2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Mining User Tasks from Print Logs

Xin Li[1], Lei Zhang[1], Ping Luo[2], Enhong Chen[1], Guandong Xu[3], Yu Zong[4], Chu Guan[1]
[1] *University of Science and Technology of China,* [2] *HP Labs China,*
[3] *University of Technology, Sydney,* [4] *West Anhui University*
[1]{*leexin, stone, cheneh, guanchu*}*@ustc.edu.cn,* [2]*ping.luo@hp.com,*
[3]*guandong.xu@uts.edu.au,* [4]*nick.zongy@gmail.com*

***Abstract*—With lots of applications emerging in World Wide Web, many interaction data from users are collected and exploited to discover user behavior or interest patterns. In this paper, we attempt to exploit a new interaction data, namely *print logs*, where each record is printing URLs selected by a user using a popular web printing tool. Users usually print web contents based on an intention (subtask or task). Apparently, mining common print tasks from print logs is able to capture users' intentions, which undoubtedly benefits many web applications, such as task oriented recommendation and behavior targeting. However, it is not an easy job to perform this due to the difficulty of URL topic representation and task formulation. To this end, we propose a general framework, named UPT (Users Print Tasks mining framework), for mining print tasks from print logs. Specifically, we attempt to leverage delicious (a social book marking web service) as an external thesaurus to expand the expression of each URL by selecting tags associated with the domain of each URL. Then, we construct a tag co-occurrence graph where similar tags can be clustered as *subtasks*. If we view each subtask as an item, then the print log is transformed to a transaction database, on which an efficient pattern mining algorithm is proposed to induce tasks. Finally, we evaluate the effectiveness of the proposed framework through experiments on a real print log.**

***Keywords*-Print Logs; Print Tasks; the Wisdom of Crowds; Clustering; Frequent Pattern Mining**

I. Introduction

World Wide Web is becoming a primary information source in our daily life. When users browse the Internet, it is very convenient for them to print out web pages they like. Thus, some IT companies provide tools for web page printing. For example, HP provides a semi-automatic tool named Smart Print[1], where users can manually select the informative areas for printing in an interactive interface. When using this tool, a sequence of URLs selected from the user are recorded with the user's permission under a certain agreement. Such users' records form a so-called *print log* data (See examples depicted in Table I). Users usually print contents initiated by an intention (subtask or task). For example, if a user wants to travel (task) to 'Beijing', she may print some information about flight booking, car rental and hotel reservation (subtasks). In one word, the print log is a valuable data and can also be used for capturing the user intention.

Table I
User Print URL Sequences

User	URL sequence
$user_1$	https://www.auto-europe.co.uk/car.cfm
	http://www.easyjet.com/it/voli-economici/Lamezia/Milano-Malpensa
$user_2$	http://www.roselladb.com/healthcare-fraud-detection.htm
	http://www.statsoft.com/solutions/medicare-fraud-detection/
.....	.....

Thus, mining common print tasks that are shared by many users from print logs is a very interesting task and has a broad application potential. We present two of the applications below. One immediate application is to facilitate user's tasks by recommendations. When a user prints out a ticket booking web page, signaling a travel task, thus we can give the user some suggestions, such as destination weather report and attractions. Such a task-oriented recommendation can greatly improve user's experience.

Additionally, print tasks can also help online advertising. Since lots of print tasks, such as travel planning or conference organization, contain commercial interests, they may help advertisers to identify potential customers and provide more relevant ads to such users. This behavioral targeting service is very helpful for all businesses targeting different types of customers.

However, mining user print tasks from print logs is not a trivial work. There are two primary challenges. The first challenge is the URL representation. Since URLs different people printed vary even when they visit the same web site, the data sparsity becomes conspicuous and get aggravated when running a conventional pattern mining algorithm. Besides, the URL is short and some of the words(e.g. "www","com") are meaningless to us. It is eager to interpret the URL for user's intention and semantically acquired for us. The second challenge is the formulation of print tasks and the design of effective algorithm for mining tasks. As the task naturally have the multi-level structure(See Figure 1) in real life, there is no existing algorithm for this task. In stead, a framework is needed to describe the hierarchy clearly.

To solve the above challenges, in this paper we propose a general framework (called Users Print Tasks mining framework) for mining print tasks from print log. Specifically, we propose a muti-level framework, i.e. task-subtask-tag hierarchy (shown in Figure 1), which can model users' tasks

[1]An extension of Web browser, the software can be downloaded from http:www.hp.com/go/smartprint

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

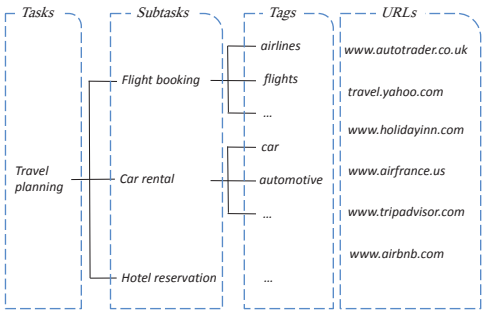

Figure 1. Task-Subtask-Tag Hierarchy (multi-level structure).

effectively. Furthermore, we enrich the semantic expression of each URL by utilizing the alternative or relevant tags associated with the domain of each URL. Then, the tags are clustered based on tag-occurrence graph to form subtasks, where tags in each subtask are semantically relevant or alternative to each other. In this manner, the print log is then transformed into a transaction database, in which each subtask is viewed as an item. Note that the task is a set of subtasks,thus mining print tasks becomes a pattern mining problem and we propose an effective algorithm for mining user task patterns. Finally, the experiments on a real print log dataset show the effectiveness of the proposed framework. In summary, we make following contributions in this paper.

