# Furion: Towards Energy-Efficient WiFi Offloading under Link Dynamics

Yi Zhang*†, Jiliang Wang †, Yuan He †, Xiaoyu Ji *, Yanrong Kang *, Daibo Liu ‡, and Bo Li *
*Department of Computer Science and Engineering, Hong Kong University of Science and Technology
†School of Software and TNLIST, Tsinghua University
‡University of Electronic Science and Technology of China
Email: {yzhangbh,xji,ykangaa,bli}@cse.ust.hk, {jiliangwang,heyuan}@tsinghua.edu.cn, dbliu.sky@gmail.com

***Abstract*—Offloading network traffic from cellular to WiFi is widely used to reduce energy consumption since WiFi is assumed to have lower power consumption than cellular. However, we find that WiFi link quality may vary significantly under user mobility. Consequently, the energy efficiency of WiFi varies and sometimes becomes even worse than that of cellular. Therefore, widely used WiFi offloading may not be beneficial or even incurs more energy consumption. To address this issue, we propose *Furion*, an energy efficient WiFi offloading scheme that exploits beneficial WiFi links on smartphones. Towards such a goal, we investigate the relationship between energy efficiency and link quality. Accordingly, we propose a practical probabilistic model to predict WiFi energy efficiency based on the dynamics of link quality. We further extend the method to different environments by exploiting contextual factors in the prediction model to improve the accuracy. Based on the model, we design an adaptive offloading scheme to optimize the energy efficiency of WiFi offloading, while also guaranteeing user experience. We have implemented *Furion* on the Android platform and conduct extensive real-world experiments. The results demonstrate that Furion achieves 34.13% improvement in energy efficiency compared with the state-of-the-arts.**

***Index Terms*—energy-efficient offloading; context-aware decision; cross-layer system; mobile environment**

## I. Introduction

Driven by the proliferation of smartphones, user's demand for data transmission is rapidly growing recently. As reported in [1], over 60% of users spend at least 63% of the time accessing network on smartphones. As a typical and commonly-used network type, cellular network, however, incurs high energy cost [2] and thus leads to a limited battery life. Therefore, reducing energy consumption is urgently needed to prolong the battery life and improve user experience.

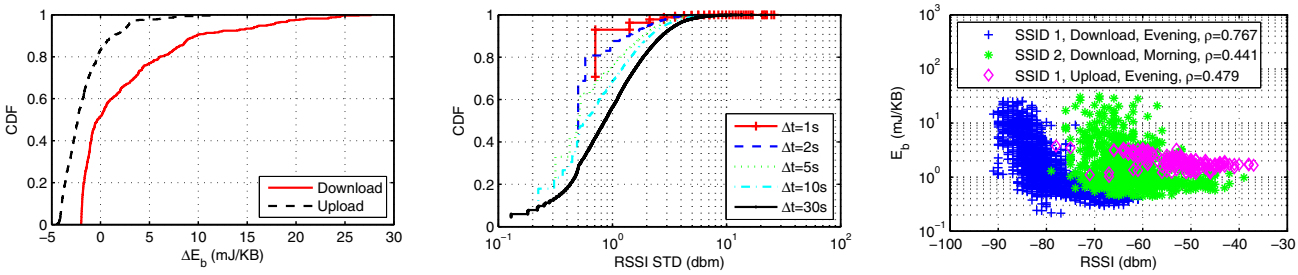
To this end, offloading traffic from cellular to available WiFi is deemed as a viable solution in many existing works [3]–[6] and also implemented as a default setting on smartphones nowadays. Compared with cellular network, WiFi access points are easy to find [5]. In addition, WiFi usually provides higher throughput and lower energy cost for the same amount of network traffic [2]. Accordingly, researchers propose various approaches to exploit the above advantages of WiFi offloading, like prefetching [5], delayed data transfer [4], [7] and collaborative network access [3], [8].

In spite of the numerous efforts on promoting WiFi offloading, whether the WiFi connections are reasonably and efficiently utilized remains an open question in the community. Many existing approaches simply presume that WiFi is always more energy efficient than cellular networks and therefore offload as much traffic to WiFi as possible. Unfortunately such an assumption is not always true. The energy efficiency of WiFi links is prone to multiple factors (e.g. channel fading, multi-path effect, etc.) and fluctuates often, especially in the mobile contexts [1], [9]. Through tests in a $2000m^2$ office area with multiple APs, we find that in 48.4% download cases and 16.6% upload cases, the energy efficiency of WiFi is lower than that of cellular network, e.g., 4G HSPA+. If the variation of link quality is not taken into account in those cases, simply offloading traffic to WiFi even incurs much more energy cost than expected.

Based on the above facts, a more appropriate scheme is to offload traffic to WiFi only when it is beneficial. Such a scheme, however, encounters several challenges. First, existing approaches often rely on data transmission over the links for link quality estimation [1], [9], [10]. However, this is inapplicable in the scenario of offloading because link quality is required to be estimated before data transmission occurs. Second, even link quality can be well estimated, it is difficult to accurately determine the corresponding energy efficiency. In different environments, e.g., different user mobility and traffic patterns, energy efficiency is not only related to link quality but also to other factors [2], [9], [11]. Last but not least, the offloading scheme should be carefully designed to optimize energy efficiency, while guaranteeing the user experience and application-level requirements, e.g., deadline of data transmission. Purely using the links with the highest energy efficiency may not be a well-considered solution.

