Privacy-Aware Personalization for Mobile Advertising

Michaela Hardt*
Twitter Inc.
San Francisco, CA, USA
mila@twitter.com

Suman Nath
Microsoft Research
Redmond, WA, USA
suman.nath@microsoft.com

ABSTRACT

Mobile advertising is an increasingly important driver in the Internet economy. We point out fundamental trade-offs between important variables in the mobile advertisement ecosystem. In order to increase relevance, ad campaigns tend to become more targeted and personalized by using context information extracted from user's interactions and smartphone's sensors. This raises privacy concerns that are hard to overcome due to the limited resources (energy and bandwidth) available on the phones. We point out that in the absence of a trusted third party, it is impossible to maximize these three variables—ad relevance, privacy, and efficiency—in a single system. This leads to the natural question: *can we formalize a common framework for personalized ad delivery that can be instantiated to any desired trade-off point?* We propose such a flexible ad-delivery framework where personalization is done jointly by the server and the phone. We show that the underlying optimization problem is NP-hard and present an efficient algorithm with a tight approximation guarantee.

Since tuning personalization rules requires implicit user feedback, such as clicks, we ask *how can we, in an efficient and privacy-preserving way, gather statistics over a dynamic population of mobile users?* This is needed for end-to-end privacy of an ad system. We propose the first differentially-private distributed protocol that works even in the presence of a dynamic and malicious set of users.

We evaluate our methods with a large click log of location-aware searches in Microsoft Bing for mobile. Our experiments show that our framework can simultaneously achieve reasonable levels of privacy, efficiency, and ad relevance and can efficiently support a high churn rate of users during the gathering statistics that are required for personalization.

1. INTRODUCTION

Mobile advertising is an increasingly important driver in the Internet economy. Experts project that US mobile advertising spending, which increased 41.2% to $1.2 billion in 2011 compared with 2010, will 'explode' in coming years due to prevalence of smartphones, geolocation, and increasingly tech-savvy consumers [11]. To maximize the return of this huge advertisement spending, Internet advertisers increasingly work to provide more targeted and personalized advertising. Online ads can be personalized on desktops as well [30, 44]; however, the potential is significantly greater for mobile advertisement, thanks to modern smartphone's capability of inferring users activities and context such as location and transportation mode from various sensors [34]. An online advertiser can use users' contexts and activities, along with their browsing and click history, to show ads preferentially to the users who are more likely to be influenced by the ads. If a user who likes Italian food (inferred from her past browsing history) is found to be walking (inferred from the accelerometer sensor data) alone (inferred from the audio data) around lunch time, she can be shown ads of popular (inferred from other users' past locations) Italian restaurants within walking distance of her current location (inferred from the GPS). Such highly targeted advertising can significantly increase the success of an ad campaign in terms of the number of resulting views, clicks or purchases on an advertised web page.

However, such personalization raises serious privacy concerns. Personalized services rely on private and possibly sensitive information about a user's preferences and current and past activities. Such information might allow for identification of the user and her activities. Current advertisement systems require users to completely trust these systems to not do anything bad with the information. This trust can easily be violated, as for instance in a confirmed case where a Google employee spied on the accounts of four underage teens for months before the company was notified of the abuses [6]. Hence, a user may not be willing to share information useful for personalization. In the previous example, the user may not be willing to reveal the fact that she is out of the office during business hours. Clicks on ads personalized based on private data can also leak information about the user [21, 29].

A personalized ad delivery system has three main components involving private user data: *statistics gathering* to learn personalization rules by interpreting users' contexts and clicks as implicit relevance feedback, *ad delivery* to select the best ad for a user based on her current context, and *billing* advertisers to collect money for impressions or clicks. Each component leads to privacy concerns that must be addressed to ensure end-to-end privacy. In this paper, we focus on the first two components. (We refer readers to [42] for a private billing component.)

Privacy-Aware Ad Delivery. Current advertising systems such as Google, Yahoo!, and Microsoft as well as many research proposals efficiently personalize ads by collecting personal data at a server [30, 44]. To address the privacy concerns of such *server-only* personalization, several prior work proposed *client-only* solutions that keeps personal information at the client device and performs personalization at the client [39]. Several recently proposed systems such as Privad [22] and Repriv [15] started to explore the interesting space between server-only and client-only solutions. Repriv [15] allows users to control how much data is shared with the server. The personalization is carried out based on this limited information at the server-side. Privad [22] places a proxy between server and client to achieve anonymity of personalization requests.

The above systems optimize for various design goals and raise one natural question: *are there any fundamental trade-offs in the design space of personalized ad delivery that these systems present?*

*Work done while at Microsoft Research.

We answer this by formalizing the task of delivering personalized ads from a server to a client as an optimization problem with three important variables: (1) privacy, i.e., how much information about the user's context is shared with the server, (2) communication efficiency, i.e., how few ads are sent to the client, and (3) utility, i.e., how useful the displayed ads are to the user in terms of revenue and relevance. We show in Section 3 that, in the absence of any trusted third party, it is impossible to maximize all three design goals simultaneously. The aforementioned previous works on personalized ad delivery present various discrete points in the trade-off space: a server-only solution achieves optimal efficiency at the cost of privacy or utility, while a client-only solution ensures optimal privacy but sacrifices efficiency or utility.

This fundamental trade-off leads to another important question: *can we formalize a common framework for personalized ad delivery that can be instantiated to any desired trade-off point?* We provide an affirmative answer to the above question with a hybrid framework where the personalization is done jointly by the ad server and the client. In our framework, we formalize the task of ad delivery as an optimization problem involving the three variables between a user and the ad server. Users can decide how much information about their sensor readings or inferred contexts they are willing to share with the server. Based on this limited information, the server selects a set of ads or search results, with bounded communication overhead, and sends them to the client. The client then picks and displays the most relevant ad based on all the private information. A key challenge in this framework is to choose the set of ads sent by the server and the ad displayed at the client in a way that maximizes utility (i.e., revenue) given constraints on efficiency (i.e., maximum communication cost) and privacy (i.e., maximum information disclosure). In other instantiations, our framework can optimize a combination of revenue and efficiency given a constraint on privacy. Such a flexible framework is extremely useful in practice as different systems may have different priorities on these variables. We show that the optimization problem is NP-hard and present an efficient greedy algorithm for hybrid personalization with tight approximation guarantees.

Note that several existing advertising systems such as Privad and location based services [10, 28, 47] use a similar principle as our framework: the client releases limited information (e.g., broad interest category, cloaked region), the server chooses a set of ads/results to disseminate to the client, and finally the client chooses the most suitable ad/result based on private information. Our contribution is to formally analyze the framework and to show how to operate in a desired point in the vast trade-off space of privacy, efficiency, and utility. We achieve the latter by letting the user and server flexibly choose constraints on privacy, efficiency, and utility. In doing so, existing personalization solutions become special cases of our flexible framework; and the framework can be configured to explore other attractive points in the trade-off space.

Privacy-Preserving Statistics Gathering. Personalized ads are chosen based on historical information about which ads users in a context clicked on i.e., context-dependent click-through rates (CTRs) of ads. However, estimating CTRs constitutes a big privacy challenge: users are often unwilling to reveal their exact contexts and even their clicks as they leak information about their contexts. We need to address this challenge in order to ensure end-to-end privacy of the overall ad service. One might use a distributed privacy-preserving aggregation protocol [1, 12, 37, 40] to compute such statistics. However, these are unsuitable for a large population of mobile users where a small fraction of users can become unavailable *during* the course of computing CTRs. For example, a user may turn off her mobile device any time or may want to answer an aggregation query only at a convenient time when her phone is being charged and connected through a local WiFi network. Another user might decline to participate in the exchange of certain messages in the protocol. Yet another user might leave or join the community of mobile users. Existing protocols [1, 12, 37, 40] do not efficiently handle such dynamics (more details in Section 5.5), making them unsuitable for estimating CTRs from mobile users. Then, *how can we, in an efficient and privacy-preserving way, gather statistics over a dynamic population?*

We answer this with a novel aggregation protocol to compute CTRs from a highly dynamic population without a trusted server. To the best of our knowledge, this is the first differentially-private protocol that computes accurate aggregations efficiently even when a fraction of participants become unavailable or behave maliciously.

Note that our results can be applied to personalize not just online advertising but also other online services based on users' fine-grained contextual information including local search. For concreteness, we consider advertising throughout the paper.

We have evaluated our algorithm with a large trace of location-aware searches in Microsoft Bing for mobile. To the best of our knowledge, this is the first empirical study of ad-serving trade-offs between privacy, efficiency, and utility with real trace. Our results show that the trade-offs between privacy, efficiency, and utility are not very strong in practice and reasonable levels of all these three goals can be achieved simultaneously. Results also show that hybrid personalization achieves much better trade-offs than server-only and client-only personalization. Finally, our statistics gathering algorithm efficiently handles large churns of users.

In summary, we make the following contributions:

► We formalize a hybrid personalization framework with three optimization variables: privacy, efficiency, and utility. We show that the optimization problem of personalizing ads based on private contexts is NP-hard and present an efficient greedy algorithm with a tight approximation guarantee (Sec. 3).

► We develop a differentially-private protocol for estimating statistics required for personalization without a trusted server. In contrast to existing protocols, our protocol can efficiently handle a dynamic set of participating users (Sec. 4).

