Computational Linguistics and Chinese Language Processing
Vol. 21, No. 1, June 2016, pp. 69-90

Identifying the Names of Complex Search Tasks with Task-Related Entities

Ting-Xuan Wang[*] and Wen-Hsiang Lu[*]

Abstract

Conventional search engines usually consider a search query corresponding only to a simple task. Nevertheless, due to the explosive growth of web usage in recent years, more and more queries are driven by complex tasks. A complex task may consist of multiple sub-tasks. To accomplish a complex task, users may need to obtain information of various task-related entities corresponding to the sub-tasks. Users usually have to issue a series of queries for each entity during searching a complex search task. For example, the complex task “travel to Beijing” may involve several task-related entities, such as “hotel room,” “flight tickets,” and “maps”. Understanding complex tasks with task-related entities can allow a search engine to suggest integrated search results for each sub-task simultaneously. To understand and improve user behavior when searching a complex task, we propose an entity-driven complex task model (ECTM) based on exploiting microblogs and query logs. Experimental results show that our ECTM is effective in identifying the comprehensive task-related entities for a complex task and generates good quality complex task names based on the identified task-related entities.

Keywords: Complex Search Task, Task Name Identification, Task-related Entity.

1. Introduction

Conventional search engines usually consider single queries corresponding only to a simple search need. In reality, however, more and more queries are driven by complex search tasks (Guo & Agichtein, 2010; Jones & Klinkner, 2008). Generally, a real-life complex search task usually has more than one sub-task to be accomplished. Therefore, users usually cannot accomplish a complex search task by submitting only a single query. Some researchers have worked to try to identify sub-tasks in order to help users deal with complex search tasks (Tan

[*] Departmenr of Computer Science and Information Engineering, National Cheng Kung University, Tainan, Taiwan
E-mail: playif@gmail.com; whlu@mail.ncku.edu.tw

et al., 2006; MacKay *et al*., 2008; Ji *et al*., 2011; Kotov *et al*., 2011; Yamamoto *et al*., 2012). Nevertheless, only identifying sub-tasks in a complex search task is not sufficient to help users who want to search the complex task name directly *e.g.*, "北京旅遊 (travel to Beijing)". Understanding complex task names for complex search tasks can help search engines deal with complex-task-based queries. A complex search task can be represented by a task event and a task topic. For example, as shown in Figure 1, a complex task "travel to Beijing," which is composed of a task topic "Beijing" and a task event "travel," has at least three sub-tasks, including "book flight ticket," "reserve hotel room," and "survey maps". Users need to issue at least three queries for each sub-task including "Beijing flight ticket," "Beijing hotel," and "Beijing map". The queries targeting a sub-task usually focus on a task-related entity, such as "flight ticket," "hotel room," and "maps". Therefore, understanding task-related entities is very important for a complex task and can help search engines provide integrated search results containing a variety of information of distinct task-related entities.

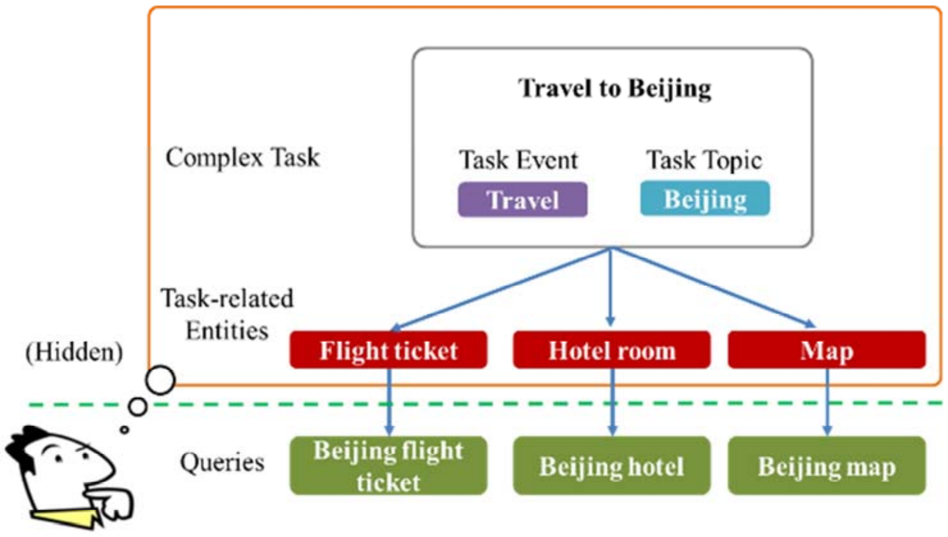

Figure 1. The structure of a complex task with task-related entities and search queries.

When users search for a complex task, we have found the users often have a task event that triggers the users to perform exploratory or comparative search behaviors, such as "prepare something," "buy something," or "travel somewhere". Furthermore, the search behaviors are usually around a certain task topic that is the subject of interest in the complex task. Users may describe the task event and task topic of their complex task with various task-related entities in microblogs, *e.g.*, Twitter or Weibo[1]. Microblogs are a miniature version

[1] Weibo: http://weibo.com

of traditional weblogs. In recent years, many users have posted and shared their life details with others on microblogs every day. Due to the post length limitation (only 140 characters for Weibo), users tend to describe only key points of their life task. Table 1 shows an example of a microblog. We find the user, who has an ongoing complex task “北京旅遊 (travel to Beijing),” mentioned two task-related entities “機票 (flight ticket)” and “飯店 (hotel)”.

Table 1. A microblog post from Weibo mentioning an ongoing complex task “travel to Beijing”

Chinese	English Translation
今天已經訂好**機票**,只剩下找間**飯店**，就等著下禮拜去**北京旅遊**了~好期待!	I have already booked a **flight** today, and I only have to find a **hotel**. I’m about to **travel to Beijing** next week - good anticipation!

To understand and model a complex search task, some researchers have analyzed long and short-term user search behavior based on a single user’s search sessions (Agichtein *et al*., 2012; Liao *et al*., 2012; Mihalkova & Mooney, 2009; Tan *et al*., 2006). Nevertheless, a single user’s search session from query logs may not be sufficient in identifying complex search tasks since complex tasks may cross search sessions or be interleaved in a single search session. In this work, we address the problem of how to help users efficiently accomplish a complex task when submitting a single query or multiple queries. To complete the scenario illustrated in Figure 1, a complex task name with several task-related entities must be identified. Basically, the problem can be divided into the following three major sub-problems.