In order to address those challenges, in this paper we propose *Furion*, an energy-efficient WiFi offloading scheme on smartphones. The key of *Furion* is to exploit the dynamics of link quality and contextual information to accurately and efficiently predict the energy efficiency of data transmission. Enlightened by the findings in [1], we find the wireless signal strength is a practical and efficient indicator on link energy efficiency. By investigating the fine-grained signal strength vari-

978-1-5090-1732-4/16/$31.00 ©2016 IEEE




(a) CDF of $\Delta E_b$ for download/upload. (b) CDF of RSSI standard deviation (STD) (c) Contextual discrimination on RSSI and $E_b$

Fig. 1: Preliminary experimental results. Fig.1a demonstrates the experimental results of WiFi energy efficiency tested in the office building. Fig.1b shows the "state" property of WiFi link dynamics. Fig.1c illustrates the improvement of contextual discrimination on the correlation between wireless signal strength and energy efficiency.

ations under user mobility, we translate continuous changes of energy efficiency of WiFi links into transitions of discrete states and further characterize them by a probabilistic model. To improve the prediction accuracy and extend the model to different environments, we incorporate the ability of contextual analysis into the probabilistic model. Accordingly, energy efficiency of WiFi links can be accurately predicted even without actual data transmission over them. Based on the model, we design an adaptive offloading scheme to exploit transmission time slots with highest energy efficiency, while taking into account the traffic patterns and the deadline requirements. We have implemented *Furion* on the Android platform and evaluate it on a variety of applications. The real world experiments demonstrate that *Furion* can achieve promising energy efficiency improvement compared with existing works.

Our contributions are summarized as follows:

- By fine-grained field tests, we validate WiFi link energy efficiency variation and quantitatively analyze its impact. We demonstrate that WiFi offloading is not always more energy efficient than cellular network, especially under mobile contexts.
- We show the practical challenges in predicting energy efficiency of WiFi offloading and design *Furion* to tackle those challenges. *Furion* utilizes a probabilistic model to predict link quality variation with efficient link quality indicator and contextual information. It leverages user contexts to discriminate impacting factors for accurate estimation of energy efficiency.
- We have implemented *Furion* on the Android platform and conduct extensive experiments to evaluate the performance of *Furion*. The results show that *Furion* saves in-average 34.13% more energy compared with the state-of-the-arts.

The rest of the paper is organized as follows: we first present the motivation and findings in Section II. Based on those findings, we describe the design of *Furion* in Section III. We show the implementation of *Furion* and evaluate its performance in real world experiments in Section IV. We discuss the related work in Section V and conclude our work in Section VI.

## II. Motivation

Revealed by many existing works [1], [2], modern cellular networks have a high power consumption. To save energy on cellular communication, research community propose that some traffic can be offloaded through WiFi to reduce their energy cost [3]–[5], [12].

### A. WiFi Link Variation

However, WiFi links are not always energy efficient. As pointed in many measurements [1], [9], WiFi link quality varies greatly in different environments. This variation, as a result, has a significant impact on the energy efficiency of data transmission over WiFi.

To quantify the impact of link quality variation on WiFi energy efficiency, we conduct comparison experiments in a $2000m^2$ office building with multiple APs. We use Samsung Galaxy Note3 as the testing platform and compare $E_b$ of WiFi with a cellular network in 52 different locations. We choose two modern network candidates for test, i.e., 802.11n for WiFi and 4G HSPA+ for the cellular network. At each testing point, we send/receive TCP packets to/from a remote server for 10 minutes.

We use $\Delta E_b = E_b(WiFi) - E_b(HSPA+)$ to compare energy efficiency at each location and plot the experimental results in Fig.1a. The results show that $E_b$ of 48.4% downloaded traffic and 16.6% uploaded traffic through WiFi are larger than that of HSPA+. The max $\Delta E_b$ in those scenarios is 27.7mJ/KB for download while 11.9mJ/KB for upload. The results clearly indicate that WiFi offloading without considering link quality may incur much more energy cost than expected. Unfortunately, such a phenomenon is ignored by most of the existing works [5], [13].

### B. Correlation between Signal Strength and Energy Efficiency

Pursuing high-quality WiFi links under different user mobility is difficult. Existing methods often rely on data transmission over a connected WiFi link to determine the energy

efficiency. Obviously, this method is not applicable for practical WiFi offloading as it needs to determine link quality before using it. Other works assume the link quality can be obtained directly [3], [14], which is also not practical in real world. Enlightened by works in [1], [15], we find that WiFi signal strength, indicated by RSSI on smartphones, is an efficient and practical link quality indicator on smartphones. On one hand, the results in [1], [15] demonstrate that $E_b$ for both uploaded and downloaded traffic decreases sharply with the increase of RSSI, indicating a strong correlation between them. On the other hand, RSSI is also a practical link parameter that can be easily obtained for WiFI APs without data transmission. Obtaining real-time RSSI incurs ignorable energy overhead [16]. It does not require specific hardware support like other metrics, e.g., Channel State Information (CSI). As a result, we propose to use RSSI as a link quality indicator to estimate the variation of energy efficiency.

### C. Modeling link dynamics

We find that within small time scales, RSSI *locality* is more significant than its *dynamics*. We plot the standard deviation (STD) of RSSI in different time intervals for all traces in Fig.1b. Interestingly, we observe that the majority of STD is below 2dbm. Even when the time interval extends to 30 seconds, there are still 81.2% of total traces with STD below 2dbm. By further examination, we find this observation is due to various reasons like physical constraints [3], e.g., users' temporal moving range is often small. Thus the properties of WiFi links are unlikely to change significantly in a short time [17]. Based on this observation, we can convert RSSI dynamics into transitions between discrete *RSSI states* in different time scales. If the transition rules between them can be estimated and updated timely, RSSI estimates can be efficiently predicted.

### D. Context-aware Energy Efficiency Estimation

Although there exists a correlation between RSSI and energy efficiency, quantifying the relationship under different conditions is also difficult. This is because the energy efficiency of WiFi can be also affected by other factors, like interference from other WiFi transmitters [18], the bandwidth of the wired link behind the AP [9] and etc. Due to the hardware limitation of smartphones, those impacting factors are nontrivial to be quantified directly. Instead, we further investigate the data and find that the impact of those factors can be indirectly discriminated by using fine-grained contextual information from built-in sensors on smartphones. For example, we can distinguish different AP loads by comparing throughput at different time, e.g., morning and evening. In addition, we can also estimate the outgoing bandwidth by comparing throughput in different BSSIDs. By combining contextual information, the correlation between RSSI and throughput would be more significant, as shown in Fig.1c. Compared with $\rho = 0.083$ in raw traces, the correlations between RSSI and energy efficiency are greatly increased, i.e., 0.767, 0.441 and 0.479, by using just three contexts. This finding motivates us that RSSI-based WiFi energy efficiency estimation under different environments can be improved by leveraging contextual information.

