Establishing Smartphone User Behavior Model Based on Energy Consumption Data

MING DING, TIANYU WANG, and XUDONG WANG, Shanghai Jiao Tong University

In smartphone data analysis, both energy consumption modeling and user behavior mining have been explored extensively, but the relationship between energy consumption and user behavior has been rarely studied. Such a relationship is explored over large-scale users in this article. Based on energy consumption data, where each users' feature vector is represented by energy breakdown on hardware components of different apps, User Behavior Models (UBM) are established to capture user behavior patterns (i.e., app preference, usage time). The challenge lies in the high diversity of user behaviors (i.e., massive apps and usage ways), which leads to high dimension and dispersion of data. To overcome the challenge, three mechanisms are designed. First, to reduce the dimension, apps are ranked with the top ones identified as typical apps to represent all. Second, the dispersion is reduced by scaling each users' feature vector with typical apps to unit ℓ_1 norm. The scaled vector becomes Usage Pattern, while the ℓ_1 norm of vector before scaling is treated as Usage Intensity. Third, the usage pattern is analyzed with a two-layer clustering approach to further reduce data dispersion. In the upper layer, each typical app is studied across its users with respect to hardware components to identify Typical Hardware Usage Patterns (THUP). In the lower layer, users are studied with respect to these THUPs to identify Typical App Usage Patterns (TAUP). The analytical results of these two layers are consolidated into Usage Pattern Models (UPM), and UBMs are finally established by a union of UPMs and Usage Intensity Distributions (UID). By carrying out experiments on energy consumption data from 18,308 distinct users over 10 days, 33 UBMs are extracted from training data. With the test data, it is proven that these UBMs cover 94% user behaviors and achieve up to 20% improvement in accuracy of energy representation, as compared with the baseline method, PCA. Besides, potential applications and implications of these UBMs are illustrated for smartphone manufacturers, app developers, network providers, and so on.

CCS Concepts: • **Human-centered computing** → **Smartphones**; • **Information systems** → **Clustering**; **Data analytics**; • **Mathematics of computing** → *Dimensionality reduction;*

Additional Key Words and Phrases: Data mining, smartphone energy consumption, user behavior modeling

ACM Reference format:
Ming Ding, Tianyu Wang, and Xudong Wang. 2021. Establishing Smartphone User Behavior Model Based on Energy Consumption Data. *ACM Trans. Knowl. Discov. Data.* 16, 2, Article 25 (July 2021), 40 pages.
https://doi.org/10.1145/3461459

Author's address: M. Ding, T. Wang, and X. Wang (corresponding author), Shanghai Jiao Tong University, No. 800 Dongchuan Road, Shanghai 200240, China; emails: dingming150@gmail.com, gunnerwang27@sjtu.edu.cn, wxudong@ieee.org.

1556-4681/2021/07-ART25 $15.00
https://doi.org/10.1145/3461459

1 INTRODUCTION

With the proliferation of mobile devices, the smartphone is one of the most widely used devices and plays an indispensable role in modern life. Smartphone energy consumption analysis is vital since 89% of users rate "long battery life" as the most important factor of user satisfaction [4]. In the meantime, smartphone user behavior analysis is also important considering it provides useful models and implications to various key players within the smartphone eco-system (e.g., app developers, users, and vendors) [21].

However, the relationship between smartphone's energy consumption and smartphone user's behavior has been rarely studied so far due to the issue of data accessibility. Specifically, smartphone manufacturers have access to energy consumption data on hardware components of different apps, while user behavior data indicating details (e.g., in the form of log file) about how apps are used is usually collected by respective app developers and hardly accessible to smartphone manufacturers. Fortunately, as smartphone energy consumption highly depends on users' interaction with their smartphones [2, 13], it is possible to explore user behavior from the perspective of energy consumption data. Such an exploration of the relationship between energy consumption and user behavior can reduce the cost of data acquisition for smartphone manufacturers. More importantly, user behavior established upon energy consumption data can enable multiple useful applications, such as: (1) refine user behavior classification from the perspective of detailed energy consumption features; (2) for users with different behaviors, locate important hardware components that have the most urgent demand for upgrades so as to improve smartphone energy efficiency; (3) for users with different behaviors, estimate the room for improvement of hardware components; and (4) assuming that user behaviors do not vary much between successive smartphone generations, predict the energy performance of the next-generation smartphone based on the user behavior explored from energy consumption data of current-generation smartphones.

In literature, there have been a number of research studies working on smartphone energy consumption modeling and smartphone user behavior mining. For the former direction, the majority of the studies focus on constructing power models for energy consumption estimation without external measurement equipment [7–9, 19, 20, 24, 26, 35]. These power models are mostly established under a few explicitly specified parameters. Therefore, it can be difficult for them to express more complex user behavior. Considerable inaccuracy may arise when implementing these models on diverse user behaviors in real-world. For the latter direction, many studies investigate user behavior from various perspectives, such as app selection and management [21], app usage frequency and duration [10, 12, 22], app revisitation pattern [17], and click events [34]. However, the perspective of energy consumption for user behavior is hardly explored in these studies. Instead, the data used in these studies for mining user behavior usually expose some explicit details about app usage, which are hardly accessible for smartphone manufacturers.

In this article, an Energy-based Smartphone User Behavior modeling approach, called E-Sub, is proposed to explore the relationship between energy consumption and user behavior. Specifically, some **User Behavior Models** (**UBMs**) that reveal which and how apps are used are extracted from energy consumption data. A small number of UBMs are preferred to control the model implementation cost for smartphone manufacturers. In other words, these extracted UBMs should have strong representation abilities so that diverse user behaviors can be covered by this small number of UBMs. In terms of energy consumption data, its structure is shown in Figure 1. The data mainly involve smartphone energy breakdown on apps for each user and app energy breakdown on hardware components for each app used by a specific user. Therefore, each smartphone user's energy consumption data are represented as a feature vector whose dimension consists of hardware components with respect to all existing apps (to ensure the dimension consistency among all users).

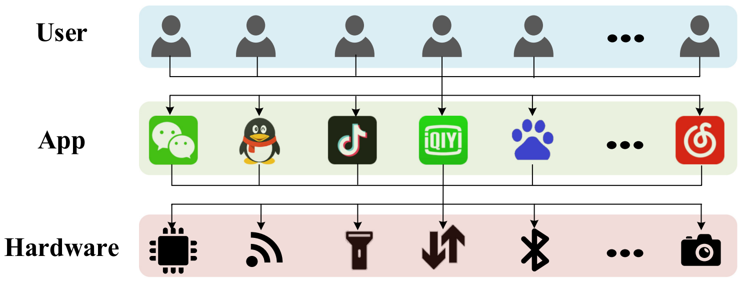


Fig. 1. The data structure of energy consumption in this article.