1. Collect queries related to the same complex search task.
2. Extract task-related entities from the collected queries.
3. Automatically identify the name of the complex search task.

The above three problems are very important but non-trivial to solve. We propose an entity-driven complex task model (ECTM) to automatically identify complex task names and task-related entities based on various web resources. To evaluate our proposed ECTM, we have conducted extensive experiments on a large dataset of real-world query logs. The experimental results show that our ECTM is able to identify a comprehensive task name for a complex task with related entities and generate good quality complex task names based on the identified task-related entities.

2. Related Work

Search session partition: Recent studies show that about 75% of search sessions are searching for complex tasks (Field & Allan, 2013). To help users deal with their search tasks, researchers have devoted effort to understand and identify tasks from search sessions. Boldi *et al.* (2008) proposed a graph-based approach to dividing a long-term search session into search tasks. Guo and Agichtein (2010) investigated the hierarchical structure of a search task with a series of search actions based on search sessions. Agichetin *et al*. (2012) conducted a comprehensive analysis of search tasks and classified them based on several aspects, such as intent, motivation, complexity, work-or-fun, time-sensitive, and continued-or-not. Beeferman and Berger (2000) proposed a graph-based iterative approach to clustering query logs. Wen *et al*. (2001) clustered similar queries based on query content and document clicks. Cui *et al*. (2011) grouped search queries from a click-through log with similar search tasks using the random walk method. Unfortunately, exploring only the query content and click-through information for query clustering may not obtain precise results since queries usually are short and ambiguous. Note that the above works only focus on single-goal task discovery, while our work focuses on identifying a complex task with relevant sub-tasks over a series of sessions.

Cross-session task prediction: Kotov *et al*. (2011) noticed that a complex task may require a user to issue a series of queries, spanning a long period of time and multiple search sessions. Thus, they addressed the problem of modeling and analyzing complex cross-session search tasks. Lucchese *et al*. (2011) tried to identify task-based sessions in query logs by semantic-based features extracted from Wiktionary and Wikipedia to overcome a lack of semantic information. Ji *et al*. (2011) proposed a graph-based regularization algorithm to predict popular search tasks and simultaneously classify queries and web pages by building two content-based classifiers. White *et al*. (2013) improved the traditional personalization methods for search-result re-ranking by exploiting similar tasks from other users to re-rank search results. Wang *et al*. (2011) addressed the problem of extracting cross-session tasks and proposed a task partition algorithm based on several pairwise similarity features. Raman *et al*. (2013) investigated intrinsic diversity (ID) for a search task and proposed a re-ranking algorithm according to the ID tasks.

Complex task search: To discriminate between simple and complex tasks, we define simple tasks as triggering only one sub-task. Complex tasks may trigger more than one sub-task. A complex task consists of several sub-tasks, and each sub-task goal may be composed of a sequence of search queries. Therefore, modeling the sub-tasks is necessary for identifying a complex task. Jones and Klinkner (2008) proposed a classification-based method to divide a single search session into tasks and sub-tasks based on the four types of features, including time, word, query log sequence, and web search. Lin *et al.* (2012) defined a search goal as an action-entity pair and utilized a web trigram to identify fine-grained search goals. Jones, Yamamoto, *et al*. (2012) proposed an approach to mining sub-tasks for a task using query clustering based on bid phrases provided by advertisers. The most important difference between our work and previous works is that we further try to identify task names with related task-intrinsic entities. To the best of our knowledge, there is no existing approach to utilizing microblogs in dealing with task identification and identifying human-interpretable names. In this work, we propose an entity-driven complex task model (ECTM) to deal with the problems mentioned above. To the best of our knowledge, there is no existing approach to utilizing multiple resources in dealing with task identification and identifying human-interpretable names for complex search tasks with various task-related entities.

3. Entity-driven Complex Task Model

3.1 System Architecture

To improve current search engines without support for complex task searches, we proposed an entity-driven complex task model (ECTM), which can automatically identify the name of a complex task and discover task-related entities behind the complex task. Figure 2 shows our ECTM architecture. The ECTM utilizes web resources, including query logs and microblogs. Given a search query that is driven by a latent complex task, there are three major stages in the ECTM to suggest integrated search results.

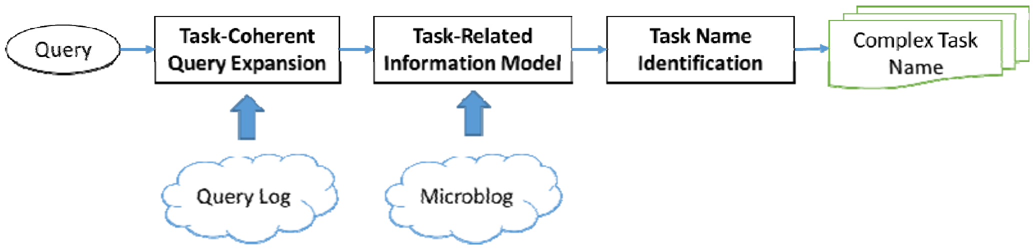

Figure 2. Architecture of our proposed entity-driven complex task model.

1. **Task-Coherent Query Expansion:** Integrate an expanded task-coherent query set to help identify the latent complex search task. In this stage, we utilize query logs and user search sessions to collect task-coherent queries.

2. **Task-Related Information Model:** Extract multiple task-related entities from a task-coherent query set, then retrieve microblog posts based on extracted task-related entities.

3. **Task Name Identification:** Identify the complex task name consisting of a task topic and a task event extracted from the retrieved microblog posts.

In the following, we describe the details of each major stage in the ECTM.

3.2 Task-Coherent Query Expansion

In fact, only using a single query is insufficient for finding the latent complex task. We thus try to extract other relevant task-coherent queries from search sessions. Although users may persistently search for the same complex task over a period of time, they may also simultaneously search for multiple interleaved tasks (Liu & Beklin, 2010; MacKay & Watters, 2008). Therefore, identifying task-coherent queries from search sessions is an important and non-trivial issue. The process for extracting task-coherent queries is described below.