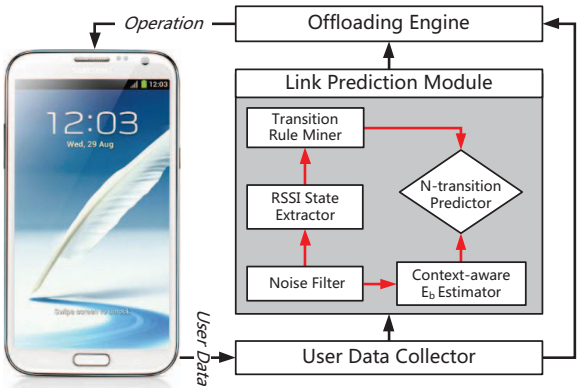



Fig. 2: The architecture of *Furion*

## III. Design of *Furion*

Based on the above mentioned findings, we propose *Furion*, a fine-grained scheme to achieve energy efficient WiFi offloading. In this section, we first present the overview of *Furion*. Then we describe each part of the design in detail.

### A. Overview

Since the energy efficiency varies in different scenarios such as user movement, the goal of *Furion* is to schedule the offloading traffic according to the energy efficiency while satisfying the deadline requirement. As shown in Fig.2, *Furion* mainly contains three components:

- **User Data Collector**: it collects and stores user data to facilitate the link prediction module and the offloading engine.
- **Link Prediction Module**: it predicts the link quality based on the information provided by the user data collector.
- **Offloading Engine**: it determines offloading schemes based on the information provided by the user data collector and the link prediction module.

In the following subsections, we will describe the design details for each of them.

### B. User Data Collector

The function of this part is recording and storing user data. It collects BSSID, time, traffic information, RSSI, acceleration, App information, and user preference. Those data are highly related to WiFi energy efficiency (BSSID, Time, Traffic pattern and RSSI) and further offloading scheme design (Acceleration, App information and Daily cellular data plan) [9], [16]. In addition, acquiring those contexts also incurs little cost [19].

### C. Link Prediction Module

This module has two goals: First, it uses information provided by the user data collector to model transition rules between *RSSI states*. Second, it derives a mapping function between *RSSI state* and energy efficiency to make real-time predictions.

As shown in Fig.2, raw traces are passed to a noise filter to filter out the noise. Then filtered traces are passed to RSSI state extractor and context-aware $E_b$ estimator for deriving *RSSI states* and RSSI-to-energy mapping respectively. Based on *RSSI states*, the transition rules are derived in the transition rule miner, which are used by N-transition predictor to make link predictions.

*1) Noise Filter:* Usually, RSSI samples collected by WiFi interface on smartphones contain "noise" due to reasons such as multipath effects [17]. The "noise" may incur errors in extracting *RSSI states* and needs to be filtered.

To preserve the RSSI locality while filtering the noise, we propose to use Nadaraya-Watson estimator [20] in the noise filter. Typically, it models the RSSI samples within a "window" as a series of random variables from a joint Probability Density Function (PDF):

$$r_i = m(i) + \epsilon_i, \quad (i = 1, ..., n) \tag{1}$$

where $r_i$ is RSSI at time $i$ and $m(\cdot)$ is a function for real data. The error $\{\epsilon_i\}$ satisfies [17]: $E(\epsilon_i) = 0, \quad V(\epsilon_i) = \sigma^2, \quad Cov(\epsilon_i, \epsilon_j) = 0 (i \neq j)$. The estimation $\hat{r_i}$ after filtering can be expressed as follows:

$$\hat{r_i} = \frac{\sum_{j=1}^{k} K_\lambda(i,j) r_j}{\sum_{i=1}^{k} K_\lambda(i,j)} \tag{2}$$

where $K_\lambda(i,j)$ is the kernel function for pairwise values $(i,j)$. Given RSSI *locality*, we use Radial Basis Function (RBF) kernel for $K_\lambda$ and the window size $\lambda$ is chosen adaptively to data distribution.

**Algorithm 1** PeakCount Algorithm

**Input:**
Filtered RSSI traces: $D$
**Output:**
*RSSI states* and their intervals;
1: $\{q_1, ..., q_k\} = PDF(D)$; //Derive peaks
2: **for** all data points $r_i \in D_p$ **do**
3: $\{\{Q_1, ..., Q_k\}, \{q_1, ..., q_k\}\} = KMeans(\{q_1, ..., q_k\}, r_i, r_{thr})$;
4: **end for**
5: **for** each $q_i \in \{q_1, ..., q_k\}$ **do**
6: $t_i = \frac{\sum_{j=1}^{k} N(C_i^j)}{k \cdot f_{sample}}, \quad \forall C_i^j \in Q_i$;
7: **end for**
8: **return** $\{q_1, ..., q_k\}$ and $\{t_1, ..., t_k\}$;

*2) RSSI State Extractor:* This part is responsible for extracting *RSSI states* from filtered traces. According to the finding of RSSI locality mentioned in Section II, we define *RSSI states* as follows: a *RSSI state* is a value for a set of continuous RSSI samples such that the STD of the distance of those samples to the state is less than a threshold $r_{thr}$. Specifically, the length of the RSSI samples is denoted as the interval for the *RSSI state*.

However, extracting *RSSI states* from RSSI samples is nontrivial. First, the number and values of *RSSI states* for the set of RSSI samples are usually unknown. Second, even if we know the exact values of *RSSI states*, clustering RSSI samples is also hard since there may be multiple feasible choices.

To address this issue, we design an efficient approximation algorithm, i.e. PeakCount, based on the PDF features of traces. According to the finding of RSSI locality mentioned in Section II, the density of RSSI samples should be high near a *RSSI state*. This leads to the peaks in the PDF of RSSI samples. By traversing along the PDF of the traces, the peaks can be obtained, i.e. $\{D_p\}$. Accordingly, we can use those peaks as the initial centroids and apply k-means to cluster RSSI samples. Based on the results, we use $q_i$ to denote the i-th *RSSI state*. Its time interval $t_i$ can be calculated as follows:

$$t_i = \frac{\sum_{j=1}^{k} N(C_i^j)}{k \cdot f_{sample}}, \tag{3}$$

where $N(C_i^j)$ is the number of samples in $j$th cluster for state $q_i$ and $f_{sample}$ is the sampling frequency. The details of PeakCount algorithm are shown in Algorithm.1.

