Abstract--The large-scale user-generated meta-data not only facilitate users in sharing and organizing multimedia content, but provide useful information to improve media retrieval and management. Personalized search serves as one of such examples where the web search experience is improved by generating the returned list according to the modified user search intents. In this paper, we exploit the annotations and propose a novel framework simultaneously considering the user and query relevance to learn to personalized content like that image search. The basic premise is to embed the user preference and query-related search intent into user-specific topic spaces. Since the users’ original annotation is too sparse for topic modeling, we need to enrich users’ annotation pool before user specific topic spaces construction. The study privacy protection in PWS applications that model user preferences as hierarchical user profiles. The propose a PWS framework called UPS that can adaptively generalize profiles by queries while respecting user-specific privacy requirements. Our runtime generalization aims at striking a balance between two predictive metrics that evaluate the utility of personalization and the privacy risk of exposing the generalized profile.

I. INTRODUCTION

The web search engine has long become the most important portal for ordinary people looking for useful information on the web. However, users might experience failure when search engines return irrelevant results that do not meet their real intentions. Such irrelevance is largely due to the enormous variety of users’ contexts and backgrounds, as well as the ambiguity of texts. Personalized web search (PWS) is a general category of search techniques aiming at providing better search results, which are tailored for individual user needs. As the expense, user information has to be collected and analyzed to figure out the user intention behind the issued query. The solutions to PWS can generally be categorized into two types, namely click-log-based methods and profile-based ones. The click-log based methods are straightforward they simply impose bias to clicked pages in the user’s query history. Although this strategy has been demonstrated to perform consistently and considerably well it can only work on repeated queries from the same user, which is a strong limitation confining its applicability. In contrast, profile-based methods improve the search experience with complicated user-interest models generated from user profiling techniques. Profile-based methods can be potentially effective for almost all sorts of queries, but are reported to be unstable under some circumstances. Although there are pros and cons for both types of PWS techniques, the profile-based PWS has demonstrated more effectiveness in improving the quality of web search recently, with increasing usage of personal and behavior information to profile its users, which is usually gathered implicitly from query history, browsing history click-through data bookmarks, user documents, and so forth. Unfortunately, such implicitly collected personal data can easily reveal a gamut of user’s private life. Privacy issues arising from the lack of protection for such data, for instance the AOL query logs scandal, not only raise panic among individual users, but also dampen the data-publisher’s enthusiasm in offering personalized service. In fact, privacy concerns have become the major barrier for wide proliferation of PWS services. To protect user privacy in profile-based PWS, researchers have to consider two contradicting effects during the search process. On the one hand, they attempt to improve the search quality with the personalization utility of the user profile. On the other hand, they need to hide the privacy contents existing in the user profile to place the privacy risk under control.

A few previous studies suggest that people are willing to compromise privacy if the personalization by supplying user profile to the search engine yields better search quality. In an ideal case, significant gain can be obtained by personalization at the expense of only a small (and less-sensitive) portion of the user profile, namely a generalized profile. Thus, user privacy can be protected
without compromising the personalized search quality. In general, there is a trade-off between the search quality and the level of privacy protection achieved from generalization. Unfortunately, the previous works of privacy preserving PWS are far from optimal. The problems with the existing methods are explained in the following observations. Only once offline, and used to personalize all queries from a same user indiscriminatingly. Such “one profile fits all” strategy certainly has drawbacks given the variety of queries. One evidence reported in is that profile-based personalization may not even help to improve the search quality for some ad hoc queries, though exposing user profile to a server has put the user’s privacy at risk. A better approach is to make an online decision on a. whether to personalize the query (by exposing the profile) and b. what to expose in the user profile at runtime. To the best of our knowledge, no previous work has supported such feature.

