# PocketWeb: Instant Web Browsing for Mobile Devices

Dimitrios Lymberopoulos, Oriana Riva, Karin Strauss, Akshay Mittal‡, Alex Ntoulas
Microsoft Research, Redmond, WA, USA
‡Indian Institute of Technology, Kanpur, India
{dlymper,oriana,kstrauss,antoulas}@microsoft.com, amittal@iitk.ac.in

## ABSTRACT

The high network latencies and limited battery life of mobile phones can make mobile web browsing a frustrating experience. In prior work, we proposed trading memory capacity for lower web access latency and a more convenient data transfer schedule from an energy perspective by prefetching slowly-changing data (search queries and results) nightly, when the phone is charging. However, most web content is intrinsically much more dynamic and may be updated multiple times a day, thus eliminating the effectiveness of periodic updates.

This paper addresses the challenge of prefetching dynamic web content in a timely fashion, giving the user an instant web browsing experience but without aggravating the battery lifetime issue. We start by analyzing the web access traces of 8,000 users, and observe that mobile web browsing exhibits a strong spatiotemporal signature, which is different for every user. We propose to use a machine learning approach based on stochastic gradient boosting techniques to efficiently model this signature on a per user basis. The machine learning model is capable of accurately predicting future web accesses and prefetching the content in a timely manner. Our experimental evaluation with 48,000 models trained on real user datasets shows that we can accurately prefetch 60% of the URLs for about 80-90% of the users within 2 minutes before the request. The system prototype we built not only provides more than 80% lower web access time for more than 80% of the users, but it also achieves the same or lower radio energy dissipation by more than 50% for the majority of mobile users.

## Categories and Subject Descriptors

H.4.m [**Information Systems**]: Information Systems Applications—*Miscellaneous*

## General Terms

Algorithms, Human Factors, Experimentation

## 1. INTRODUCTION

With recent advances in large touch screens and widespread data networks, smartphones are rapidly gaining popularity. They are the most convenient device to access the web, and according to a recent study [22], mobile devices are expected to surpass desktop web browsing in the next 4 years. Mobile phone user's experience has come a long way in the past decade, but such devices still face high network latencies and limited battery life, which can make the mobile experience frustrating.

Luckily, memory capacity is still experiencing healthy improvements [15], and can be used to mitigate the two previous limitations. Surplus memory can be used to store data that is brought to the mobile device when network conditions are favorable. The data can be later accessed at low latency and low energy cost. Based on this observation we have proposed the concept of Pocket Cloudlets [19], *i.e.*, bringing part of cloud services into mobile phones to reduce latency and energy consumption, with the added benefit of significantly reducing the load on the server side as well. We demonstrated the concept using a search service: a set of popular search queries and results is loaded onto the phone at night, when the phone is charging, to speed up searches during the next day. However, in that work we limited ourselves to search and did not address actual web content. While search results change slowly (they can be considered static on a daily basis), web content can change quickly during a single day.

To enable a faster mobile web browsing experience on the currently available mobile infrastructure, this paper proposes an intelligent web content prefetcher that downloads web content on the mobile device at appropriate times, anticipating a user's future web accesses. Perhaps the biggest challenge with prefetching web content compared to search queries and results is that web content is *dynamic*. While the mapping of search queries to search results remains relatively stable over days or even weeks, web content changes frequently. News and social network web sites change continuously, such that the nightly update approach does not work as well as it does for search.

A naïve approach to achieving high prefetching accuracy is to prefetch as much content we can as often as possible. However, this approach is not practical due to constraints in battery capacity and limited network bandwidth. Downloading all frequently accessed web pages of a user every 2 minutes might ensure a lightning-fast web browsing experience, but the battery would not last very long. At a high level, we can consider a prefetch of a web page as unsuccessful or "wasted" either because it happened too long ago (and thus the content on the device is stale), or because the user did not end up explicitly requesting the web page at all. In either of these cases, we end up using some phone's resources without realizing any gains. Therefore, our goal is to achieve timely prefetches without increasing (and possibly decreasing!) energy consumption.

We first analyze the web access traces from roughly 8,000 mobile users over a period of 3 months and show three well-defined


*ASPLOS'12,* March 3–7, 2012, London, England, UK.

patterns. First, users often visit a small set of web pages from their phones, which they tend to repeatedly visit over time. Second, user accesses are often periodic and happen at given time windows. For example, a user may check the Facebook updates every 30 minutes and the CNN news right when she wakes up and during an afternoon break. Third, users often access content in bursts. For example, when the user checks the news, she may also check the weather forecast and current stock prices.

Prior work in web content prefetching has focused on simply correlating sets of web sites accessed together and using this information to prefetch the set when the first page of a set is accessed. In this work, we study properties of mobile web accesses that go beyond these sequential features. We take advantage of the spatiotemporal access patterns obtained from our analysis and use machine learning techniques to create a model that can be used to predict both *what* web pages a user is likely to request as well as *when* these requests are likely to occur. By learning how each individual user accesses the web over time, the phone can proactively download web content before the user explicitly attempts to access it, thus enabling an instant mobile browsing experience.

Our experimental evaluation with real user datasets shows that we can accurately prefetch 60% of the URLs for about 80-90% of the users within 2 minutes before the request. Furthermore, the proposed approach not only provides more than 80% lower web access time for more than 80% of the users, but it also achieves the same or lower radio energy dissipation by more than 50% for the majority of mobile users.

In summary, this paper makes the following contributions:

- Provides a detailed analysis of web browsing on mobile phones from 8,000 users, showing widely disparate behaviors from user to user but a strong spatiotemporal structure for individual users.
- Formulates the problem of web content prefetching as a binary access prediction problem in machine learning, where the features are derived from the observable structure of mobile web browsing. We use stochastic gradient boosting techniques for this purpose, which allow us to provide insight into which features are the most relevant to access prediction.
- Experimentally evaluates the accuracy of the proposed approach for each user by creating individual user models with a portion of the accesses in each trace, and testing their performance with the remaining portion. Quantifies, using a prototype implementation, the impact of the proposed approach on the web access time and radio power consumption with respect to the state-of-the-art.

# 2. MOBILE WEB BROWSING ANALYSIS

Central to our work is the ability to understand and model effectively the user browsing behavior. We start our study by first providing a description of the data set that we used, and continue with the results of our analysis both *on aggregate* across users and *individually* per user.

## 2.1 Data Set

We used the mobile web access logs of 8,000 users across the United States over a 3-month period. The users were randomly selected among a larger number of users that opted to download and install the Bing application or to enable the pre-installed Bing toolbar on their mobile phones. Users' phones varied from high-end smartphones (*e.g.*, iPhone, Android, Blackberry) to low-end featurephones (*e.g.*, LG and Samsung devices with custom operating systems). From the total of 8,000 users in our dataset, half are smartphone users and half are featurephone users. To get a deeper understanding on user behavior, each of the two sets of users was further split into 4 different classes (low, medium, high, and extreme volume classes) based on the monthly volume of web accesses ([20-40), [40,140), [140,460), [460,$\infty$) respectively). In the logs we analyzed, the information on each web access included unique user identifier, full path of accessed URL, and access timestamp.

## 2.2 Repeatability of Mobile Web Accesses

We first study the repeatability of mobile web accesses. We compute the number of times that any user will be visiting a new unique URL (*i.e.*, a full path URL that has not been visited before) in the next access. We show the results in Figure 1(a) across volume classes and device types. Approximately 40% to 60% of the smartphone users, for the low and extreme volume classes, are likely to visit a new URL 20% of the time. In other words, 80% of the URL visits are repeated visits for roughly half of the smartphone users. We also observe that users in higher volume classes are more likely to repeat visits than users in lower volume classes. Finally, although the trends are similar for featurephones, the overall repeated visits are higher when compared to smartphone users. Intuitively, featurephone users that have to interact with devices with constrained user interfaces and hardware capabilities are more likely to access the web in a more conservative way compared to smartphone users. They tend to explore the web less and focus more on accessing web content they really need to access.

To examine the repeatability of URL visits in more detail, we also compute, for each user, the cumulative URL volume for the top URLs that the user accesses. The result is shown in Figure 1(b). The numbers on the horizontal axis represent the top most frequently visited URLs (these URLs might be different across users). The vertical axis shows the cumulative fraction of the total URL visits that the number of the most frequently visited URLs is responsible for across users. Notably, across user classes, the most frequently visited URL accounts for about 50% of the overall user's URL visits. In other words, a single URL is responsible for approximately half of a typical user's URL requests. However, we also observe that the overall URL volume varies across users. More specifically, there are users for whom more than 90% of their total URL volume can be attributed to a single URL, and users for which the most frequent URL corresponds to less than 10% of their total volume. It is therefore important for any prefetching technique to take into account the individual characteristics of every user.