Given an input query q and query logs Q, we first separate queries in the query logs into search sessions by the time gap of 24 hours. We extract search sessions containing the input query q and obtain a set of sessions S_q. To extract task-coherent queries Q_t from the session set S_q, we employ a log-linear model (LLM) with the following three useful features.

Average Query Frequency: Generally, the frequency of a query reflects its popularity. To avoid a long session resulting in high query frequency, we calculate the normalized query frequency as:

$$f_{AQFrequency}(q_t) = \frac{1}{|\boldsymbol{S}_{q_t}|} \times \sum_{s \in \boldsymbol{S}_{q_t}} \frac{freq(q_t, s)}{|s|} \quad (1)$$

where $freq(q_t,s)$ is the frequency of the query q_t in the session s, $\boldsymbol{S}_{q_t}$ is the sessions containing q_t, $|s|$ is the number of queries in the session s, and $|\boldsymbol{S}_{q_t}|$ is the number of sessions containing query q_t in the set $\boldsymbol{S}_{q_t}$.

Session Coverage: The queries occurring in several sessions are candidates in terms of task-coherence. To collect queries occurring in many sessions, we use average session frequency, which can be calculated as follows:

$$f_{ASFrequency}(q_t) = \exp\left(\frac{|\boldsymbol{S}_{q_t}|}{|\boldsymbol{S}_q|}\right) \quad (2)$$

where $|\boldsymbol{S}_q|$ is the number of sessions containing the input query q in the set $\boldsymbol{S}_q$, $|\boldsymbol{S}_{q_t}|$ is the number of sessions containing query q_t in the set $\boldsymbol{S}_{q_t}$, and $\exp(\cdot)$ is the exponential function.

Average Query Distance: Since queries that close to the input query in a search session may have high task-coherence degree for the latent complex task. We thus use normal distribution to estimate the task-coherence for each query:

$$f_{AQDistance}(q_t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{d^2}{2\sigma^2}} \quad (3)$$

where σ is standard deviation (which is set to 6.07, according to our training dataset, see Section 4.1.2), d is the average number of queries between q_t, and input query q in sessions.

We employ a log-linear model to calculate the probability of each candidate task-coherent query based on the features described above:

$$P(q_t; \boldsymbol{W}) = \frac{\exp(\sum_{i=1}^{|F|} w_i f_i(q_t))}{Z(\boldsymbol{Q}_t)} \quad (4)$$

where $\boldsymbol{Q}_t$ is the set of all candidate queries in the session set $\boldsymbol{S}_q$, $|F|$ is the number of used feature functions $f_i(q_t)$, $\boldsymbol{W}$ is the set of weighting parameters w_i of feature functions, and $Z(\boldsymbol{Q}_t)$ is a normalizing factor set to the value $Z(\boldsymbol{Q}_t) = \sum_{q_t \in \boldsymbol{Q}_t} \exp(\sum_{i=1}^{|F|} w_i f_i(q_t))$.

3.3 Task-Related Information Model

Sometimes, complex search task names will not occur in the expanded query set $\boldsymbol{Q}_t$; therefore, we cannot directly identify a complex task name from the query set $\boldsymbol{Q}_t$. In this stage, we expand the content of $\boldsymbol{Q}_t$ by extracting task-related entities and use the entities to retrieve microblog posts from a microblog search service, such as Weibo[2].

3.3.1 Task-Related Entity Extraction

We first try to extract task-related entities from expanded task-coherent query set $\boldsymbol{Q}_t$. In fact, the queries in the query set $\boldsymbol{Q}_t$ usually consist of a task topic and a task-related entity. For example, a query "北京機票 (Beijing flight ticket)" contains a topic "北京 (Beijing)" and an entity "機票 (flight ticket)". To realize the Part-Of-Speech (POS) of the task-related entity in the query set $\boldsymbol{Q}_t$, we generated statistics on 2000 queries randomly selected from a query log. The entities were labeled with a POS tag by a Chinese segmentation and tagging tool. Table 2 shows the results of the POS tag distribution of queries. We find that most entities are common nouns (87.5%), such as cellphone or flight ticket. For the POS of the task topic, we find that 78.9% of task topics are proper nouns and 19.8% are common nouns. Therefore, we extract task-related entities from the query set $\boldsymbol{Q}_t$ by extracting all common nouns in each set of queries and select the top-frequency proper noun as a candidate task topic. We thus can obtain a candidate task topic and a list of task-related entities $\boldsymbol{E}_t$ ordered by the occurrence frequency.

Table 2. The POS tag distribution of task entities and task topics

POS tag	Entity	Topic
Common Noun	87.5%	19.8%
Proper Noun	7.3%	78.9%
Others	5.2%	1.3%

3.3.2 Task-Related Microblog Retrieval

To identify a complex task name that does not occur in queries, the basic idea is to collect microblog posts from microblog search engines based on the given task-related entities. According to our observations, a microblog post containing most of the task-related entities may also contain the task name (see the example in Table 1). Unfortunately, a microblog post may contain only a portion of the entities required for a complex task. To overcome the above problem, we identify pseudo queries based on all subsets containing two or three entities from

[2] http://s.weibo.com/weibo/%E5%8C%97%E4%BA%AC%E6%97%85%E9%81%8A

top-*n* entities of $\boldsymbol{E}_t$. To make sure that each pseudo query is relevant to the candidate topic *t*, we combine the candidate topic *t* with each pseudo query and retrieve a set of microblog posts via microblog search engines.

3.4 Task Name Identification

Based on the identified task-related entities and microblog posts in the previous stage, we aim to identify a complex task name. To identify a suitable task name, we utilize conditional random field (CRF) (Lafferty *et al.*, 2001) to automatically label each term in a microblog post with a task-semantic tag.