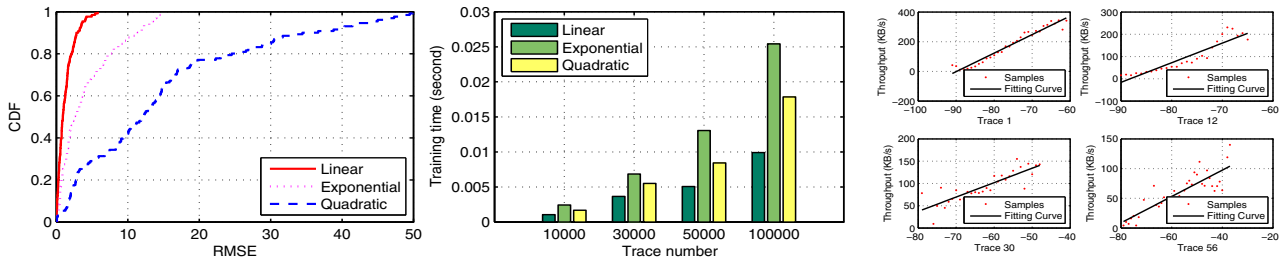
*3) Transition Rule Miner:* With state information provided by RSSI state extractor, we then derive the corresponding transition rules. There are different RSSI states at different locations [9], [21]. The probability of state transition between two states is relatively stable due to user mobility patterns and physical constraints. Thus we propose to use Markov chain to model the transitions among states. Markov chain is a generalized time series analyzer, which assumes no prior distribution on samples. In addition, the limited *RSSI state* space, which ranges from $-30$dbm to $-95$dbm, makes deriving transition matrix fast at runtime.

In the rule miner, deriving transition rules is equal to deriving transition matrix $\mathbf{P}$. Based on the *RSSI states*, the transition matrix $\mathbf{P} = [a_{ij}]^{m \times m}$ can be estimated as follows:

$$a_{i,j} = \frac{N(q_i, q_j)}{\sum_{\forall k} N(q_i, q_k)} \tag{4}$$

in which $N(q_i, q_j)$ is the number of instances that $q_j$ appears right after $q_i$. The transition matrix $\mathbf{P}$ will be passed to N-transition predictor to make real-time predictions.

*4) Context-aware $E_b$ estimator:* For each *RSSI state*, we need to derive the corresponding energy efficiency to determine the offloading scheme. However, achieving this goal is challenging due to different impacting factors in different environments. To address this issue, we leverage contextual information to discriminate external impacting factors. According to the measurements in [9], [16], we use the tuple {Time, BSSID, Traffic pattern} as the context information to efficiently differentiate the factors that impact WiFi energy efficiency. We will present the validation tests of those factors in Section IV. Apart from their effectiveness, acquiring those




(a) CDF of *RMSE* on Linear, Exponential and Quadratic regression models.

(b) Training time

(c) Linear curve fitting on four traces with different contexts

Fig. 3: The performance of different regression models.

contextual information also incurs low energy cost [19], which makes them applicable on smartphones. Notice that other factors can also be used in our model if they can be efficiently obtained and have significant impacts on energy efficiency.

Then, we determine the mapping function between *RSSI state* and energy efficiency by leveraging context tuple {Time, BSSID, Traffic pattern}. Based on the power equation in [2], it is equal to modeling the relationship between *RSSI state* and throughput $s$. Since there is no prior knowledge about the relation between RSSI and $s$, we thus resort to regression analysis to characterize the mapping function. We select three regression models for estimation as listed in Table.I. The performance for those three regression models is evaluated based on the *root mean square error*, i.e.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(s^* - s(q_i))^2} \tag{5}$$

where $s^*$ is the actual throughput and $s(q_i)$ is estimated throughput at $q_i$. A small RMSE indicates a tight fit of the model to the data. In addition, we should also consider the training time of the model. If the training time is too long, then it is still inapplicable on smartphones.

The tests of three models are conducted on the dataset [22] and the model parameters are selected to minimize *RMSE* value. We plot the CDF of RMSE and training time in Fig.3a and Fig.3b. In addition, we also list the mean values in Table.I. We find that mean RMSE of linear model is 1.27 which is the least one compared with 4.27 for exponential and 14.71 for quadratic. In addition, its training time is also the shortest on datasets with different traces. Fig.3c shows 4 random chosen traces and their fitting curve by the linear model. Despite of the diversity of contexts, the linear regression model fits well with all sample sets.

Based on the above result, the mapping function between *RSSI state* $q_i$ and energy efficiency $E_b$ can be expressed as:

$$E_b(q_i) = \frac{\beta}{m \cdot q_i + n} + \alpha \tag{6}$$

*5) N-transition Predictor:* This part makes link prediction based on the transition matrix **P** and linear regression model

TABLE I: Regression model test

| Model | Expression | RMSE[1] | Training[2] |
|---|---|---|---|
| Linear | $s(q_i) = m \cdot q_i + n$ | 1.27 | 0.197s |
| Exponential | $s(q_i) = m \cdot \exp(q_i) + n$ | 4.27 | 0.411s |
| Quadratic | $s(q_i) = m \cdot {q_i}^2 + n \cdot q_i + c$ | 14.71 | 0.337s |

[1] Mean RMSE value on all tests
[2] Mean training time on a laptop with i7-3610QM

derived from context-aware $E_b$ estimator. The main problem here is that we need to make link prediction before a certain deadline. The time to deadline may contain multiple *RSSI states*. Thus the link before the deadline cannot be predicted with a single transition.

To address this issue, we resort to the properties of **P**: if we denote $\mathbf{P}^n$ as the power n of **P**, then the entry $a_{ij}^{(n)}$ of the matrix $\mathbf{P}^n$ gives the probability that the Markov chain, starting at state $q_i$, will be in state $q_j$ after n transitions.