► We evaluate effectiveness and robustness of our solution on a large click log of location-aware searches in Microsoft Bing for mobile. Our results illustrate trade-offs between privacy, communication efficiency and utility in personalized ad delivery (Sec. 5).

2. THE FRAMEWORK AND DESIDERATA

2.1 The Framework

Our framework has three classes of participants: The *users* who are served ads (also referred to as clients) in their mobile contexts, the *advertisers* who pay for clicks on their ads, and the *ad service provider* (also referred to as the *server*) who decides which ads to display and is paid for clicks by the advertisers. The framework works in two (potentially parallel) phases.

► **Statistics Gathering.** In this phase, the server gathers various statistics (e.g., click-through-rates of various ads) from clients. This phase happens periodically in the background. (Sec. 4.)

► **Ad-delivery.** In this phase, the server uses statistics gathered in the previous phase and user's current context to select and deliver personalized ads to the user. We allow users to decide how much information about their sensor readings or inferred contexts they are willing to share with the server. Based on this limited information, the server selects a set of ads or search results, with bounded communication overhead, and sends them to the user. The user then

Figure 1: Framework.

picks and displays the most relevant ad based on all the private information. (Sec. 3.)

Privacy Guarantee. Statistics gathering and ad delivery use private data differently—statistics gathering uses historical context and click data from many users, while ad delivery uses a user's current context. Therefore, we use different, but state of the art, privacy guarantees for them. For the gathering of statistics, a user can decide whether or not to participate. If she decides to participate then she is guaranteed differential privacy [13], i.e., that her data will hardly affect the statistics. Such a strong privacy guarantee is needed in practice since statistics are often shared and used extensively. Differential privacy has been widely used (see for instance [23, 24, 31, 33, 37, 40, 43]). However, it seems to be incompatible with personalization that requires a single user's current context (instead of aggregate statistics). Therefore, in the spirit of many existing personalization systems and the modus operandi in many mobile applications [15, 26, 30, 44], we ensure user privacy through limited information disclosure. The information disclosure about a user in context c can be limited by generalizing the user's context obtaining $\hat{c}$ and only sending $\hat{c}$ to the server, e.g. instead of revealing that the user is *doing Yoga* in *Dolores Park*, the user only discloses to be *exercising*. The generalization of context is done over a hierarchy described later. For a context c that can be generalized to $\hat{c}$ we write $c \to \hat{c}$. The question is what is the right level of generalization of a context? We can let the user decide how to generalize her context with the help of existing tools from the UI community (e.g. [41]). Alternatively, we can guarantee ℓ-diversity [32] to protect the privacy of a user's sensitive contexts. ℓ-diversity has been used in location-based systems [3, 9, 45] to protect location privacy. The privacy of a generalized context is measured as (1) its number of descendants and (2) its ratio of non-sensitive descendants to sensitive descendants. The user can specify a minimum privacy requirement and we choose an appropriate cut in the context hierarchy so that (1) and (2) are satisfied and each leaf node has exactly one ancestor on the cut to which it is generalized, before it is sent off to the server. This introduces uncertainty of an adversary about whether the user is in a sensitive context or not.

Figure 1 illustrates our framework. The server periodically computes various click-through-rates (CTRs) over the entire population offline. A CTR of an ad is defined as the number of clicks on the ad divided by the number of times it is shown (impressions). These CTR values are used to estimate relevance of ads to various contexts. During ad-delivery time, the user's phone determines her context to be doing Yoga in Dolores Park in San Francisco. However, the user decides to share only the fact that she is exercising. Based on this limited information and CTR values, the ad server selects and returns two ads. The more relevant one is chosen by the phone based on user's private information (doing Yoga in Dolores Park in San Francisco) and displayed to the user.

2.2 Desiderata

Our desiderata include goals of the individual participants as well as general system requirements. Informally, we want the advertising system to provide good privacy, utility (revenue/relevance), and performance (communication efficiency/scalability/robustness). Since ad delivery and statistics gathering phases are different in nature, we have slightly different desiderata for them.

2.2.1 *Desiderata for Ad Delivery*

We have three design goals for ad delivery: Privacy, efficiency, and revenue/relevance (utility).

► **Privacy.** In order to protect privacy of sensitive contexts, the user would like to limit the amount of information about her mobile context that is sent to the server.

► **Efficiency.** The ad serving system should be efficient both in terms of communication and computational cost—the user wants ads fast and without draining much battery power on her mobile device, while the ad service provider wants to run his system at low operating cost. For simplicity, we focus on communication cost between the server and a client since it is the most dominant cost of serving ads to mobile devices. Our results can be extended to consider computational cost of the server and the client as well.

► **Revenue and Relevance.** The ad service provider seeks to maximize revenue. The user is only interested in relevant ads. The goal of the ad service provider is to display an ad from a given set of ads $\mathcal{A}$ that maximizes the expected revenue. For a click on ad a, the ad service provider is being paid p_a from the advertiser. Clearly, not all users click on an ad. We denote by $\mathsf{CTR}(a|c)$ the context-dependent click-through-rate, i.e., the fraction of users who actually clicked on it in context c among those who were served the ad in context c. The expected revenue of displaying an ad a to a user in context c is $p_a \cdot \mathsf{CTR}(a|c)$. We view clicks as an indicator for relevance: users who are interested in an ad click on it. Maximizing relevance means maximizing the expected number of clicks by displaying to a user in context c the ad a with the highest context-dependent $\mathsf{CTR}(a|c)$.

2.2.2 *Desiderata for Statistic Gathering*

We have following goals in the statistics gathering phase.

► **Privacy in the Absence of a Trusted Server.** We do not assume the availability of a trusted server to collect all user data to compute statistics. Without a trusted server, we need a distributed aggregation protocol that protects user privacy, even under adversarial scenarios such as when a fraction of the participants behave maliciously, send bogus messages, or collude with each other. This requirement sets our work apart from previous work on publishing privacy-preserving statistics that all assume a trusted third party (see [4] and the references therein).

► **Scalability.** We need to scale the computation to a large number of users and contexts.

► **Robustness to a Dynamic User Population.** With a large number of transient mobile phone users, not all of them are available and willing to engage in all rounds of our protocol. Users decide which queries they are willing to answer and when (e.g., when the phone is being charged and connected through a WiFi network). Therefore, our protocol should be able to deal with a dynamic user population without sacrificing privacy or scalability.

With the above desiderata in mind, rest of the paper answers the following questions mentioned in Introduction:

- What are the trade-offs in the design space of personalized ad delivery? How can we instantiate the above framework to any desired optimal trade-off point?

- How can we, in an efficient and privacy-preserving way, gather required statistics over a dynamic population?

3. PRIVACY-AWARE AD DELIVERY

In this section, we investigate a fundamental trade-off between our goals and show how to deliver ads with a desired trade-off.

3.1 The P-E-R Trade-offs

Our three design variables—Privacy, Efficiency, and Relevance—are conflicting. Without a trusted third party, *optimizing all three design goals simultaneously is impossible*. Consider the task of showing only one ad to the user. Then, in case of minimum information disclosure (i.e., the user does not send any information about her context) and highest communication efficiency (i.e., the server may only return a single ad), the server needs to choose the ad without any knowledge of user's context. Whatever the server does yields suboptimal relevance and revenue, as long as there is an ad whose CTR depends on the context. If we want to improve the relevance, either the user needs to send some information to the server, or the server needs to send more than one ad for the user to perform local personalization.

If we drop any of our three design goals the problem becomes easy. If there were no concerns about privacy, we could use a *server-only* scheme, where the user sends her context c to the ad service provider, who serves the ad that maximizes the expected revenue, i.e., $p_a \cdot \mathsf{CTR}(a|c)$. This is a very efficient scheme that maximizes revenue. If there were no efficiency concerns, we could use a *client-only* scheme, where the server simply sends all ads $\mathcal{A}$ so that the user can pick the ad that maximizes expected revenue.[1] It has been estimated that due to ad churn this requires sending 2GB of compressed ads per month [22]. Alternatively, we could use expensive cryptographic protocols for private information retrieval [16]. No user information is disclosed to the server and optimal revenue is achieved, but performance is bad. Finally, if there were no financial incentive and no interest in relevant ads, one could stop serving ads altogether to avoid any concerns regarding efficiency and privacy. In practice, one has to find reasonable trade-offs between the three design goals.

3.2 Optimizing Ad Delivery

In our framework, the user gets to decide what information about her context to share with the server. Based on this information the server selects some k ads $A \subset \mathcal{A}$ that are sent to the user. Here, the parameter k determines the communication cost. Computation cost can also be included in k if needed. The user then picks one ad from A to display. The set of ads and the ad to display should be chosen in a way that maximizes revenue.

Our flexible framework can be optimized for various objective functions over privacy, efficiency, and revenue. For concreteness, we now assume that there are constraints on both information disclosure (determined by users) and communication cost (determined based on current network load); we seek to maximize revenue under these constraints. We will discuss alternative objective functions in Sec. 3.3.2.

3.2.1 Client-Side Computation

For a given set of ads A chosen by the server, a client in context c maximizes the revenue by selecting the ad

$$a^* = \arg\max_{a \in A} p_a \cdot \mathsf{CTR}(a|c).$$

[1] *If we had a trusted third party*, it could collect private information from the client and all ads from the server, and send the best ad to the client. This would maximize all three variables, ignoring communication overhead between the server and the third party.