3.4.1 Automatically Labeling of Task Name

To realize the structure of a complex task name, we annotated 244 distinct complex task names from 513 search sessions (the details of the annotation process will be described in Section 4.1.2). Table 3 shows the statistics of the structure distribution. We found most task names consist of a task topic and a task event, such as “購買三星手機 (buy Samsung cellphone),” where “三星 (Samsung)” is the task topic, and “購買手機 (buy cellphone)” is the task event. We also find that the task event usually is composed of a transitive verb (*i.e.*, buy) and an event object (*i.e.*, cellphone). Nevertheless, some events are intransitive verbs needing no event object, such as “英語學習 (English learning),” where ”學習 (learning)” is an intransitive verb in Chinese. Therefore, we define the two types of events as Event 1 and Event 2, where Event 1 consist of a transitive verb (E_{1V}) and an object (E_{1O}), and Event 2 is only an intransitive verb. We aim to automatically label each term in a microblog post with one of the five task-semantic tags, *T* (topic), E_{1V} (Event 1), E_{1O} (Event object), E_2 (Event 2), and *O* (Others).

Table 3. The statistics of complex task name structure distribution

Task Name Pattern	Percentage	Example
Topic + Event 1	54.92%	購買三星手機 (Buy Samsung cellphone)
Topic + Event 2	40.16%	英語學習 (English learning)
Others	4.92%	--

To automatically label each term with a task-semantic tag, we employ a supervised probabilistic graphical model, conditional random field (CRF), which is suitable to predict the latent structure of sentences [10]. CRF can predict sequences of labels for sequences of terms in a sentence. We use a popular CRF implementation “CRF++,”[3] which can adopt multiple features for each term. In the following, we describe the two types of features for complex task name identification, including term-based and post-based features.

3.4.2 Features for Complex Task Name Identification

(1) Term-based features

There are five term-based features proposed in this work.

Stop word: A stop word usually is unimportant and not a task name. We consider stop word as a binary feature to indicate if the term is a task name.

Candidate topic: In the previous stage (see Section 3.3.1), we extracted a candidate topic from task-coherent query set $\boldsymbol{Q}_t$. Therefore, we can utilize the candidate topic as a binary feature for indicating if a term is a task topic.

Term frequency: Generally, a term in a post with high frequency may indicate the term is more important than other terms in the post. The term frequency is normalized by dividing the largest frequency of terms in the post. The normalized term frequency is divided into three ranges, including high [1, 0.8), middle (0.2, 0.8], and low [0, 0.2].

Document frequency: A term occurring in several search-result posts may indicate the term is a task topic or a task event. The reason is that the search-result posts usually are related to a certain complex task. We normalize the document frequency by dividing the post number of search results. The normalized document frequency is divided into three intervals, including high [1, 0.8), middle (0.2, 0.8], and low [0, 0.2].

POS tag: According to our observation, the POS tag of a task topic usually is a proper noun (N_p) or a common noun (N_c), and the POS tag of a task event usually is a transitive verb (V_t) + common noun (N_c) or an intransitive verb (V_i). To enhance the accuracy of the CRF model, we only use four types of POS tag “V_i,” “V_t,” “N_c,” and “N_p” and others are labeled as “Others”.

(2) Post-based Features

According to our observation, a post describing a complex task usually is more important or popular for an author or the author’s friends. For example, when users write posts talking about their wedding, they will receive more attention than other ordinary posts. Therefore, we try to calculate a post importance score based on four post features, including descriptive, interactive, attractive, and influential degrees, according to the metadata of microblog posts. Figure 3 shows a real example of a microblog post collected from Weibo. We can see that there is some metadata on the post, such as the click times of “like,” “share,” and “comment”.

[3] CRF++: http://crfpp.googlecode.com/svn/trunk/doc/index.html

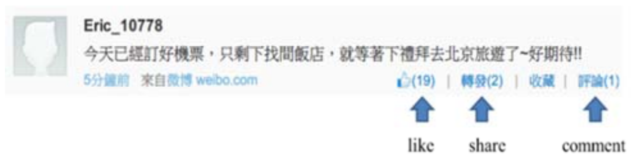
Eric_10778
今天已經訂好機票，只剩下找間飯店，就等著下禮拜去北京旅遊了~好期待!!
5分鐘前 來自微博 weibo.com
(19) | 轉發(2) | 收藏 | 評論(1)

Figure 3. A real example post containing various metadata, such as the times of "like," "share," and "comment"

Descriptive degree: Generally, a complex task needs more words to describe a variety of subtasks. Thus, we assume that a microblog post p with longer context can provide more content about the complex task. Nevertheless, some spam posts may contain long text with repeated terms. Therefore, we calculate the entropy to represent post descriptive degree:

$$F_{Descriptive}(p) = -\sum_{w \in Term(p)} P(w) \log_2 P(w) \tag{5}$$

where *Term*(p) is a set of terms in post p and $P(w)$ is the occurrence probability of term w.

Interactive degree: If a post has many "comments," we assume the post is more interactive. An interactive post has higher probability of mentioning a complex task. We formulate the interactive degree of a post as:

$$F_{Interactive}(p) = \frac{CommentCount(p)}{\max_{p_i \in P} CommentCount(p_i)} \tag{6}$$

where *CommentCount*(p) is the "comment" number of a post p and where $\max(\cdot)$ function returns the max "comment" number of a post p_i in the post set P.

Attractive degree: If a post receives many "likes," we assume the post is more attractive. An attractive post has higher probability to mention a complex task. We formulate the attraction of a post as:

$$F_{Attractive}(p) = \frac{LikeCount(p)}{\max_{p_i \in P} LikeCount(p_i)} \tag{7}$$

where *LikeCount*(p) is the "like" number of a post p and where $\max(\cdot)$ function returns the max "like" number of a post p_i in the post set P.

Influential degree: If a post was shared many times, we assume the post is more influential. An influential post has higher probability to mention a complex task. We formulate the influential degree of a post as:

$$F_{Influential}(p) = \frac{ShareCount(p)}{\max_{p_i \in P} ShareCount(p_i)} \tag{8}$$

where *ShareCount*(p) is the "share" number of a post p and where $\max(\cdot)$ function returns the max "share" number of a post p_i in the post set P.

Since the CRF model only accept features for terms (not for posts), we need to transform the post importance score to term importance score. The basic idea is that, if a term occurs often in more important posts, we can assume the term is important. Therefore, we calculate average term importance based on post importance as follows:

$$f_{TermImportance}(w) = \frac{\sum_{p_i \in P_w} PostImportance(p_i)}{|P_w|} \tag{9}$$

where *PostImportance*(p_i) can be replaced by one of the above four feature functions, P_w is a set of posts containing the term *w*, and $|P_w|$ is the number of posts in the set P_w. We further normalize $f_{TermImportance}(w)$ as follows:

$$f_{NormalizedTermImportance}(w) = \frac{f_{TermImportance}(w)}{\max_{w_j \in W} f_{TermImportance}(w_j)} \tag{10}$$

where $\boldsymbol{W}$ is the set of all terms. Finally, we divide the normalized term importance score into three ranges, high [1, 0.8), middle (0.2, 0.8], and low [0, 0.2].