- We address the user task mining through a new user interaction dataset, namely print logs and propose a framework called UPT (Users Print Tasks mining framework) based on the proposed user intent hierarchy of "Tags-Subtasks-Tasks" to mine user task patterns.
- We extend the semantics of each URL by utilizing a third-party website *delicious.com* as external thesaurus, which is proven to be practical. This work has a very good reference value for short text extension.
- We evaluate our approach on a large dataset of over 16,000 users and the experiments clearly demonstrate that the tasks and subtasks mined by our method are meaningful.

The rest of the paper is organized as follows. We describe the related work in Section IV and show the mining framework in Section II. We report the experimental study in Section III and finally conclude the paper in Section V.

II. The Proposed UPT Mining Framework

A. The Framework Overview

As shown in Figure 2, the framework can be divided into three steps.

Step 1. We use *delicious.com* to fetch tags for each URL user printed. Once a URL is given, we extract the domain and input it into *delicious.com api*[2] and get the returned top-10 popular tags[3] to express each URL. Those tags do reflect the URL's topical information.

Step 2. Given the data representation set $\mathcal{R}$, in which each URL has its representative tags, we then build up a

[2]Delicious.com api: https://delicious.com/developers

[3]Due to the api limitation, only top 10 tags are returned from the website.

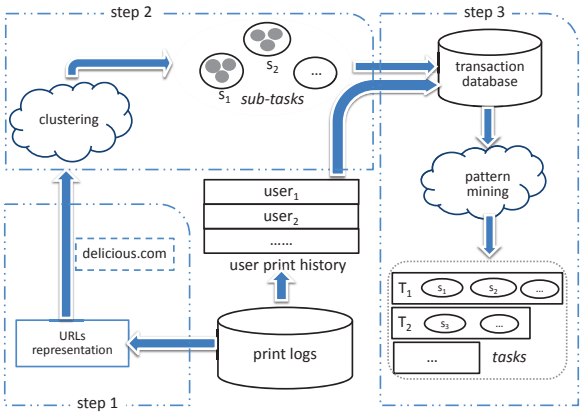

Figure 2. The Mining Framework.

tag co-occurrence graph $\mathcal{G}$. The vertex is the tag collection and the edges count for the number of co-occurrence of two tags. Based on this graph, similar tags can be clustered as *subtasks*.

Step 3. We manually mark each mined subtask or use the most frequent one in a subtask to represent each subtask. Then the print logs can be transformed to a transaction database if each subtask is viewed as an item. Based on the database, we combine two efficient algorithms MAFIA [3] and DOFIA [6] to mine *dominant* and *frequent* patterns by devising a hybrid algorithm named DOMAFIA for more diverse tasks.

In the following subsections, we will detail each step of the proposed framework.

B. The URL Representation

There are several ways to represent URL. First of all, web page content can be crawled according to the URL and we can extract the terms in pages as representations. However we do not choose this method due to two reasons. First, as time goes by, some URLs may be unavailable due to some reasons and we can not get the content, let alone extracting keywords. Second, it's not wise to extract all keywords from the whole content because most of web pages contain lots of ads and other information irrelevant to user's intention, thus leading to noisy concepts. Moreover, it's very time-consuming.

In contrast to keyword extraction, adopting the third party thesaurus to expand semantics of each URL has been well recognized as an alternative way. There are a variety of approaches to fulfill this, e.g., some use ODP [4], one kind of taxonomy concepts to represent a certain URL. However, in this paper we leverage *delicious.com api*, one kind of folksonomy, to fetch tags for each URL. The reasons are as follows. First, delicious.com has a broader coverage than ODP due to the mass contributions. We count the coverage rate of the both methods in our dataset and find that the delicious.com tags cover more URLs at 61.44% while that by ODP is only 46.82%. Second, ODP uses a hierarchy ontology scheme for organizing website lists which is constructed and maintained by a community of volunteer

[4]Open Directory Project: http://www.dmoz.org/

Table II

(a) Domain and Tags

Domain	Tags
auto-europe.co.uk	car, travel, car-rental, rental, car-hire, holiday, cheap, carhire, hire
easyjet.com	travel, flights, europe, lowcost, airlines, cheap, tickets, transport
travelodge.co.uk	hotel, uk, accommodation, london, travelodge, übernachtung
travelrepublic.co.uk	travel, holidays, flights, hotels, Cheap_holidays, cheap, thomascook

(b) Example of Subtasks and Tags

Subtasks	Tags
flight booking	airfare, airlines, flights
hotel reservation	hotel, reservations
car rental	auto, automotive, car

editors. Nevertheless, the returned tags from *delicious.com* were annotated by the folks users reflecting "the wisdom of crowds", which would have broader and richer semantics. Third, ODP looks much like a public library catalog for websites which has a fixed hierarchy structure and a few categories while *delicious.com* is more flexible and adaptable to unknown categories at appropriate level of granularity.