3.2.2 Server-Side Computation

The server needs to determine the best k ads to send to the user given only the partial information $\hat{c}$ it has. Suppose that the server has information not only on click-through-rates, but also on the frequency of each context. If this is all the information the server has, then from its point of view the expected revenue of sending a set A of ads to the user depends on the user's true context c; it is $\max_{a \in A} p_a \cdot \mathsf{CTR}(a|c)$. Since the server knows only the generalized context $\hat{c}$, it considers the probability of each of the possible contexts $c' \to \hat{c}$ and the expected revenue of A in this context c'. With this limited information the expected revenue of a set of ads A for a generalized context $\hat{c}$ is

$$\mathrm{E}[\mathsf{Revenue}(A|\hat{c})] = \sum_{c:c \to \hat{c}} \Pr[c|\hat{c}] \cdot \max_{a \in A} p_a \cdot \mathsf{CTR}(a|c).$$

It is the server's task to select the set A^* of k ads from $\mathcal{A}$ that maximizes the expected revenue, given only the generalized context $\hat{c}$ of the user, i.e.,

$$A^* = \arg \max_{A \subset \mathcal{A}: |A| = k} \mathrm{E}[\mathsf{Revenue}(A|\hat{c})]$$

Finding these k ads is NP hard as we will show in the next section. However, we can employ approximation techniques to efficiently select a set of k ads with revenue close to the optimal revenue.

3.2.3 Instantiations of the Framework

Our framework encompasses client-side personalization by setting $\hat{c}$ to the most generalized context that does not leak any information about the client's true context. In this case the personalization takes place exclusively on the client side. Our framework also encompasses server-side personalization by setting $k = 1$ in which case the client simply displays the ad sent by the server without further personalization. However, higher revenue can be achieved in our framework when the server sends back $k > 1$ results.

Additional Constraints. While high revenue and high relevance of ads are related goals, they are not the same. Suppose the ad service provider receives a request from a user in context c. Suppose further there are two ads a_1, a_2 with $\mathsf{CTR}(a_1|c) = 0.1, \mathsf{CTR}(a_2|c) = 0.9$ and $p_{a_1} = \$0.1, p_{a_2} = \$0.01$. Ad a_1 has the higher expected revenue but a_2 is more relevant. While displaying a_1 maximizes short-term revenue it might not be the best long-term strategy. Recent work has found that the time users spend viewing ads depends on the predictability of the quality of the ads [5]. Our framework can reconcile relevance and short-term and long-term revenue goals by adding a constraint on CTR.

3.3 Ad Selection Algorithms

We now explain how client and server can efficiently compute their parts of the optimization to jointly choose the best set of ads that achieve a desired trade-off. We consider a specific instantiation of the optimization problem where the user fixes her privacy requirement; the client and the server then try to maximize revenue for a given bounded communication complexity k. At the end of the section we discuss extensions and alternatives.

The client can quickly compute the equation in Sec. 3.2.1, since the number of ads from which the client picks one is small ($\leq k$).

However, the server's task—to select a set of k ads from $\mathcal{A}$ that maximize the expected revenue given only the generalized context $\hat{c}$ of the user—is much more demanding. In fact, a reduction from the maximum coverage problem shows:

PROPOSITION 3.1. *For a generalized context $\hat{c}$ it is NP-hard to*

Algorithm 1 Greedy algorithm for selecting ads maximizing the expected revenue.

Greedy(ads $\mathcal{A}$, generalized context $\hat{c}$, threshold k)
Init $A = \emptyset$
while $|A| < k$ **do**
  for $a \in \mathcal{A}$ **do**
    $b_a \leftarrow \mathrm{E}[\mathsf{Revenue}(A \cup \{a\}|\hat{c})] - \mathrm{E}[\mathsf{Revenue}(A|\hat{c})]$
  $A \leftarrow A \cup \{\mathrm{argmax}_a b_a\}$
return A.

select the revenue-maximizing set of k ads A^* such that:

$$A^* = \arg \max_{A \subset \mathcal{A}:|A|=k} \sum_{c:c \to \hat{c}} \Pr[c|\hat{c}] \cdot \max_{a \in A} p_a \cdot \mathsf{CTR}(a|c)$$

Moreover, the maximum coverage problem cannot be approximated within $\frac{e}{e-1} - o(1)$ assuming $P \neq NP$ [14].

3.3.1 Approximation Algorithm

Algorithm 1 shows a greedy algorithm, called Greedy, that constructs a set A of k ads incrementally. It starts with A empty and in each round, the ad that increases the expected revenue the most is added to A.

Interestingly, the output of this simple greedy algorithm approximates the optimal value to within a factor of $(1 - 1/e)$. Although the greedy algorithm is known to provide such a guarantee for the maximum coverage problem [25], our problem is considerably more complex: In the coverage problem a set either *fully* covers an element or not. In our case an ad a can *partially* "cover" a context c that can be generalized to $\hat{c}$. Thus a new analysis is required. We first define a *benefit function* of adding a set A' to a set A:

$$\mathrm{B}(A, A') = \mathrm{E}[\mathsf{Revenue}(A \cup A'|\hat{c})] - \mathrm{E}[\mathsf{Revenue}(A|\hat{c})].$$

The benefit function has the nice property (proved in page 99 of [18]):

FACT 3.2. *The benefit function is submodular, i.e., for all sets of ads $A_1 \subseteq A_2$ and for all A, $B(A_1, A) \geq B(A_2, A)$.*

However, due to the complex nature of our problem, the submodularity property alone does not imply our approximation guarantee.

Let $a_1, \ldots, a_k$ be the k ads chosen by Greedy in the order they were chosen. To simplify the analysis, we define the benefit of the i^{th} ad to be b_i and the expected revenue after adding the first l ads to be $b(l) = \sum_{i=1}^{l} b_i$. Similarly, let $a_1^*, \ldots, a_k^*$ be the k optimal ads in any fixed order. We define the benefit of the i^{th} ad to be b_i^* and the expected revenue after adding the first l ads to be $b^*(l) = \sum_{i=1}^{l} b_i^*$.

LEMMA 3.3. *$\forall l \in [k]$: $b_l \geq \frac{b^*(k) - b(l-1)}{k}$.*

PROOF. The benefit of adding $a_1^*, \ldots, a_k^*$ to $a_1, \ldots, a_{l-1}$ is at least $b^*(k) - b(l-1)$:

$$\mathrm{B}(\{a_1, \ldots, a_{l-1}\}, \{a_1^*, \ldots, a_k^*\}) \geq b^*(k) - b(l-1)$$

It is also equal to $\sum_{i=0}^{k} \mathrm{B}(\{a_1, \ldots, a_{l-1}\} \cup \{a_1^*, \ldots, a_{i-1}^*\}, \{a_i^*\})$. Thus, it follows from an averaging argument that $\exists i, 1 \leq i \leq k$: $\mathrm{B}(\{a_1, \ldots, a_{l-1}\} \cup \{a_1^*, \ldots, a_{i-1}^*\}, \{a_i^*\}) \geq \frac{b^*(k) - b(l-1)}{k}$. By submodularity this implies that $\exists i, 1 \leq i \leq k : \mathrm{B}(\{a_1, \ldots, a_{l-1}\}, \{a_i^*\}) \geq \frac{b^*(k) - b(l-1)}{k}$.

Since the greedy algorithm in round l selected the ad a_l that maximizes $\mathrm{B}(\{a_1, \ldots, a_{l-1}\}, \cdot)$, the benefit of that ad, b_l, has to be at least $\frac{b^*(k) - b(l-1)}{k}$ which completes the proof. □

We use this lemma to prove the following by induction.

LEMMA 3.4. *$\forall l \in [k]$: $b(l) \geq (1 - (1 - 1/k)^l) b^*(k)$.*

PROOF. Proof by induction on l.
$l = 1$**.** Lemma 3.3 tells us that $b_1 \geq \frac{b^*(k)}{k} = (1 - (1-1/k)^1) b^*(k)$.
$l \to l + 1$**.**

$$b(l+1) = b(l) + b_{l+1} \geq b(l) + \frac{b^*(k) - b(l)}{k}$$
$$= \frac{b^*(k)}{k} - b(l)(1 - 1/k) \geq \frac{b^*(k)}{k} - (1 - (1 - 1/k)^l) b^*(k)(1 - 1/k)$$
$$= (1 - (1 - 1/k)^{l+1}) b^*(k)$$

The first inequality follows from Lemma 3.3 and the second follows from the induction hypothesis. □

The main theorem on the approximation guarantee follows.

THEOREM 3.5. *The greedy algorithm approximates the optimal value to within a factor of $(1 - 1/e)$.*

PROOF. By Lemma 3.4 we have that

$$b(k) \geq (1 - (1 - 1/k)^k) b^*(k) \geq (1 - 1/e) b^*(k)$$

□

3.3.2 Extensions

Alternate Objective Function. So far, we tried to maximize revenue under hard constraints on both the amount of information disclosure and the communication cost k. Instead, one might consider the communication cost as a variable and include it in an objective function that maximizes the value of (revenue $-\alpha k$).

Consider an alternate objective function that maximizes the value of (revenue $-\alpha k$). A simple solution is to run Greedy for all values of k and pick the outcome that maximizes our new objective function. However, by exploiting the submodularity of the benefit function, we can maximize the new objective function more efficiently. All we have to do is to replace the **while** condition in Algorithm 1 by a new one that checks whether the current value of $\mathrm{E}[\mathsf{Revenue}(A)] - \alpha \cdot |A|$ is increasing.