By obtaining $a_{ij}^{(n)}$, we can make predictions as follows: if a user is moving, within n transitions, the expectation time to meet the appropriate *RSSI states* can be calculated as [23]:

$$E(t) = \frac{1 - G(q_k)}{Z(n)} \sum_{i=1}^{n}\sum_{j=1}^{m} G(q_j) \cdot a_{kj}^{(i)} \cdot t_L^{kj}(i) \tag{7}$$

where $Z(n)$ is a normalization factor to ensure the sum of probabilities equals to 1. $G$ is a function indicating whether WiFi energy efficiency of $q_j$ is higher than that of cellular:

$$G(q_j) = \begin{cases} 1, & E_b(q_j) \le E_b^c \\ 0, & otherwise \end{cases} \tag{8}$$

where $E_b^c$ is the cellular energy efficiency. $t_L^{kj}(i)$ is the time interval of the most probable state transition sequence from starting state $q_k$ to state $q_j$ calculated. Without loss of generality, if no appropriate *RSSI states* exist within n transitions, we set $E(t) = \infty$ to avoid collision with $G(q_k) = 1$. Accordingly, the expectation of link capacity for $E(t) \neq \infty$ can also be calculated as follows:

$$C = \frac{1 - G(q_k)}{Z(n)} \sum_{i=1}^{n}\sum_{j=1}^{m} G(q_j) \cdot a_{kj}^{(i)} \cdot s(q_j) \cdot t_j + G(q_k) \cdot s(q_k) \cdot t_k \tag{9}$$

where $s(q_j)$ is the estimated throughput for $q_j$ by linear regression model and $t_j$ is the time interval for $q_j$. By default, we set $n = 3$ to limit the prediction error.

If a user is static, the link quality is also likely to remain stable. Then we have:

$$E(t) = \begin{cases} 0, & E_b(q_k) \leq E_b^c \\ \infty, & otherwise \end{cases} \quad (10)$$

Accordingly, we set $C = t_k \cdot s(q_k)$ to avoid overestimation. This component periodically obtains acceleration from user data collector to update the prediction for guaranteeing accuracy when user moves [16].

**Algorithm 2** The adaptive offloading algorithm

**Input:**
Prediction: $E(t)$, $C$
Current RSSI: $r_{now}$
User data budget: $P_{daily}$
App information: $\Delta t_{min}$, $D$

**if** $offload(A)$ equals to 0 **then**
  Determine $(t_{off}, N_{off})$ based on user preference;
**else**
  **if** $C \geq D$ **then**
    $(t_{off}, N_{off}) = (E(t), WiFi)$;
  **else**
    $\Delta t = \frac{D - E(C)}{s(r_{now})}$;
    $(t_{off}, N_{off}) = (E(t) - \Delta t, WiFi)$;
  **end if**
**end if**
EXEC_TRANSMIT($t_{off}$,$N_{off}$);

### D. Offloading Engine

The offloading engine determines the offloading scheme based on the information provided by user data collector and link prediction module.

To exploit links with high energy efficiency while satisfying user requirement, we design an adaptive offloading algorithm. The details are shown in Algorithm.2. First, it is required to determine the size and deadline of the upcoming data transmission. Without loss of generality, we denote the nearest deadline as $\Delta t_{min}$ based on information provided by user data collector. The deadline information can be obtained by mining historic usage pattern [22] or manually set by users. Bearing the deadline in mind, we can extract the total transmission size $D$ before current deadline as:

$$D = \sum_{\forall App_j \in A} d(App_j) \quad (11)$$

Then, the offloading engine checks whether there exists appropriate *RSSI states* before $\Delta t_{min}$, indicated by $offload(A)$:

$$offload(A) = \begin{cases} 1, & E(t) \leq \Delta t_{min} \\ 0, & otherwise \end{cases} \quad (12)$$

Based on $offload(A)$, the offloading scheme is determined as follows:

1: $offload(A) = 0$: it indicates no appropriate WiFi is available. The engine determines not to use WiFi offloading.

TABLE II: App Information

| App ID | App name | Description | Real-time | Mean $d$ |
|---|---|---|---|---|
| 1 | Browser | Web | High | Small |
| 2 | KingSoft Store | App Downloader | Low | Large |
| 3 | Renren | Social | Medium | Medium |
| 4 | QQ Music | Online Music | Medium | Medium |
| 5 | WeChat | Social | Medium | Medium |
| 6 | YY | Communication | Medium | Small |
| 7 | QQ | Communication | Medium | Small |
| 8 | Phoenix Times | News | Low | Small |
| 9 | Baidu App | App Downloader | Low | Large |
| 10 | Google Play | App Downloader | Low | Large |
| 11 | Mobile DOTA | Online Game | High | Small |
| 12 | Tencent News | News | Low | Small |
| 13 | Weibo | Web/Social | Medium | Medium |

Meanwhile, the user can also set his/her preference according to contextual information. For example, if a user has limited cellular budget (less than $D$) and prefers to use WiFi, the user can set such a preference in the engine.

2: $offload(A) = 1$, it indicates appropriate WiFi links are available before $E(t)$. Then the engine will determine the time for data transmission such that efficient links can be fully utilized and $A$ is finished before the deadline.

To cope with WiFi link dynamics, the above mentioned schemes are updated in real time according to the information provided by the user data collector and link prediction module.