**Contributions:**

The existing methods do not take into account the customization of privacy requirements. This probably makes some user privacy to be overprotected while others insufficiently protected. For example, in, all the sensitive topics are detected using an absolute metric called surprisal based on the information theory, assuming that the interests with less user document support are more sensitive. However, this assumption can be doubted with a simple counterexample: If a user has a large number of documents about “sex,” the surprise of this topic may lead to a conclusion that “sex” is very general and not sensitive, despite the truth which is opposite. Unfortunately, few prior works can effectively address individual privacy needs during the generalization. Many personalization techniques require iterative user interactions when creating personalized search results. They usually refine the search results with some metrics which require multiple user interactions, such as rank scoring, average rank, and so on. This paradigm is, however, infeasible for runtime profiling, as it will not only pose too much risk of privacy breach, but also demand prohibitive processing time for profiling. Thus, we need predictive metrics to measure the search quality and breach risk after personalization, without incurring iterative user interaction. The above problems are addressed in our UPS (literally for User customizable Privacy-preserving Search) framework. The framework assumes that the queries do not contain any sensitive information, and aims at protecting the privacy in individual user profiles while retaining their usefulness for PWS. As illustrated in Fig. 1, UPS consists of a nontrusty search engine server and a number of clients. Each client (user) accessing the search service trusts no one but himself/herself. The key component for privacy protection is an online profiler implemented as a search proxy running on the client machine itself. The proxy maintains both the complete user profile, in a hierarchy of nodes with semantics, and the user-specified (customized) privacy requirements represented as a set of sensitive-nodes. The framework works in two phases, namely the offline and online phase, for each user.

Hierarchical user profile is constructed and customized with the user-specified privacy requirements. The online phase handles queries as follows: 1. when a user issues a query qi on the client, the proxy generates a user profile in runtime in the light of query terms. The output of this step is a generalized user profile Gi satisfying the privacy requirements. The generalization process is guided by considering two conflicting metrics, namely the personalization utility and the privacy risk, both defined for user profiles. 2. Subsequently, the query and the generalized user profile are sent together to the PWS server for personalized search. 3. The search results are personalized with the profile and delivered back to the query proxy. 4. Finally, the proxy either presents the raw results to the user, or re-ranks them with the complete user profile. UPS is distinguished from conventional PWS in that it 1) provides runtime profiling, which in effect optimizes the personalization utility while respecting user’s privacy requirements; 2) allows for customization of privacy needs; and 3) does not require iterative user interaction. Our main contributions are summarized as following: We propose a privacy-preserving personalized web search framework UPS, which can generalize profiles for each query according to user-specified privacy requirements. Relying on the definition of two conflicting metrics, namely personalization utility and privacy risk, for hierarchical user profile, we formulate the problem of privacy-preserving personalized search as _Risk Profile Generalization, with itsNP-hardness proved. We develop two simple but effective generalization algorithms, GreedyDP and GreedyIL, to support runtime profiling. While the former tries to maximize the discriminating power (DP), the latter attempts to minimize the information loss (IL). By exploiting a number of heuristics, GreedyIL outperforms GreedyDP significantly. We provide an inexpensive mechanism for the client to
decide whether to personalize a query in UPS. This decision can be made before each runtime profiling to enhance the stability of the search results while avoiding the unnecessary exposure of the profile. Our extensive experiments demonstrate the efficiency and effectiveness of our UPS framework.

II. RELATED WORK

In this section, we overview the related works. We focus on the literature of profile-based personalization and privacy protection in PWS system.

Profile-Based Personalization:
Previous works on profile-based PWS mainly focus on improving the search utility. The basic idea of these works is to tailor the search results by referring to, often implicitly, a user profile that reveals an individual information goal. In the remainder of this section, we review the previous solutions to PWS on two aspects, namely, the representation of profiles, and the measure of the effectiveness of personalization. Many profile representations are available in the literature to facilitate different personalization strategies. Earlier techniques utilize term lists/vectors or bag of words to represent their profile. However, most recent works build profiles in hierarchical structures due to their stronger descriptive ability, better scalability, and higher access efficiency. The majority of the hierarchical representations are constructed with existing weighted topic hierarchy/graph, such as ODP1, and so on. Another work in builds the hierarchical profile automatically via term-frequency analysis on the user data. In our proposed UPS framework, we do not focus on the implementation of the user profiles. Actually, our framework can potentially adopt any hierarchical representation based on taxonomy of knowledge. As for the performance measures of PWS in the literature, Normalized Discounted Cumulative Gain is a common measure of the effectiveness of an information retrieval system. It is based on a human graded relevance scale of item-positions in the result list, and is, therefore, known for its high cost in explicit feedback collection. To reduce the human involvement in performance measuring, researchers also propose other metrics of personalized web search that rely on clicking decisions, including Average Precision (AP), Rank Scoring, and Average Rank. We use the Average Precision metric, proposed by Dou et al., to measure the effectiveness of the personalization in UPS. Meanwhile, our work is distinguished from previous studies as it also proposes two predictive metrics, namely personalization utility and privacy risk, on a profile instance without requesting for user feedback.