This modification works correctly because the following argument shows that as we increase k, our new objective function increases until at some point it starts to decrease and never increases again. Suppose in round k' the expected revenue of $A = \{a_1, \ldots, a_{k'}\}$ minus $\alpha \cdot k'$ is not increasing any longer, i.e.,

$$\mathsf{Revenue}(\{a_1, \ldots, a_{k'}\}) - \alpha k'$$
$$\leq \mathsf{Revenue}(\{a_1, \ldots, a_{k'-1}\}) - \alpha(k' - 1).$$

At this point the benefit of adding $a_{k'}$ is at most α. Due to submodularity, the benefit of any future ad being added to A can only be smaller and thus will never lead to an increase of the objective function.

Additional Constraints. We can incorporate a constraint on ad relevance by setting the CTR to zero whenever it is below a certain threshold. Then, no ad with CTR below this threshold will ever be displayed at the client.

Advertisers' Control. Our algorithm can incorporate additional restrictions posed by advertisers on the contexts in which their ads are being displayed. Very much like advertisers for sponsored results in Web search can bid on keywords in a query, our advertisers can bid on contexts of users. To make sure the ad is only displayed on these contexts, we can make the payment p_a context-dependent and set it to 0 for all but the contexts the advertiser bids on.

4. PRIVATE STATISTICS GATHERING

The optimization framework described in previous section uses various statistics; in this section we describe how to obtain those in a private way with the desiderata mentioned in Section 2.2.2. The main mechanism we employ to build a scalable and robust protocol is to use a server and a proxy: The server is responsible for key distribution and the computation of the final result while the proxy is responsible for aggregation and anonymization. For example, the ad network server can employ Verisign as the proxy. The idea of using two servers to build secure protocols has been used previously [2, 16, 22] in different applications; we use it here for privacy-preserving aggregation.

In our setting, each user keeps a history of what ads she has viewed/clicked that, for privacy reason, is stored on user's local device. The server then, with the help of the proxy, uses our protocol to compute statistics necessary for ad delivery: the probability distribution over contexts, $\Pr[c]$, and the context-dependent click-through rates, $\mathsf{CTR}(a|c)$. Both can be estimated by counting how many users were in a specific context c and viewed / clicked on a specific ad a. Hence we start with privacy-preserving computation of count queries.

4.1 Assumptions and Privacy Preliminaries

We assume secure, reliable, and authenticated communication channels between servers and users. In addition, we make the following two key assumptions, similar to those made in previous works [8, 37, 40].

1. Honest-but-Curious Servers. *Server and proxy honestly follow the protocol. They are curious but do not collude with anyone.*[2]

2. Honest Fraction of Users. *At most a t fraction of users are malicious or unavailable during the protocol. This means, at least a fraction of $1-t$ users honestly follow the protocol. The honest users can be curious but they do not collude with anyone.*

We aim to ensure user privacy with respect to all participants in the distributed protocol. There are many different ways to define privacy in data publishing. We refer the reader to an excellent survey [7]. For the purpose of this paper, we work with ϵ-differential privacy [13]. The idea behind this strong guarantee is that whether or not the contexts and clicks of a single user were used in the computation hardly affects the released outcome. Therefore, a user, given the choice of whether or not to supply her data has hardly any incentive to withhold it. The parameter ϵ determines how much the outcome may be affected.

In the absence of a trusted server, we need to generate noise required to ensure differential privacy in a distributed manner. In this paper we adopt the probabilistic relaxation (ϵ, δ)-differential privacy [31], for which noise can be generated in a distributed way. The parameter δ bounds the probability that a privacy breach (according to ϵ-differential privacy) occurs. For $\delta = 0$ this definition is equivalent to ϵ-differential privacy. (ϵ, δ)-differential privacy of a count query can be realized by adding Gaussian noise and Gaussian noise with variance σ^2 can be generated in a distributed manner by N parties, by summing up N independent random samples from the Gaussian distribution with variance σ^2/N. More recently, Ács et al. [1] have shown how to generate Laplace noise in fully distributed way, by constructing Laplace distribution as the sum over i.i.d. samples from the Gamma distribution. Our protocol can easily adopt this technique and ensure ϵ-differential privacy as well.

Notation. Consider a user activity log L containing the data of a set of users U. We can restrict the log L to the data of a subset of the users $U' \subset U$, denoted by $L_{U'}$. If U' contains users not in U, we define $L_{U'}$ to be $L_{U \cap U'}$. Consider a distributed protocol M involving a set of participants P. Note that the set of users and the set of participants can be overlapping. We define the *view* of a subset of participants $P' \subset P$ in the execution of M on input L, denoted by $V_{P'}$, to be a random variable for all messages received and sent by a participant in P'. For a non-participant we define the view to be the output of the distributed protocol. The set of participants can be partitioned into two sets: P_m of malicious participants and P_h of honest but possibly unavailable participants. We have that $P = P_h \cup P_m$ and $P_h \cap P_m = \emptyset$.

DEFINITION 1. *A distributed protocol M with participants P satisfies (ϵ, δ)-distributed probabilistic differential privacy of the users if for all user activity logs L and all (non-) participants p the following holds. In case p is malicious let P'_m denote the set of malicious participant colluding with p which are a subset of the malicious participants P_m. Otherwise let P'_m be $\{p\}$. There exist randomized algorithms M' and R so that (a) $M'(L_{U \setminus (P_m \cup \{p\})})$ preserves (ϵ, δ)-probabilistic differential privacy and (b) R generates the distribution of the view $V_{P'_m}$ given $M'(L_{U \setminus (P_m \cup \{p\})})$ and $L_{P_m \cup \{p\}}$, i.e., $V_{P'_m}$ and $R(M'(L_{U \setminus (P_m \cup \{p\})}), L_{P_m \cup \{p\}})$ are identically distributed.*

The definition considers the view of a (non-)participant p. This view contains messages sent and received by any participant colluding with p, denoted by P'_m. In case p is not malicious P'_m is $\{p\}$. Privacy means that this view can be simulated from an output that preserves probabilistic differential privacy of the users who are not malicious. This output is generated by some algorithm M' from the users' input that are neither malicious nor equal to p ($L_{U \setminus (P_m \cup \{p\})}$). We do not attempt to protect the privacy of malicious participants. This is indeed impossible, since the adversary controlling them could always send a message containing L_{P_m} which breaches their privacy. Moreover, we do not protect p's privacy against herself. The simulation takes this output as well as the input from all malicious participants to produce the same view. The second input is necessary for a simulation to be possible at all.

A central building block of our protocol is a procedure for computing a sum over user values. We will use it to compute how many users clicked on an ad a in context c.

4.2 A Privacy-Preserving, Distributed Count

4.2.1 Our counting protocol

Protocol 1 describes our protocol $\mathsf{Count}(t, \sigma^2)$. Each user u_i for $i = 1, \ldots, N$ holds a bit b_i. The protocol computes a noisy version of the sum $\sum b_i$. The parameter p is a sufficiently large prime number, t is an upper bound on the fraction of malicious or unavailable users, and σ^2 is the amount of noise. If more than t fraction of users turn out to be malicious or unavailable, the privacy guarantee degrades and/or the protocol needs to be aborted before Step 4 and restarted (with a larger value for t). As t increases, the share of noise each participant adds to his or her bit increases.

Efficiency. The number of messages exchanged in Count is linear in the number of users, as in the most efficient previous solutions [1, 37, 40]. Messages across Count computations can be batched.

Robustness. Unlike previous protocols [37, 40], Count successfully terminates as long as at least $(1-t)N$ users send messages to server and proxy. Unlike these previous protocols, our protocol does not expect the secrets of participating users to add up to a predefined constant. Rather, it lets users independently choose their own secrets k_i (Step 1) and uses secrets of only the users who have participated throughout the entire protocol. Unlike [1], our

[2]The assumption may be relaxed by using trusted hardware [8].

Protocol 1 Robust, distributed count computing a privacy-preserving version of the sum over all private user bits b_i.

$\mathsf{Count}(\sigma^2, t)$

1. Each user i with bit b_i samples $k_i \in \{0, \ldots, p-1\}$ i.i.d.
2. Each user i samples r_i from $\mathcal{N}(\sigma^2/((1-t)N-1))$.
3. Each user i uses 2-Phase-Commit to atomically send k_i to the server and $m_i = b_i + \lfloor r_i \rfloor + k_i \mod p$ to the proxy.
4. The proxy sums up all incoming messages m_i. It forwards $s = \sum m_i \mod p$ to the server.
5. The server subtracts from s the random numbers k_i ($\mod p$) it received and releases the result $\sum b_i + r_i$.

protocol can tolerate failures of users *during* its execution. When Count is executed multiple times, it suffices that for each execution, at least $(1-t)N$ possibly different users participate. Thus, our protocol can deal with unavailable users much more efficiently than previous protocols.