Due to the above reasons, we choose delicious tags as representations and process the raw data by wiping out the URLs without returned tags. Table II(a) shows the tag representations of each domain given in Table I. After that, we got 14,465 independent URL domains that receive returned tags from *delicious.com* with each domain possessing 8.93 tags in average. Here we use domains rather than simple URLs due to the data sparsity problem.

C. Mining Subtasks

After deriving tag representations from *delicious.com* for each URL, we mine subtasks because the tag representation for a domain may cover more than one topic. A subtask is a subset of tags and it has two characteristics. First, each tag in a subtask is similar and alternative to each other. Second, different subtasks corresponds to different steps in a task. Then a graph-based clustering approach [14] is formed here to mine subtasks.

1) Tag Co-occurrence Graph: Heuristically, when we put an URL to *delicious.com* and receive a collection of tags. Some of tags are related to each other to a certain extent such as synonymy and containment relationship. For example, in Table II(a), we can observe the correlation of tags in domain "auto-europe.co.uk".

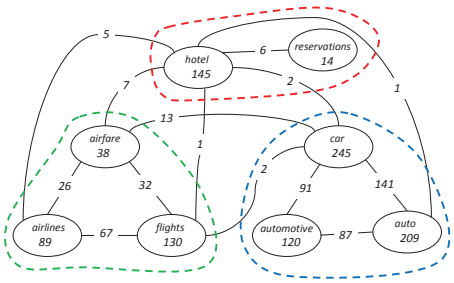

Figure 3. Example of Co-Occurrence Graph

For all domains, we can build up a tag co-occurrence graph $\mathcal{G}$, where each vertex t stands for a tag and an edge (t_1, t_2) appears if they co-occur in a domain. Two tags t_1 and t_2 are said to **co-occur** if the corresponding tag collection to a domain contains both of them. Hence, we denote the co-occurrence frequency by $c(t_1, t_2)$. For each vertex in $\mathcal{G}$, we count the tags frequency of t, denoted by $freq(t)$.

Figure 3 shows an example of the tags co-occurrence graph in our real dataset. There are eight tags on the graph belonging to three different subtasks, which are listed in Table II(b). We manually label the subtasks or use the most frequent one in a subtask to represent the subtask. Intuitively, we get two observations from Figure 3. (1) Tags in the same subtask often co-occur more frequently than the two in different subtasks. For example, the co-occurrence of tags in subtask *car rental* is much larger than the co-occurrence of tags from different subtasks. (2) Using only co-occurrence has some limitations thus leading to some exceptions. For instance, in subtask *hotel reservation*, $c(hotel, reservations)$ is only 6, while $c(hotel, airfare)$ is more than that. According to rule 1, *hotel* and *airfare* belong to the same subtask, which is not proper. In order to obtain the subtasks more accurately, we should extract more features on graph $\mathcal{G}$ to cluster tags.

2) Clustering: In addition to co-occurrence frequency, we extract another feature to measure distance between two tags thereby doing clustering. The feature is inspired by information entropy. Heuristically, since tags in a subtask are alternative to each other, they may have similar neighborhood on graph $\mathcal{G}$.

Neighborhood of a Tag Let T be the collection of all tags, i.e., T = $\{ t_1, t_2, ..., t_n \}$. The neighborhood of a tag t_i in the graph $\mathcal{G}$ is a n-dimensional vector $\vec{v_i}$ = ($v_i[1]$,$v_i[2]$,...,$v_i[n]$),where

$$v_i[j] = \begin{cases} \frac{c(t_i,t_j)}{\sum_{k \neq i} c(t_i,t_k)} & i \neq j \\ 0 & i = j \end{cases} \quad (1)$$

Certainly, we have $\sum_{j=1}^{n} v_i[j]$ = 1. And the neighborhood vector $\vec{v_i}$ denote the distribution of neighborhood on graph $\mathcal{G}$. We use a modification of the Jensen-Shannon divergence [1] to measure the distance between two tags, which is a symmetric and smoothed version of Kullback-Leibler divergence.

Jensen-Shannon Divergence Given two probability distribution P and Q, the Jensen-Shannon divergence is

$$JSD(P \parallel Q) = \frac{1}{2} D(P \parallel M) + \frac{1}{2} D(Q \parallel M) \quad (2)$$

where $M = \frac{1}{2}(P+Q)$, the Kullback-Leibler divergence $D(P \parallel Q) = \sum_x P(x) log \frac{P(x)}{Q(x)}$.