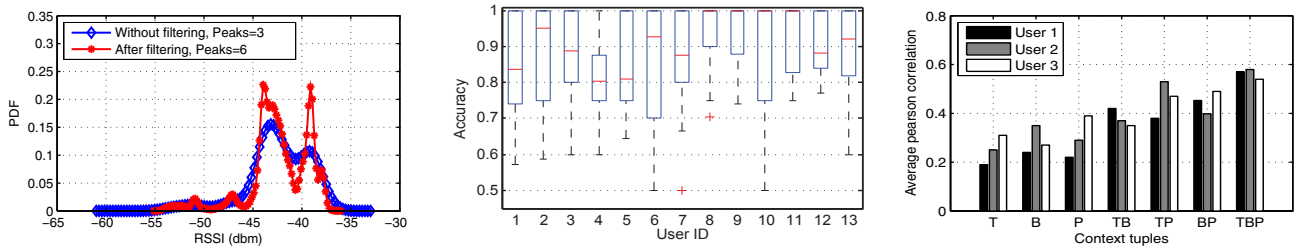
## IV. Evaluation

### A. Experimental settings

We have implemented *Furion* on three modern smartphones, i.e., Galaxy Note3, HTC One X and HTC M8. We enable PSM to decrease the sensing cost and eliminate energy consumed in idle listening. The operating system is Android as we need permissions to call kernel functions. We set the training frequency as three times a day to update the models. After training, we release precious storage by deleting used traces. Given the hardware difference between those platforms, we conduct independent comparison experiments on each platform. Considering the diversity of usage patterns, each experiment consists of 13 volunteers with the age between 20 and 70 and lasts for one week. By conducting off-line tests on user traces under different scenarios, we compare the performance of two existing approaches with *Furion*:

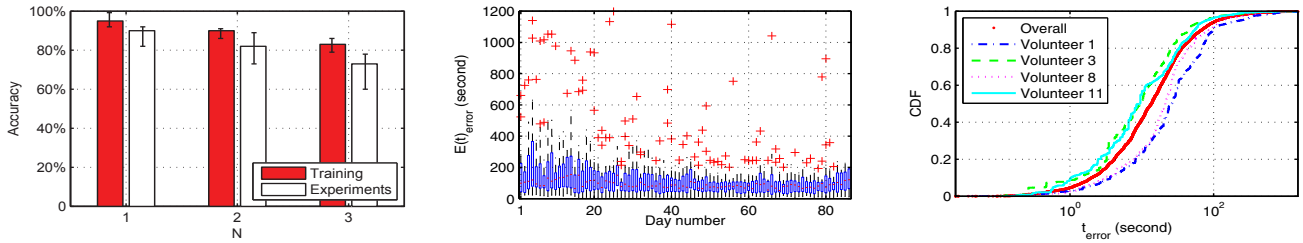
- *eTime* [14]: it utilizes a local greedy algorithm to determine the offloading decision, where the correlation between user mobility and link dynamics is not considered.
- *Wiffler* [5]: it is a best-effort WiFi offloading scheme. The data will be immediately transmitted if users are within WiFi coverage.

In addition, we also derive the optimal results, i.e. oracle, as the benchmark by using the global information recorded in each experiment. By default, we focus on comparing the




(a) The performance of noise filter (b) Accuracy of PeakCount algorithm (c) On context tuple selection

Fig. 4: Evaluation of link prediction module. Fig.4a illustrates the performance of noise filter. Fig.4b presents the accuracy of PeakCount algorithm. In Fig.4c, we use "T" for time, "B" for BSSID, and "P" for traffic pattern for simplification.




(a) State prediction accuracy (b) $E(t)_{error}$ evaluation (c) $t_{error}$ evaluation

Fig. 5: Evaluation of offloading engine. Specifically, $E(t)_{error}$ refers to the inter-state prediction error, while $t_{error}$ refers to the intra-state prediction error.

energy consumption of those mechanisms in data transmission. Through the experiments, we also find that most users only use a very small set of Apps. To obtain reliable experimental results, we focus on 13 most used Apps in different categories as listed in Table.II.

### *B. Evaluation of link prediction module*

First, we analyze the performance of noise filter. Using a user trace with 5 hour RSSI samples as an example, we show in Fig.4a that three more *RSSI states* can be derived by PeakCount algorithm after filtering the noise. This is because by filtering out the noise, the PDF peaks are more significant, considering the property of RSSI locality mentioned in Section II. Therefore, more fine-grained information about link dynamics can be obtained, which helps determine offloading schemes.

Then, we examine the accuracy of PeakCount algorithm on extracting RSSI states. In order to derive the ground truth, we conduct offline analysis through manual examination and significant difference check. Denoting $Q^*$ as the true *RSSI states* and $Q$ as the states derived by PeakCount, we define the accuracy of PeakCount algorithm as follows:

$$Accuracy = \frac{|Q^* \cap Q|}{|Q^*|} \tag{13}$$

As shown in Fig.4b, we observe the average accuracy is 87.59%. It indicates that PeakCount can guarantee high accuracy in extracting fine-grained RSSI states.

In addition, we also examine whether context tuple {Time, BSSID, Traffic pattern} can efficiently discriminate the external impacting factors. Since the relation between RSSI and energy efficiency can be characterized by a linear model, we use Pearson parameter to examine the correlation between energy efficieny and RSSI [16]. If the value is high, then it indicates that current context tuple is efficient. We plot the average Pearson correlation under different context tuples of three users in Fig.4c. The results show that the context tuple {Time, BSSID, Traffic pattern} gives over 0.5 average pearson correlation for all tested users, which is the highest among all testing tuples. As a result, utilizing {Time, BSSID, Traffic pattern} can efficiently eliminate the error in modeling the relation between RSSI and energy.

### *C. Evaluation of offloading engine*

In this subsection, we examine the performance of offloading engine, which is affected by two factors: link quality prediction accuracy and state interval estimation accuracy.

Link quality prediction accuracy determines whether delayed transmission can achieve high energy efficiency. First, we check whether appropriate RSSI states can be accurately captured, i.e., $E(t) \neq \infty$ if $q_j \in Q^*$ within N-transitions according to Eq.7. As shown in Fig.5a, we observe that within 3 state transitions, the average link quality prediction accuracy is high, with 89.33% for training and 81.67% in experiments. In addition, we also quantify the inter-state prediction error $E(t)_{error}$, which is the time interval between $E(t)$ and the actual time of meeting appropriate RSSI states. We plot the results of total 86 effective daily traces from 13 volunteers in Fig,5b. The figure shows that *Furion* can also achieve

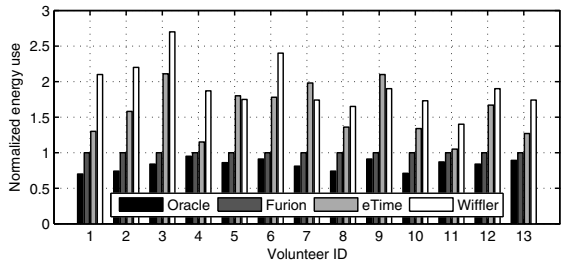



Fig. 6: Energy usage comparison for different volunteers.