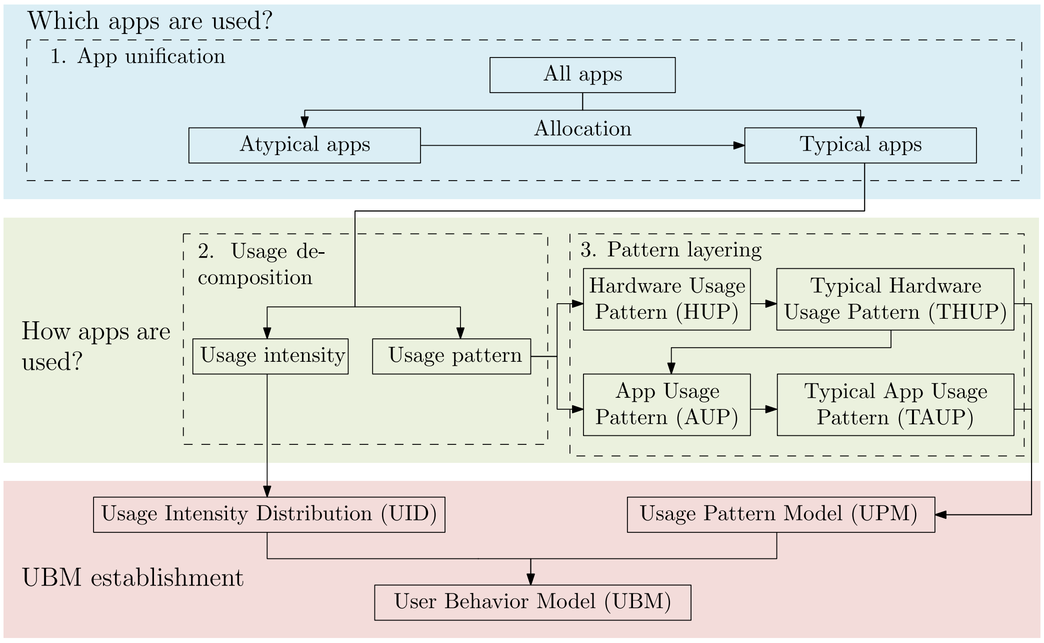


Fig. 2. Three key mechanisms of E-Sub.

However, due to the high diversity of user behavior, users' feature vectors characterize: (1) high dimension; (2) high dispersion. High dimension comes from the massive apps used by large-scale users (2.5 million apps in Google Play Store by the second quarter of 2019 [11]). Moreover, app energy breakdown on hardware components further expands the dimension. In the meantime, a high dispersion comes from users' different usage ways of each app. Such different usage ways are caused by: (1) enriched app functionalities [2] and users' respective preference for these functionalities; and (2) different usage time. Regarding users' feature vectors, these two factors result in a wide range of direction and magnitude of vectors, respectively.

In our proposed approach E-Sub, three mechanisms are designed to address the challenge of modeling user behavior from highdimensional and highly dispersed energy consumption data. These mechanisms are illustrated in Figure 2.

(1) The first mechanism is **App Unification**. To reduce data dimension, all apps are first ranked in descending order according to a normalized indicator, app typicality, that measures app's

importance in smartphone normal daily usage across the entire spectrum of users. Then a threshold is set regarding the cumulative sum of app typicality and apps lying in front of the threshold are identified as typical apps. Moreover, to maintain the completeness of energy consumption information produced by atypical apps (i.e., those apps locating behind the threshold), their energy consumption is redistributed to typical apps according to their similarity of energy consumption breakdown on hardware components. With this mechanism, the data dimension is reduced because user's feature vector only has dimensions related to these typical apps.

(2) The second mechanism is **Usage Decomposition**. To reduce data dispersion brought about by various vector ℓ_1 norms (i.e., sum of the vector elements), each user's feature vector is scaled to unit-ℓ_1-norm vector [25]. The scaled feature vector becomes usage pattern, which is represented as energy percentage breakdown on the typical apps. Usage pattern reflects the user's preference and choice for app functionalities, regardless of how intensive apps are used (i.e., the differences regarding usage time are eliminated by scaling). Meanwhile, the ℓ_1 norm of vector before scaling is considered as usage intensity, which is proportional to the usage time of the smartphone. Combining the two aspects above, the analysis of how the typical apps are used can be decomposed into branches of usage pattern analysis and usage intensity analysis.

(3) The third mechanism is **Pattern Layering**. After the previous two mechanisms, usage pattern becomes the user's new feature vector, which has reduced data dimension (i.e., representation with typical apps) and partially reduced data dispersion (i.e., unit-ℓ_1-norm vector). To deal with the data dispersion brought about by various vector directions, clustering algorithms are employed hierarchically upon the usage patterns (i.e., first dealing with hardware components of each typical app, then dealing with typical apps) so that the data dimension for each clustering is further controlled and easy to interpret. The overall idea is to group closely located usage patterns together and represent the in-group usage patterns with corresponding cluster centroids for data dispersion reduction. In the upper layer, **Hardware Usage Patterns** (**HUPs**) are clustered, which are represented as an app energy percentage breakdown on hardware components. It is noted that HUP is studied for individual typical apps, thus users' usage pattern can be divided into multiple HUPs according to the typical apps that they use. For each typical app, cluster centroids are thus identified as preliminary **Typical HUPs** (**THUPs**), which characterize users' typical preference for app functionalities. To reduce the redundancy of these preliminary THUPs, they are then filtered by similarity (i.e., Euclidean distance) to obtain ultimate THUPs. In the lower layer, **App Usage Patterns** (**AUPs**) are clustered, which are represented as smartphone energy percentage breakdown on ultimate THUPs. For each user's usage pattern, AUP can be obtained by summing energy percentages over all hardware components of each ultimate THUP. Cluster centroids are identified as **Typical AUPs** (**TAUPs**), which characterize users' typical degrees of preference on those ultimate THUPs.

After going through the three mechanisms, eventually, the UBM can be established. Each UBM is composed of a **Usage Intensity Distribution** (**UID**) that results from investigating the usage intensity, and a **Usage Pattern Model** (**UPM**) that is developed by consolidating ultimate THUPs and TAUPs. Therefore, the established UBMs are able to provide details about which apps are typically used with the mechanism of app unification and how apps are typically used with the rest two mechanisms. Such details bridge smartphone energy consumption and user behavior.