Based on Jensen-Shannon divergence, we adopt the following measurement distance proposed in [21] since that one of the salient feature of Jensen-Shannon divergence is that it avoids the situation Q(x)=0 in Kullback-Leibler divergence *D*. Of course, the Jensen-Shannon divergence is non-negative. Specifically,

$$dist(t_i,t_j)=\frac{1}{2}(v_i[j]log\frac{2v_i[j]}{v_i[j]+v_j[i]}+v_j[i]log\frac{2v_j[i]}{v_i[j]+v_j[i]}) +\frac{1}{2}\sum_{k\neq i,j}(v_i[k]log\frac{2v_i[k]}{v_i[k]+v_j[k]}+v_j[k]log\frac{2v_j[k]}{v_i[k]+v_j[k]}) \quad (3)$$

The equation can be divided into two parts. The first part calculates the distance between t_i and t_j to each other, while the second part calculates the distance of each neighborhood of t_i and t_j to them. Now we get the feature *distance* (Equation 3) and next we will see how it works.

Previously, we point out the drawback of co-occurrence frequency in observation (2) when introducing the definition of co-occurrence, which can be eliminated if the *distance* is considered. When setting a distance threshold 0.35, we melt those edges whose value is larger than the threshold. By Equation 3, $dist(airfare, hotel)$=0.398 and $dist(hotel, reservations)$=0.342 so we cut the edge between vertexes *airfare* and *hotel*.

Based on this graph, thus we conduct an effective method for clustering tags. Of course, more complex models or algorithms are alternative to our approach and can be embedded into the framework, which is not our focus. To be specific, for graph $\mathcal{G}^*$, where we remove the unimportant edges whose distance is lower than the user specific threshold η, several connected components (clusters) by using *Union Find Sets* [2] is obtained finally. According to our experimental validation, the algorithm shows better result when parameter η is set to 0.35.

D. Mining Tasks

Task is essentially different from subtask. Mining subtasks is based on tags by expanding each URL with external thesaurus thus each mined subtask (cluster) represents a unique semantics. However, print log is "the wisdom of crowd", which implies user tasks. As mentioned before, a task consists of several subtasks but the method mining subtasks can not be applied to mine tasks. There are two critical reasons. First, tags in subtasks are alternative to each other while subtasks in a task are not similar, even vary enormously. *subtasks* and *tasks* are two different concepts, which can not be obscured. Second, the co-occurrence graph is based on the hypothesis that the less *distance* value the edge has the closer two vertexes will be. Since the first reason blocks us to build the graph, we cannot do further clustering to obtain tasks.

Consequently, we formulate the problem of mining tasks from print logs to a pattern mining problem. Generally speaking, MAFIA [3] or DOFIA [6] are often taken to mine maximal frequent patterns or dominant qualified patterns. MAFIA can reduce redundant patterns by selecting long patterns, however patterns mined overlap a lot. DOFIA derive patterns who occupied a lot in its supporting transactions while it have the problem of less diversity. In our work, mining task not only need patterns (Tasks) to be shared by more users, but also make demands on diversity. Thus we propose a hybrid algorithm named DOMAFIA for more diverse tasks by combining advantages from two efficient algorithms MAFIA and DOFIA.

We first transform the user printing URL transaction database to tag transaction database by using *delicious.com* returned tags as representations for each URL. Then tags in each transaction is mapping to a subtask label to form a new subtask transaction database. After wiping out the duplicated subtask labels of each transaction, we gain the clean subtask transaction database, which is fit for mining tasks.

1) Method DOMAFIA: DOMAFIA is short for **Do**minant and **Ma**ximal **F**requent **I**temset **A**lgorithm, which is a hybrid algorithm by merging two metrics support and occupancy. Support is defined as a conventional constraint for pruning in association rule mining, while occupancy is defined to measure how a pattern occupied in its supporting transactions. The definition of support and frequent patterns are basic concepts for pattern mining.

Table III
TRANSACTION DATABASE EXAMPLE

Trans. No.	Items
t_1	a b d
t_2	a b c e
t_3	b c
t_4	a b d e
t_5	a b e

Definition: (Support) The support of X is defined as $\sigma(X)=|\mathcal{T}_X|/|\mathcal{T}|$, where $\mathcal{T}_X$ is the transactions that contain itemset X and $\mathcal{T}$ is the whole transactions. Given a user specific threshold α ($0<\alpha\leq 1$), X is said to be frequent if $\sigma(X)\geq\alpha$.

Let us take pattern $\{a,b\}$ in Table III as example. Its supporting transactions are t_1, t_2, t_4 and t_5 thus $support(\{a,b\})=4/5=0.8$. We call an itemset the maximal frequent pattern if none of its immediate supersets is frequent. Hence, in order to get long patterns, we mine maximal frequent subtasks. In the above case, pattern $\{a,b,e\}$ is maximal frequent patterns when setting $\alpha=0.6$.

In [6], a property *occupancy* is proposed to evaluate the portion an itemset occupies in the transactions it appears in, which is shown below.

Definition: (Occupancy) The occupancy of X is defined as $\phi(X)=average(|X|/|t|:t\in\mathcal{T}_X\})$, where function average() derives the average value of all items in the set.

Harmonic occupancy and *Arithmetic occupancy* are defined respectively according to the different average function employed. In our paper, we use the arithmetic occupancy while same techniques can also be employed if considering harmonic occupancy. Also see $\{a,b,e\}$ in Table III, we can easily calculate $\phi(\{a,b,e\})=(3/3+3/4+3/4)/3=0.83$. We say an itemset X a dominant pattern if its occupancy $\phi(X)$ is no less than a user specific threshold β. Thus if X meets both the requirements for frequent and dominant, we say the itemset X a dominant and frequent pattern.