Privacy Protection in PWS System

Generally there are two classes of privacy protection problems for PWS. One class includes those treat privacy as the identification of an individual, as described. The other includes those consider the sensitivity of the data, particularly the user profiles, exposed to the PWS server. Typical works in the literature of protecting user identifications (class one) try to solve the privacy problem on different levels, including the pseudo identity, the group identity, no identity, and no personal information. Solution to the first level is proved to fragile. The third and fourth levels are impractical due to high cost in communication and cryptography. Therefore, the existing efforts focus on the second level. Both and provide online anonymity on user profiles by generating a group profile of k users. Using this approach, the linkage between the query and a single user is broken. In the useless user profile (UUP) protocol is proposed to shuffle queries among a group of users who issue them. As a result any entity cannot profile a certain individual. These works assume the existence of a trustworthy third-party anonymizer, which is not readily available over the Internet at large. Viejo and Castell a use legacy social networks instead of the third party to provide a distorted user profile to the web search engine. In this scheme, every user acts as a search agency of his or her neighbors. They can decide to submit the query on behalf of who issued it, or forward it to other neighbors. The shortcomings of current solutions in class one is the high cost introduced due to the collaboration and communication. The solutions in class two do not require third-party assistance or collaborations between social network entries. In these solutions, users only trust themselves and cannot tolerate the exposure of their complete profiles an anonymity server. In Krause and Horvitz employ statistical techniques to learn a probabilistic model, and then use this model to generate the near-optimal partial profile. One main limitation in this work is that it builds the user profile as a finite set of attributes, and the probabilistic model is trained through predefined frequent queries. These assumptions are impractical in the context of PWS. Xu et al. Proposed a privacy protection solution for PWS based on hierarchical profiles. Using a user-specified threshold, a generalized profile is obtained in effect as a rooted subtree of the complete profile. Unfortunately, this work does not address the query utility, which is crucial for the service quality of PWS. For comparison, our approach takes both the privacy requirement and the query utility into account.

A more important property that distinguishes our work from is that we provide personalized privacy protection in PWS. The concept of personalized privacy protection is first introduced by Xiao and Tao in Privacy-Preserving Data Publishing (PPDP). A person can specify the degree of privacy protection for her/his sensitive values by specifying “guarding nodes” in the taxonomy of the
sensitive attribute. Motivate by this, we allow users to customize privacy needs in their hierarchical user profiles. Aside from the above works, a couple of recent studies have raised an interesting question that concerns the privacy protection in PWS.

III. PROPOSED SYSTEM

In Proposed System We propose a novel personalized image search framework by simultaneously considering user and query information. The user’s preferences over images under certain query are estimated by how probable he/she assigns the query-related tags to the images.

1) A Ranking based Multi-correlation Tensor Factorization model is proposed to perform annotation prediction, which is considered as users’ potential annotations for the images;

2) The introduce User-specific Topic Modeling to map the query relevance and user preference into the same user-specific topic space. For performance evaluation, two resources involved with users’ social activities are employed. Experiments on a large-scale Flickr dataset demonstrate the effectiveness of the proposed method.

ADVANTAGES OF PROPOSED SYSTEM:

- A ranking based tensor factorization model named RMTF is proposed to predict users’ annotations to the images.
- To better represent the query-tag relationship, we build user-specific topics and map the queries as well as the users’ preferences onto the learned topic spaces.

IV. PRELIMINARIES and PROBLEM DEFINITION

In this section, we first introduce the structure of user profile in UPS. Then, we define the customized privacy requirements on a user profile. Finally, we present the attack model and formulate the problem of privacy preserving profile generalization. For ease of presentation, Table 1 summarizes all the symbols used in this paper.