We implement our E-Sub approach with the energy consumption data collected from 18,308 distinct Android users on a daily basis. The data involve a total of 173,917 days and 2,991,315 app usage records on 12 major hardware components. Based on the training part of data (80% of all the data), 33 UBMs are established. The performance of these UBMs is evaluated in two aspects: (1) behavior coverage, measuring UBMs' ability to cover diverse user behaviors; and (2) energy discrepancy, measuring the weighted root mean squared error due to representing users' energy consumption with these small number of UBMs. Based on the test part of data (the rest 20% of all the data), it turns out that the established 33 UBMs achieve 93.97% regarding behavior coverage and up to 20% improvement regarding energy discrepancy compared with the baseline method, **Principal Components Analysis (PCA)** [16] (i.e., directly reducing the data dimension from the original feature vectors and performing clustering for once). Moreover, E-Sub outperforms all of its variants (i.e., removing or modifying certain mechanisms), which further validates the necessity and effectiveness of the designed three mechanisms in E-Sub.

The contributions of this article are summarized as follows:

— We propose E-Sub, as a novel approach to explore the relationship between smartphone energy consumption and user behavior. In our approach, a small number of UBMs can be established from the large-scale energy consumption data featuring high dimension and high dispersion. Through comparing it with the baseline method and various simplified versions of E-Sub, we validate the best performance of E-Sub and the effectiveness of the three mechanisms in E-Sub.
— We experiment our E-Sub approach on the real-world data from large-scale users. Compared with the previous studies that conducted experiments on a relatively small number of participants or simulated data, the large-scale real-world data allow us to fully explore diverse user behaviors and obtain more general user behavior models.
— We propose several metrics that can be potentially leveraged by future researchers to explore more on related topics in an unsupervised manner. These metrics involve: (1) indicators that quantify how important an app is; (2) comprehensive performance scores for clustering; and (3) metrics that evaluate the performance of the extracted UBMs quantitatively.
— Our manifold experiment results provide significant implications and quantitative evidence about smartphone usage to various stakeholders within the smartphone eco-system. Specifically, we demonstrate three potential applications for smartphone manufacturers showing how our established UBMs can be used to guide the hardware upgrades for better energy performance.

The rest of this article is organized as follows. In Section 2, related work about smartphone energy consumption modeling and user behavior mining are summarized and analyzed. In Section 3, an overview of our data, as well as major terms and notations used in this article, is provided. In Section 4, the approach of E-Sub is elaborated step by step. Intermediate results are also demonstrated at the end of each step, which serve as an instant validation of the effectiveness of each step. Final results of UBM establishment are presented at the end of this section. In Section 5, the performance evaluation and performance comparison are reported. In Section 6, several potential applications by leveraging our established UBMs are demonstrated. In Section 7, insightful implications that can be derived from our results are summarized for various key players in smartphone eco-system. In Section 8, conclusion and future work are discussed.

2 RELATED WORK

The prevalent usage of smartphones potentially leads to massively available smartphone data and inspires a lot of research work on smartphone data analysis. Among these works, energy consumption modeling and user behavior mining are the two major research topics.

2.1 Energy Consumption Modeling

A number of studies have focused on building energy consumption models for smartphone's hardware components and apps. Chen et al. [9] modeled the energy draw behavior of Modem under normal usage at the granularity of event. Zhang et al. [35] proposed a method for automated generation of energy consumption model. The method infers system variables and power coefficients of major hardware components. Lee et al. [19] further modeled energy consumption of CPU, Display, and WiFi without detailed knowledge of the smartphone hardware. Such a modeling exploits apps' hardware access profile. Mittal et al. [24] emulated the energy consumption of the Display, Network, and CPU through energy consumption modeling and then resource scaling. Rather, Pathak et al. [26] proposed a program entity level energy profiler for smartphone apps. Li et al. [20] developed a source line level energy profiler through statistical modeling of measured hardware energy consumption. Chen et al. [8] proposed a hybrid model for energy breakdown estimation among activities and smartphone components. These modeling techniques to large degrees facilitate the subsequent analysis (e.g., prediction) of the smartphone's energy consumption. However, as argued in [29], these established models are subject to inaccurate estimates of energy consumption when some smartphone components and apps manifest complex energy consumption behavior (i.e., associated with more parameters and possible non-linearity). In fact, the diversity and personalization of modern usage of smartphones can often lead to such complex behavior beneath.

Instead, Rattagan et al. [29] first identified asynchronous energy consumption behavior through affinity propagation clustering. Then symbolic regression is utilized for fitting to address the non-linearity. Moreover, Jiang et al. [15] studied the relationship between user behavior and energy consumption based on 20 smartphone users. Various levels of correlation are demonstrated between phone usage and factors such as time of day, user's location, and remaining battery power. Carroll et al. [7] developed energy consumption models for distinct smartphone usage scenarios (e.g., text messaging, voice call, web browsing), and further investigated the day-to-day energy consumption under some usage patterns (e.g., suspend, casual,regular). Their work demonstrates an evident relationship between smartphone energy consumption and user behavior. In some cases, the performance of energy consumption modeling benefits from the awareness of distinct behavior. However, behavior explored in their work is relatively limited and manually defined. The generalization of such energy consumption models to a wider range of smartphone users still remains a problem.

2.2 User Behavior Mining

In the meantime, many studies have investigated user behavior mining in various aspects. Li et al. [21] explored patterns of app usage in terms of app management activities (i.e., installation, updating, and uninstallation) and app network traffic. Do et al. [12] established a bag-of-apps model to represent daily users according to their frequency (i.e., no-use, low-use, middle-use, high-use), time period (i.e., night, morning, afternoon, evening) and launched apps (i.e., Voice, SMS, Internet, Camera, Gallery). An author-topic model was then utilized to discover usage patterns and retrieve relevant users. Cao et al. [6] mined the association rule between context (i.e., holiday, day period, time range, speed) and user interaction record (e.g., listening to music) from context log. Zhao et al. [36] represented each smartphone user with a 232-dimensional sparse vector (29 for app category $\times$ 4 for time period $\times$ 2 for holiday / workday indicator). They finally discovered 382 types of users through a two-step clustering and analyzed these user types by ranking features. Wang et al. [34] employed a hierarchical clustering approach to mine the most popular user behavior from clickstream data. For the outcomes of hierarchical clustering, higher-level clusters represent a more general behavior, while lower-level clusters further identify a relatively minor behavior.

Table 1. The Form of App Usage Records for Each Day of Each User

UserID	StartTime	EndTime	TopType	App	T	Audio	FrontCam	···	GPU	Modem	CPU	···	Display
1	2018/5/8 00:00	2018/5/9 00:00	1	com.tencent.mm	136.16	14.34	0.00	···	0.09	236.76	98.12	···	420.95
					⋮								
1	2018/5/8 00:00	2018/5/9 00:00	20	com.android.contacts	4.53	0.00	0.00	···	0.02	0.00	1.76	···	14.87

The unit of T is min, while the unit of hardware energy consumption is mAh.