Table IV
EXAMPLES OF SUBTASKS

Subtask Label	Tags
book hotel	accommodation, accomodation, booking, hotel, hotels
cheap airline	airtravel, airline, airlines, flights, fly, low cost, tickets, voyage
travel auto	travel, vacation, autoverhuur, travel auto, rentalcar
buy something	shop, shopping, store
rent house	apartment, apartments, rent, rental, rentals
home developer	developers, homebuying, houses, property, real estate, realtors, realty
sports game	athletics, basketball, football, nfl, scores, soccer, sport, sports
baseball	baseball, mlb(Major League Baseball)
news	daily, media, news
game machine	games, ign, nintendo, playstation, ps2, ps3, wii, xbox, xbox360
sports tv program	deporte, deportes, espn, futbol
discount buy	aldi, angebote, discounter, kaufen, purchase, schickeshops, supermarkt
buy books	ebook, ebooks, books, ankauf, bucher, gebraucht, verkaufen
thesaurus	archive, dictionary, encyclopedia, libraries, thesaurus, wikipedia
job	career opportunities, cv, headhunter, jobsites, knoxville area, resume
computer	tweak, tweaks, vista, win7, windows7

Table V
EXAMPLES OF TASKS

No.	Tasks	Subtasks
1	travel planning	book hotel, airline, travel auto, buy something
2	real estate	rent house, real estate or home developers
3	sport	sports game, baseball, news, game, tv program
4	shopping	discount buy, buy books, thesaurus, buy something

Algorithm 1: DOMAFIA

input : the subtask transaction data $\mathcal{T}^{trans}$, , subtask lexicographic ordering $\mathcal{T}$, minimal support α , minimal occupancy β and quanlity balance parameter λ
output: task patterns $\mathcal{P}$

$Root.head \leftarrow empty$;
$Root.tail \leftarrow \mathcal{T}$;
$DFS(Root)$;
$DFS(Node)$ **begin**
  for $subtask \in Node.tail$ **do**
    $t_{tmp} \leftarrow Node.head \cup subtask$;
    if $t_{tmp}.support > \alpha$ **then**
      $Node_{tmp}.head \leftarrow t_{tmp}$;
      remove subtask from$Node_{tmp}.tail \leftarrow Node.tail$;
      $Children \leftarrow Node_{tmp}$;
  for $node \in Children$ **do**
    $DFS(node)$;
  if $Children.size = 0 and Node.head is\ not\ in P_{candidate}$ **then**
    if $Node.occupancy > \beta$ **then**
      $Node.quality = Node.support + \lambda \times Node.occupancy$;
      add$Node.head$to$P_{candidate}$;
$P \leftarrow P_{candidate}$;

According to the definition of support and occupancy, we can calculate the quality of an itemset as follows.

Definition: (Quality) The quality of an itemset X is defined as $q(X) = \sigma(X)+\lambda\phi(X)$, where λ is a user defined balance weight parameter between support and occupancy ranges from 0 to ∞.

In addition, [6] show that incorporating occupancy into frequent pattern mining can give high quality pattern recommendation. Given minimal frequency threshold and minimal occupancy threshold, we mine Top-k dominant and frequent patterns.

Based on our experiment, though DOFIA outperforms the MAFIA in occupancy, there do exist some tasks which overlap a lot in the returned top-k dominant and frequent subtasks. We aim to obtain longer tasks while we wish the task we derived could have a high portion (occupancy) in the whole transaction. Thus we combine MAFIA and DOFIA to develop a hybrid algorithm named DOMAFIA in order to solve the problem of tasks overlapping and lift the whole quality of the result. The algorithm detail is shown in Algorithm 1. The returned patterns by DOMAFIA are ranked by their quality in descending order. In comparison, MAFIA and DOFIA are two baselines, of which the results are ranked according to support and quality value respectively.

III. EXPERIMENTAL EVALUATION

We evaluated our approach on a real print log dataset from a commercial company. The print log contains a period of time dataset with over 100,000 records from more than 16,000 users, where 29,740 unique domains are related. After using *delicious.com* tags as representation, more than 26,000 different tags are obtained in total. From the print log, we derived more than 1,000 subtasks (clusters) and hundreds of tasks (patterns). Table VI gives a summary of our data. To avoid bias,the subtask and task results are mixed and each user study case is judged by at least four judgers.

Table VI
PRINT LOG STATISTICS

users	tags	domains	subtasks	tasks(DOMAFIA)
16,041	26,533	29,740	1043	114

A. Examples of Subtasks and Tasks

Due to space limitation, we only show some examples of the subtasks and tasks in Table IV and V respectively. The complete list of our result can be found on an anonymous website[5].We manually label each subtask and task.