4.3 Privacy

Following Definition 1, the protocol preserves privacy.[3]

THEOREM 4.1. *For users with real values $b_i^{(1)}, \ldots, b_i^{(d)}$ Protocol $\mathsf{Count}(\sigma^2, t)$ can be used repeatedly to compute $\sum b_i^{(1)}, \ldots, \sum b_i^{(d)}$ with noise added to protect privacy. Let s denote the L_2-sensitivity of $\sum b_i^{(1)}, \ldots, \sum b_i^{(d)}$. Consider $\epsilon \leq 1$ and $\sigma^2 \geq 2s^2 \ln(4/\delta)/\epsilon^2$. The protocol guarantees (ϵ, δ)-probabilistic differential privacy in the presence of up to a fraction of t unavailable or malicious users.*

The proof relies on Gaussian noise to protect the privacy. In the case of a trusted server, it is well know that adding Gaussian noise protects privacy. In particular, we can sanitize the output of any real-valued function f : ad logs $\to \mathbb{R}^d$ by adding Gaussian noise to $f(L)$. The standard deviation of the noise depends on the L_2-*sensitivity* of f which describes how much the value of the function can change if a single user's data is deleted from the input. This change is measured by the L_2-norm.

PROPOSITION 4.2. *For $\epsilon \leq 1$ and $\sigma^2 \geq s^2 2\ln(4/\delta)/\epsilon^2$, adding Gaussian noise with variance σ^2 to a function f with L_2-sensitivity s gives (ϵ, δ)-probabilistic differential privacy.*

This theorem has been established for δ-approximate ϵ-indistinguishability [12] and extends to our definition, which is stronger [19]. Now, when we consider all the participants of the protocol we use the addition of Gaussian noise as M' in Definition 1 and show that their view can be generated from it. Details can be found in [18].

Additional Guarantees. Our protocol also provides some guarantees in case either server or proxy (but not both) are corrupted by an adversary (but not colluding with any user or the other server). If the proxy is corrupted by an adversary, we guarantee that the adversary will not be able to learn information that breaches privacy. This guarantee holds since the proxy sends only the very last message of the protocol upon which no further action is taken. Similarly, if the server is corrupted we guarantee that the adversary will not be able to learn information that breaches privacy. The adversary may send any value to the proxy from which $\sum k_i$ will be subtracted. The output will be random in case the value is not a sum that contains exactly one term for each received message m_i. In this case privacy is trivially preserved. Otherwise the value contains sufficient noise to preserve privacy. These guarantees are meaningful with regard to adversaries who want to learn private information. We do not guarantee that a malicious adversary cannot breach privacy: an adversary corrupting the proxy could release the keys, which would allow the honest-but-curious server to learn $b_i + r_i$, which would breach privacy.

[3]Our protocol can also guarantee ϵ-differential privacy if Laplace noise is generated in a distributed way, following techniques described in [1].

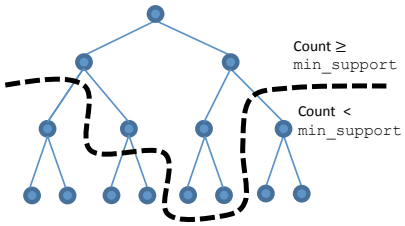


Figure 2: Hierarchy H over contexts.

4.4 Employing Count to compute CTRs

Given Count, the above statistics can be estimated well for contexts with a lot of user data (e.g., clicks). However, for a large number of contexts sufficient click data may not be available. For such rare or new contexts, the estimates can be noisy or not even be defined. We suggest estimating $\Pr[c]$ and $\mathsf{CTR}(a|c)$ for a rare context c based on contexts *similar* to c for which we have enough click data. Coming back to our example from Section 2, if we do not have enough click data for users who were skating in Central Park, we might use clicks from users in close-by locations who were doing some sort of physical activity ($\tilde{c}$) to estimate the statistics for the context c. This helps us increase coverage of targeted ads, albeit at the possible cost of lower quality ads. To define similarity over contexts, we reuse the hierarchies over which users generalize their context attributes. The context hierarchy is built by merging various attribute hierarchies; a concrete example will be shown in Section 5.1. This hierarchy tells us, for each generalized context, which attribute to generalize next.[4] Given some rare context, we can generalize it until we have a sufficient number of clicks for the generalized context. With these clicks we estimate the CTRs. The parameter `min_support` specifies how many clicks are sufficient for robust estimates. Figure 2 shows a hierarchy H over contexts with leaf nodes being exact contexts and intermediate nodes being generalized contexts. It shows a cut-off through the hierarchy so that all (generalized) contexts above the cut-off have at least `min_support` many clicks in the training data for descendant contexts. The contexts below the threshold need to be generalized before estimating their CTRs.[5]

Possible Solutions. One possible way to employ Count to obtain privacy-preserving statistics for various contexts is to compute noisy counts for all possible contexts and all possible ads. Another alternative approach, with better utility, would be to use multi-dimensional histograms [43, 23]. However, all these approaches have a running time at least linear in the number of possible contexts, rendering them infeasible. Moreover, a large part of the computation is wasteful, since, as mentioned before, statistics computed for rare contexts are almost meaningless.

To address this problem, we opt for a simple top-down approach

[4]It is recommend but not required that users generalize the contexts they send to the server to a node in the hierarchy.

[5]Note that there are other ways to define similarity, for example using the lattice structure imposed by the attributes' hierarchies. Our experimental results show only a minor effect on quality when using a fixed combined hierarchy as opposed to a lattice structure.

Algorithm 2 Privacy-preserving Estimates.

Estimates(context-driven ad log, noise scale λ, threshold min_support, contribution bound m, hierarchy H)
for each user **do**
  Delete all but m views or clicks on ads and their contexts of this user from the ad log.
return TopDown(ad log, root(H), λ, min_support)

Algorithm 3 Top-Down computation of noisy statistics.

TopDown(context-driven ad log, node v in the hierarchy, noise scale λ, threshold min_support)
$\mathcal{A}' =$ set of ads with bids on context of v
for $a \in \mathcal{A}'$ **do**
  $\text{clicks}_{a,v} =$ Count (# of clicks on a in v in ad log)
  $\text{no\_clicks}_{a,v} =$ Count (# of views of a w/o clicks in v)
  release $\widehat{\text{CTR}}(a|v) = \frac{\text{clicks}_{a,v}}{\text{clicks}_{a,v}+\text{no\_clicks}_{a,v}}$
$\text{count}_v =$ Count (# of appearances of node v appears)
release count_v
if $\text{count}_v > \text{min\_support}$ **then**
  for each child w of v **do**
    return TopDown(ad log, w , λ, min_support)

that can efficiently deal with sparse data by identifying and pruning the computations for rare contexts. The solution requires using a context hierarchy that specifies similarity of contexts. Such a top-down algorithm has been used recently to find frequent signatures by gradually expanding the prefix of signatures with high noisy counts [33]. We adapt it to compute CTRs over a context hierarchy.

A Top-Down Algorithm. To compute privacy-preserving CTRs for the generalized contexts in the hierarchy H, algorithm TopDown starts at the root and moves down the hierarchy. For each traversed node v and for each ad a, it estimates $\text{CTR}(a|v)$ by calling Count to compute how often users in a descendant context of v have clicked (or only viewed) a. The results of this computation is referred to as $\text{clicks}_{a,v}$ ($\text{no\_clicks}_{a,v}$, resp.). The estimated click-through-rate is then simply $\widehat{\text{CTR}}(a|v) = \frac{\text{clicks}_a}{\text{clicks}_{a,v}+\text{no\_clicks}_{a,v}}$. TopDown also computes the total number of times a descendant of v appears in the ad log and adds noise to this count. If the count is above min_support then the algorithm recurses on v's children, otherwise all descendants are pruned. We note that the accuracy of Estimates can be further improved by using the post-processing techniques of Hay et al. to make sure the counts of all children add up to the parent's count [24]. To bound the sensitivity and to guarantee differential privacy, we limit the number of entries per user in the ad log. Estimates deletes from the ad log all but m random entries per user and then calls TopDown.

We now analyze the privacy and efficiency of our algorithm. Let a denote the maximum number of ads bidding on the same context. We denote by height(H) the height of the hierarchy and by branch(H) the maximum number of children for a node in H. Estimates makes $O(a+\text{branch}(H))\cdot\text{height}(H)\cdot N\cdot m/\text{min\_support})$ calls to Count with high probability and is thus independent of the number of contexts. Moreover, when we use Count in Estimates we can batch all the messages in one level in the hierarchy.

In Estimates we employ Count to obtain noisy estimates of $\text{clicks}_{a,v}$, $\text{no\_clicks}_{a,v}$, and count_v.

COROLLARY 4.3 (PRIVACY). *Consider any $\epsilon \leq 1$. Let σ^2 be at least $6\cdot height(H)\cdot m^2\cdot\log(4/\delta)/\epsilon^2$. When* Estimates *employs* Count(t,σ^2) *as a subroutine for counting* $\text{clicks}_{a,v}$, $\text{no\_clicks}_{a,v}$, count_v, *it guarantees (ϵ,δ)-probabilistic differential privacy. A fraction of t unavailable or malicious users during each call of* Count(t,σ^2) *can be tolerated.*

A proof and a utility analysis can be found in Appendices A.1, A.2.

This completes our discussion of the statistics gathering in our framework. Next, we illustrate with an example, how it fits together with the online component of ad targeting described in Section 3.