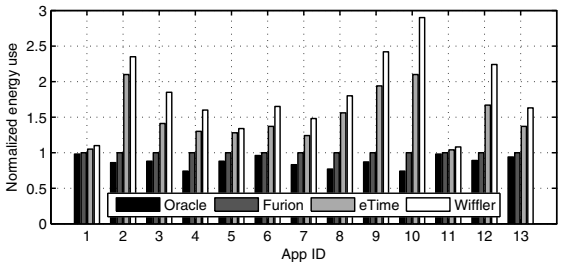
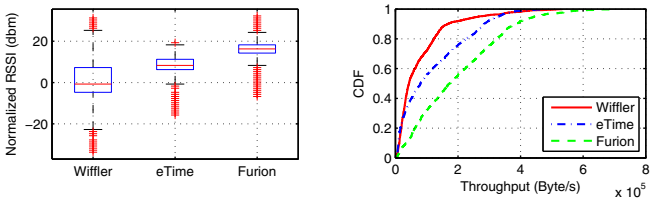



Fig. 7: Energy usage comparison on 13 most prevalent Apps.




(a) RSSI usage comparison (WiFi) (b) Throughput comparison

Fig. 8: Performance comparison of *Wiffler, eTime* and *Furion* on RSSI utilization and throughput.

competitive numerical results, with the average $E(t)_{error}$ below 180 seconds.

Then, we examine the accuracy of state interval estimation. Using $t_{error}$ to denote the difference between the estimated state interval $t_i$ and the actual value, we plot the results in Fig.5c. It shows that average $t_{error}$ of total traces is 56.78 seconds, which is often negligible compared with multiple minutes time interval of *RSSI states*. In addition, over 40% of $t_{error}$ can be efficiently limited below 30 seconds, which demonstrates the offloading engine is also able to accurately estimate link capacity for predicted RSSI states.

### D. Overall performance of Furion

In this subsection, we compare the performance of *Furion* in energy saving with other three mechanisms, i.e. eTime, Wiffler and oracle, under different scenarios.

First, we analyze the normalized energy use for each volunteer. We plot the results in Fig.6. Given the diversity of usage patterns among different users, *Furion* achieves 26.52% (2% to 47.9%) more energy saving compared with *eTime* while 37.8% (10.37% to 55.56%) compared with *Wiffler*. In addition, we also observe that the average gap between *Furion* and oracle is less than 17%. The results show that, by exploring fine-grained link properties, *Furion* can efficiently utilize WiFi links with good quality for data offloading. Thus, it can achieve near optimal energy consumption in WiFi offloading.

Second, we also analyze the performance of *Furion* on different Apps. Choosing 13 most commonly used Apps, we plot the normalized energy use in Fig.7. We observe that the energy efficiency improvement of *Furion* over other two mechanisms is also significant. In average, *Furion* can achieve 33.09% (1% to 52.4%) more energy saving compared with *eTime*, while 44.54% (7.41% to 65.52%) compared with *Wiffler*. The performance variation of *Furion* is mainly due to the various traffic and usage patterns for different Apps. For instance, the improvement is little for those Apps which have high real-time requirement and small transmission sizes, e.g., Mobile DOTA. While for those which are more delay tolerant and have large data sizes, e.g., Google Play, the energy saving is much more significant. With large deadlines, WiFi RSSI variation is more significant, e.g., may vary -90dbm to -30dbm, which provides more potential for implementing energy-efficient offloading [4]. In words, the energy efficiency improvement of *Furion* is significant compared with existing works, and also robust under different scenarios.

To further examine why *Furion* can achieve more energy saving than existing works, we also conduct fine-grained analysis on user traces. First, we compare the RSSI usage between *Furion* and the other two mechanisms in the same environments. As shown in Fig.8a, we find *Furion* improves RSSI experienced in data transmissions by 16.3dbm compared with *Wiffler*, while 9.7dbm compared with *eTime*. This is because the majority of Apps are not real-time, which provides a potential in rescheduling its data transmission. By predicting RSSI dynamics according to user mobility, *Furion* can accurately and efficiently find WiFi links with high energy efficiency before the deadline. Accordingly, we notice that by improving the utilization of WiFi links with good RSSI, the throughput in data transmission also increases, which is shown in Fig.8b. In average, the throughput in *Furion* is 152% higher than that in *Wiffler*, while 67% higher than that in *eTime*. The results imply that *Furion* can save a significant portion of transmission time for the same amount of data and thus, battery life can be extended to satisfy more network access.

## V. Related Work

Optimizing the energy consumption of communication on smartphones has been a research hotspot in recent years. In this section, we first introduce the related work in network energy measurement and then look into the WiFi offloading approach.

**Measurement.** The works in [1], [2], [9], [15], [24], [25]

conduct extensive experiments to analyze network usage on smartphones. Huang et al. [2] illustrate that energy efficiency variation is common among different network types. Ding et al. [1] collect usage data from 3785 volunteers and find that links with poor quality are common and severely affect the energy efficiency of different network types. Some fine-grained measurements are also made to analyze the factors that affect network energy efficiency [9], [15], [16]. The results from those measurements imply that there exists a great potential in optimizing network usage.

**WiFi offloading.** Offloading traffic to WiFi is recognized as one viable solution in many existing works [3]–[6], [10] and also implemented as the default setting on modern smartphones. To optimizing the energy efficiency of WiFi offloading, several approaches have been proposed [3], [5], [8]. Balasubramanian et al. [5] propose a trajectory-based WiFi offloading strategy by predicting the access availability. Ding et al. [3] propose a collaborative offloading scheme among cellular operators, WiFi service providers and end-users. Shu et al. [14] resort to queuing theory and design a local greedy scheme to achieve energy efficient WiFi offloading. However, most existing works ignore the correlation between WiFi link quality variation and user mobility. Therefore, the performance of their works is hard to be guaranteed in practice. To address this issue, we propose *Furion*, an energy-efficient WiFi offloading scheme by exploiting link dynamics with mobile contexts. *Furion* investigates the correlation between link dynamics and user mobility, and can efficiently and accurately predict the energy efficiency of WiFi links. Hence, the energy efficiency of WiFi offloading can be further improved by *Furion*, compared with existing works.