4.1 User Profile

Consistent with many previous works in personalized web services, each user profile in UPS adopts a hierarchical structure. Moreover, our profile is constructed based on the availability of a public accessible taxonomy, denoted as R, which satisfies the following assumption. Assumption 1. The repository R is a huge topic hierarchy covering the entire topic domain of human knowledge. That is, given any human recognizable topic t, a corresponding node (also referred to as t) can be found in R, with the subtree as the taxonomy accompanying t. The repository is regarded as publicly available and can be used by anyone as the background knowledge. Such repositories do exist in the literature, for example, the ODP Wikipedia and so on. In addition, each topic t ∈ R is associated with a repository support, denoted by which quantifies how often the respective topic is touched in human knowledge. If we consider each topic to be the result of a random walk from its parent topic in R, we have the following recursive equation: \( \text{sup}_R(t) \propto X \text{t02Ct}_R;R \text{sup}_R(t0); \delta t \). Equation (1) can be used to calculate the repository support of all topics in R, relying on the following assumption that the support values of all leaf topics in R are available. Assumption 2. Given a taxonomy repository R, the repository support is provided by R itself for each leaf topic. In fact, Assumption 2 can be relaxed if the support values are not available. In such case, it is still possible to “simulate” these repository supports with the topological structure of R. That is, can be calculated as the count of leaves Based on the taxonomy repository, we define a probability model for the topic domain of the human knowledge. In the model, the repository R can be viewed as a hierarchical partitioning of the universe (represented by the root topic) and every topic t ∈ R stands for a random event. The conditional probability (s is an ancestor of t) is defined as the proportion of repository support: \( \text{Pr}(s \mid t) \). Thus, \( \text{Pr}(t) \) can be further defined as where is the root topic which has probability 1. Now, we present the formal definition of user profile. A user profile H, as a hierarchical representation of user interests, is a rooted subtree of R. The notion rooted subtree is given in Definition Given two trees S and T, S is a rooted subtree of T if S can be generated from T by removing a node set X \( \_ T \) (together with subtrees) from T. A diagram of a sample user profile is illustrated in Fig. 2a, which is constructed based on the sample taxonomy repository in Fig. 2b. We can observe that the owner of this profile is mainly interested in Computer Science and Music, because the major portion of this profile is made up of fragments from taxonomies of these two topics in the sample repository. Some other taxonomies also serve in comprising the profile, for example, Sports and Adults. Customized Privacy Requirements Customized privacy requirements can be specified with a number of sensitive-nodes (topics) in the user profile, whose disclosure (to the server) introduces privacy risk to the user. It must be noted that user’s privacy concern differs from one sensitive topic to another. In the above example, the user may hesitate to share her personal interests (e.g., Harmonica, Figure Skating) only to avoid various advertisements. Thus, the user might still tolerate the exposure of such interests to trade for better personalization utility. However, the user may never allow another interest in topic Adults to be disclosed. To address the difference in privacy concerns,
we allow the user to specify a sensitivity for each node $s$ 2 $S$. As the sensitivity values explicitly indicate the user’s privacy concerns, the most straightforward privacy preserving method is to remove sub trees rooted at all sensitive-nodes whose sensitivity values are greater than a threshold. Such method is referred to as forbidding. However, forbidding is far from enough against a more sophisticated adversary. To clearly illustrate the limitation of forbidding, we first introduce the attack model which we aim at resisting. Attack Model Our work aims at providing protection against a typical model of privacy attack, namely eavesdropping. As shown in Fig. 3, to corrupt Alice’s privacy, the eavesdropper Eve successfully intercepts the communication between Alice and the PWS-server via some measures, such as man-in-the-middle attack, invading the server, and so on. Consequently, whenever Alice issues a query $q$, the entire copy of $q$ together with a runtime profile $G$ will be captured by Eve. Based on $G$, Eve will attempt to touch the sensitive nodes of Alice by recovering the segments hidden from the original $H$ and computing a confidence for each recovered topic, relying on the background knowledge in the publicly available taxonomy repository $R$. Note that in our attack model, Eve is regarded as an adversary satisfying the following assumptions: Knowledge bounded. The background knowledge of the adversary is limited to the taxonomy repository $R$. Both the profile $H$ and privacy are defined based on $R$. Session bounded. None of previously captured information is available for tracing the same victim in a long duration.

In other words, the eavesdropping will be started and ended within a single query session. The above assumptions seem strong, but are reasonable in practice. This is due to the fact that the majority of privacy attacks on the web are undertaken by some automatic programs for sending targeted (spam) advertisements to a large amount of PWS-users. These programs rarely act as a real person that collects prolific information of a specific victim for a long time as the latter is much more costly. If we consider the sensitivity of each sensitive topic as the cost of recovering it, the privacy risk can be defined as the total (probabilistic) sensitivity of the sensitive nodes, which the adversary can probably recover from $G$. For fairness among different users, we can normalize the privacy risk with $P$ $s$ $2$ $S$ $\text{sen}$$\delta$ $s$, which stands for the total wealth of the user. Our approach to privacy protection of personalized web search has to keep this privacy risk under control.