Jones et al. [17] presented a smartphone revisitation analysis that reveals clusters of apps and users sharing similar revisitation patterns. Krieter et al. [18] developed an approach to analyze user behavior by automatically generating log files from mobile screen recordings in a privacy-friendly way. Methods of computer vision and machine learning were utilized to process screen recordings. Rashid et al. [28] presented a study on the smartphone features preferences and usage patterns (e.g., phone calls, SMS, social apps) among users with various demographics (e.g., age, gender, education). Data were collected via questionnaires and statistical methods were utilized to identify key smartphone features. Silva et al. [31] mined app usage patterns of 5,342 smartphone users mainly concerning how, when, and where they interact with the mobile apps. General patterns (e.g., frequency, duration, diversity, data traffic), temporal patterns, and location of each app access were exploited. In general, the work above explored user behavior based on various types of smartphone data that expose explicit details about users, such as time slot, clickstream, and revisitation. While such data are available for the app developers and users themselves, it can be difficult or requires high cost for manufacturers to access such types of data.

3 DATA OVERVIEW

The dataset used in this article was collected by a leading provider of smart devices in China. The dataset is collected from a certain model of smartphone product, and it involves 183,080 days of smartphone usage data evenly from 18,308 distinct Android users during May 1st, 2018 and May 10th, 2018. The time stamp of the dataset is every 24 hours, i.e., from 00:00 of the previous day to 00:00 of the current day.

As a routine, before data collection, smartphone users are asked, and they agree to share smartphone usage data to help manufacturers improve product performance. However, due to issues of user privacy, the dataset used for analysis has been desensitized and, therefore, does not contain any personal information, but smartphone energy consumption data. Due to the logging overhead, the top 20 most energy-consuming apps (including third-party apps, system apps, and system processes) are recorded per day. And the form of such records in the dataset is given in Table 1. Each user is assigned with a random User ID (*UserID*). For each user, the usage data is composed of multiple rows, where we call each row as a record. Each record of a user contains the usage information of a certain app (*App*) on a certain day (*StartTime*, *EndTime*): daily app ranking in terms of energy consumption (*TopType*), daily app usage time (T), and daily app energy breakdown on 12 major hardware components (*Audio, Front Camera, Flashlight, Bluetooth,* **Global Navigation Satellite System**(**GNSS**), *Sensor,* ***Graphic Processing Unit (GPU)***, *Modem, CPU, WiFi, Rear Camera, Display*).

After data preprocessing, the dataset is left with 18,306 users, 173,917 valid days, and in total 2,991,315 records from 20,964 distinct apps. We then divide the dataset into the training set and test set. It is supposed to avoid splitting days of the same users into both training and test sets; otherwise, potential behavior consistency over days might lead to information leakage. We proceed with the following. First, sort the dataset by multiple keys: *UserID*, *StartTime*, and *TopType*. Thus,

Table 2. Major Terms Used in This Article

Term	Abbreviation	Description
Hardware Usage Pattern	HUP	App's energy percentage breakdown on hardware components, indicating how an app is used by a single user (Section 4.2).
Typical Hardware Usage Pattern	THUP	Cluster centroid of HUPs, indicating the typical pattern of using an app (Section 4.2).
App Usage Pattern	AUP	Smartphone's energy percentage breakdown on ultimate THUPs, indicating user's degrees of preference on ultimate THUPs (Section 4.4).
Typical App Usage Pattern	TAUP	Cluster centroid of AUPs, indicating the typical pattern of degrees of preference on ultimate THUPs (Section 4.4).
User Group	/	The group of users that share the same TAUP.
Usage Intensity Distribution	UID	The outcome of fitting smartphone energy consumption within a specific user group by normal distribution, indicating the usage intensity distribution within that user group (Section 4.5).
Usage Pattern Model	UPM	Smartphone's energy percentage breakdown on hardware components of all ultimate THUPs for a specific user group, i.e., a consolidation of TAUPs and ultimate THUPs. It reflects typical usage patterns of smartphone usage for a specific user group (Section 4.6).
User Behavior Model	UBM	A series of concrete user behaviors for a specific user group (i.e., both usage pattern and usage intensity), reflecting smartphone's energy breakdown on hardware components of all ultimate THUPs and usage time of these ultimate THUPs (i.e., a union of UID and UPM).

records of the same user on different days are arranged consecutively. Then, label the first 80% records as the training set and the last 20% records as the test set, where additional measures are taken to guarantee that records of the user at the 80% boundary are not split. The UBMs are extracted from the training set whereas the evaluation of these UBMs' performance is conducted upon the test set that is composed of out-of-sample users. Moreover, as our goal is establishing UBMs from daily user behavior instead of exploring its temporal patterns, we thus consider data records of the same user on different days as distinct users. Henceforth, we assume that our dataset involves 173,917 distinct users in the rest of this article.

The major terms used in this article are summarized in Table 2 and the major mathematical notations are listed in Table 3.

4 E-SUB APPROACH

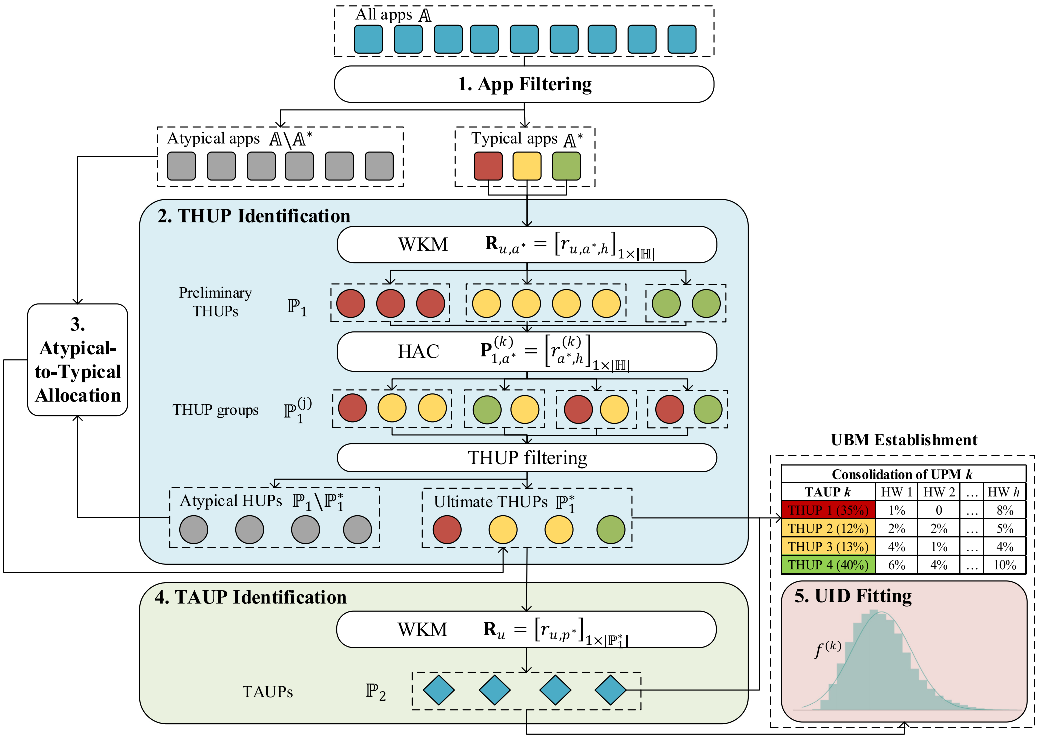
Our proposed E-Sub is implemented with five steps, as shown in Figure 3. The three key mechanisms mentioned previously are intertwined in these steps. Specifically, the first step (i.e., app filtering) and the third step (i.e., atypical-to-typical allocation) constitute the app unification mechanism; the second step (i.e., THUP identification) and the fourth step (i.e., TAUP identification) analyze usage pattern and constitute the pattern layering mechanism; the fifth step (i.e., UID fitting) studies usage intensity and constitutes the usage decomposition mechanism together with the second and the fourth step. UBM establishment demonstrates our final result, which is a union of UPM (i.e., a consolidation of THUP and TAUP) and UID, which are intermediate results of previous steps.