## 2.3 Targeted vs. Untargeted web Accesses

From the analysis so far we infer that the URLs that a user visits fall into two classes: there is a small number of frequently visited URLs, and a long tail of infrequently visited URLs. In order to investigate the properties of these two kinds of URLs we classify them as *targeted* and *untargeted* URLs. We define a *targeted* URL to be one which was visited by the user at least 5 times in a month. We chose this threshold by closely analyzing the user web access logs. We found that smaller thresholds, such as 3, could be too permissive and cause 50% of the extreme volume users to have more than 50 targeted URLs. On the other hand, higher thresholds such as 10 could be too aggressive and cause 30% of the low volume users to have 0 targeted URLs.

We analyze the volume of web accesses generated by targeted and untargeted URLs. As Figure 2 shows, although the targeted accesses are only slightly more than the untargeted accesses for the

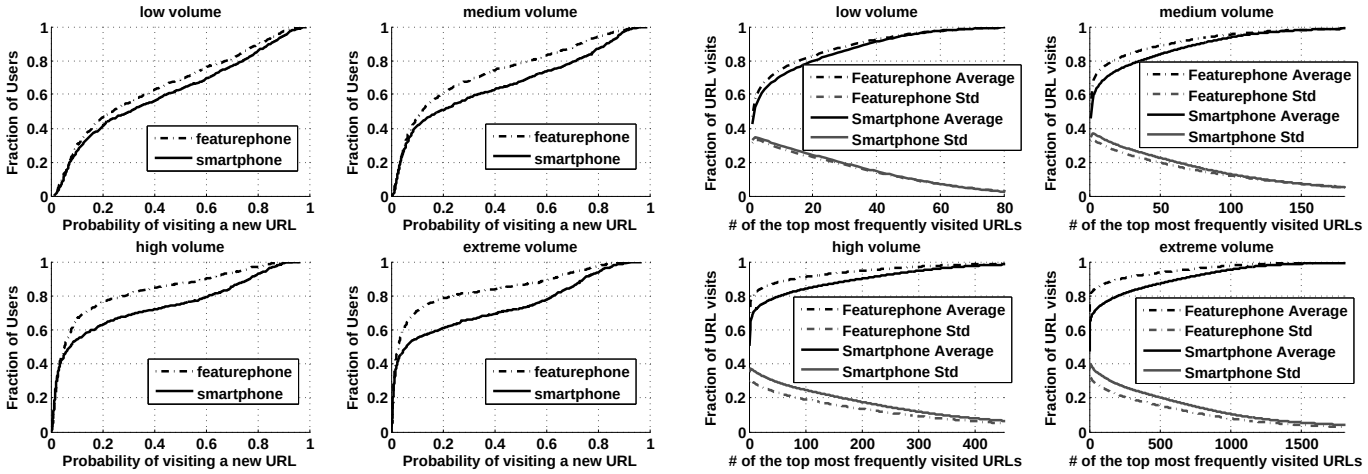


**Figure 1: (a) Repeatability of mobile URL visits. (b) Average and standard deviation of the cumulative URL volume that the top most frequently visited URLs are accountable for.**

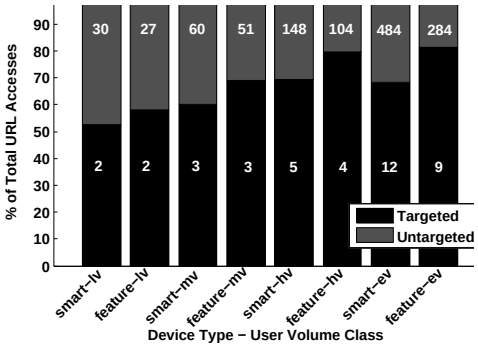


**Figure 2: Breakdown of total URL accesses into targeted (URLs that have been accessed at least 5 times in a month) and untargeted. The white numbers in each bar plot represent the average number of unique targeted/untargeted URLs.**

low volume users, the URL accesses are dominated by targeted accesses for the remaining classes of users. For example, for high volume users, targeted accesses account for 70% of the total smartphone URL accesses.

Figure 2 also provides more insight on the average number of *unique* targeted URLs across the different volume classes and device types (number indicated inside the bars in Figure 2). For low and medium volume users and for both featurephones and smartphones, the average number of unique targeted URLs is roughly 2 and 3 respectively. For high and extreme volume classes, it increases to 9 and 12 for featurephones and smartphones respectively. In other words, 2 to 12 unique URLs are, on average, responsible for more than 70% of a user's web accesses. Hence, enabling the mobile device to properly model *when* and *which* of the small number of targeted URLs will be accessed by the user is of paramount importance for an effective prefetching policy.

## 2.4 Timing of Mobile Web Accesses

Web pages are constantly updated. For a prefetching technique to be effective, it needs to predict *when* the user's web accesses will take place. Hence, we study the temporal access patterns of our users. Figure 3 shows the elapsed time between consecutive smartphone web accesses for targeted, untargeted and combined (targeted and untargeted) URL visits. Approximately 35% to 50% of targeted URL visits across the 4 volume classes occur within 12 minutes (0.2 hours in Figure 3) of the last targeted URL visit. Additionally, 25% to 40% (depending on the volume class) of targeted URL visits take place within 6 minutes (0.1 hours in Figure 3) of the last targeted URL visit. Hence, a targeted URL access can serve as a good predictor of the time at which a next targeted URL access will occur.

According to Figure 3, untargeted URL visits tend to be more concentrated in time when compared to targeted URLs. Approximately 70% to 80% of untargeted URL visits (as opposed to 35% to 50% of targeted URL visits) take place within 12 minutes of the last untargeted URL visit. In other words, when mobile users explore the web, they tend to visit many more URLs within a short amount of time as compared to when visiting targeted content. We can leverage this information to improve the accuracy of prefetching and save battery resources by not prefetching targeted content when the user is about to visit untargeted URLs.

## 2.5 A Peek into Individual Users

In addition to relative timing, we also study the role of absolute timing in mobile web browsing (*e.g.*, time of day when URL accesses occur). In general, knowing when to expect URL accesses can drive content prefetching. Figure 4 shows the timestamps within a day of all URL accesses that 4 random smartphone users performed over 3 months.

The variance in mobile web access patterns across the 4 users is striking. The URL visits of users 1 and 4 are dominated by untargeted URLs. Most likely, a web content prefetching technique will have difficulty in modeling these users' web browsing patterns accurately, as it has no way of predicting the untargeted accesses. Interestingly, however, users 1 and 4 access web pages from their phones at given time intervals within the day (*e.g.*, user 4's accesses are only between 6am and 9am, 9pm and 11pm, and midnight and 2am). On the other hand, for users 2 and 3, web accesses are dominated by a small set of targeted URLs (2 targeted URLs for user

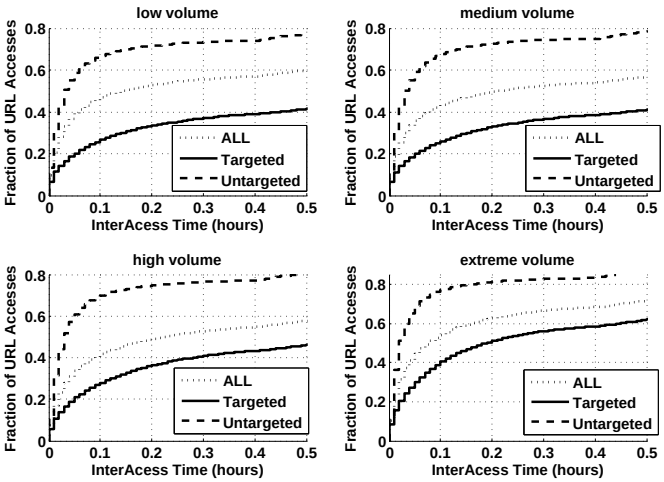


**Figure 3: Time elapsed between consecutive smartphone web accesses when all, targeted, or untargeted URLs are considered. The trends are identical for featurephones (not shown).**

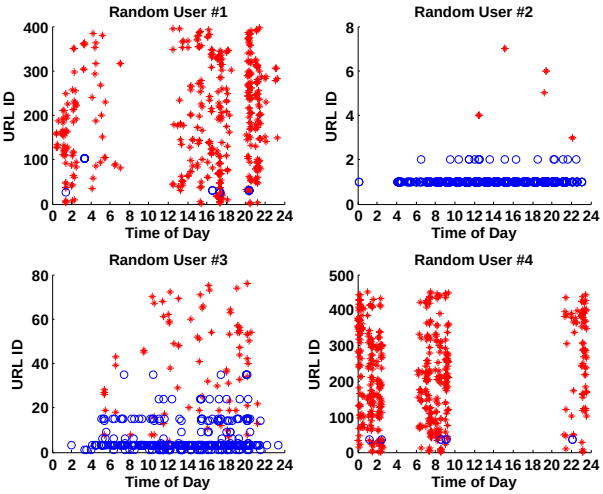


**Figure 4: Web accesses of 4 representative smartphone users in the high volume class. All accesses over the 3 months are projected into a single day. The red stars represent untargeted URL visits. The blue circles represent targeted URL visits.**

2 and 7 for user 3). More importantly, the single most frequently visited targeted URL is responsible for the majority of that user's URL visits. Additionally, both users 2 and 3 tend to continuously access this single targeted URL periodically throughout the day. If a prefetching policy can predict *when* the targeted URL will be accessed by the users based on their periodic accesses, it can be very effective in providing an instant mobile browsing experience.