Generalizing User Profile the inadequacy of forbidding operation. In the sample profile in Fig. 2a, Figure is specified as a sensitive node. Thus, $rsbtrS$; $H\rho$ only releases its parent Ice Skating. Unfortunately, an adversary can recover the subtree of Ice Skating relying on the repository shown in Fig. 2b, where Figure is a main branch of Ice Skating besides Speed. If the probability of touching both branches is equal, the adversary can have 50 percent confidence on Figure. This may lead to high privacy risk if $\text{sen}$$\delta$Figure$P$ is high. A safer solution would remove node Ice Skating in such case for privacy protection. In contrast, it might be unnecessary to remove sensitive nodes with low sensitivity. Therefore, simply forbidding the sensitive topics does not protect the user’s privacy needs precisely.

To address the problem with forbidding, we propose a technique, which detects and removes a set of nodes $X$ from $H$, such that the privacy risk introduced by exposing $G$ $\frac{1}{4}$ $rsbtrX$; $H\rho$ is always under control. Set $X$ is typically different from $S$. For clarity of description, we assume that all the subtrees of $H$ rooted at the nodes in $X$ do not overlap each other. This process is called generalization, and the output $G$ is a generalized profile. The generalization technique can seemingly be conducted during offline processing without involving user queries. However, it is impractical to perform offline generalization due to two reasons: 1. The output from offline generalization may contain many topic branches, which are irrelevant to a query. A more flexible solution requires online generalization, which depends on the queries. Online generalization not only avoids unnecessary privacy disclosure, but also removes noisy topics that are irrelevant to the current query.

A query $qa$ $\frac{1}{4}$ “K-Anonymity,” which is a privacy protection technique used in data publishing, a desirable result of online generalization might be $Ga$, surrounded by the dashed ellipse in Fig. 2a. For comparison, if the query is $qb$ $\frac{1}{4}$ “Eagles,” the generalized profile would better become $Gb$ contained in the dotted curve, which includes two possible intentions (one being a rock band and the other being an American football team Philadelphia Eagles). The node sets to be removed are $Xa$ $\frac{1}{4}$ $f$Adults; $Privacy$; $Database$; $Develop$; $Arts$; $Sportsg$, and $Xb$ $\frac{1}{4}$ $f$Adults; $Computer$ $Science$; $Instrument$; $Ice$ $Skatingg$, respectively. 2. It is important to monitor the personalization utility during the generalization. Using the running example, profiles $Ga$ and $Gb$ might be generalized to smaller rooted subtrees. However, overgeneralization may cause ambiguity in the personalization, and eventually lead to poor search results. Monitoring the utility would be possible only if we perform the generalization at runtime. We now define the problem of privacy-preserving generalization in UPS

![Diagram](https://via.placeholder.com/150)
as follows, based on two notions named utility and risk. The former measures the personalization utility of the generalized profile, while the latter measures the privacy risk of exposing the profile.

V. UPS PROCEDURES

In this section, we present the procedures carried out for each user during two different execution phases, namely the offline and online phases. Generally, the offline phase constructs the original user profile and then performs privacy requirement customization according to user-specified topic sensitivity. The subsequent online phase finds the Optimal Risk Generalization solution in the search space determined by the customized user profile. As mentioned in the previous section, the online generalization procedure is guided by the global risk and utility metrics. The computation of these metrics relies on two intermediate data structures, namely a cost layer and a preference layer defined on the user profile. The cost layer defines for each node \( t \in H \) a cost value \( c(t) \), which indicates the total sensitivity at risk caused by the disclosure of \( t \). These cost values can be computed offline from the user-specified sensitivity values of the sensitive nodes. The preference layer is computed online when a query \( q \) is issued. It contains for each node \( t \in H \) a value indicating the user’s query-related preference on topic \( t \).