As shown in Table IV, we can get several observations. First, the subtask topics span a broad spectrum of life, including news, sports, travel and so on. Second, tags in a subtask are alternative or relevant to each other as we

[5]https://www.dropbox.com/s/x2e7zn0fuhb46nx/completelist.zip

Table VII
(a) Subtasks Homogeneity

Subtask Size	#Subtasks	#Tags	#Inconsistent Tags	Homogeneity
(0,10]	166	706	22	96.9%
(10,20]	24	350	21	94%
(20,∞)	10	404	9	97.7%
overall	200	1460	52	96.2%

(b) Subtasks Heterogeneity

	assessor 1	assessor 2	assessor 3	assessor 4	average
#"yes"	5	8	7	19	9.75
#"no"	495	492	493	481	490.25
Heterogeneity	99%	98.4%	98.6%	96.2%	98.05%

Table VIII
TOP-5 TASKS MINED BY MAFIA AND DOFIA

Method	Tasks
MAFIA	book hotel, cheap airline, travel, electronic, buy something
	book hotel, cheap airline, travel, electronic
	book hotel, cheap airline, travel, thesaurus
	book hotel, cheap airline, travel, food or cooking related
	book hotel, cheap airline, travel, news
DOFIA	get discount, buy book, thesaurus, cd or dvd, buy something
	car, buy something, london, review
	suchmaschine(primary search engine)
	finance like bank or stock or invest, france
	thesaurus, bookstore, buy something

Table IX
STATISTICS FOR TASK CATEGORIES OF THREE METHODS

category	shopping	finance	media	sports	travel	food	home	computer	kids	music	job	uncertain
MAFIA	1	0	0	0	94	0	0	1	0	2	0	2
DOFIA	25	12	13	13	20	2	2	8	0	0	0	8
DOMAFIA	9	8	10	9	17	11	7	11	3	2	4	9

expected. For example, in subtask *cheap airline* we cluster *airline* and *low cost* together and attach an advanced label *cheap airline* rather than simple *airline*, through which it can describe users' intentions well. In this case, we get to know that people tend to buy cheap airline ticket. Third, the divergence of two subtasks is evident. They overlap rarely.

In Table V, the selected tasks are listed. The task size, which is the number of subtasks in a set, ranges from 2 to 5 in the table. We can easily find clues and relationships between subtasks in a task. As example No.1 shows, if we are planning a travel, we need to buy flight ticket, better on discount (*cheap airline*) and book hotel. Moreover, shopping before setting out to prepare necessaries is reasonable. Renting a car when arriving at a new city seems a wise choice if you could drive. Other tasks listed in the table can also be explained easily under a scenario.

B. Quality of Subtasks

In order to evaluate the quality of the subtasks, we sample 200 subtasks from the cluster results and manually label them, all of which appeared in the result of tasks. According to the measurement philosophy of clustering, we will illustrate homogeneity and heterogeneity of the sampled subtask dataset. We perform a user study with four participants who all have background in computer science exclude the authors themselves.

Homogeneity shows whether tags in a subtask belongs to the same semantics or not. The experimental process is as follows. The mined subtasks are shown to assessor one by one, where tags that are in semantically disharmony with other tags are picked out. The number of disharmonious tags in the subtask is recorded and then accumulated according to the three subtask size categories, i.e., the number of tags in a subtask. Finally the sum is listed in column 4 table VII(a). As an example of inconsistence, we point out the subtask *buy books*. Tag *ebook* and tag *bucher* seems talking about different things. It is more rational to eliminate tag *bucher* from the subtask. We define all the tags like *bucher* in subtasks as *inconsistent tags*.

Heterogeneity measures how well a subtask is separated from another subtask. We call two subtasks *insufficient clustered* if they should be merged. We use the same sampled subtask dataset in measuring homogeneity and develop a evaluation system for four assessors to make their decisions and record their judgements. In the system, a pair of subtasks is showed each time and each assessor need to judge wether the two subtasks should be merged or not. On account of the subtask number reaching to 200 thus making the pair number to 19900, we randomly select 500 pairs for assessors' evaluation. Assessors say "yes" or "no" to the pair, afterwards the number of "yes" and "no" are recorded and heterogeneity is calculated as number of "no" divided by 500, the number of randomly selected subtask pair.

We show the result of homogeneity and heterogeneity in Table VII(a) and VII(b) respectively. The statistics in Table VII(a) shows that there are a few inconsistent tags in subtasks and the average percentage of inconsistence is less than 5%. In addition, Table VII(b) gives the evaluation from four assessors. The average heterogeneity ratio is 98.05%. It powerfully demonstrates that the mined subtasks with high internal homogeneity and high external heterogeneity are meaningful.

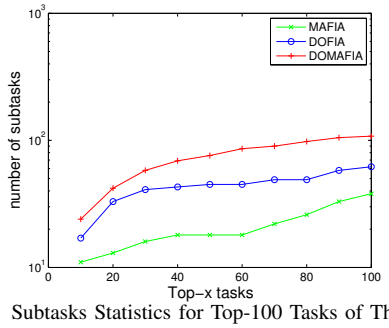

Figure 4. Subtasks Statistics for Top-100 Tasks of Three Methods

C. Quality of Tasks

We devise a hybrid algorithm named DOMAFIA by combining MAFIA and DOFIA to mine tasks from the subtask transaction database. In comparison, three algorithms are employed and the results are ranked in a descending order according to different metrics. For MAFIA, we rank the tasks in considering *support* value of each task while *quality* is used to measure tasks in DOFIA and DOMAFIA result. Top-5 tasks of both MAFIA and DOFIA are shown in Table VIII, where DOMAFIA have same top-5 tasks with DOFIA. Intuitively, we can clearly derive the conclusion that DOFIA shows better diversity than MAFIA. We carefully check the difference between these methods and look deep into the results in both subtask and task view.