(1) The first step is app filtering. Apps are divided into two sets after filtering: typical apps $\mathbb{A}^*$ and atypical apps $\mathbb{A}\backslash\mathbb{A}^*$.
(2) The second step is THUP identification. In this step, two standard clustering techniques are integrated to identify typical hardware usage patterns: (1) **Weighted K-Means** (**WKMs**)

Table 3. Major Mathematical Notations Used in This Article

Notation	**Description**
$\mathbb{U}$	Set of users
$\mathbb{A}/\mathbb{A}^*$	Set of all apps / typical apps
$\mathbb{H}$	Set of hardware components
$\mathbb{P}$	$\mathbb{P}_{1,a}$: set of preliminary THUPs from the app $a \in \mathbb{A}^*$
	$\mathbb{P}_1 = \bigcup_{a\in\mathbb{A}^*} \mathbb{P}_{1,a}$: set of preliminary THUPs of all typical apps
	$\mathbb{P}_1^{(j)}$, $j = \{1, 2, \ldots, K\}$: set of j-th THUP group, where K is the total number of THUP groups
	$\mathbb{P}_1^*$: set of all ultimate THUPs, where $\lvert\mathbb{P}_1^*\rvert = K$
	$\mathbb{P}_2$: set of TAUPs
D	$\mathbf{D}_1 = \left[d_{u,p}\right]_{\lvert\mathbb{U}\rvert\times\lvert\mathbb{P}_1\rvert}$: a matrix with $d_{u,p} \in \{0, 1\}$ ($d_{u,p} = 1$ indicates that user u uses the functionality described by THUP p)
	$\mathbf{D}_2 = [d_{u,a}]_{\lvert\mathbb{U}\rvert\times\lvert\mathbb{A}\rvert}$: a matrix with $d_{u,a} \in \{0, 1\}$ ($d_{u,a} = 1$ indicates that user u uses the app a)
L	$\mathbf{L}_a = \left[l_{u,a,k}\right]_{\lvert\mathbb{U}\rvert\times 1\times\lvert\mathbb{P}_{1,a}\rvert}$: a matrix with $l_{u,a,k} \in \{0, 1\}$ ($l_{u,a,k} = 1$ indicates that the app a used by user u is partitioned to the k-th THUP of the app a)
	$\mathbf{L} = \left[l_{u,k}\right]_{\lvert\mathbb{U}\rvert\times\lvert\mathbb{P}_2\rvert}$: a matrix with $l_{u,k} \in \{0, 1\}$ ($l_{u,k} = 1$ indicates that user u is partitioned to the k-th user group)
T (min)	$T_{u,a}$: user u's usage time on app a
E (mAh)	$E_{u,a,h}(E_{u,p,h})$: user u's energy consumption on hardware h of app a (of THUP p)
	$E_{u,a} = \sum_{h\in\lvert\mathbb{H}\rvert} E_{u,a,h}$: user u's energy consumption on app a
	$E_{u,p} = \sum_{h\in\lvert\mathbb{H}\rvert} E_{u,p,h}$: user u's energy consumption on THUP p
	$E_{u,h} = \sum_{a\in\lvert\mathbb{A}\rvert} E_{u,a,h} = \sum_{p\in\lvert\mathbb{P}_1\rvert} E_{u,p,h}$: user u's energy consumption on hardware h
	$E_u = \sum_{a\in\lvert\mathbb{A}\rvert} E_{u,a} = \sum_{p\in\lvert\mathbb{P}_1\rvert} E_{u,p}$: user u's smartphone energy consumption
E	$\mathbf{E}_u = \left[E_{u,a,h}\right]_{1\times(\lvert\mathbb{A}\rvert\lvert\mathbb{H}\rvert)}$: feature vector of user u's energy consumption on hardware components with respect to apps
R	$\mathbf{R}_{u,a} = \left[r_{u,a,h}\right]_{1\times\lvert\mathbb{H}\rvert}$: feature vector of user u's HUP of app a, where $r_{u,a,h} = E_{u,a,h}/E_{u,a}$
	$\mathbf{R}_{u,p} = \left[r_{u,p,h}\right]_{1\times\lvert\mathbb{H}\rvert}$: feature vector of user u's HUP that is similar to THUP p, where $r_{u,p,h} = E_{u,p,h}/E_{u,p}$
	$\mathbf{R}_u = \left[r_{u,p}\right]_{1\times\lvert\mathbb{P}_1^*\rvert}$: feature vector of user u's AUP, where $r_{u,p} = E_{u,p}/E_u$
P	$\mathbf{P}_{1,a}^{(k)} = \left[r_{a,h}^{(k)}\right]_{1\times\lvert\mathbb{H}\rvert}$: feature vector of the k-th preliminary THUP of the app a, where $r_{a,h}^{(k)}$ is a function of $E_{u,a}$, $r_{u,a,h}$, $d_{u,a}$, and $l_{u,a,k}$
	$\mathbf{P}_{1,p} = \left[r_{p,h}\right]_{1\times\lvert\mathbb{H}\rvert}$: feature vector of an ultimate THUP p, where $r_{p,h}$ equals to $r_{a,h}^{(k)}$ of a corresponding preliminary THUP
	$\mathbf{P}_2^{(k)} = \left[r_p^{(k)}\right]_{1\times\lvert\mathbb{P}_1^*\rvert}$: feature vector of the k-th TAUP, where $r_p^{(k)}$ is a function of E_u, $r_{u,p}$, and $l_{u,k}$