These preference values are computed relying on a procedure called query topic mapping. Specifically, each user has to undertake the following procedures in our solution: 1. offline profile construction, 2. offline privacy requirement customization, 3. online query-topic mapping, and 4. online generalization. Offline-1. Profile Construction. The first step of the offline processing is to build the original user profile in a topic hierarchy \( H \) that reveals user interests. We assume that the user’s preferences are represented in a set of plain text documents, denoted by \( D \). To construct the profile, we take the following steps: 1. Detect the respective topic in \( R \) for every document \( d \in D \). Thus, the preference document set \( D \) is transformed into a topic set \( T \). 2. Construct the profile \( H \) as a topic-path trie with \( T \), i.e., \( H \) rooted subtree of \( T \). 3. Initialize the user support \( sup_{H(t)} \) for each topic \( t \in T \) with its document support from \( D \), then compute \( sup_{H(t)} \) of other nodes of \( H \) with (4). There is one open question in the above process.

5.1 SECURE EXTRANET

The introduce the two critical metrics for our generalization problem. Then, we present our method of online decision on personalization. Finally, we propose the generalization algorithms Metrics.: The purpose of the utility metric is to predict the search quality (in revealing the user’s intention) of the query \( q \) on a generalized profile \( G \). The reason for not measuring the search quality directly is because search quality depends largely on the implementation of PWS search engine, which is hard to predict. In addition, it is too expensive to solicit user feedback on search results. Alternatively, we transform the utility prediction problem to the estimation of the discriminating power of a given query \( q \) on a profile \( G \) under the following assumption.

Although the same assumption has been made in to model utility, the metric in that work cannot be used in our problem settings as our profile is a hierarchical structure rather than a flat one. Given a hierarchical profile \( G \) and a query \( q \), we can intuitively expect more discriminating power when \( \), \( \). To propose our model of utility, we introduce the notion of Information Content (IC), which estimates how specific a given topic \( t \) is. Online Decision: To Personalize or Not The results reported in demonstrate that there exist a fair amount of queries called distinct queries, to which the profile-based personalization contributes little or even reduces the search quality, while exposing the profile to a server would for sure risk the user’s privacy. To address this problem, we develop an online mechanism to decide whether to personalize a query.

The basic idea is straightforward— if a distinct query is identified during generalization, the entire runtime profiling will be aborted and the query will be sent to the server without a user profile. We identify distinct queries using the discriminating power (defined in Section 5.1). Specifically, remember that the personalization utility is defined as the gain in DP when exposing the generalized profile with the query. Thus, we consider the distinct queries as those with good DP even when the client does not expose any profile.

A predefined threshold, then \( q \) is considered a distinct query. The benefits of making the above runtime decision are twofold: 1. It enhances the stability of the search quality; 2. It avoids the unnecessary exposure of the user profile. 5.3 The Generalization Algorithms We start by introducing a brute-force optimal algorithm, which is proven to be NP-hard. Then, we propose two greedy algorithms, namely the GreedyDP and GreedyIL. 5.3.1 The Brute-Force Algorithm The brute-force algorithm exhausts all possible rooted subtrees of a given seed profile to find the optimal generalization. The privacy requirements are respected during the exhaustion. The subtree with the optimal utility is chosen as the result. Although the seed profile \( G_0 \) is significantly smaller than \( H \), the exponential computational complexity of brute-force algorithm is still unacceptable. Formally, we
have the following theorem whose proof is given in the appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TKDE.2012.201.

The _RPG problem (Problem 1) is NP-hard. 5.3.2 The GreedyDP Algorithm Given the complexity of our problem, a more practical solution would be a near-optimal greedy algorithm. As preliminary, we introduce an operator _t_!_l that indicates the removal of a leaf topic t from a profile. Formally, we denote by Gi _t_! _l Gi the process of pruning leaf t from Gi to obtain Gi_p. Obviously, the optimal profile G _ can be generated with a finite-length transitive closure of prune-leaf. The first greedy algorithm GreedyDP works in a bottom-up manner. Starting from G0, in every ith iteration, GreedyDP chooses a leaf topic t 2 T Gi d_q for pruning, trying to maximize the utility of the output of the current iteration, namely Gi_p. During the iterations, we also maintain a bestprofile-so-far, which indicates the Gi_p having the highest discriminating power while satisfying the _risk constraint. The iterative process terminates when the profile is generalized to a root-topic. The best-profile-so-far will be the final result (G _) of the algorithm. The main problem of GreedyDP is that it requires recomputation of all candidate profiles (together with their discriminating power and privacy risk) generated from attempts of prune-leaf on all t 2 T Gi d_q. This causes significant memory requirements and computational cost. 5.3.3 The GreedyIL Algorithm The GreedyIL algorithm improves the efficiency of the generalization using heuristics based on several findings. One important finding is that any prune-leaf operation reduces the discriminating power of the profile. In other words, the DP displays monotonicity by prune-leaf. Formally, we have the following theorem:

The above finding motivates us to maintain a priority queue of candidate prune-leaf operators in descending order of the information loss caused by the operator. Specifically, each candidate operator in the queue is a tuple like op ¼ ht : IL ot; Gi_pí, where t is the leaf to be pruned by op and IL ot; Gi_p indicates the IL incurred by pruning t from Gi. This queue, denoted by Q, enables fast retrieval of the bestso-far candidate operator. Theorem 2 also leads to the following heuristic, which reduces the total computational cost significantly.