## 2.6 Summary and Key Findings

The analysis of real web access logs from 8,000 users has highlighted different aspects of mobile web browsing behavior that are critical to content prefetching:

- A small number of targeted URLs is responsible for the majority of a user's URL visits. Predicting these targeted URL accesses can have a huge impact on the user's browsing experience.
- Targeted URL accesses are clustered in time. Mobile users tend to access them in batches within short time windows. Hence, recent targeted URL accesses can be strong indicators of future URL visits.
- Untargeted URL accesses are significantly more clustered in time than targeted URL accesses. Recent untargeted URL accesses can help us decide about the type of future URL accesses (targeted vs. untargeted).
- URL accesses exhibit strong temporal properties that help us prefetch content in a *timely* manner. Prefetching based only on past sequences (or, more generally, sets) of web accesses can be inefficient as it might not provide enough information to decide *when* to prefetch.
- Mobile web browsing behavior across users can vary greatly in the type and number of accessed URLs as well as the timing of URL accesses.

Thus, a data-driven content prefetching technique that takes advantage of the underlying spatiotemporal patterns of each individual user's web browsing behavior is required to enable timely and accurate content prefetching.

# 3. CONTENT PREFETCHING AS A LEARNING PROBLEM

Mobile web browsing behavior exhibits several spatial and temporal properties. To enable timely prefetching of web content, the prefetching scheme needs to carefully model and learn all these different properties for each individual user. However, optimally combining these properties is not straightforward.

Our approach is inspired by the web search community, where multiple hundreds or thousands of features are combined to rank web documents (*i.e.*, URLs). In a web search scenario, the user submits a query and the search engine ranks a set of URLs to show the most relevant ones higher up in the result page. The ranking problem is often formulated as a click prediction problem, where for every related URL, the engine has to estimate the probability of a user click on that URL. The higher the probability, the higher the rank of the URL. To create the click prediction model, search engines leverage, among other signals, web search click logs. For every URL displayed to the user, various features are computed, encoding information about the user, the query, the URL or all of them. Each feature vector is labeled as a click (if the URL was clicked) or a non-click (if it was not clicked). The click prediction model is then trained using millions of these labeled feature vectors.

In web content prefetching, the web search click logs are replaced by the user's web access logs. The URLs are no longer web search results, but the targeted URLs identified in the user's web access logs. The click prediction model is now turned into a web access prediction model whose role is to assign, at any given time, an access probability to each targeted URL. The higher the probability, the more likely the user is to request access to this URL. The features used to train the prediction model are the most critical elements of the modeling process and are directly extracted from the user's web access logs.

Conversely to web search, in web content prefetching the user does not explicitly submit a query. Thus, to determine *when* the mobile device should make a web access prediction, we follow an event-driven approach where the mobile device makes web access predictions as a result of certain user actions. For instance, the

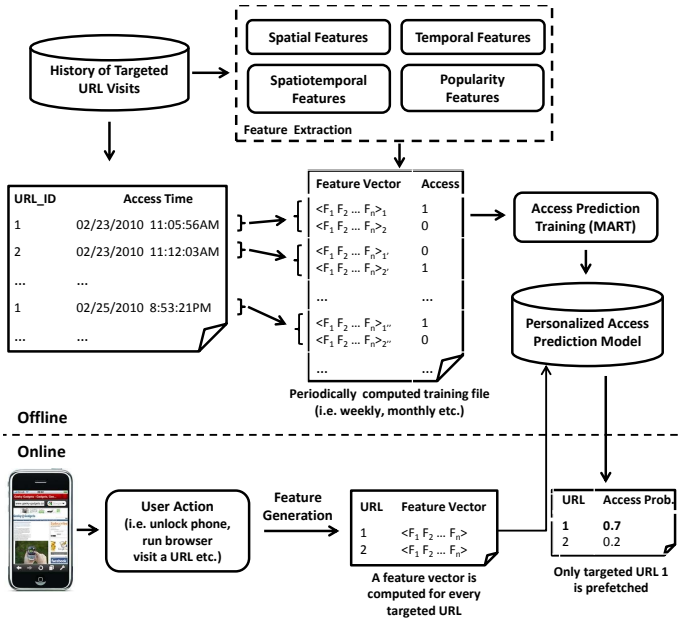


**Figure 5: We specify the problem of web content prefetching as an access prediction problem. In this example, the user has two targeted URLs.**

mobile device makes a prediction every time the user unlocks the phone, activates the browser or visits a URL. Depending on the resulting probabilities, the phone decides whether to prefetch any content.

Figure 5 provides an overview of our web access prediction approach. Offline, the mobile device records user web accesses, including automatic page refreshes, and periodically (*e.g.*, weekly, monthly) uses this information to build a web access prediction model for the user. This model can be built on the mobile device itself, but most likely would be built on users' desktops or the cloud. First, a set of features is extracted for every targeted URL in a user's web logs. The role of these features is to encode the underlying structure of mobile web browsing behavior in terms of spatial and temporal properties, as described in Section 2. Then, the web access logs are mapped to a set of feature vectors that are annotated as accesses or non-accesses.

Using these labeled feature vector traces, the system trains a model that leverages state-of-the-art stochastic gradient boosting techniques [32], creating a decision tree that supports time ranges, with access probabilities in its leaves. The goal of the training phase is to combine the different features provided as input to enable the model to maximize the number of correct predictions in the test data set. Online, user actions such as unlocking the phone may trigger a prediction. A feature vector is then generated for every targeted URL, and the prediction model is invoked with each of these feature vectors as input. For every feature vector and thus every targeted URL, the model assigns an access probability. Depending on these probabilities and a pre-configured threshold, none, one or more targeted URLs are fetched, along with associated images, css, javascript, etc.

Formulating the problem of content prefetching as a binary access prediction problem and leveraging state-of-the-art stochastic gradient boosting techniques provides a significant advantage. The ability of these modeling tools to consume a large number of features and exploit their dependencies to maximize access prediction enables us to virtually encode any possible information as a feature without being limited by the nature of the modeling tool (*e.g.*, Markov model). More importantly, it allows us to compare the effectiveness of different features in the access prediction task to discover which features best capture mobile web browsing patterns.

## 3.1 Web Access Prediction with MART

We use MART [32], a learning tool based on *M*ultiple *A*dditive *R*egression *T*rees, widely deployed in commercial systems for web search and advertisement ranking. MART is based on the stochastic gradient boosting approach [14].

To construct the predictive model, MART takes as input historical data on web accesses, which is partitioned into training and validation sets, where the size of the latter is typically $1/5^{th}$ or $1/6^{th}$ of the former. The training data set is used to build the model, while the validation set is used to provide an unbiased error estimate. Each entry contains a set of $n$ features, $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$,[1] that might be related to a spatial parameter $s$ (*e.g.*, the previously accessed URL), a temporal parameter $t$ (*e.g.*, time of day), a spatiotemporal parameter $b$ (*e.g.*, time since this URL was last accessed), or a popularity feature $p$ (*e.g.*, how popular this URL across all accessed URLs), in conjunction with a label $a$ which records the user's action. The training data is fed into MART to build a classification model, $\mathcal{M}$, which is used to estimate the probability of access $p_{\mathcal{M}}(a|s,t,b,p)$. In our tests, we use the gradient-descent as the optimization technique, and use binary decision trees as the fitting function.

Once model $\mathcal{M}$ is trained, MART reports a relative ordering of all features $\mathcal{F}_q$, which indicates the "*relative feature importance*" when making a prediction. The larger the number of decision tree branches associated to a feature, the higher the importance of that feature. The most important feature has an importance of 1, while other features have a relative importance between 0 and 1. The relative importance of feature values allows us to quantify and compare the impact of different features on the prediction accuracy.

## 3.2 Feature Generation

Feature generation is one of the most critical steps of the modeling process. Its role is to provide as accurate and discriminative information as possible about the mobile web browsing behavior of individual users. We leverage the findings from our large scale user study to drive the feature generation process. According to our analysis, mobile web browsing behavior exhibits strong *spatial*, *temporal*, and *spatiotemporal* structure. As a result, we introduce 4 sets of features, shown in Table 1, that focus on capturing this structure. Overall, for a user with $k$ targeted URLs, $2 \times k + 11$ features are computed. A detailed description of each feature is provided in Table 1.