3.4.3 Complex Task Name Composition

Based on the task-semantic tagging results of the CRF model, we can calculate the highest frequency task-semantic tagging terms, including e_{1V}, e_{1O}, e_2, and *t*, for each type of task-semantic tag E_{1V}, E_{1O}, E_2, and *T*, respectively. To compose a semantically suitable complex task name *c*, we use a rule-based algorithm that considers the frequency and POS of each task-semantic tagging term. Figure 4 shows the algorithm of complex task name composition (CTNC), which combines a task topic and a task event into a complex task name, where *Freq*(*e*) is the frequency of an event *e* and *POS*(*t*) is the POS tag of a topic *t*. We first compare the term frequency of a transitive event e_{1V} and an intransitive event e_2. If the frequency of e_2 is greater than e_{1V}, CTNC simply returns a complex task name composed of <t+e_2>, *e.g.*, "北京<*t*>旅遊<*e*2> (Beijing<*t*> travel<e_2>)". Otherwise, if the topic *t* is a common noun, CTNC returns a complex task name composed of <e_{1V}+t>, *e.g.*, "學習<e_{1V}>英語<*t*> (learn<e_{1V}> English<*t*>)". Otherwise, if the topic *t* is a proper noun, CTNC returns a complex task name composed of e_{1V}+t+e_{1O}, *e.g.*, "購買<e_{1V}>三星< *t* >手機<e_{1O}> (buy<e_{1V}> Samsung< *t* > cellphone<e_{1O}>)".

```
Algorithm: complex task name composition
Input: task-semantic tagging terms $e_{1V}$, $e_{1O}$, $e_2$, $t$
Output: A complex task name $c$
If $Freq(e_{1V}) < Freq(e_2)$ Then return $t + e_2$
Else If $POS(t)$ is "common noun" Then return $e_{1V} + t$
Else return $e_{1V} + t + e_{1O}$
```

Figure 4. The algorithm of complex task name composition

4. Experiments

4.1 Experimental Setup

4.1.1 Dataset

We use a month of query logs from the Sogou search engine as our dataset. The query logs contain 21,422,773 records with 3,163,170 distinct queries. We group these query records into search sessions according to user ID. Since a complex task may span a period of time, we used 24 hours as the time gap to segment search sessions, which resulted in 264,360 search sessions.

4.1.2 Data Labeling

In this work, we employed three annotators to label each query with a task-related entity and a latent complex task name. Since the query logs are diverse and often ambiguous, heuristically labeling the task-related entities and task names for each query may lead to inconsistent results. To identify reasonable and consistent training/testing data for evaluating our ECTM, a formal annotation method procedure should be provided. In the following, we describe the guidelines for annotators on how to label a task-related entity and a complex task name for each search query.

In general, a search query should give one entity that users focus on. Annotators thus only focus on queries containing exactly one entity and discard other queries.

To better interpret task-related entities and task names for queries, annotators are encouraged to exploit external resources, *e.g.*, clicked pages, search results for queries, or query context (*i.e.*, other queries in the same search session).

Since a complex task should be determined based on the whole search session, annotators should complete the labelling of all task-related entities in a search session before they begin to label task names.

From the 264,360 obtained sessions, we randomly labeled 5,142 sessions with task names and entities. For each task, we further examined the labeled results, and unified the similar task names annotated by different annotators. For instance, "北京旅遊 (travel to Beijing)" and "北京旅行 (trip to Beijing)" would be unified to "北京旅遊 (travel to Beijing)". Table 4 shows an example search session of our labeled results. In fact, it is not easy to find a search session containing a good complex search task search intent. Each query belonging to a complex search task was labeled with a task-related entity. After excluding the tasks containing less than three entities and some controversial tasks, we obtained only 523 complex tasks from 513 sessions. We found that, although there were many interleaving simple tasks in one session, few complex tasks occurred in the same session. In other words, we found users seldom deal with two complex tasks simultaneously within the same period of time. The statistics of the labeled results are shown in Table 5. On average, there are 5.68 (2972 / 523) entities per task in our labeled dataset.

Table 4. An example search session with annotated task-related entities for each query. These search sessions obviously searched for the complex search task "結婚準備 (prepare for wedding)".

Chinese		English Translation	
Query	**Task-related Entity**	**Query**	**Task-related Entity**
結婚選購首飾	首飾	Wedding jewelry purchase	Jewelry
結婚首飾	首飾	Wedding jewelry	Jewelry
結婚禮服	禮服	Order wedding dress	Dress
結婚戒指展示	戒指	Wedding ring gallery	Ring
黃金戒指	戒指	Gold ring	Ring
購買黃金戒指	戒指	Buy gold ring	Ring

Table 5. The statistics of our labeled results for complex tasks

Data Type	Total Count	Distinct Count
Complex Task	523	244
Query	3,741	1,715

Training data: We sampled 100 complex tasks as the training dataset, which contained 724 queries and 424 distinct entities. For the log-linear model (LLM) used in task-coherent query expansion, we identified pairwise training data according to our labeled data set. Each query pair indicates if two queries belong to the same complex task. Therefore, we could obtain 4950 (*i.e.*, the combination number C_2^{100}) query pairs to train LLM. For the CRF model used in task name identification, we manually labeled 30 microblog posts retrieved for each complex task; thus, there were 3000 microblog posts for training the CRF model.

Testing data: We used the remaining 423 complex tasks as the testing dataset, which contained 3017 queries and 1426 distinct entities. For each testing complex task, we randomly selected a query from the testing task as the input query.

4.1.3 Task Name Quality Labeling

Since the automatically identified task name may be semantically the same as our annotated task name but represented in different lexicons, we cannot simply judge each identified task name by an automatic keyword matching approach. To overcome the above problem, we employed three judges to give scores independently for identified task names of all compared methods (the details of the compared methods will be described in Section 4.1.5). To ensure the fairness for all compared methods, we designed a labeling procedure.

