting search task continuation. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 315–324. ACM, 2012.

[2] P. Anick. Using terminological feedback for web search refinement: a log-based study. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 88–95. ACM, 2003.

[3] D. Cai, X. He, X. Wang, H. Bao, and J. Han. Locality preserving nonnegative matrix factorization. In *IJCAI'09*, pages 1010–1015, 2009.

[4] L. D. Catledge and J. E. Pitkow. Characterizing browsing strategies in the world-wide web. *Computer Networks and ISDN systems*, 27(6):1065–1073, 1995.

[5] M. Chang, D. Goldwasser, D. Roth, and V. Srikumar. Structured output learning with indirect supervision. In *ICML'10*, 2010.

[6] W. W. Cohen and J. Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 475–480. ACM, 2002.

[7] S. Cucerzan and E. Brill. Spelling correction as an iterative process that exploits the collective knowledge of web users. In *Proceedings of EMNLP*, volume 4, pages 293–300, 2004.

[8] T. Finley and T. Joachims. Supervised clustering with support vector machines. In *Proceedings of the 22nd international conference on Machine learning*, pages 217–224. ACM, 2005.

[9] D. He, A. Göker, and D. J. Harper. Combining evidence for automatic web session identification. *Information Processing & Management*, 38(5):727–742, 2002.

[10] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[11] R. Jones and K. L. Klinkner. Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 699–708. ACM, 2008.

[12] R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *Proceedings of the 15th international conference on World Wide Web*, pages 387–396. ACM, 2006.

[13] A. Kotov, P. N. Bennett, R. W. White, S. T. Dumais, and J. Teevan. Modeling and analysis of cross-session search tasks. *SIGIRŠ11*, pages 5–14, 2011.

[14] Z. Liao, Y. Song, L.-w. He, and Y. Huang. Evaluating the effectiveness of search task trails. In *Proceedings of the 21st international conference on World Wide Web*, pages 489–498. ACM, 2012.

[15] C. Lucchese, S. Orlando, R. Perego, F. Silvestri, and G. Tolomei. Identifying task-based sessions in search engine query logs. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 277–286. ACM, 2011.

[16] X. Luo. On coreference resolution performance metrics. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 25–32. Association for Computational Linguistics, 2005.

[17] F. Radlinski and T. Joachims. Query chains: learning to rank from implicit feedback. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 239–248. ACM, 2005.

[18] D. Shen, J.-T. Sun, Q. Yang, and Z. Chen. Building bridges for web query classification. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 131–138. ACM, 2006.

[19] C. Silverstein, H. Marais, M. Henzinger, and M. Moricz. Analysis of a very large web search engine query log. In *ACM SIGIR Forum*, volume 33, pages 6–12. ACM, 1999.

[20] A. Spink, M. Park, B. Jansen, and J. Pedersen. Multitasking during web search sessions. *Information Processing & Management*, 42(1):264–275, 2006.

[21] V. Vapnik. *The nature of statistical learning theory.* springer, 1999.

[22] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *ICML'01*, pages 577–584, 2001.

[23] R. W. White and S. M. Drucker. Investigating behavioral variability in web search. In *Proceedings of the 16th international conference on World Wide Web*, pages 21–30. ACM, 2007.

[24] C.-N. J. Yu and T. Joachims. Learning structural svms with latent variables. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1169–1176. ACM, 2009.