The *spatial features* encode sets of mobile web accesses accessed together (Figure 1), and record information about the immediately previously accessed URL (either that it was to an untargeted URL, or to which targeted URL it was). The *temporal features* record the periodicity of targeted web accesses (Figure 4), and the time of day or day of the week in which these accesses occur. The *spatiotemporal features* combine spatial and temporal properties to encode information such as the time elapsed between a targeted URL access and the immediately previous access (Figure 3). These features can provide invaluable information to the model to decide *what* targeted URL and *when* to prefetch, enabling timely prefetches. The *popularity features* encode the popularity of each

[1]As the model's goal is to predict targeted web accesses, the feature vectors are generated only for targeted web accesses or non-accesses. However, the feature values encode information about both targeted and untargeted web accesses.

| Feature class | Feature name | Definition | Example for $t_x$=$t_1$ in Figure 6 |
|---|---|---|---|
| spatial | $t_i$_*Targeted*$^*$ | whether the previous access was to the targeted URL $t_i$, for $i \in [1,k]$ | $t_1 = 0; t_2 = 0$ |
| | *Untargeted* | whether the previous access was an access to an untargeted URL | 1 |
| temporal | *TimeOfDay* | time of day at which the access occurred | $11h15m$ |
| | *AvgTime* | average access time over all accesses to the same targeted URL $t_x$ | $9h06$ |
| | *TimeStdev* | standard deviation over all access times for the same targeted URL $t_x$ | $55m$ |
| | *Weekend* | whether the access happened during a weekend | 1 |
| spatio-temporal | *TimeSinceTargeted* | time elapsed since any targeted URL was last accessed | $1d3h15m$ |
| | $t_i$_*TimeSinceTargeted*$^*$ | time elapsed since the targeted URL $t_i$ was last accessed, for $i \in [1,k]$ | $t_1 = 1d3h15m$; $t_2 = 1d3h14m$ |
| | *TimeSinceUntargeted* | time elapsed since any untargeted URL was last accessed | $2m$ |
| | *TimeSinceAccess* | time elapsed since the targeted URL $t_x$ was last accessed | $1d3h15m$ |
| | *AvgInterAccessTime* | average time between two consecutive accesses to the targeted URL $t_x$ | $1d12h31m$ |
| | *InterAccessTimeStdev* | standard deviation of inter-access times for the targeted URL $t_x$ | $9h16m$ |
| popularity | *PopAmongTargeted* | popularity of the targeted URL $t_x$ over all $k$ targeted URLs | 75% |
| | *PopAmongAll* | popularity of the targeted URL $t_x$ over all targeted and untargeted URLs | 37.5% |

**Table 1: Features used for training MART. In the table, we assume the feature vector describes an access to a targeted URL $t_x$ of a user with $k$ targeted URLs. Feature names marked with * correspond to $k$ features, one per targeted URL.**

targeted URL, and can be used to decide which targeted URLs will most likely be accessed. These features can be particularly important, given that the number of targeted URLs varies anywhere from 1 to 10, but usually 2 or 3 of these URLs are responsible for most targeted URL accesses (Figures 2 and 4).

## 3.3 Training for Timely Prefetches

To understand how features are computed, and training, validation and test files are generated, consider the log of an example user in Figure 6. Web logs are first split into training, validation and test with a data volume ratio of 70%:10%:20%, respectively. We initially process the training data only to identify the targeted URLs of the user, namely $t_1, t_2, \ldots, t_k$ – URLs that have been accessed more than 5 times a month. For the example user in Figure 6, $k = 2$. After having identified the targeted URLs, the training, validation and test data sets are generated as follows.

Starting from the first web access in the logs, we divide user's web accesses into multiple *access epochs* with a duration of $D$ minutes each. For instance, Figure 6 shows 4 different access epochs, where $D = 5$. For each access epoch, we compute $k$ feature vectors (Table 1), one for every targeted URL. The vectors for targeted URLs accessed in the epoch are labeled as accessed. All other vectors are labeled as non-accessed.

If a targeted URL is accessed multiple times within an epoch, only one feature vector is computed for the first access to that URL within the epoch. This biases the model towards the first access, which improves timeliness. The duration of the access epoch $D$ defines the freshness of the prediction, and, thus, we call it the *freshness threshold*. Whenever a prediction is made in the beginning of an access epoch and a URL is prefetched, its content is considered fresh for the duration $D$ of the access epoch. In that way, we implicitly assume that the web content update rate is less than 1 update per $D$ minutes.

The labeled feature vectors in the training file are used offline to train the access prediction model. The labeled feature vectors in the test data set are used to evaluate the performance of the model. In particular, for each feature vector in the test file, we retrieve the access probability from the model. An access probability higher than 0.5 corresponds to prefetching the targeted URL (this probability threshold could be adjusted dynamically depending on network and battery conditions). The success or not of the prediction depends on the label of the feature vector in the test file. If it was marked as accessed, the prefetching was successful, otherwise the prefetching was unsuccessful.

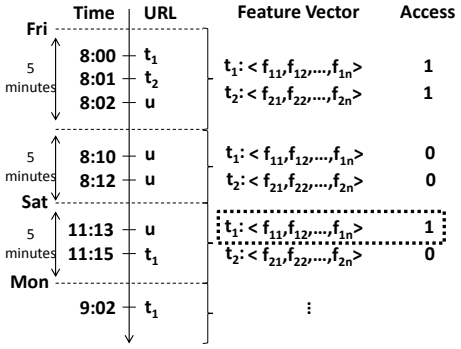


**Figure 6: Example of web accesses trace and the resulting web accesses and non-accesses events generated. Dashed lines delimit access epochs. The feature vector values for $t_1$ in the $3^{rd}$ access epoch are shown in the last column of Table 1.**

# 4. EVALUATION

We evaluate our approach using logs of all 8,000 users in our dataset. We split the dataset into training, validation and test sets, and use the test set to measure the accuracy in predicting both web accesses and non-accesses. To evaluate the freshness of the prefetches, we need to know the content update rate of each web page. Unfortunately, this is hard to reliably estimate because we cannot know all the updates for the pages in our dataset and because several pages require user authentication to be accessed (*e.g.*, Facebook). To provide a notion of prefetch timeliness we leverage the concept of *freshness threshold* previously introduced. Different freshness thresholds allow us to evaluate the ability of the model to timely prefetch content for different time thresholds and, thus, for different content update rates. In the experiments, we use three different thresholds: 2, 5 and 10 minutes. To study the impact of different features on the model's prediction accuracy, we train models with either all or a subset of the proposed features. Overall, for each user we train 6 different models (2 sets of features and 3 freshness thresholds). Across all users, we train a total of 48,000 prediction models.

Finally, we use our prototype implementation of the proposed prefetching approach, shown in Figure 7, to provide an estimate on the power consumption overhead and the web access time gains achieved. The preferred freshness threshold is manually selected by the user.

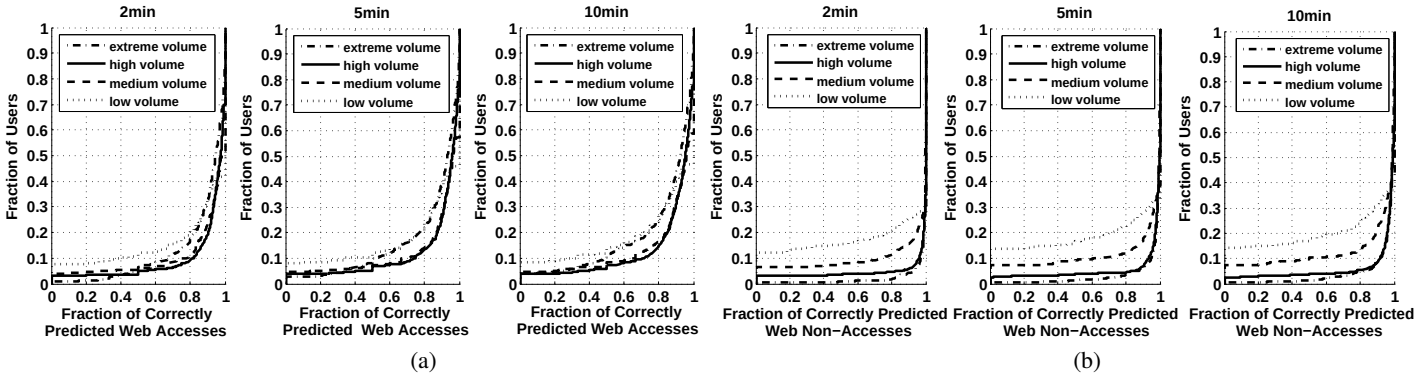


**Figure 8: CDF of successful predictions of (a) web accesses and (b) web non-accesses for smartphone users across different volume classes and freshness thresholds. The results for featurephone users are very similar and are not shown in the interest of space.**

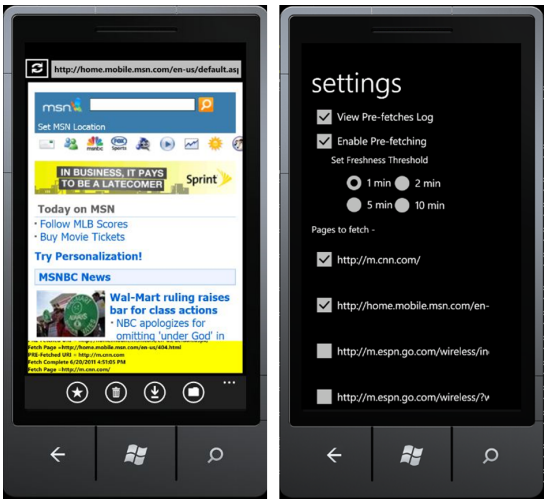


**Figure 7: Web access prediction system prototype running on Windows Phone 7.**

## 4.1 Model Prediction Accuracy

We start by evaluating the accuracy of our approach. Our goal is to maximize the number of successful prefetches and minimize the number of unsuccessful prefetches. In all the experiments, we decide to prefetch a web page when the access probability returned by the model is higher or equal to 0.5.