By viewing in task level, we manually check the category of each task in top-100 list returned by the three methods. Based on our knowledge, we classify the tasks into categories and keep controversial tasks to the category "uncertain". The statistics are shown in Table IX, from which the distribution of the tasks by MAFIA is heavily skewed while DOFIA and DOMAFIA give an relative balanced one. Besides, DOFIA shows more categories than MAFIA while DOMAFIA covers three more categories than DOFIA.

By viewing at subtask level, we count top-x tasks of three methods when x changes from 0 to 100 with step length set to 10. Figure 4 shows the result, where x axis is the top-x tasks and y-axis is the counted subtask number in top-x tasks. We can find out that DOMAFIA covers more subtasks and then DOFIA and MAFIA followed. With the further increase of the number of tasks, three curves become approaching and the value on y-axis reachs to the whole number of the subtasks we mined.

IV. Related Work

One related work is the query suggestions on search logs. Under the assumption that two items should be clustered together if they co-occur a lot in click-through bipartite, most of work [7], [8], [9], [10] use different methods to find similar queries of user's input as representations of user's intention in order to give suggestions. However, search logs and print logs are very different. Print logs is lack of abundant users' queries and click data, while search logs records users' click-through histories to construct a bipartite by taking co-occur queries into consideration. It is a great challenge to mine tasks from print logs using little information of URLs. Due to the above reasons, we can't adopt methods on search logs to represent user's intention. Instead, we try *delicious.com api* to fetch tags for each URL domain as representations, which addresses the problem of poor representability in print logs itself.

Another related work is the task mining on search logs. Recent years, many companies provide search engine tools or web browser tools for organizing users' tasks. From the time order, Compaq *Search Pad* [11], Microsoft *Search Bar* [12] and *Yahoo! Search Pad* [13] are three representative example for mining tasks. However, These tools for mining tasks focus on single user's history while our work mines "the wisdom of crowd" from print logs.

Compared to the well-studied search log, the print log dataset seems more clean and less noisy, which reflects user's intention more directly with a set of printed URLs. This is mainly because users need to pay for the printing suppliers and therefore they usually only print those necessary information and significant contents, which can reflect users' intentions. Moreover, search log often consists of lots of queries and their corresponding clicked URLs, in which only a small part of clicked URLs are really useful to users. In contrast, the print log studied in this paper, which was collected from Smart Print, one plug-in of Bing search bar by Microsoft, provides a more useful and direct means to reveal users' intentions. For example, user 2 in Table I printed out web pages according to his judgment from the returned results by querying "healthcare fraud detection" in the bing search engine[6], where the listed printing URLs are ranked 3 and 9 respectively in the search result. To the best of our knowledge, our work is the first attempt in mining user print tasks on print logs.

Social tags are frequently utilized as a auxiliary source for understanding users. Recommend System performance is raised with embedded social tags [22], [23]. Li *et al.* [24] coined an efficient algorithm to mine frequent patterns with social tags. However, using social tags to derive user behavior is not systematically studied. Since tag clustering is an essential part of mining subtasks, upon which Wang *et al.* [15] proposed a new approach detecting the similar tags from software document by using a newly proposed similarity metric. Besides, Papadopoulos *et al.* [16] presented a graph-based clustering algorithm to identify related tags in folksonomies. Liu *et al.* [17] modeled the relationship among tags in a photo application. Wang *et al.* [18] incorporated the social network information into tag-based music style clustering. Radelaar *et al.* [19] detected semantically synonymous tags with Flickr data. Most of the tags clustering algorithm on graph mainly focus on a

[6]Note that the Smart Print is installed in bing search engine bar in IE browser.

specific area or dataset, instantiated in photo graph. We do tag clustering on print logs which is significantly different from those graph data and any graph clustering method can be embedded into our framework.

As pattern mining plays an important role in mining tasks, a conventional pattern mining algorithm is proposed by Agrawal [4]. Following this, many newly algorithm were developed to mining diverse patterns. Dough [3] mine maximal frequent patterns while Zhang *et al.* [6] incorporate a new constraint called "occupancy" into the lexicographic tree for pruning. We absorb the advantaged of both and proposed a new algorithm DOMAFIA for stemming with two constraints.(See Subsection II-D).

V. Conclusion and Future Work

In this paper, we exploit a new user interaction data, namely print logs, which is recognized cleaner than search logs. Specifically, we attempt to mine user tasks form print logs, which undoubtedly benefits many applications such as tasks-oriented recommendations and behavior targeting. Thus, we formulate the problem of mining user tasks and then propose a general mining framework UPT (User Print Tasks mining framework) for this problem. Finally, the empirical study shows that our mined subtasks and tasks are reasonable and promising. It is worth mentioning that for some domains such as *www.amazon.com* with broad coverage of URL, our approach can only extract the common sense embedded in the domain, e.g., online shopping but cannot differentiate the specific intention of users, e.g., *www.amazon.com/phone*. Hence, information under the domain should be taken into consideration and this could be left for future work.