EXAMPLE 4.4. *The server has a list of all users, all contexts c, a hierarchy of these contexts, and all ads a. It periodically estimates the values of $\Pr[c]$ and $\widehat{\text{CTR}}(a|c)$, for all c and a, by executing the* Estimates *algorithm. While doing so, it walks top-down through the context hierarchy. Whenever* Count *is invoked, it asks all users to submit their counts (either views or clicks of a in v or simply appearances of v) to the proxy. The proxy does not need to know (but may know) what is being counted. Note that* Estimates *might not be able to compute $\widehat{\text{CTR}}(a|c)$, for all $c \to \hat{c}$. Some contexts, like skating in Central Park, might be to sparse in which case they we will be substituted by the lowest ancestor for which an estimate is available (such as exercising in Central Park).*

*Now suppose the ad server wants to deliver ads to a user who is currently running in Central Park of New York. Due to privacy concerns, the user discloses to the server only that she is exercising in New York City $\hat{c}$. The server then uses Algorithm 1 (*Greedy*) to decide which ads to send to this user. For this, it uses precomputed values of $\Pr[c]$ and $\widehat{\text{CTR}}(a|c)$, for all $c \to \hat{c}$ and all ads a that are displayed in such contexts. Finally the client device selects the best ad based on the user's actual context (running in Central Park).*

5. EXPERIMENTS

We now empirically answer the following important questions with a real trace: (1) How strong are the trade-offs between privacy, efficiency, and utility in practice? (i.e., is it possible to achieve reasonable levels all three design goals simultaneously?) (2) How does our client-server joint optimization compare with client-only or server-only personalization? and (3) How robust is our statistics gathering algorithm with dynamic population? Before answering the questions, we first describe our experimental setup.

5.1 Experimental Setup

Dataset. Ideally, we would like to evaluate our algorithms with real usage traces from a context-aware ad service. However, since no such real systems exist, we emulate such a system by using a trace of location-aware searches in Microsoft Bing for mobile.[6] The trace has a schema: ⟨*user-ID, query, user-location, business-ID*⟩. Each record in the trace describes an event of a user issuing a query from a location and then clicking on a business. The trace consists of 1,519,307 records. In our evaluation we focus on clicks to "Food & Dining" businesses, which constitute the largest category of business in the trace. We also filter out any user with fewer than three clicks in the trace, as we cannot generate an interest profile for such a user. This leaves us with 116,432 unique user-IDs. We use the first 90% of the trace as training data and the remainder to evaluate our framework and to compute targeted ads (i.e., businesses).

Context. We use the above trace to emulate a context-aware ad service as follows. We assume that each business with id i has an ad with the same id i, and hence our goal is to deliver target business-IDs to the users. Ideally, we would like to use context based on the sensor readings of smart phones for personalization. However, this

[6] http://m.bing.com

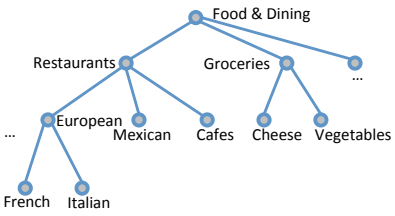


Figure 3: Hierarchy over businesses.

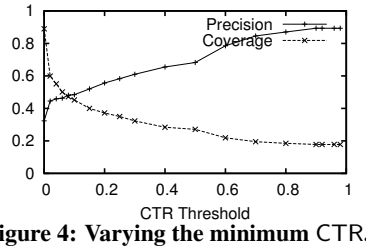


Figure 4: Varying the minimum CTR.

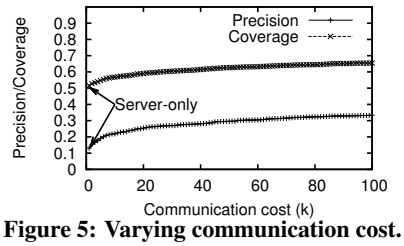


Figure 5: Varying communication cost.

information is not present in our trace and we therefore limit our evaluation to contexts consisting of the following set of attributes.

► Location: The user's location as latitude and longitude.

► Interest: A multi-set of business-IDs the user clicked on before.

► Query: The search query the user sends.

Attribute Generalization. To limit information disclosure, we let users generalize context attributes according to fixed hierarchies.

► Location: We use five levels of generalization for user location, depending on how many decimal points we truncate from her latitude and longitude. More specifically, `Level-i` location, $0 \leq i \leq 5$ of a user is her latitude and longitude, after keeping all, 4, 3, 2, 1, and 0 decimal points respectively.

► Interest: We generalize user interest using a fixed hierarchy for the businesses, as shown in Figure 3. In `Level-0`, `Level-1`, and `Level-2`, the interest set contains business categories, generalized business categories, and only the most general business category ("Food and Dining"), respectively, of the user's clicks.

► Query: Again, we use the business hierarchy to generalize the query at three levels. `Level-0` is the exact query issued by the user, `Level-1` is the business category of the clicked business, and `Level-2` is the generalized category of the business.

For all attributes, `Level-i` is more general, and hence more privacy-preserving, than `Level-j` for $i > j$. As a short-hand, we use (x, y, z) to denote (`Level-`x location, `Level-`y interest, `Level-`z query).

Context Hierarchy. We combine the attribute hierarchies into a context hierarchy. We generalize one attribute at a time using the following sequence: $(0,0,0) \rightarrow (0,0,1) \rightarrow (0,1,1) \rightarrow (1,1,1) \rightarrow (1,2,1) \rightarrow (2,2,1) \rightarrow (3,2,1) \rightarrow (3,2,2) \rightarrow (4,2,2)$. As an example, consider the context at level $(0,0,0)$

⟨(61.22913, -149.912044), [B-ID2011, B-ID124], "Starbucks"⟩.

Generalizing each attribute one level yields, at level $(1,1,1)$,

⟨(61.2291, -149.9120), [Peruvian Restaurants, Wine], "Coffee"⟩.

Generalization does not just provide privacy, but also helps personalization with sparse data. For example, in our dataset there are $\approx$ 100,000 queries that appear only once. It is impossible to personalize search results for these queries because we have not seen the same query before. With generalization, we reduce the number of such queries by an order of magnitude. We address the sparsity of other context attributes similarly. This increases the coverage.

Metrics. We use the following two metrics for our prediction.

► **Precision:** The fraction of targeted ads in our framework on which users actually click. Precision is an indicator of relevance.

► **Coverage:** The fraction of contexts for which our framework computes and displays a targeted business.

The higher the precision and coverage values, the better the performance of our framework. We report average precision and coverage for 1,000 random contexts from the testing data; the averages become fairly stable after 1,000 predictions.

Parameters. Unless otherwise stated, we use the following default configuration. For limited information disclosure, we use $(4,2,2)$ generalization. We set the upper bound on communication complexity, k, to be 10, the threshold on click-through rate to be 0.3, and the threshold on support to be 2.

5.2 Evaluating Trade-offs

Effect of CTR Threshold. The CTR threshold trades off precision and coverage, see Figure 4. For a high value of the CTR threshold, an ad will be shown only if it is highly relevant. Thus, this increases the precision of our algorithm and improves the relevance of the displayed ads. On the other hand, a high threshold reduces the number of ads displayed and with that the number of clicks and the revenue. Interestingly, as we can see, *high levels of both precision (0.48) and coverage (0.47) can be achieved simultaneously*.[7]

Effect of Communication Complexity. Figure 5 shows the effect of increasing the communication complexity k (i.e. having the server return more ads to the client) on precision and coverage. We expect both to improve with increasing k since the client can choose an ad from a larger set. The graph shows further that *increasing k has diminishing returns*. In the beginning the precision and coverage increase quickly with every additional ad being sent, however, as more ads are sent, the increase in precision and coverage becomes smaller.

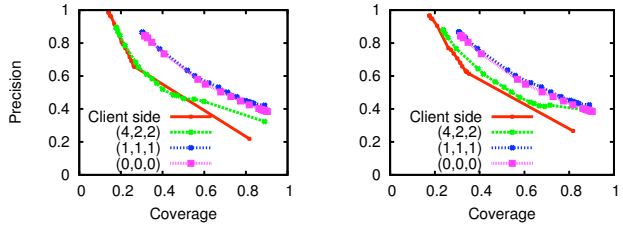
Effect of Information Disclosure. Figure 6 shows the precision and coverage (for various CTR thresholds) of our framework with various levels of generalization. As expected, precision and coverage of our framework increases as more specific context information is sent to the server. Interestingly, *limited privacy does not hurt utility in a significant way*; as shown in the graph, precision and coverage values are very close for limited privacy (shown as $(1,1,1)$) and no privacy (shown as $(0,0,0)$).

Trading-off Constraints. To see how communication overhead affects the performance of our framework, we increase k from 10 to 50 in Figures 6(a) and (b). The graphs show that *privacy can be improved without hurting utility by a small increase in the communication cost*. For example, when $k = 10$, a privacy level of $(4,2,2)$ does not achieve a precision of at least 0.85 and a coverage of at least 0.3. But it does, when increasing k to 50. Overall, we conclude that reasonable levels of limited information disclosure, efficiency, and relevance can be achieved simultaneously.

5.3 Comparison with Other Strategies

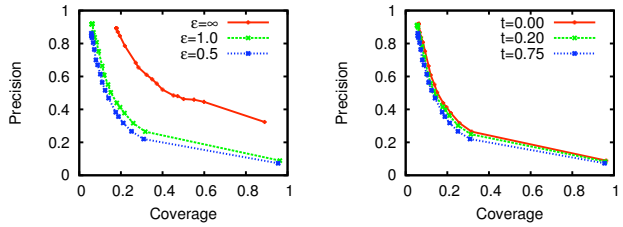
Server-only Personalization. Here, the server performs personalization based on the limited private information it has and sends only one ad to the client. As shown in Figure 5, this strategy gives a precision of 0.12. We can do much better with our optimization: When instead sending 5 ads and letting the client pick the most relevant one, the precision rises by 35%.