### *4.1.1 Prefetching Accuracy*

Figure 8(a) shows the trained model accuracy in predicting smartphone users' web accesses across different volume classes, and freshness thresholds. We first consider the freshness threshold of 2 minutes. For roughly 80-90% of the users, the prediction model correctly predicts the targeted URL that will be accessed in the next 2 minutes at least 80% of the time. In other words, 80% of the targeted URL accesses of approximately 80-90% of the users can be prefetched within 2 minutes of the user actually accessing them. To put things in perspective, 80% of the targeted URL accesses correspond to 55%-60% of the all URL accesses (targeted and untargeted) for low and medium volume users and to 70% for high and extreme volume users (Figure 2(a)).

The prefetching accuracy across all user volume classes is quite similar. In general, the smaller the web access history, the harder it is to make predictions because the repeatability of URL accesses is lower and low volume users visit the web less frequently, thus not providing enough information for the training phase. Still, the models for low volume users achieve 80% or higher accuracy in 80% of the cases. On the other hand, for high and extreme volume models it is difficult to achieve 100% accuracy in all cases because despite the rich traces of these users, some exhibit a random behavior.

The performance of the prediction model does not vary significantly across freshness thresholds. Higher freshness thresholds (5 and 10 minutes) cause only a slight degradation in the prediction accuracy. For instance, for the high volume class, the percentage of users for which the model successfully predicts at least 80% of their web accesses reduces from 90% (2 minutes) to 87% (10 minutes).

The accuracy for featurephone users (omitted due to space constraints) is consistently slightly better than that for smartphone users. For 95% of the featurephone users (compared to 90% for smartphones) we can successfully predict at least 80% of a given user's web accesses. This is due to the higher repeatability of URL accesses for featurephone users compared to smartphone users (Figure 1).

Overall, across all user volumes, device types and freshness thresholds, at least 80% of a user's targeted accesses can be successfully predicted for approximately 90% of the users. In practice, the proposed content prefetching technique can provide instant browsing experience for 80% of a user's targeted accesses while guaranteeing that the prefetched content will be at most 2 minutes old.

### *4.1.2 False Positives*

Correctly predicting web accesses helps provide an instant mobile browsing experience, but incorrectly predicting non-accesses to web pages can result in unnecessary prefetches (false positives) that can drain the battery of the mobile device. Figure 8(b) shows the fraction of successful non-access predictions of the trained model in the case of smartphone users, across different volume classes and freshness thresholds. For the 2-minute freshness threshold, the model successfully predicts non-accesses at least 80% of the time for anywhere between 90% (high volume class) and 80% (low volume class) of the users. Again, the lower the volume class, the lower the non-accesses prediction accuracy.

Figure 10 provides more insights on the ability of the proposed prefetching technique to accurately predict web non-accesses. It shows the CDF of the actual number of unsuccessful prefetches across smartphone users in the medium and high volume classes for

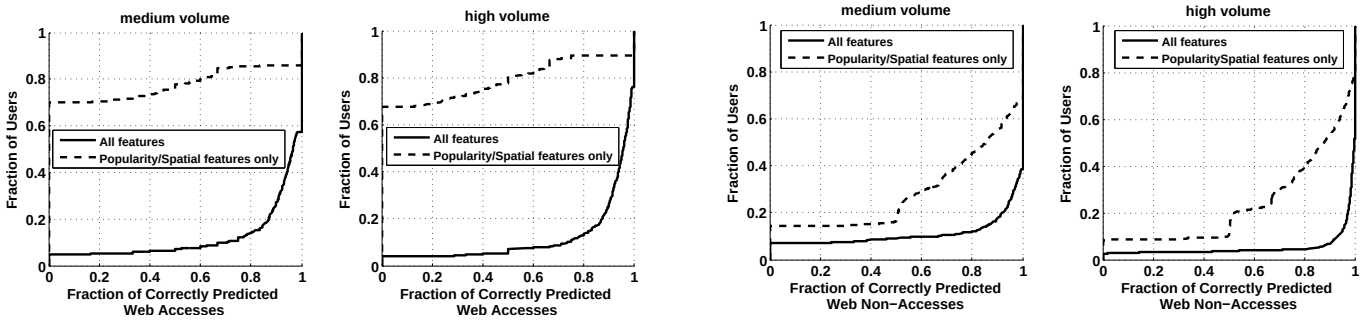


**Figure 9: CDF of successful predictions of web accesses, and web non-accesses when all features as well as no temporal or spatiotemporal features are used in the training phase, for the 5-minute freshness threshold and smartphone users. Results are similar across freshness thresholds, device types, and user volume classes, and are not shown in the interest of space.**

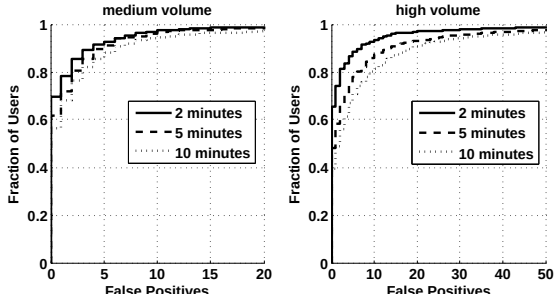


**Figure 10: CDF of the number of unsuccessful prefetches for medium and high volume smartphone users, and across different freshness thresholds. The results for featurephones and the rest user volume classes are not shown in the interest of space.**

different freshness thresholds. For approximately 90% of the users, the number of false positives (unsuccessful prefetches) is less than 4, 5, 10, and 100 for the low, medium, high and extreme volume classes respectively. Given that the test dataset of every user contained accesses over a period of approximately 20 days, these numbers correspond to less than one (low, medium and high volume classes) and approximately 5 (extreme volume class) unsuccessful prefetches per day. As a result, the proposed prefetching approach has minimal impact on the battery life of the mobile device for the vast majority of mobile users. However, if necessary to limit battery life degradation, the number of prefetches could be throttled by a daily threshold and the probability threshold above which a page is prefetched could be adjusted dynamically.

However, the tail of the CDF plots in Figure 10 can be relatively long, especially for extreme volume users. In general, there is a small percentage of users (less than 2% across all volume classes), for which the number of false positives can significantly increase. These are users with a random or unpredictable web access behavior. To prevent the number of unsuccessful predictions from negatively impacting the battery life of the mobile device, the prefetching mechanism can monitor, over time, its own access prediction accuracy and accordingly disable any prefetching attempts when the number of false positives exceeds a daily threshold. In that way, even these users will not experience any noticeable degradation in their mobile devices' battery life.

| Feat. Class | Feat. Name | Feature Relative Rank 1 | 2 | 3 | 4–7 | 8–14 |
|---|---|---|---|---|---|---|
| spat. | *$t_i$_Targeted* | 0.0 | 0.8 | 1.4 | 48.6 | 49.3 |
| | *Untargeted* | 0.0 | 0.4 | 1.1 | 11.6 | 86.9 |
| temp. | *TimeOfDay* | 2.2 | 5.8 | 8.6 | 82.2 | 1.2 |
| | *AvgTime* | 0.0 | 0.0 | 0.0 | 0.7 | 99.3 |
| | *TimeStdev* | 0.0 | 0.0 | 0.0 | 0.4 | 99.6 |
| | *Weekend* | 0.0 | 0.0 | 1.2 | 42.6 | 56.2 |
| spat.-temp. | *TimeSinceTargeted* | 56.7 | 22.4 | 13.8 | 7.0 | 0.1 |
| | *$t_i$_TimeSinceTargeted* | 4.2 | 6.5 | 22.5 | 54.0 | 12.7 |
| | *TimeSinceUntargeted* | 24.2 | 46.2 | 15.1 | 14.5 | 0.0 |
| | *TimeSinceAccess* | 2.7 | 5.9 | 24.1 | 66.0 | 1.4 |
| | *AvgInterAccessTime* | 0.0 | 0.0 | 0.0 | 0.1 | 99.9 |
| | *InterAccessTimeStdev* | 0.0 | 0.0 | 0.0 | 0.7 | 99.3 |
| pop. | *PopAmongTargeted* | 10.0 | 11.9 | 12.2 | 60.1 | 5.8 |
| | *PopAmongAll* | 0.0 | 0.0 | 0.0 | 11.6 | 88.4 |

**Table 2: Feature importance for high volume smartphone users. Each cell reports the percentage of users for which a certain feature was ranked $1^{st}$, $2^{nd}$, $3^{rd}$, between $4^{th}$ and $7^{th}$ or between $8^{th}$ and $14^{th}$. Results are consistent across all volume classes and phone types.**

### *4.1.3 Feature Importance*

In this section, we leverage the relative importance of features reported by MART (described in detail in the previous section) to study the impact of different features on the performance of the web access prediction model.