[1] is implemented for each typical app across the app users based on $\mathbf{R}_{u,a}$ (i.e., app energy percentage breakdown on hardware components), from where cluster centroids constitute the set of THUPs of all typical apps $\mathbb{P}_1 = \bigcup_{a\in\mathbb{A}^*} \mathbb{P}_{1,a}$; and (2) **Hierarchical Agglomerative Clustering** (**HAC**) [30] is then applied across $\mathbb{P}_1$ and these THUPs are clustered into THUP groups $\mathbb{P}_1^{(j)}, j = \{1, 2, \ldots, K\}$. In general, WKM identifies cluster centroids of all typical apps and HAC groups similar centroids. Then, from each THUP group $\mathbb{P}_1^{(j)}$, an ultimate THUP is selected and the rest are regarded as atypical HUPs. Therefore, K ultimate THUPs constitute $\mathbb{P}_1^*$ and atypical HUPs belong to $\mathbb{P}_1 \backslash \mathbb{P}_1^*$.


All apps $\mathbb{A}$
1. App Filtering
Atypical apps $\mathbb{A}\backslash\mathbb{A}^*$
Typical apps $\mathbb{A}^*$
2. THUP Identification
WKM $\mathbf{R}_{u,a^*} = [r_{u,a^*,h}]_{1\times|\mathbb{H}|}$
Preliminary THUPs $\mathbb{P}_1$
HAC $\mathbf{P}^{(k)}_{1,a^*} = [r^{(k)}_{a^*,h}]_{1\times|\mathbb{H}|}$
THUP groups $\mathbb{P}^{(j)}_1$
THUP filtering
Atypical HUPs $\mathbb{P}_1\backslash\mathbb{P}^*_1$
Ultimate THUPs $\mathbb{P}^*_1$
3. Atypical-to-Typical Allocation
4. TAUP Identification
WKM $\mathbf{R}_u = [r_{u,p^*}]_{1\times|\mathbb{P}^*_1|}$
TAUPs $\mathbb{P}_2$
UBM Establishment
5. UID Fitting
$f^{(k)}$

Consolidation of UPM *k*				
TAUP *k*	HW 1	HW 2	…	HW *h*
THUP 1 (35%)	1%	0	…	8%
THUP 2 (12%)	2%	2%	…	5%
THUP 3 (13%)	4%	1%	…	4%
THUP 4 (40%)	6%	4%	…	10%

Fig. 3. The overall framework of E-Sub.

(3) The third step is atypical-to-typical allocation. To maintain the completeness of energy consumption after the filtering processes in the first two steps, both atypical apps and atypical HUPs of typical apps are allocated to their most similar ultimate THUPs.

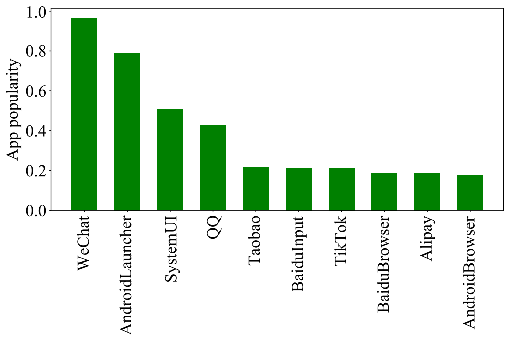
(4) The fourth step is TAUP identification. In this step, WKM is implemented across all users based on $\mathbf{R}_u$ (i.e., smartphone energy percentage breakdown on ultimate THUPs). Cluster centroids form the set of TAUPs $\mathbb{P}_2$ and users are accordingly divided into $|\mathbb{P}_2|$ user groups. For each user group, the behavior of in-group users without usage intensity is represented by a common UPM which is established by consolidating the TAUP and ultimate THUPs. The UPM appears in the form of smartphone energy percentage breakdown on hardware components with respect to all ultimate THUPs.

(5) The fifth step is UID fitting. For every single UPM, a UID is accordingly fitted by normal distribution, upon smartphone energy consumption of in-group users.

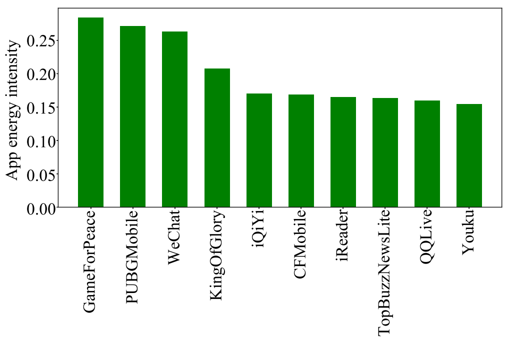
Finally, UBMs are established as a union of UPMs and corresponding UIDs.

4.1 Step 1: App Filtering

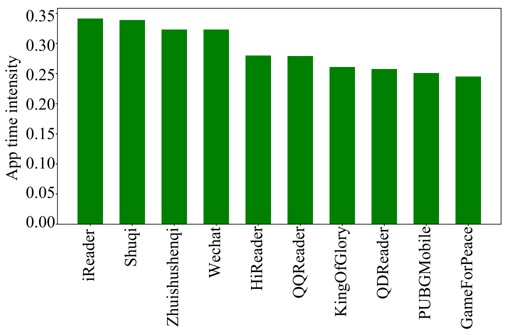
The issue of huge data dimension exists due to the large number of involved apps in the dataset. To reduce data dimension, we explicitly represent user behavior only with the most "typical" apps. Notably, this does not mean we discard the remaining apps from the dataset. In fact, the energy consumption of the remaining apps will be allocated to these "typical" apps in the later step. Naturally, the "typical" apps are considered to be those that can cover the majority of normal smartphone daily usage so as to maintain most of original smartphone usage information.



(a) Top 10 apps in terms of λ_a.



(b) Top 10 apps in terms of μ_a.



(c) Top 10 apps in terms of ν_a.

Fig. 4. Top 10 apps with respect to individual indicators.

4.1.1 Method Description. There are several indicators to measure apps' contributions in smartphone daily usage.

— App popularity λ_a is calculated as the percentage of users who use an app a over the entire users:

$$\lambda_a = \frac{\sum_{u \in \mathbb{U}} d_{u,a}}{|\mathbb{U}|}, \tag{1}$$

where the value of $d_{u,a}$ is binary and $d_{u,a} = 1$ indicates that user u uses app a.