[7]Precisions and coverages close to 0.5 are considered high in predicting user clicks. Our numbers are higher than the ones reported for other personalization techniques [46].



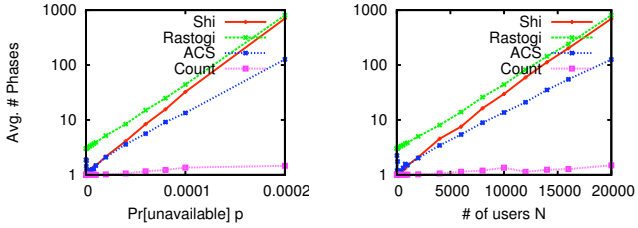
(a) $k = 10$ (b) $k = 50$

Figure 6: Varying information disclosure.



(a) Effect of noise ϵ (b) Effect of robustness t

Figure 7: Differentially-private estimates.



(a) $N = 10{,}000$ (b) $p = 0.0001$

Figure 8: Varying user population.

Client-only Personalization. Here, the client sends only the query to the server, which then sends k ads matching the query to the client. The client chooses the best ad based on the exact user context. Precision and coverage of this strategy are also shown in Figure 6 with the label "Client-side". As shown, our optimization can provide better utility than the client-only strategy. For example, for a target precision of 0.75, the client-side strategy can achieve coverage of 0.2, while our framework with $(1, 1, 1)$ generalization can achieve a coverage of 0.4, an increase of $2\times$.

5.4 Privacy-Preserving CTRs

In the following experiments, we fix the maximum number of contributions per user, $m = 4$. Moreover, we found it beneficial to limit how far TopDown goes down in the hierarchy. Such a limit reduces the amount of noise added to each count. This is important for training data as small as ours. Therefore, we chose an aggressive limit of 1.

Efficiency. When we run Estimates on our trace, a user has to send roughly 1MB on average. Many of the count queries can be batched. On average, a user participates in two batches for all CTR computations. We feel this communication cost is acceptable.

Accuracy. Figure 7(a) shows how precision and coverage of our framework degrades when we increase the differential privacy guarantee (by decreasing ϵ). We fixed $\delta = 0.01$. As a point of comparison, the figure also draws the precision and coverage curve when using the exact, non-private statistics ($\epsilon = \infty$). We can see that to achieve a precision of 0.6, the coverage of our framework using non-private statistics is much higher than the coverage of our framework using the ϵ-differentially private statistics (i.e., 0.3 vs 0.1). This is the price we have to pay for a privacy guarantee. The exact value of the privacy parameter ϵ (1 vs. 0.5) has a minor effect on the utility. We expect the cost of privacy to decrease with a larger user population. Moreover, we can avoid such negative impact on utility by paying the price of privacy in terms of communication overhead k—as shown in Figure 6, the utility can be improved by using a higher value of k.

5.5 Robustness of Statistics Gathering

Figure 7(b) shows the effect of varying t (fraction of malicious or unavailable users) on precision and coverage for $\epsilon = 1.0$. We see that the parameter t has only a mild effect. Even when 75% of the users could be unavailable or malicious ($t = 0.75$) the utility is almost the same as when all users are available and honest.

We compare our Count protocol with three existing distributed, differentially-private count protocols: RASTOGI [37], SHI [40], and ÁCS [1]. Briefly, RASTOGI [37] and SHI [40] start with a setup phase in which an honest server generates secrets and distributes them to users such that the secrets of all users add up to a constant. After this setup phase, a series of count queries can be computed in a privacy-preserving manner assuming that all available users in the setup phase participate in the aggregation phase. However, when a single user becomes unavailable, no further queries can be answered until a new setup is performed or the user returns. Thus, for a query to be successfully answered, the setup phase followed by the aggregation phase must be repeated until they both run on the same stable set of users. Recently, Ács et al. [1] proposed an efficient protocol that can tolerate failures of up to a predefined number of users *before* running the aggregation phase; however, the phase must be repeated if any user fails *during* its execution.

We compare the robustness by modeling users' unavailability as a simple random process. Suppose a *phase* denotes the time it takes for the server to send a message to all users or for all users to send messages to the server. Let p denote the probability that a user is unavailable to participate in a given phase. We measure the average number of phases required to complete a query, as it indicates the latency and communication complexity of a protocol.

It should be pointed out, though, that this compares only one aspect of these protocols. They also widely differ in the assumptions they make. Our protocol requires two honest-but-curious servers. RASTOGI and SHI require an honest server for the key set-up, while ÁCS provides privacy without any such assumptions.

Figure 8 illustrates the effects of unavailability on communication complexity. We run 1000 queries and report the average number of phases per query for different protocols. As shown, all the protocols cost close to their optimal number of phases when the probability of being unavailable (p) and the number of users (N) are small. However, *unlike our protocol, the costs for all three protocols increase exponentially with N and p* (note the log scale of the graphs). For $p \geq 0.0001$ (corresponding to less than only 10 seconds a day) in (a) or $N \geq 1000$ (much fewer than users of popular online services) in (b) the protocols become impractical. This shows that unlike our protocol, the three protocols are impractical for online services with dynamic users.

6. RELATED WORK

Targeted Advertising. Closest to our privacy-aware ad serving framework are the works of [15, 22, 27]. Repriv [15] verifies that applications only access the limited information about a user that was granted and proposes techniques for client only personalization. Privad [22] and the work by Juels [27] anonymize user profiles. Neither work explains how ads should be chosen based on limited user information by the ad server and based on more private information on the client. Thus, there is a potential benefit of

integrating our framework into these systems to trade off privacy, efficiency and utility.

Location-Based Services. Several location-based services (LBS) protect anonymity of users and privacy of their locations [26]. A user's location in the query is replaced by a broader region [20] with at least k users in it [38]. This approach is extended to general contexts beyond location [36]. Following the principles of ℓ-diversity [32] a region is broad enough if it's area is large enough and its ratio of sensitive area to total area is low enough (see for instance [9]). Most solutions (e.g., [35]) follow a hybrid approach in which, given a generalized context, the server returns a superset of the results [26], which can lead to high communication cost. Notable exceptions approximate the results and allow to trade off efficiency and accuracy [10, 28, 47]. We follow the same goals of LBS: privacy, utility and efficiency. While LBS focus on nearest neighbor queries measuring utility as proximity, our work focuses on target advertisements measuring utility as revenue or ad relevance. Previous techniques cannot be applied to this problem.

Privacy-Preserving Distributed Count Protocols. Previous work on distributed counting protocols [1, 12, 37, 40] provides strong privacy guarantees. Early work by Dwork et al. [12] required the exchange of a number of messages quadratic in the number of users. This is prohibitively expensive in our setting. Follow-up work by Rastogi et al. [37], Shi et al. [40], and Ács et al. [1] reduced the number of messages to be linear in the number of users. In Section 5.5, we empirically showed that even under modest assumptions on user dynamics, all these existing protocols need to repeat various phases impractically high numbers of times. This is problematic in our setting with a large number of transient mobile users.

Follow-up work by Chen et al. [8] extended our Count protocol in order to guarantee accuracy and prevent malicious users from arbitrarily altering the result. This is achieved by using the Goldwasser-Micali bit encryption scheme [17] and adding noise at the proxy.

7. CONCLUSION

We have addressed the problem of personalizing ad delivery to a smart phone, without violating user privacy. We showed that the problem of selecting the most relevant ads under constraints on privacy and efficiency is NP-hard and proposed a solution with a tight approximation guarantee. We also proposed the first differentially-private distributed protocol to compute various statistics required for our framework even in the presence of a dynamic and malicious set of participants. Our experiments on real click logs showed that reasonable levels of privacy, efficiency, and ad relevance can be achieved simultaneously.

8. REFERENCES

[1] Gergely Ács and Claude Castelluccia. I have a dream!: differentially private smart metering. In *Proceedings of the 13th international conference on information hiding (IH)*, 2011.

[2] Gagan Aggarwal, Mayank Bawa, Prasanna Ganesan, Hector Garcia-Molina, Krishnaram Kenthapadi, Rajeev Motwani, Utkarsh Srivastava, Dilys Thomas, and Ying Xu. Two can keep a secret: A distributed architecture for secure database services. In *CIDR*, 2005.

[3] Bhuvan Bamba, Ling Liu, Peter Pesti, and Ting Wang. Supporting anonymous location queries in mobile environments with privacygrid. In *WWW*, 2008.

[4] Thorben Burghardt, Klemens Böhm, Achim Guttmann, and Chris Clifton. Anonymous search histories featuring personalized advertisement - balancing privacy with economic interests. *Transactions on Data Privacy*, 4(1):31–50, 2011.

[5] Georg Buscher, Susan T. Dumais, and Edward Cutrell. The good, the bad, and the random: an eye-tracking study of ad quality in web search. In *SIGIR*, 2010.

[6] A. Chen. Gcreep: Google engineer stalked teens, spied on chats. http://gawker.com/5637234, September 2010.

[7] Bee-Chung Chen, Daniel Kifer, Kristen LeFevre, and Ashwin Machanavajjhala. Privacy-preserving data publishing. *Foundations and Trends in Databases*, 2(1-2):1–167, 2009.

[8] Ruichuan Chen, Alexey Reznichenko, Paul Francis, and Johannes Gehrke. Towards statistical queries over distributed private user data. In *NSDI*, 2012.