Table 2 shows the relative ranking for the spatial, temporal, spatiotemporal and popularity features for high volume smartphone users (results are similar for other device types and volume classes). Spatiotemporal features have the highest relative importance. For 88% of the high volume users, at least one of the spatiotemporal features is ranked $1^{st}$. Popularity features follow next, with 10% of the users having them as the most important features. Pure temporal features are very rarely ranked $1^{st}$ (2% of the cases), and instead are at the $4^{th}$ or worse positions in most cases. Spatial features come last: they almost never qualify as one of the top three most important features. This clearly demonstrates the spatiotemporal structure of mobile web browsing behavior. Approaches that predict web accesses solely based on past sequences of web accesses fail to capture the temporal structure of mobile web browsing, one of its most important underlying properties based on these results.

Feature importance is only a relative comparison that indicates the usage frequency of each feature in the prediction model. To better study the impact of different features in prediction accuracy, we trained two different MART models for every user. One uses all the proposed features (Table 1) while the other uses only spatial

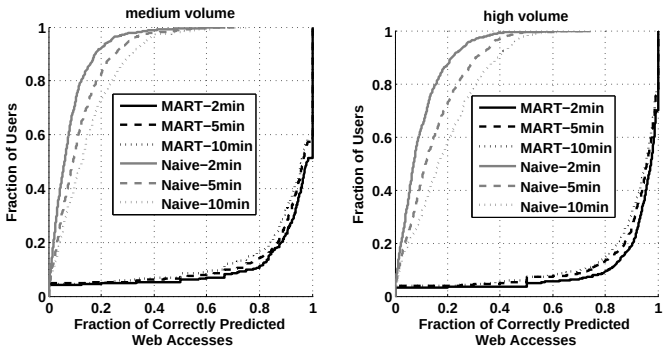


**Figure 11: CDF of successful predictions of web accesses for MART and the naïve sequential-based prefetching scheme, for smartphone users. The results for featurephone users and the other volume classes are identical, and are not shown in the interest of space.**

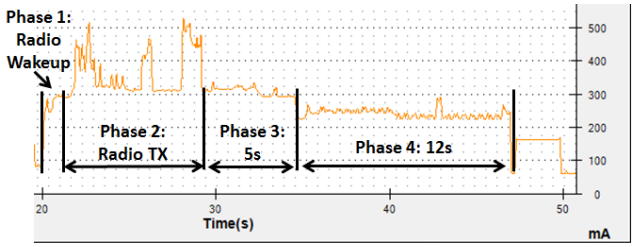


**Figure 12: Current consumption (@4.2V) of a Samsung Focus running Windows Phone 7 while transmitting UDP packets. The actual transmission (phase 2) accounts for only a portion of the radio's power consumption (phases 1,3 and 4).**

and popularity features. Figure 9 shows the prediction accuracy for both web accesses and web non-accesses achieved by each model. Across all volume classes, the model that ignores all temporal and spatiotemporal features achieves drastically lower prediction accuracy. Its access prediction accuracy is lower than 5% for more than 60% of the users while the MART model leveraging all the features achieves more than 90% prediction accuracy for the same percentage of users. Similarly, the MART model that leverages all the features is able to always achieve higher non-accesses prediction accuracy. Note that the gap in prediction accuracies between the two models is significantly smaller for non-accesses predictions. This is expected because of the significantly higher number of web non-accesses events in each test file (*i.e.*, for every access epoch, even though no targeted URLs were accessed, a web non-access event is recorded for all targeted URLs).

### *4.1.4 Comparison to a Naïve Sequential Approach*

Figures 1 and 3 suggest that most web accesses are clustered in space and time – most accesses follow shortly after a first access. For this reason, a naïve approach that fetches all targeted URLs on a periodic basis would be very inefficient from a power consumption point of view. Instead, we consider a naïve approach that prefetches every targeted URL as soon as the user accesses one of the targeted URLs. This approach would avoid the need to build a web access model for every user. To evaluate its effectiveness, we apply it on all 8,000 web access logs in our data set. Similarly to our approach, we use the three different freshness thresholds to indicate the validity of the prefetched data. If a user accesses a targeted URL before the freshness threshold has expired, it does not trigger a new round of prefetches.

Figure 11 shows the web access prediction accuracy achieved by both the naïve sequential approach and MART across different volume classes and freshness thresholds. The MART approach significantly outperforms the naïve sequential approach. For about 80% of the users, the naïve sequential approach achieves a prediction accuracy of less than 20% (medium volume class) and 30% (high volume class). For the same percentage of users, the proposed access prediction model always achieves accuracy higher than 80%.

The difference in performance lies in the fact that a large fraction of users has only 1 or 2 targeted URLs that they constantly visit. For these users, the naïve approach always fails to prefetch the first access, thus missing approximately half of the targeted URL visits. On the other hand, the access prediction model leverages both spatial and temporal information to predict both *which* targeted URL will be accessed and *when*, thus providing for higher performance.

## 4.2 System Prototype Analysis

We have implemented our prefetching approach in a system prototype running on Windows Phone 7, and is currently being used daily by 10+ people in our research group. On the phone, we built a web browser application (shown in Figure 7) which monitors web accesses and constantly uploads the collected traces to a server running on Windows Azure. The server uses MART to compute the user's prediction model based on the user's logged data, and it periodically updates it as the user's web browsing history grows. The cloud server pushes the MART model to the phone, such that each time a user opens the phone web browser, the local model is invoked and targeted web content is prefetched. In the following, we use the phone prototype to provide an estimate on the power consumed by our approach and the access time gains achieved.

In order to achieve this, an accurate estimation of both the power consumption model and the radio's transmission/reception time for downloading different web pages is needed. We adopt the power state model of modern 3G radios that has already been studied and extensively measured in the literature [28, 13, 6]. As Figure 12 shows, there are 4 distinct states that the 3G radio can be in. First, the radio incurs an energy overhead every time it has to wake up from the sleep state (phase 1 in Figure 12). Then the actual transmission/reception of data, which is the most power consuming state, takes place (phase 2 in Figure 12). After completion of data exchange, the radio spends approximately 17 seconds in two states with different power profiles (phases 3 and 4 in Figure 12). When the radio starts exchanging data while in phases 3 and 4, no extra energy cost is introduced. Note, however, that with respect to energy, every data transmission incurs a very high energy overhead in terms of the startup cost and especially the 17 seconds tail effect of 3G radios (phase 1 and phases 3 and 4 in Figure 12 respectively). Currently, web accesses that take place more than 17 seconds far apart, will always incur this energy overhead. However, with web content prefetching multiple web pages are simultaneously downloaded at the beginning of each browser session incurring this energy overhead only once. As a result, successful predictions could even lead to reduced energy dissipation.

Besides the power consumed, to accurately estimate energy dissipation, we need to estimate the time it takes to download different web pages (phase 2 in Figure 12). We use lightweight pages such as facebook.com and cnn.com, whose size varies from 5 up to 40 kBytes, and heavyweight pages such as nytimes.com and timesofindia.com, whose size varies from 50 up to 240 kBytes to measure the time that the radio spends transmitting/receiving bytes. As Table 3 shows, fetching 5 lightweight pages of increasing size

| Batch size | Lightweight Avg time (ms) [Stdev] | Heavyweight Avg time (ms) [Stdev] |
|---|---|---|
| 1 | 110 [22] | 7734 [480] |
| 2 | 120 [21] | 23340 [3026] |
| 3 | 306 [27] | 44452 [1451] |
| 4 | 382 [19] | 51562 [2597] |
| 5 | 468 [33] | 76180 [1129] |

**Table 3: Time the 3G radio has to spend actively transmitting or receiving data for downloading lightweight and heavyweight web pages with a batch size varying from 1 to 5 pages. Measurements executed on a Samsung Focus running Windows Phone 7.**

in a batch takes less than half second, which is much less than the startup time of a regular phone web browser. This means that 5 lightweight web sites can be prefetched by the time the browser is ready for use. Fetching heavyweight sites can take much longer, but also in this case the prefetching approach can help hiding such delay. Assuming a user's targeted sites are all heavyweight pages, if the first accessed page cannot be prefetched by the time the phone browser has loaded, the remaining targets will be still prefetched ahead with large savings in access time. Assuming a user might spend 1 or 2 minutes on the first page, this gives plenty of time for fetching 5 or more heavyweight pages before the user actually requests access to them. Moreover, targeted web sits are fetched in order of MART access probability to minimize the access delay.