VI. Acknowledgements

This research was supported by grants from National Key Technology Research and Development Program of the Ministry of Science and Technology of China(Grant No.2012BAH17B03), Anhui Programs for Science and Technology Development(Grant No.1301022064), International Science and Technology Cooperation Program of Anhui Province(Grant No.1303063008), Fundamental Research Funds for the Central Universities(Grant No. WK0110000042) and Science and Technology Development of Anhui Province(Grant No.13Z02008-5), Nature Science Research of Anhui(Grant No.1208085MF95), the Nuture Science Foundation of Anhui Education Department(Grant No.KJ2012A273, KJ2012A274), Technology Foundation for Selected Overseas Chinese Scholar.

References

[1] J. Lin. Divergence measures based on the sannon entropy. In *IEEE Transactions on Information Theory*, 37(1), 1991.

[2] T.H. Cormen and C.E. Leiserson and R.L. Rivest and C. Stein. Chapter 21: Data structures for Disjoint Sets. *Introduction to Algorithms (Second ed.)*, MIT Press, pp. 498C524, 2001.

[3] D. Burdick and M. Calimlim and J. Gehrke. MAFIA: a maximal frequent itemset algorithm for transactional databases. In *International Conference of Data Engineering*, 2011.

[4] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. *VLDB*, 1994.

[5] J. Han and J. Pei and Y. Yin. Mining Frequent Patterns without Candidate Generation. In *SIGMOD*, 2000.

[6] L. Tang and L. Zhang and P. Luo and M. Wang. Incorporating Occupancy into Frequent Pattern Mining for High Quality Pattern Recommendation. In *KDD*, 1996.

[7] R. Baeza-Yates and C. Hurtado and M. Mendoza. Query Recommendation using Query Logs in Search Engines. *EDBT Workshop*, 2004.

[8] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *KDD*, 2000.

[9] H. Cao and D. Jiang and J. Pei and Q. He and Z. Liao and E. Chen and H. Li. Context-aware query suggestion by mining click-through and session data. *KDD*, 2008.

[10] J.R. Wen and J.Y Nie and H.J. Zhang. Clustering User Queries of a Search Engine. *WWW*, 2001.

[11] K. Bharat. SearchPad: explicit capture of search context to support Web search. *Computer Networks*, Pages 493C501, 2000.

[12] D. Morris and M.R. Morris and G. Venolia. SearchBar: A Search-Centric Web History for Task Resumption and Information Re-finding. In *CHI*,2008.

[13] D. Donato and F. Bonchi. Do You Want to Take Notes? Identifying Research Missions in Yahoo! Search Pad. In *WWW*, 2010.

[14] R. Balasubramanyan and F. Lin and W.W. Cohen. Node Clustering in Graphs: An Empirical Study. In *NIPS Workshop*, 2010.

[15] S. Wang, D. Lo and L. Jiang. Inferring semantically related software terms and their taxonomy by leveraging collaborative tagging. In *ICSM*, 2012.

[16] S. Papadopoulos and Y. Kompatsiaris and A. Vakali. A Graph-based Clustering Scheme for Identifying Related Tags in Folksonomies. In *DAWAK*, 2010.

[17] Y. Liu and F. Wu and Y. Zhang and J. Shao and Y. Zhuang. Tag Clustering and Refinement on Semantic Unity Graph. In *ICDM*, 2011.

[18] D. Wang and M. Ogihara. Potential Relationship Discovery in Tag-Aware Music Style Clustering and Artist Social Networks. In *ISMIR*, 2011.

[19] J. Radelaar and A.J. Boor and D. Vandic and J.W. Dam and F. Hogenboom and F. Frasincar. Improving the Exploration of Tag Spaces Using Automated Tag Clustering. In *ICWE*, 2011.

[20] D. Lo and S.C. Khoo and C. Liu. Efficient Mining of Iterative Patterns for Software Specification Discovery. In *KDD*, 2007.

[21] H. Mousselly-Sergieh and M. Doller and E. Egyed-Zsigmond and G. Gianini and H. Kosch and J.M. Pinon. Tag Relatedness Using Laplacian Score Feature Selection and Adapted Jensen-Shannon Divergence. In *MMM*, pp 159-171, 2014.

[22] T. Zhou and H. Ma and M.R. Lyu and I. King. UserRec: A User Recommendation Framework in Social Tagging Systems. In *AAAI*, 2010.

[23] G. Xu and Y. Gu and P. Dolog and Y. Zhang and M. Kitsuregawa. SemRec: A Semantic Enhancement Framework for Tag Based Recommendation. In *AAAI*, 2011.

[24] X. Li and L. Zhang and E. Chen and Y. Zong and G. Xu. Mining Frequent Patterns in Print Logs with Semantically Alternative Labels. In *ADMA*, 2013.