[9] Reynold Cheng, Yu Zhang, Elisa Bertino, and Sunil Prabhakar. Preserving user location privacy in mobile data management infrastructures. In *Privacy Enhancing Technologies*, 2006.

[10] Chi-Yin Chow, Mohamed F. Mokbel, Joe Naps, and Suman Nath. Approximate evaluation of range nearest neighbor queries with quality guarantee. In *SSTD*, 2009.

[11] Direct Marketing News. Mobile marketing to 'explode' in 2012. `http://www.dmnews.com/mobile-marketing-to-explode-in-2012/article/222991/`, January 2012.

[12] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, volume 4004, 2006.

[13] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.

[14] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45:314–318, 1998.

[15] Matthew Fredrikson and Benjamin Livshits. REPRIV: Re-imagining content personalization and in-browser privacy. In *IEEE Symposium on Security and Privacy (Oakland)*, 2011.

[16] William Gasarch. A survey on private information retrieval. *Bulletin of the EATCS*, 82:72–107, 2004.

[17] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[18] Michaela Götz. *On User Privacy in Personalized Mobile Services*. PhD thesis, Cornell University, 2012.

[19] Michaela Götz, Ashwin Machanavajjhala, Guozhang Wang, Xiaokui Xiao, and Johannes Gehrke. Publishing search logs - a comparative study of privacy guarantees. *TKDE*, 99(PrePrints), 2011.

[20] Marco Gruteser and Dirk Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In

MobiSys, 2003.

[21] Saikat Guha, Bin Cheng, and Paul Francis. Challenges in Measuring Online Advertising Systems. In *IMC*, 2010.

[22] Saikat Guha, Alexey Reznichenko, Kevin Tang, Hamed Haddadi, and Paul Francis. Serving Ads from localhost for Performance, Privacy, and Profit. In *HotNets*, 2009.

[23] Moritz Hardt and Guy Rothblum. A multiplicative weights mechanism for interactive privacy-preserving data analysis. In *FOCS*, 2010.

[24] Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. Boosting the accuracy of differentially private histograms through consistency. *PVLDB*, 3(1):1021–1032, 2010.

[25] Dorit Hochbaum and Anu Pathria. Analysis of the Greedy Approach in Problems of Maximum k-Coverage. *Naval Research Logistics*, 45(6):615–627, 1998.

[26] Christian S. Jensen, Hua Lu, and Man Lung Yiu. *Location Privacy Techniques in Client-Server Architectures*, pages 31–58. Springer-Verlag, 2009.

[27] Ari Juels. Targeted advertising ... and privacy too. In *CT-RSA*, 2001.

[28] Ali Khoshgozaran and Cyrus Shahabi. Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy. In *SSTD*, 2007.

[29] Aleksandra Korolova. Privacy violations using microtargeted ads: A case study. In *PADM*, 2010.

[30] Andreas Krause and Eric Horvitz. A utility-theoretic approach to privacy and personalization. In *AAAI*, 2008.

[31] Ashwin Machanavajjhala, Daniel Kifer, John M. Abowd, Johannes Gehrke, and Lars Vilhuber. Privacy: Theory meets practice on the map. In *ICDE*, 2008.

[32] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. ℓ-diversity: Privacy beyond k-anonymity. *TKDD*, 1(1), 2007.

[33] Frank McSherry and Ratul Mahajan. Differentially-private network trace analysis. In *SIGCOMM*, 2010.

[34] Emiliano Miluzzo, Cory T. Cornelius, Ashwin Ramaswamy, Tanzeem Choudhury, Zhigang Liu, and Andrew T. Campbell. Darwin phones: The evolution of sensing and inference on mobile phones. In *MobiSys*, 2010.

[35] Mohamed F. Mokbel, Chi-Yin Chow, and Walid G. Aref. The new casper: A privacy-aware location-based database server. In *ICDE*, 2007.

[36] Linda Pareschi, Daniele Riboni, Alessandra Agostini, and Claudio Bettini. Composition and generalization of context data for privacy preservation. In *PerComm*, 2008.

[37] Vibhor Rastogi and Suman Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *SIGMOD*, 2010.

[38] Pierangela Samarati and Latanya Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, CMU, SRI, 1998.

[39] Xuehua Shen, Bin Tan, and ChengXiang Zhai. Implicit user modeling for personalized search. In *CIKM*, 2005.

[40] Elaine Shi, T-H. Hubert Chan, Eleanor Rieffel, Richard Chow, and Dawn Song. Privacy-preserving aggregation of time-series data. In *NDSS*, 2011.

[41] Eran Toch, Justin Cranshaw, Paul Hankes Drielsma, Janice Y. Tsai, Patrick Gage Kelley, James Springfield, Lorrie Cranor, Jason Hong, and Norman Sadeh. Empirical models of privacy in location sharing. In *Ubicomp*, 2010.

[42] Vincent Toubiana, Helen Nissenbaum, Arvind Narayanan, Solon Barocas, and Dan Boneh. Adnostic: Privacy preserving targeted advertising. In *NDSS*, 2010.

[43] Xiaokui Xiao, Guozhang Wang, and Johannes Gehrke. Differential privacy via wavelet transforms. In *ICDE*, 2010.

[44] Yabo Xu, Ke Wang, Benyu Zhang, and Zheng Chen. Privacy-enhancing personalized web search. In *WWW*, 2007.

[45] Mingqiang Xue, Panos Kalnis, and Hung Keng Pung. Location diversity: Enhanced privacy protection in location based services. In *LoCA*, 2009.

[46] Jun Yan, Ning Liu, Gang Wang, Wen Zhang, Yun Jiang, and Zheng Chen. How much can behavioral targeting help online advertising? In *WWW*, 2009.

[47] Man Lung Yiu, Christian S. Jensen, Xuegang Huang, and Hua Lu. Spacetwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile services. In *ICDE*, 2008.

APPENDIX

A. PROOFS

A.1 Proof of Corollary 4.3

PROOF. Consider two neighboring click logs L, L' where L' is obtained from L by adding or deleting the data of a single user. Consider the hierarchy H consisting of height(H) levels where level i contains 2^i many nodes. We denote by $c_{l,j}$ ($c'_{l,j}$, respectively) the count of the j^{th} node at level l in L (L', respectively). We denote by $c_{l,j,a,1}$ ($c'_{l,j,a,1}$) the count of the clicks on a in the j^{th} node at level l and by $c_{l,j,a,0}$ ($c'_{l,j,a,0}$) the count of the views of a in the j^{th} node at level l that did not result in clicks in L (L', respectively).

Within a level of the hierarchy the L_2-sensitivity of each count is at most m. Overall the square of the L_2-sensitivity is at most

$$\sum_{l=0}^{\text{height}(H)-1} \sum_{j=1}^{2^l} (c_{l,j} - c'_{l,j})^2 \tag{1}$$

$$+ \sum_{a} (c_{l,j,a,1} - c'_{l,j,a,1})^2 + (c_{l,j,a,0} - c'_{l,j,a,0})^2 \tag{2}$$

$$\leq \sum_{l=0}^{\text{height}(H)-1} 3 \cdot m^2 \tag{3}$$

$$= 3\text{height}(H)m^2 \tag{4}$$

Thus, the L_2-sensitivity is bounded by $\sqrt{(3\text{height}(H)}m$. From Theorem 4.1 it follows that choosing $\sigma^2 \geq 3\text{height}(H)m^2 2\log(4/\delta)/\epsilon^2$ guarantees (ϵ, δ) probabilistic differential privacy. □

A.2 Utility Analysis of Algorithm 2

We define the utility of an estimate $\widehat{\mathsf{CTR}}(a|v)$ by comparing it to the true click-through-rate:

$$\frac{\texttt{clicks\_true}_{a,v}}{\texttt{clicks\_true}_{a,v} + \texttt{no\_clicks\_true}_{a,v}}$$

We say the estimate is (α, β)-accurate if with probability at least β:

$$\widehat{\mathsf{CTR}}(a|v) \geq \frac{\texttt{clicks\_true}_{a,v} - \alpha}{\texttt{clicks\_true}_{a,v} + \texttt{no\_clicks\_true}_{a,v} + 2\alpha} \tag{5}$$

$$\widehat{\mathsf{CTR}}(a|v) \leq \frac{\texttt{clicks\_true}_{a,v} + \alpha}{\texttt{clicks\_true}_{a,v} + \texttt{no\_clicks\_true}_{a,v} - 2\alpha} \tag{6}$$

Consider our protocol Estimates (and its parameters N, t, σ as in Corollary 4.3) and suppose all users respond truthfully. Then the computed estimates are (α, β)-accurate for

$$\beta \leq 1 - 3/(2\pi\alpha)\sigma\sqrt{(N/((1-t)N-1))}\exp^{-\alpha^2((1-t)N-1)/(2N\sigma^2)}.$$

The proof follows from a Gaussian tail bound that we can use to bound the probabilities of

$$|\texttt{clicks\_true}_{a,v} - \texttt{clicks}_{a,v}| > \alpha \text{ and} \tag{7}$$

$$|\texttt{no\_clicks\_true}_{a,v} - \texttt{no\_clicks}_{a,v}| > \alpha \tag{8}$$

However, it may happen that the algorithm does not output an estimate $\widehat{\mathsf{CTR}}(a|v)$ for some a, v if the noisy count of context v or some ancestor in the context hierarchy is less than the `min_support`.