In the following we leverage the measurements in Table 3 and Figure 12 to estimate web access times and radio energy dissipation when replaying back user web access logs.

### 4.2.1 Web Access Time

We define web access time as the time that elapses between a user requesting to visit a web page and the web page being available on the mobile device. Note that web access time here refers to the radio download time (phase 2 in Figure 12) that might be delaying the access to the web page. In all of our experiments, we assume a browser loading time of 3 seconds. We estimated this is roughly the time it takes for the browser to load and the user to make a web page request. As a result, every radio communication delay that takes place within these first 3 seconds is not considered in the computation of web access time.

By replaying back the web access logs of all 8,000 users, we compute the overall web access times in the case of a state of the art browser and in the case of the proposed prefetching scheme. Figure 13 shows the CDF of the web access time improvement achieved by the prefetching scheme for both lightweight and heavyweight pages. In both cases, the proposed prefetching scheme is able to completely eliminate web access time for 30%, 60%, and 80% of the users depending on the different volume class. For all these users, the proposed model was able to prefetch the web pages the user wished to access within the past two minutes, enabling an instant browsing experience. Furthermore, more than 90% of the users across all volume classes, experienced a reduction in their web access times by more than 60%. Only in the case of heavyweight pages, there is less than 2% of users that might experience an increase of up to 20% in their web access times. This is the small set of users where the number of false positives might increase substantially (Figure 10). When many heavyweight pages are simultaneously downloaded, the 3G radio gets congested and the overall web page download might be delayed (the time to simultaneously download 5 heavyweight pages is higher than sequentially downloading the individual pages as shown in Table 3). To prevent

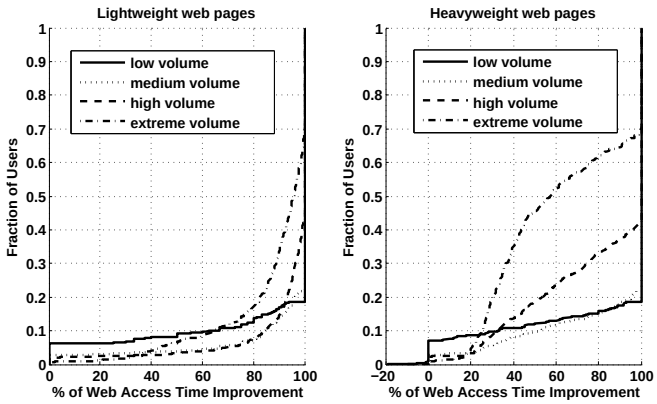


**Figure 13: CDF of the web access time (time that the user needs to wait for the radio to download a web page) improvement across all smartphone users when web access prediction is used. Negative values indicate an increase in the user response time. The trends are identical for featurephone users, and are not shown in the interest of space.**

the number of unsuccessful predictions from negatively impacting the web access time, the prefetching mechanism can monitor, over time, its own access prediction accuracy and accordingly disable any prefetching attempts when the number of false positives exceeds a daily threshold. In that way, even these users will not experience any noticeable performance degradation.

### 4.2.2 Radio Energy Dissipation

To quantify the effect of the proposed prefetching scheme on the power consumption of the mobile device, we also compute the total radio energy dissipation due to web accesses for all 8,000 users. Figure 14 shows the CDF of the radio energy improvement achieved by the prefetching scheme when assuming both lightweight and heavyweight pages are accessed by the users. Surprisingly, for anywhere between 75% and 98% of users across volume classes and web page types, the radio energy dissipation remains the same or is reduced when the web content prefetching approach is used. In other words, the proposed approach can enable instant web browsing experience for the majority of the users while maintaining the same or even significantly reducing radio energy dissipation. This is achieved because of the high energy cost that phases 1, 3, and 4 in Figure 12 introduce. When prefetching multiple web pages simultaneously the number of times that the radio needs to wake up from its sleep state and incur these energy costs can be drastically reduced. This can be clearly seen in Figure 15 where the CDF of the radio wake ups is shown when the prediction scheme is used or not used. By bundling web page downloads together, the prefetching scheme needs to wake up the radio significantly fewer times compared to state-of-the-art browsers, resulting in substantial energy savings.

On the other hand, only 2% to 18% of users in the case of lightweight pages, and 15% to 25% of the users in the case of heavyweight pages across different volume classes experience an increased radio energy dissipation. As expected, the impact on radio energy dissipation seems to be higher in the case of heavyweight pages since the false positives incur a higher energy penalty when compared to lightweight pages. For less than 2% of the users the radio energy dissipation can be increased by more than 150% to 200%. To prevent the number of unsuccessful predictions

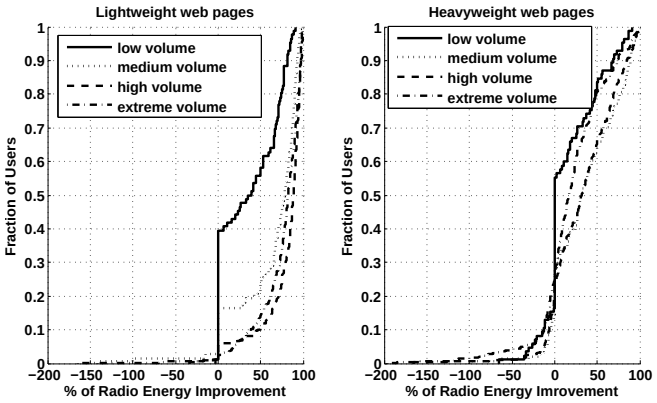


**Figure 14: CDF of the radio energy dissipation improvement across all smartphone users when the web access prediction is used. Negative values indicate an increase in radio's energy dissipation. The trends are identical across volume classes and device types, and are not shown in the interest of space.**

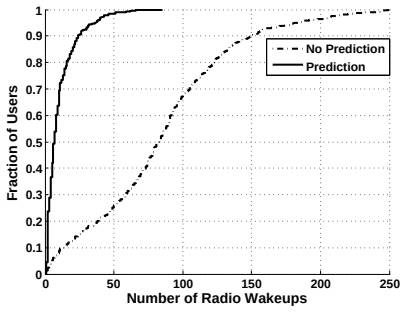


**Figure 15: CDF of the number of radio wake ups for high volume smartphone users when the proposed prediction scheme is used or not used. The trends are identical across volume classes and device types, and are not shown in the interest of space.**

from negatively impacting radio's energy dissipation, the prefetching mechanism can disable any prefetching attempts when the number of false positives exceeds a daily threshold.

# 5. RELATED WORK

Our work is inspired by Pocket Cloudlets [19]. In that work, the authors make the observation that memory capacity can be traded for access time and battery life in online services. It demonstrates the concept with a search cloudlet, where the most popular search queries and results among a large population are transfered to mobile phones at night, when they are connected to the network and charging. In addition, the search cloudlet also caches and ranks higher those pages clicked by the user. However, web content has a much more dynamic behavior than search results, and can be highly personalized. We address these challenges by carefully training an energy-thrifty, personalized web access prediction model capable of doing timely prefetches.

An important contribution of our work is the data analysis we conducted. There have been various studies [3, 7, 11, 26] on characterizing the workload of web clients (session duration, content popularity, size of web responses) with the aim of improving server performance through query caching, server scheduling and TCP channel management. The results of these studies cannot be directly used in our context because *i)* many of these studies are relatively old, *ii)* a large portion of them has focused on improving system performance in wired networks, and *iii)* we analyze web access behavior from a client perspective, rather than a server point of view, to build user-specific models for content prefetching.

Studies in the HCI community on web page revisitation [8, 29, 21, 10, 9] are also relevant to our work. The main goal of this body of research is browser and navigation tool design for usability. These studies monitor access behavior on desktop computers, and even more recent work has mostly focused on desktop revisitation patterns [24, 1, 2]. Our data analysis is instead focused on mobile web browsing behavior, which shows distinct access patterns and user intents. We particularly focus on studying how targeted web sites vary across mobile users and how temporal aspects such as absolute time of web accesses and inter-access times can be used in the prediction. Not surprisingly, there are lessons learned from the desktop world analysis that can be used in our context. For instance, Adar et al. [1] observe that revisitation rates for desktops range between 50% and 80% [1], and we found a similar pattern for smartphones. They also study the temporal behavior of revisits to the same page over a large number of websites and correlate the revist time to the type of web content (*e.g.*, fast revisits are common in web pages such as shopping, spam, pornography, while slow revisits are common for pages related to weekend activity, software updates, etc.). A feature related to the category of targeted web sites could be added to our model. Overall, all these studies suggest their findings can be applied to caching and prefetching in browsers, but have not addressed nor implemented and evaluated the problem of effectively modeling the web browsing behavior on a per-user basis.