The second finding is that the computation of IL can be simplified to the evaluation. The reason is that, referring to the second term remains unchanged for any pruning operations until a single leaf is left (in such case the only choice for pruning is the single leaf itself). Furthermore, consider two possible cases as being illustrated in Fig. 4: is a node with no siblings, and t is a node with siblings. The case C1 is easy to handle. However, the evaluation of IL in case C2 requires introducing a shadow sibling4 of t. Each time if we attempt to prune t, we actually merge t into shadow to obtain a new shadow leaf shadow0, together with the preference of t, i.e., Prðt; G _jq for pruning t incur risk. Finally, we have the following heuristic, which significantly saves the computation of IL ot; G. It can be seen that all terms in (16) can be computed efficiently. Heuristic dpðt; Gi; G _j q for pruning t incur risk. The third finding is that, in case C1 described above, prune-leaf only operates on a single topic t.

Thus, it does not impact the IL of other candidate operators in Q. While in case C2, pruning t incurs recomputation of the preference values of its sibling topics. Therefore, we have Heuristic 3. Once a leaf topic t is pruned, only the candidate operators pruning t’s sibling topics need to be updated in Q. In other words, we only need to recompute the IL values for operators attempting to prune t’s sibling topics. Algorithm 1 shows the pseudocode of the GreedyIL algorithm. In general, GreedyIL traces the information loss instead of the discriminating power. This saves a lot of computational cost. In the above findings, Heuristic 1 (line 5) avoids unnecessary iterations. Heuristics 2 (line 4, 10, 14) further simplifies the computation of IL. Finally, Heuristics 3 (line 16) reduces the need for IL-recompilation.

Customized Privacy Requirements Customized privacy requirements can be specified with a number of sensitive-nodes (topics) in the user profile, whose disclosure (to the server) introduces privacy risk to the user. It must be noted that user’s privacy concern differs from one sensitive topic to another. In the above example, the user may hesitate to share her personal interests (e.g., Harmonica, Figure Skating) only to avoid various advertisements. Thus, the user might still tolerate the exposure of such interests to trade for better personalization utility. However, the user may never allow another interest in topic Adults to be disclosed. To address the difference in privacy concerns, we allow the user to specify a sensitivity for each node s 2 SAs the sensitivity values explicitly indicate the user’s privacy concern, the most straightforward privacy preserving method is to remove sub trees rooted at all sensitive-nodes whose sensitivity values are greater than a threshold. Such method is referred to as forbidding. However, forbidding is far from enough against a more sophisticated adversary. To clearly illustrate the limitation of forbidding, we first introduce the attack model which we aim at resisting.

Attack Model Our work aims at providing protection against a typical model of privacy attack, namely eavesdropping. As shown in Fig. 3, to corrupt Alice’s privacy, the eavesdropper Eve successfully intercepts the communication between Alice and the PWS-server via some measures, such as man-in-
themiddle attack, invading the server, and so on. Consequently, whenever Alice issues a query q, the entire copy of q together with a runtime profile G will be captured by Eve. Based on G, Eve will attempt to touch the sensitive nodes of Alice by recovering the segments hidden from the original H and computing a confidence for each recovered topic, relying on the background knowledge in the publicly available taxonomy repository R. Note that in our attack model, Eve is regarded as an adversary satisfying the following assumptions: Knowledge bounded. The background knowledge of the adversary is limited to the taxonomy repository R. Both the profile H and privacy are defined based on R. Session bounded. None of previously captured information is available for tracing the same victim in a long duration.