— App energy intensity μ_a is calculated as the percentage of app a's energy consumption over the energy consumption of all apps used by one user, averaged on all users using app a:

$$\mu_a = \frac{\sum_{u \in \mathbb{U}} d_{u,a} \left(\frac{E_{u,a}}{\sum_{i \in \mathbb{A}} d_{u,i} E_{u,i}} \right)}{\sum_{u \in \mathbb{U}} d_{u,a}}. \tag{2}$$

— App time intensity ν_a is calculated as the percentage of app a's usage time over the usage time of all apps used by one user, averaged on all users using app a:

$$\nu_a = \frac{\sum_{u \in \mathbb{U}} d_{u,a} \left(\frac{T_{u,a}}{\sum_{i \in \mathbb{A}} d_{u,i} T_{u,i}} \right)}{\sum_{u \in \mathbb{U}} d_{u,a}}. \tag{3}$$

In Figure 4, top 10 apps are shown with respect to the three individual indicators (λ_a, μ_a, ν_a), respectively. It turns out that app ranking differs with respect to different indicators. For λ_a (Figure 4(a)), system apps (e.g., Android Launcher, System UI, Android Browser) and utility apps (e.g., Baidu Input) rank the highest. For μ_a (Figure 4(b)), game apps (e.g., Game for Peace, PUBG Mobile, King of Glory) and video apps (e.g., QQ Live, Youku) rank the highest. For ν_a (Figure 4(c)), E-book apps (e.g., iReader, Shuqi, Zhuishushenqi) and game apps (e.g., PUBG Mobile, Game for Peace) rank the highest.

It is obvious that each indicator is dominated by certain app categories and none of these indicators are capable of individually selecting apps that characterize the main smartphone normal daily use. To ensure that our selected "typical" apps are not dominated by system apps, system processes, or third-party apps from a single app category, and can also cover the majority of smartphone usage scenarios, we define an aggregative indicator, app typicality I_a, as a combination of the three individual indicators mentioned above (Equation (4)).

$$I_a = \frac{\lambda_a(\omega_1\mu_a + \omega_2\nu_a)}{\sum_{i\in\mathbb{A}}\lambda_i(\omega_1\mu_i + \omega_2\nu_i)}, \tag{4}$$

where ω_1, ω_2 are normalized weights for energy and time intensity, respectively (i.e., $\omega_1 + \omega_2 = 1$). We set $\omega_1 = \omega_2 = 0.5$ by default because we consider energy and time intensity as equally important in measuring app typicality.

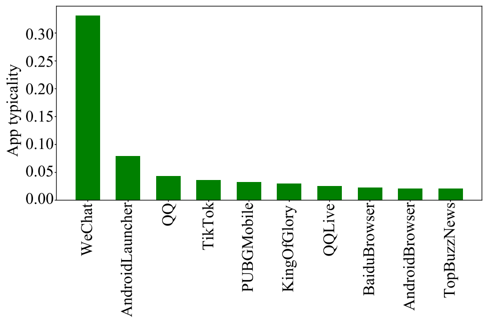
Reasons of designing the combination as such are as follows. First, there is a multiplication between app popularity λ_a and the other two intensity metrics (μ_a and ν_a). It is because λ_a is a group-based indicator while μ_a and ν_a are individual-based indicators. A bad example is that an app can have a very small group of target users (i.e., small λ_a), while its target users have an extremely high dependence on this app (i.e., large μ_a or ν_a). Lalamove (or Huolala) is one of such a type, which is intensively used by only logistics industry workers. Because of their low popularity, this type of apps is not supposed to be a good choice for typical apps. Therefore, instead of regarding the three components in parallel, we treat app popularity as a multiplicative factor before the two intensity indicators are considered. Intuitively, multiplying the group-based indicator λ_a is able to transform the two individual-based indicators μ_a and ν_a to become group-based, which is consistent with the goal of selecting typical apps as those who play an essential role in the daily life with respect to all users.

Second, μ_a and ν_a are combined as a weighted sum because these two factors have similar attributes and we assume that these two factors are equally important in determining app typicality. According to our experiment results, many system apps appear to be relatively high in time intensity while low in energy intensity, thanks to their well-managed energy consumption optimization. Meanwhile, there are also apps with high energy intensity while relatively low time intensity, such as game, video, and navigation apps. It is obvious that no matter system apps or energy-consuming third-party apps, they are all supposed to be taken into account in selecting typical apps.

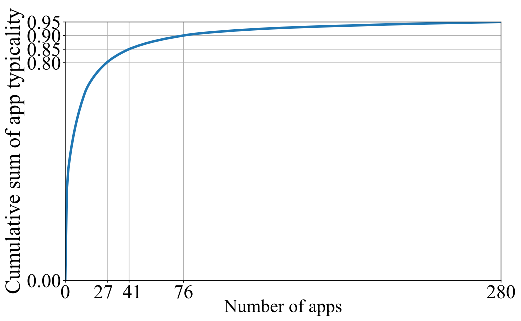
Third, we normalize the formula in the end because a threshold needs to be set to distinguish typical and atypical apps. Without normalization, such a threshold depends on the absolute value of the three components (λ_a, μ_a, ν_a) of each app. Therefore, normalization helps prevent the threshold setting from being affected by the data itself.

To this end, typical apps can be explained as those popular apps that are able to cover the majority of both energy consumption and usage time for smartphone normal daily use.

4.1.2 Intermediate Results. The results of app filtering are shown in Figure 5 and Table 4. In Figure 5(a), top 10 apps with respect to I_a are illustrated. WeChat has exclusively high app typicality $I_{WeChat} = 0.33$. This can easily be understood as WeChat ranks top with respect to all the three individual indicators. To divide the typical and atypical apps, a threshold is set on the cumulative sum of app typicality after all these apps are arranged in descending order according to I_a (Figure 5(b)). It is reported that 27 apps can cover 80% app typicality, 41 apps for 85%, 76 apps for 90%, and 280 apps for 95%. The relative gain in app typicality decreases drastically as the number



(a) Top 10 apps in terms of I_a.



(b) Cumulative sum of I_a.

Fig. 5. Results of app filtering.

Table 4. The Number of Typical Apps in Each Category and Some Exemplary Apps

Category	**Number of Apps**	**Example Apps**
System	15	Android Launcher, Android Contacts, System UI
Video	12	TikTok, QQ Live, iQiYi
Game	10	Game For Peace, King of Glory, Kaixinxiaoxiaole
News	9	TopBuzz News, Sina, QQ News
E-book	6	iReader, Zhuishushenqi, Shuqi
Instant message	5	WeChat, QQ, Ding Talk
Music	5	Kugou, QQ Music, NetEase Cloud Music
Browser	4	Baidu Browser, Android Browser, UC Mobile
Shopping	3	Taobao, Jingdong, Vip Shop
Navigation	2	Baidu Map, Amap
Downloader	2	Baidu Netdisk, Xunlei Downloader
Finance	1	Tonghuashun
Camera	1	Meiyan Camera
Transportation	1	Didi

of apps increases. Moreover, according to the statistical results from App Annie [3], smartphone users have 60~90 apps installed on their smartphones on average. Both the relative gain and the statistical results support us that it is reasonable to identify typical apps as the top 76 apps lying in front of the threshold 90%. We denote each typical app as a^* and the set of all typical apps as $\mathbb{A}^*$. In Table 4, the number of typical apps as well as exemplary ones are displayed according to app categories. Selecting according to the aggregative indicator of I_a, it is proved that typical apps cover a wide range of app categories. It ensures that UBMs established with these typical apps can comprehensively characterize smartphone normal daily use.