## VI. Conclusion

WiFi offloading provides great potential in satisfying ubiquitous applications on smartphones. However, by investigation, existing works fall short in achieving energy efficient WiFi offloading due to the unawareness of the variation of link quality. In this paper, we propose *Furion*, an energy-efficient scheme to achieve energy efficient WiFi offloading based on accurate link quality prediction and efficient energy efficiency estimation under different scenarios. Extensive evaluation shows that *Furion* can achieve 34.13% improvement in energy efficiency, compared with the state-of-the-arts.

## Acknowledgements

The research was support in part by grants from RGC under the contracts 615613, 16211715 and C7036-15G (CRF), a grant from NSF (China) under the contract U1301253, NSFC under Grants 61572277, 61532012, 61529202, and National Science Fund for Excellent Young Scientist No. 61422207.

## References

[1] N. Ding, D. Wagner, X. Chen, Y. C. Hu, and A. Rice, "Characterizing and modeling the impact of wireless signal strength on smartphone battery drain," in *Proceedings of ACM SIGMETRICS*, 2013, pp. 29–40.

[2] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, "A close examination of performance and power characteristics of 4g lte networks," in *Proceedings of ACM MobiSys*, 2012, pp. 225–238.

[3] A. Y. Ding, B. Han, Y. Xiao, P. Hui, A. Srinivasan, M. Kojo, and S. Tarkoma, "Enabling energy-aware collaborative mobile data offloading for smartphones," in *Proceedings of IEEE SECON*, 2013, pp. 487–495.

[4] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong, "Mobile data offloading: how much can wifi deliver?" in *Proceedings of ACM CoNEXT*, 2010, pp. 26–37.

[5] A. Balasubramanian, R. Mahajan, and A. Venkataramani, "Augmenting mobile 3g using wifi," in *Proceedings of ACM MobiSys*, 2010, pp. 309–322.

[6] M. V. Barbera, S. Kosta, A. Mei, and J. Stefa, "To offload or not to offload? the bandwidth and energy costs of mobile cloud computing," in *Proceedings of IEEE INFOCOM*, 2013, pp. 1285–1293.

[7] J. Huang, F. Qian, Z. M. Mao, S. Sen, and O. Spatscheck, "Screen-off traffic characterization and optimization in 3g/4g networks," in *Proceedings of ACM IMC*, 2012, pp. 357–364.

[8] J. Manweiler and R. Roy Choudhury, "Avoiding the rush hours: Wifi energy management via traffic isolation," in *Proceedings of ACM MobiSys*, 2011, pp. 253–266.

[9] A. Patro, S. Govindan, and S. Banerjee, "Observing home wireless experience through wifi aps," in *Proceedings of ACM MobiCom*, 2013, pp. 339–350.

[10] Y. Zhang, J. Wang, Y. He, Y. Kang, B. Li, and Y. Liu, "Q-offload: Quality aware wifi offloading with link dynamics," in *Proceedings of IEEE Real-Time Systems Symposium (RTSS)*, 2015, pp. 239–248.

[11] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Predictable 802.11 packet delivery from wireless channel measurements," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 159–170, 2011.

[12] K. Fahad R. Dogar, Peter Steenkiste, "Catnap: Exploiting high bandwidth wireless interfaces to save energy for mobile devices," in *Proceedings of ACM MobiSys*, 2010.

[13] N. Ristanovic, J.-Y. Le Boudec, A. Chaintreau, and V. Erramilli, "Energy efficient offloading of 3g networks," in *Proceedings of IEEE MASS*, 2011, pp. 202–211.

[14] P. Shu, F. Liu, H. Jin, M. Chen, F. Wen, and Y. Qu, "etime: energy-efficient transmission between cloud and mobile devices," in *Proceedings of IEEE INFOCOM*, 2013, pp. 195–199.

[15] A. Schulman, V. Navda, R. Ramjee, N. Spring, P. Deshpande, C. Grunewald, K. Jain, and V. N. Padmanabhan, "Bartendr: a practical approach to energy-aware cellular data scheduling," in *Proceedings of ACM MobiCom*, 2010, pp. 85–96.

[16] A. Rahmati and L. Zhong, "Context-for-wireless: context-sensitive energy-efficient wireless data transfer," in *Proceedings of ACM MobiSys*, 2007, pp. 165–178.

[17] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge university press, 2005.

[18] J. Huang, G. Xing, G. Zhou, and R. Zhou, "Beyond co-existence: Exploiting wifi white space for zigbee performance assurance," in *Proceedings of IEEE ICNP*, 2010, pp. 305–314.

[19] S. Nath, "Ace: exploiting correlation for energy-efficient and continuous context sensing," in *Proceedings of ACM MobiSys*, 2012, pp. 29–42.

[20] E. A. Nadaraya, "On estimating regression," *Theory of Probability & Its Applications*, vol. 9, no. 1, pp. 141–142, 1964.

[21] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: wireless indoor localization with little human intervention," in *Proceedings of ACM MobiCom*, 2012, pp. 269–280.

[22] Y. Zhang, Y. He, X. Wu, Y. Liu, and W. He, "Netmaster: Taming energy devourers on smartphones," in *Proceedings of IEEE ICPP*, 2014, pp. 301–310.

[23] R. R. Sarukkai, "Link prediction and path analysis using markov chains," *Computer Networks*, vol. 33, no. 1, pp. 377–386, 2000.

[24] P. Deshpande, X. Hou, and S. R. Das, "Performance comparison of 3g and metro-scale wifi for vehicular network access," in *Proceedings of ACM IMC*, 2010, pp. 301–307.

[25] C. Yoon, D. Kim, W. Jung, C. Kang, and H. Cha, "Appscope: Application energy metering framework for android smartphone using kernel activity monitoring." in *USENIX Annual Technical Conference*, 2012, pp. 387–400.