Web content prefetching has been studied extensively in the context of desktop browsing [17, 20, 23, 25]. However, these approaches are not directly applicable to the mobile world because they do not take into account the constraints of latency and energy consumption imposed by mobile devices. In the context of mobile devices, prefetching has been used to support disconnected operation [18, 33], or to reduce access latency and power consumption [6, 33, 4]. Specifically, to minimize energy consumption when prefetching web content, the content is prefetched more aggressively when the available bandwidth changes to higher rates [6] or content with high access probability, low update rate, a small data size, and a high retrieval delay is fetched with higher priority [33]. Other approaches to reducing power consumption use proxies to push new content to the mobile client only when the portion of the web page of interest to the user is updated, and make use of batch updates [4]. These techniques are orthogonal to our work, in the sense that they look at specific optimizations which could be integrated into our system to reduce power consumption. Our primary goal is not to minimize energy consumption, but proposing a web content prefetching model which respects the power constraints of mobile devices.

There exists a number of techniques for predicting a user's web accesses. Most previous work is based on two assumptions: first the users' behavior can be captured *only* by sequential or set patterns and, second, *all* users show similar behaviors. Hence, most work uses variations of the Markov model [27], *i.e.*, they attempt to determine the probability of a user accessing web content based on previously accessed content. Dependency graph algorithms consider only first-order dependencies [16], while more sophisticated approaches, such as prediction-by-partial match, use higher-order dependencies [23]. Yet, these algorithms are trained with the set of accessed content units only, and do not take the temporal structure of web accesses into account. In our work, we do not focus only on sequential or set patterns, but we also use additional features such

as the relative and absolute timing of web accesses, which allow us to achieve much higher prefetching accuracy. Moreover, we show that each user's web access patterns are different, and we address this variability by building personalized models.

Since web content is highly dynamic, timeliness is an important aspect in prefetching it [4, 30, 31]. Most previous approaches that considered timeliness of content delivery rely on subscription-based prefetching. Subscription-based prefetching assumes that the user manually subscribes to content of interest, which is updated at a proxy or the client itself whenever it is modified, and the network and/or battery conditions are favorable [30, 31]. Our work does not rely on the user to provide any information about interests or to create subscriptions. Some approaches operate in a link-based fashion and prefetch content that is linked from the content currently being viewed by the user [12]. Although automatic, these approaches may result in high bandwidth consumption and untimely requests, especially when the currently viewed content has a lot of outgoing links, thus it is not suitable for mobile devices.

Another line of work relies on community-based profiles for content access prediction [5, 6, 17, 23, 25]. Communities is an interesting dimension to prefetching, but in mobile web browsing, the community data is not very helpful due to the high variability of web browsing behavior across users.

# 6. CONCLUSIONS

Trading surplus memory capacity for lower latency and battery life is desirable. Prior work was only capable of taking advantage of this surplus memory for relatively static content. This work provides the missing piece that enables mobile devices to handle dynamic content. We presented a study of web access patterns of 8,000 users over a 3-month period. We found that users do not browse the web randomly from their phones. Instead, we found a strong spatiotemporal signature in mobile web browsing. Based on these observations and the fact that each user behaves differently, we presented a framework for extracting a web access prediction model for each *individual* user. By predicting users' web accesses, we can provide a faster web browsing experience, with the same or lower radio energy dissipation. Our experimental evaluation on real datasets showed that for about 80-90% of the users we can accurately prefetch 60% of the URLs within 2 minutes before the request. Our results demonstrate the great potential of the proposed methodology in empowering users to enjoy an instant mobile web browsing experience.

# 7. REFERENCES


[1] E. Adar, J. Teevan, and S. T. Dumais. Large scale analysis of web revisitation patterns. In *Proc. of CHI*, 2008.

[2] E. Adar, J. Teevan, and S. T. Dumais. Resonance on the web: web dynamics and revisitation patterns. In *Proc. of CHI*, pages 1381–1390, 2009.

[3] A. Adya, P. Bahl, and L. Qiu. Analyzing the browse patterns of mobile clients. In *Proc. of SIGCOMM Workshop on Internet Measurement*, 2001.

[4] T. Armstrong, O. Trescases, C. Amza, and E. de Lara. Efficient and transparent dynamic content updates for mobile clients. In *Proc. of MobiSys*, 2006.

[5] A. Balasubramanian, B. Levine, and A. Venkataramani. Enhancing interactive web applications in hybrid networks. In *Proc. of MobiCom*, 2008.

[6] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: a measurement study and implications for network applications. In *Proc. of IMC*, pages 280–293, 2009.

[7] P. Barford, A. Bestavros, A. D. Bradley, and M. Crovella. Changes in web client access patterns: Characteristics and caching implications. *World Wide Web*, 2(1–2):15–28, 1999.

[8] L. D. Catledge and J. E. Pitkow. Characterizing browsing strategies in the world-wide web. In *Proc. of the 3rd World-Wide Web conference on Technology, tools and applications*, pages 1065–1073, 1995.

[9] A. Cockburn, S. Greenberg, S. Jones, B. Mckenzie, and M. Moyle. Improving web page revisitation: Analysis, design and evaluation. *IT and Society J.*, 1:159–183, 2003.

[10] A. Cockburn and B. McKensie. What do web users do? an empirical analysis of web use. *Int. J. Hum.-Comput. Stud.*, 54:903–922, June 2001.

[11] C. Cunha, A. Bestavros, and M. Crovella. Characteristics of www client-based traces. Technical Report TR-95-010, 1995.

[12] D. Duchamp. Prefetching hyperlinks. In *Proc. of USENIX Symp. on Internet Technologies and Systems*, 1999.

[13] H. Falaki, D. Lymberopoulos, R. Mahajan, S. Kandula, and D. Estrin. A first look at traffic on smartphones. In *Proc. of IMC*, pages 281–287, 2010.

[14] J. H. Friedman. Stochastic gradient boosting. *Comput. Stat. Data Anal.*, 38(4):367–378, 2002.

[15] ITRS Working Group. International technology roadmap for semiconductors 2009 report. Technical report, 2009.

[16] Z. Jiang and L. Kleinrock. Web prefetching in a mobile environment. *IEEE Personal Communications*, 5(5), 1998.

[17] F. Khalil, J. Li, and H. Wang. Integrating recommendation models for improved web page prediction accuracy. In *Australasian Conference on Computer Science*, 2008.

[18] A. Komninos and M. Dunlop. A calendar based internet content pre-caching agent for small computing devices. *J. of Personal and Ubiquitous Computing*, 12(7), 2008.

[19] E. Koukoumidis, D. Lymberopoulos, K. Strauss, J. Liu, and D. Burger. Pocket cloudlets. In *Proc. of ASPLOS*, 2011.

[20] E. P. Markatos and C. E. Chronaki. A top-10 approach to prefetching on the web. In *Proc. of INET*, 1998.

[21] B. McKenzie and A. Cockburn. An empirical analysis of web page revisitation. In *Proc. of HICSS*, volume 5, 2001.

[22] Mongoose Metrics. Mobile Devices Surpass Desktop Web Browsing in Five to Ten Years, 2010. http://www.mongoosemetrics.com/press_05012010.php.

[23] A. Nanopoulos, D. Katsaros, and Y. Manolopoulos. A data mining algorithm for generalized web prefetching. *IEEE Trans. on Knowledge and Data Engineering*, 2003.

[24] H. Obendorf, H. Weinreich, E. Herder, and M. Mayer. Web page revisitation revisited: Implications of a long-term click-stream study of browser usage. In *Proc. of CHI*, 2007.

[25] V. N. Padmanabhan and J. C. Mogul. Using predictive prefetching to improve world wide web latency. *SIGCOMM Comput. Commun. Rev.*, 26(3), 1996.

[26] V. N. Padmanabhan and L. Qiu. The content and access dynamics of a busy web site: findings and implications. *SIGCOMM Comput. Commun. Rev.*, 30:111–123, 2000.

[27] J. Pitkow and P. Pirolli. Mining longest repeating subsequences to predict world wide web surfing. In *Proc. of USENIX*, pages 139–150, 1999.

[28] F. Qian, Z. Wang, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. Characterizing radio resource allocation for 3g networks. In *Proc. of IMC*, pages 137–150, 2010.

[29] L. Tauscher and S. Greenberg. How people revisit web pages: empirical findings and implications for the design of history systems. *Int. J. Hum.-Comp. St.*, 47:97–137, 1997.

[30] A. Thawani, S. Gopalan, V. Sridhar, and K. Ramamritham. Context-aware timely information delivery in mobile environments. *The Computer Journal*, 50(4), 2007.

[31] E. Vartiainen, V. Roto, and A. Popescu. Auto-update: a concept for automatic downloading of web content to a mobile device. In *Proc. of Mobility*, 2007.

[32] Q. Wu, C. J. C. Burges, K. M. Svore, and J. Gao. Ranking, boosting, and model adaptation. Technical Report MMSR-TR-2008-109, Microsoft Research, 2008.

[33] L. Yin and G. Cao. Adaptive power-aware prefetch in wireless networks. *IEEE Trans. on Wireless Communications*, 3(5), 2004.