1. Merge all identified task names from different methods into a large task name list.
2. Remove the duplicated task names in the task name list.
3. Shuffle the task names in order to hide the information of the original rank of different methods.

In order to give a relevance score, the judges could look at our pre-labeled task names and survey the information of the complex tasks. The score for each task name should be 0, 1, or 2. We define the criterion of giving a relevance score as follows.

Bad (score of 0): A bad complex task name is irrelevant to the complex task or is semantically unsuitable.

Fair (score of 1): A fair task name is semantically suitable but the judges cannot determine whether the task name can represent the complex task.

Good (score of 2): A good task name is semantically suitable and semantically the same as the pre-labeled task name.

Since there were three judges (thus, we had three relevance scores for each task), we needed to decide the final relevance scores for evaluating performance of the compared methods. We used the majority decision to decide the final relevance score for each task. For instance, when a name was labeled with relevance scores 1, 0, and 0, the final relevance score would be 0. If a task name was labeled with three distinct relevance scores 0, 1, and 2, the final relevance score would be 1. We only considered a task name with a relevance score of 2 as a correct task name.

4.1.4 Evaluation Metrics

Inclusion rate: The inclusion rate is to evaluate the fraction of the top *n* identified complex task names that include at least one correct complex task name. The equation of inclusion rate is given as follows:

$$InclusionRate = \frac{|inclusion(T)|}{|T|} \tag{11}$$

where $|T|$ is the number of testing tasks (*i.e.*, 423) and where $|inclusion(T)|$ is the number of testing tasks that can find at least one correct task name in top *n* identified complex task names.

Mean reciprocal rank (MRR): Since we only need one correct task name for each testing data, we use MRR, which only considers the rank of the first returned correct task name for each testing data task. To calculate the MRR, the equation is as follows:

$$MRR = \frac{1}{|T|}\sum_{i=1}^{|T|}\frac{1}{rank(i)} \tag{12}$$

where $|T|$ is the number of testing tasks and $rank(i)$ is the rank of the first identified correct task name of the i_{th} testing task.

Normalized discounted cumulative gain (NDCG): To realize the overall quality of the top *n* identified task name, we also use *NDCG* to evaluate the performance for different methods. According to scores of identified task name, we first have to calculate the *DCG* as follows:

$$DCG = score_1 + \sum_{i=2}^{n}\frac{score_i}{\log_2(i)} \tag{13}$$

where *n* is the number of identified task names and $score_i \in \{0,1,2\}$ is the relevance score of the top-i task name. In order to normalize the *DCG* value from 0 to 1, the DCG divided by *IDCG*, called *NDCG*, is defined as follows:

$$NDCG = \frac{DCG_n}{IDCG_n} \tag{14}$$

where *IDCG* can be calculated the same as *DCG* with an ideal rank, which was ranked by labeled scores.

4.1.5 Task Name Identification Method Comparison

LRM_SERP (linear regression model with search engine result snippet): This method was proposed by Zeng *et al.* (2004) to identify salient phrases from search-result snippets. Salient phrases are identified using several regression models with five proposed features, including TFIDF, phrase length, intra-cluster similarity, cluster entropy, and phrase independence. We used the linear regression model (LRM) with the five features proposed in their work to extract salient phrases as task names from search-result snippets. Since using only the testing input query to collect search-result snippets may not be fair for identifying complex task names, we used our produced pseudo queries with task-related information model to collect search-result snippets from a web search engine (see Section 3.3). The weight for each feature was set as in Zeng *et al.*'s work.

LRM_MB (linear regression model with microblog): We also tried to adopt a linear regression model with microblog posts in order to compare the resources of SERP and microblog based on the five features proposed in Zeng *et al.*

LRM_MB+ (linear regression model with microblog plus): This was used to compare the performance of the linear regression model (LRM) and with our proposed microblog features with quantified values in LRM_MB.

ECTM (entity-driven complex task model): This is our proposed entity-driven complex task model, which utilizes microblog as extending data for a task-coherent query set. We used all features proposed in this work for training a CRF model to identify the name of complex task. The only difference between LRM_MB+ and our method is that the former first extracts bigrams and trigrams from posts as candidate phrases before using LRM to determine their correctness, and our method uses CRF to directly label each term in the posts with a task-semantic tag (*i.e.*, topic or event).

4.1.6 Parameter Selection

For each testing task, we need to determine the number of the top n selected task-related entities to produce pseudo queries for retrieving microblog posts. Since we use all subsets containing two or three entities of the top n entity set to produce pseudo queries, the entity number is critical to the number of produced pseudo queries (*i.e.*, the total number of pseudo queries is C_2^n plus C_3^n, where C_k^n is the k-combination of the entity set containing n entities), also making it critical to computational time. To realize how many entities should be selected for each testing task to achieve the best performance of identifying a complex task name, we randomly selected 15 complex tasks from the testing dataset to make the preliminary experiment. For each pseudo query, we retrieved the top 20 microblog posts and search-result snippets via a microblog search engine and a web search engine, respectively. We calculated the average top 3 inclusion rate for each testing task of LRM_SERP and LRM_MB. Figure 5 shows the different number of entities for producing pseudo queries along with the average top 3 inclusion rate. We can see both LRM_SERP and LRM_MB achieved better inclusion rates when using five entities to compose 20 (*i.e.*, $C_2^5 + C_3^5$) pseudo queries. Generally, a small number of entities achieved a worse inclusion rate since the retrieved microblog posts or search-result snippets for a complex task were insufficient. When the number of selected entities was greater than five, the number of unrelated entities also increased, resulting in a worse inclusion rate. To understand the correct number of microblog posts that should be used in our training data, we also conducted a preliminary experiment. Figure 6 shows that, when we used 30 posts, our ECTM could achieve the best precision. Therefore, in the following experiments, we used the top five entities for all methods when identifying pseudo queries.