5.2 EFFICIENCY OF GENERALIZATION ALGORITHMS

To study the efficiency of the proposed generalization algorithms, we perform GreedyDP and GreedyIL algorithms on real profiles. The queries are randomly selected from their respective query log. We present the results in terms of average number of iterations and the response time of the generalization. Fig. 6 shows the results of the experiment. For comparison, we also plot the theoretical number of iterations of the Optimal algorithm. It can be seen that both greedy algorithm outperform Optimal. GreedyDP bounds the search space to the finite-length transitive closure of prune-leaf. GreedyIL further reduces this measure with Heuristic 1. The greater the privacy threshold $\epsilon$, the fewer iterations the algorithm requires. The advantage of GreedyIL over GreedyDP is more obvious in terms of response time, as Fig. 6b shows. This is because GreedyDP requires much more recomputation of DP, which incurs lots of logarithmic operations. The problem worsens as the query becomes more ambiguous. For instance, the average time to process GreedyDP for queries in the ambiguous group is more than 7 seconds. In contrast, GreedyIL incurs a much smaller real-time cost, and outperforms GreedyDP by two orders of magnitude.

5.3 Scalability of Generalization Algorithms

We study the scalability of the proposed algorithms by varying 1) the seed profile size (i.e., number of nodes), and 2) the data set size (i.e., number of queries). For each possible seed profile size (ranging from 1 to 108), we randomly choose 100 queries from the AOL query log, and take their respective R0p as their seed profiles. All leaf nodes in a same seed profile are given equal user preference. None of previously captured information is available for tracing the same victim in a long duration.

5.4 Effective Analysis of Personalization

In this experiment, we evaluate the real search quality on commercial search engines using our UPS framework. The search results is reranked with the generalized profile output by GreedyIL over 50 target users. The final search quality is evaluated using the Average Precision of the click records of the users, which is defined as where $li$ is the $i$th relevant link identified for a query, and $n$ is the number of relevant links.
respectively. The GreedyIL has a $\_ ¼ 0.0$ and online decision mechanism disabled. From the results of both search engines, we can observe that improvements of the search quality for Medium Queries and Ambiguous Queries are much more significant than that of Distinct Queries. In particular, the personalization on Distinct Queries of Yahoo results reduces the average performance from 73.4 to 66.2 percent. This is because some irrelevant profile topics (noises) are added. The results demonstrate that profile-based personalization is more suitable for queries with small DPðq; RÞ: RP. Fig. 10 shows the results of search quality by varying the $\_$ threshold. It is observed that the average precision of FusionRank increases rapidly when $\_$ grows from 0.0 to 0.1. Then, further increasing $\_$ (in effect exposing more specific topics) will only improve the search quality marginally. Moreover, the AP of FusionRank based on Yahoo (Fig. 10a) has a significant drop when $\_ > 1/4.3$. A comparison between the personalization results of ODP and Yahoo reveal that, although the original ODPRank (AP $\_ ¼ 0.373\%$) is poorer than the original Yahoo-Rank (AP $\_ ¼ 46.7\%$), personalization on ODP will generate better ranking than that on Yahoo.

VI. CONCLUSION

This paper presented a client-side privacy protection framework called UPS for personalized web search. UPS could potentially be adopted by any PWS that captures user profiles in a hierarchical taxonomy. The framework allowed users to specify customized privacy requirements via the hierarchical profiles. In addition, UPS also performed online generalization on user profiles to protect the personal privacy without compromising the search quality. We proposed two greedy algorithms, namely GreedyDP and GreedyIL, for the online generalization. Our experimental results revealed that UPS could achieve quality search results while preserving user’s customized privacy requirements. The results also confirmed the effectiveness and efficiency of our solution. For future work, we will try to resist adversaries with broader background knowledge, such as richer relationship among topics (e.g., exclusiveness, sequentiality, and so on), or capability to capture a series of queries (relaxing the second constraint of the adversary in Section 3.3) from the victim. We will also seek more sophisticated method to build the user profile, and better metrics to predict the performance (especially the utility) of UPS.

REFERENCES


Author’s Details:

Mrs. K. K. Kavitha, MCA, M.Phil.,
Head of the Department,
Department of Computer Science,
Selvamm Arts and Science College ( Autonomous),
kavithakkcs@gmail.com

Mrs. C. Anitha, MCA., M.Phil.,
Assistant Professor,
Department of Computer Science,
Selvamm Arts and Science College ( Autonomous),
anithapnk1@gmail.com

M. Uma,
Student,
Department of Computer Science,
Selvamm Arts and Science College ( Autonomous)
umamahes.raj@gmail.com