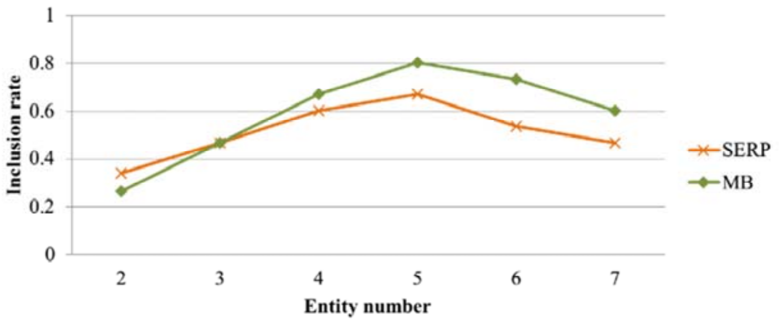

Figure 5. The top 3 task name inclusion rate of different number of entities for producing pseudo queries

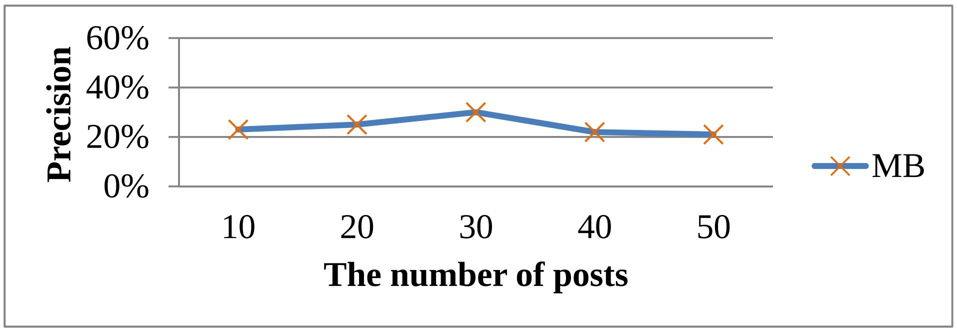

Figure 6. The precision for different number of microblog posts (MB) in complex task name identification.

4.2 Results of Task Name Identification

Table 6 shows the overall results of task name identification using different methods. Generally, the four methods achieved adequate top 5 inclusion rate (0.68, 0.72, 0.79, and 0.83 for LRM_SERP, LRM_MB, LRM_MB+, and ECTM, respectively). We found LRM_MB using a microblog is better than LRM_SERP using an SERP in identifying the complex task name. According to our analysis, microblog posts contain more task names, even in tail posts. On the contrary, only a few top-ranked search result snippets contain complex task names. The reason is that microblog posts are identified by users; thus, they have more likelihood to talk about a real-life complex task. Therefore, LRM_MB can achieve better performance than LRM_SERP. We also compared the performance between using and not using our proposed features in a microblog post (*i.e.*, LRM_MB and LRM_MB+). Using microblog features can slightly improve the overall performance since important posts usually mention a complex task. Our proposed ECTM using CRF to automatically label task-semantic tags for each term can improve the performance significantly, and it achieved Top-1 MRR of 0.57.

To realize whether the identified complex task names are semantically suitable, some example of top-1 identified task names are shown in Table 7. For the query “北京機票 (Beijing flight ticket),” only our ECTM identified correct task name “北京旅遊 (Beijing travel)”. LRM_SERP identified incorrect task name “旅行社旅行 (travel agency travel),” which is not semantically suitable. The reason is that search-result snippets sometimes are not represented as complete natural language sentences. Therefore, when we try to extract a bigram or trigram, there are some combinations that are not semantically suitable. LRM_MB and LRM_MB+ identified incorrect task names “訂購自由行 (reserve independent travel)” and “國外旅遊網訂 (online ordering to travel in foreign countries),” which are semantically suitable but not very related with Beijing travel. The reason is that the above two methods do not consider the task topic when identifying complex task names. According to our observation, the task topic of a complex task usually occurs in almost every search query. For the testing task query “深圳會計待遇 (Shenzhen accounting salary),” only LRM_MB+ identified the correct task name “申請會計工作 (apply for accounting work),” and our ECTM identified an incorrect task name “申請深圳工作 (apply for Shenzhen work)”. According to our analysis, ECTM identified an incorrect task topic “深圳 (Shenzhen),” which is a location name in China and occurs many times in a task-coherent query set (see Section 3.2). Nevertheless, the correct task topic should be “會計 (accounting)”, which specifies the career searched by the user. As a result, we find our ECTM may identify incorrect task names when the task topic is not the most frequent term in the task-coherent query set.

Table 6. The overall complex task identification performance comparison of four methods

Method	**MRR**			**NDCG**			**IR**		
	Top 1	**Top 3**	**Top 5**	**Top 1**	**Top 3**	**Top 5**	**Top 1**	**Top 3**	**Top 5**
LRM_SERP	0.37	0.41	0.47	0.22	0.35	0.31	0.37	0.45	0.68
LRM_MB	0.42	0.52	0.54	0.27	0.46	0.42	0.42	0.65	0.72
LRM_MB+	0.48	0.54	0.59	0.39	0.54	0.53	0.48	0.68	0.79
ECTM	**0.57**	**0.63**	**0.66**	**0.45**	**0.61**	**0.58**	**0.57**	**0.71**	**0.83**

Table 7. The example of top-1 complex task names generated by four compared methods, where * sign indicates the generated complex task name is incorrect

Testing task query	**LRM_SERP**	**LRM_MB**	**LRM_MB+**	**ECTM**
北京機票 (Beijing flight ticket)	*旅行社旅行 (Travel agency travel)	*訂購自由行 (Reserve independent travel)	*國外旅遊網訂 (Online ordering to travel in foreign countries	**北京旅遊 (Beijing travel)**
手機最新報價 (This newest prices of cellphones)	*中華門號續約 (Renewal cellphone number of Chunghwa)	*組裝電腦 (DIY computer)	*購買配件 (Buy accessories)	**購買手機 (Buy cellphones)**
深圳會計待遇 (Shenzhen accounting salary)	*考到深圳車牌 (Get Shenzhen license plated)	*深圳會計兼職 (Shenzhen part-time job of account)	**申請會計工作 (Apply for Shenzhen work)**	*申請深圳工作 (Apply for accounting work)

5. Discussion

For our proposed ECTM, we discuss two issues of producing pseudo queries with a candidate topic and the limitations of task name composition in the following.

(1) Producing pseudo queries with a candidate topic: In general, pseudo queries containing a candidate topic can achieve better precision when retrieving microblog posts. If the pseudo queries do not contain a topic, it is hard to find the correct topic for the task name. For example, two tasks "北京旅遊 (travel to Beijing)" and "青島旅遊 (travel to Tsingtao)" may have many of the same task-related entities, such as "機票 (flight tickets)," "飯店 (hotel)," and "地圖 (maps)". Therefore, all of the microblog posts containing these entities may be retrieved and result in our ECTM identify an incorrect task name. Nevertheless, when the identified candidate topic is incorrect, our task name identification model usually is unable to identify a correct task name. How to improve the precision of identifying a task topic is still an important issue.

(2) The limitations of task name composition: Our proposed ECTM can identify task names composed of correct task-semantic tags effectively. Nevertheless, some identified task names that contain an event object may have semantic flaws. For instance, we identified a complex task name "治療北京打鼾 (treat Beijing snore)," which is composed of an event "治療 (treat)," a topic "北京 (Beijing)," and an event object "打鼾 (snore)," based on our definition of task name composition strategy, which considers only POS patterns. Actually, the more semantically suitable task name is "北京治療打鼾 (Beijing treat snore)" that is composed of a topic "北京 (Beijing)," an event "治療 (treat)," and an event object "打鼾 (snore)". In this work, we still cannot provide a perfect solution for composing task names.

6. Conclusion and Future Work

In this work, we proposed an entity-driven complex task model (ECTM), which addressed the problem of improving the user experience when searching for a complex task. We exploited various web resources, including query logs, microblogs, and search-result snippets, to enhance the performance of our ECTM. To identify a human-interpretable complex task name from short-content queries, we utilized microblog posts and investigated several useful features to train the CRF model to automatically identify complex task names. Experimental results show that ECTM efficiently identifies complex task names with various task-related entities. Nevertheless, there are still some problems that need to be solved. In the future, we will try to investigate other useful features to improve the task name identification when dealing with real-life complex task queries.

References

Agichtein, E., White, R. W., Dumais, S. T., & Bennett, P. N. (2012). Search, Interrupted: Understanding and Predicting Search Task Continuation. In *Proc. of SIGIR 2012*.

Beeferman, D. & Berger, A. (2000). Agglomerative Clustering of a Search Engine Query log. In *Proc. of KDD '00*, 407-416.

Boldi, P., Bonchi, F., Castillo, C., Donato, D., Gionis, A., & Vigna, S. (2008). The Query-Flow Graph: Model and Applications. In *Proc. of CIKM 2008.*

Cui, J., Liu, H., Yan, J., Ji L., Jin R., He, J., Gu, Y., Chen, Z., & Du, X. (2011). Multi-view Random Walk Framework for Search Task Discovery from Click-through Log. In *Proc. of CIKM 2011*.

Feild, H. & Allan, J. (2013). Task-Aware Query Recommendation. In *Proc. of SIGIR 2013*.

Guo, Q. & Agichtein, E. (2010). Ready to Buy or Just Browsing? Detecting Web Searcher Goals from Interaction Data. In *Proc. of SIGIR 2010*.

Ji, M., Yan, J., Gu, S., Han, J., He, X., Zhang, W. V., & Chen, Z. (2011). Learning Search Tasks in Queries and Web Pages via Graph Regularization. In *Proc. of SIGIR 2011*.

Jones, R., & Klinkner, K. (2008). Beyond the Session Timeout: Automatic Hierarchical Segmentation of Search Topics in Query Logs. In *Proc. of CIKM '08*, 699-708.

Kotov, A., Bennett, P. N., White, R. W., Dumais, S. T., & Teevan, J. (2011). Modeling and Analysis of Cross-Session Search Tasks. In *Proc. of SIGIR 2011*.

Lafferty, J., Mccallum, A., & Pereira, F. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proc. of ICML*, 282-289.

Liao, Z., Song, Y., He, L.-W., & Huang, Y. (2012). Evaluating the Effectiveness of Search Task Trails. In *Proc. of WWW 2012*.

Lin, T., Pantel, P., Gamon, M., Kannan, A., & Fuxman, A. (2012). Active Objects: Actions for Entity-Centric Search. In *Proc. of WWW 2012*.

Liu, J & Belkin, N. J. (2010). Personalizing Information Retrieval for Multi-Session Tasks: The Roles of Task Stage and Task Type. In *Proc. of SIGIR 2010*.

Lucchese, C., Orlando, S., Perego, R., Silvestri, F., & Tolomei, G. (2011). Identifying Task-based Sessions in Search Engine Query Logs. In *Proc. of WSDM 2011*.

MacKay, B. & Watters, C. (2008). Exploring Multi-Session Web Tasks. In *Proc. of CHI '08*, 1187-1196.

Mihalkova, L. & Mooney, R. (2009). Learning to Disambiguate Search Queries form Short Sessions. In *Proc. of ECML '09*, 111-127.

Raman, K., Bennett, P. N., & Collins-Thompson, K. (2013). Toward Whole-Session Relevance: Exploring Intrinsic Diversity in Web Search. In *Proc. of SIGIR 2013*.

Tan, B., Shen, X., & Zhai, C. (2006). Mining Long-Term Search History to Improve Search Accuracy. In *Proc. of KDD '06*, 718-723.

Wang, T.-X., & Lu, W.-S. (2011). Identifying Popular Search Goals behind Search Queries to Improve Web Search Ranking. In *Proc. of AIRS 2011*.

Wen, J.-R., Nie, J.-Y., & Zhang, H.-J. (2001). Clustering User Queries of Search Engine. In *Proc. of WWW 2001*.

White, R. W., Chu, W., Hassan, A., He, X., Song, Y., & Wang, H. (2013). Enhancing Personalized Search by Mining and Modeling Task Behavior. In *Proc. of WWW 2013*.

Yamamoto, T., Sakai, T., Iwata, M., Yu, C., Wen, J.-R., & Tanaka, K. (2012). The Wisdom of Advertisers: Mining Subgoals via Query Clustering. In *Proc. of CIKM 2012*.

Zeng, H.-J., He, Q.-C., Chen, Z., Ma, W.-Y., & Ma, J. (2004). Learning to Cluster Web Search Results. In *Proc. of SIGIR 2004*.