4.2 Step 2: THUP Identification

The ever-growing demands of smartphone users have motivated app developers to enrich app functionalities [2]. Given an app, it becomes increasingly uncertain which app functionalities are used by smartphone users. Therefore, merely identifying typical apps is far from characterizing the user behavior with highly dispersed preferences for app functionalities. Fortunately, it is possible to specify how a typical app is used based on the energy consumption data without the needs for detailed log information. As an app calls relevant hardware components to perform certain functionality, we can use app energy percentage breakdown on hardware components, namely HUP, to represent a specific app functionality. For example, Kugou (a music app) users who prefer sodcasting usually demonstrate a high energy percentage on the Audio component. In the meantime, with the availability of WiFi access point, users tend to show a higher energy percentage on the WiFi component and a lower energy percentage on the Modem component.

4.2.1 Method Description. Mathematically, user u's HUP ofapp a is denoted as an $|\mathbb{H}|$-dimensional vector

$$\mathbf{R}_{u,a} = [r_{u,a,h}]_{1\times|\mathbb{H}|}, \tag{5}$$

whose elements $r_{u,a,h} = \frac{E_{u,a,h}}{E_{u,a}}$ is the app a's energy percentage on the hardware component h. Notably, $\mathbf{R}_{u,a}$ has unit ℓ_1 norm because each element of the vector is in the form of a percentage. Thus, the dispersion brought about by various ℓ_1 norm of vectors has already been eliminated.

As clustering techniques are widely used to discover the cluster of data samples that share similar characteristics in multi-dimensional space [36], users' HUPs are analyzed with a WKM-HAC hybrid algorithm, together with a filtering process. For the first layer clustering WKM, it reduces the complexity of data distribution by identifying cluster centroids. For the second layer clustering HAC , it is applied to those previously identified cluster centroids and automatically determines the final number of clusters. There are two advantages of such a design: (1) HAC serves as a fine-tuner of WKM, which helps to identify a better number of clusters K; (2) this design allows us to cluster apps one by one instead of clustering as a whole, which helps improve the interpretability of the results.

WKM. Every typical app is supposed to go through WKM and preliminary THUPs are identified as centroids of HUP clusters. K-Means is the algorithm whose objective function exactly minimizes the total Euclidean distance between all training samples and their corresponding cluster centroids. WKM extends the traditional K-Means by assigning a weight to each data point, which indicates the contribution of the data point towards identifying cluster centroids. In our scenario, the weight is designed as the app energy consumption $E_{u,a}$, as HUPs (i.e., app energy percentage breakdown on hardware components) with high usage intensity are considered to contribute more by shortening the distance between in-group HUPs and the corresponding cluster centroid preliminary THUP. Therefore, by employing the weights, WKM can result in less energy discrepancy when representing HUPs with the corresponding preliminary THUP, which is exactly the objective function of WKM for each typical app $a^* \in \mathbb{A}^*$:

$$\min \sum_{k=1}^{K_{a^*}} \sum_{u\in\mathbb{U}} d_{u,a^*} l_{u,a^*,k} E_{u,a^*} ||\mathbf{R}_{u,a^*} - \mathbf{P}_{1,a^*}^{(k)}||_2, \tag{6}$$

where

— $l_{u,a^*,k}$ are binary with 1 indicating user u, who uses the app a^*, falls in the cluster k;

— $\mathbf{P}_{1,a^*}^{(k)}$ is the feature vector of kth preliminary THUP of app a^*, whose element is calculated as the weighted average of $r_{u,a^*,h}$:

$$r_{a^*,h}^{(k)} = \frac{\sum_{u \in \mathbb{U}} d_{u,a^*} l_{u,a^*,k} E_{u,a^*} r_{u,a^*,h}}{\sum_{u \in \mathbb{U}} d_{u,a^*} l_{u,a^*,k} E_{u,a^*}}; \tag{7}$$

— K_{a^*} is the number of clusters of app a^*, which is determined by a clustering performance score q_{a^*}, given as

$$q_{a^*} = 0.3 \times (1 - \mathrm{mdh}_{a^*}) + 0.3 \times s_{a^*} + 0.3 \times e_{a^*} + 0.1 \times c_{a^*}, \tag{8}$$

where

- mdh_{a^*} is app a^*'s Mean Discrepancy on Hardware components, which is calculated as the weighted root mean squared error between in-group HUPs and corresponding preliminary THUP (Equation (9)). It is noticed that the nominator is the same as the objective function of WKM (Equation (6)). Therefore, mdh_{a^*} can be utilized to measure the performance of WKM:

$$\mathrm{mdh}_{a^*} = \frac{\sum_{k=1}^{K_{a^*}} \sum_{u \in \mathbb{U}} d_{u,a^*} l_{u,a^*,k} E_{u,a^*} \|\mathbf{R}_{u,a^*} - \mathbf{P}_{1,a^*}^{(k)}\|_2}{\sum_{k=1}^{K_{a^*}} \sum_{u \in \mathbb{U}} d_{u,a^*} l_{u,a^*,k} E_{u,a^*}}. \tag{9}$$

- s_{a^*} is the Silhouette Coefficient after normalization [23], measuring how similar a data point is to its own cluster (compactness) compared with other clusters (separation).
- e_{a^*} is the Shannon's Entropy [27] after normalization, whose probability is the normalized number of users in each cluster. The high value of Shannon's entropy signifies that the clustering outcome has a uniform distribution regarding the number of users across clusters.
- c_{a^*} is the Clustering Complexity that penalizes the large value of K_{a^*}. It is defined as

$$c_a = \frac{K_{\max} - K_{a^*}}{K_{\max}}, \tag{10}$$

where $K_{\max}$ is the upper limit of the number of clusters. We set $K_{\max} = 6$ by default.

The design of q_{a^*} is inspired by [36]. We do not directly utilize the off-the-shelf metrics (e.g., Silhouette Coefficient, Davies–Bouldin Index, Calinski–Harabasz score) because performance evaluation of these internal metrics are not completely consistent with that of ours. Rather, q_{a^*} is designed to comprehensively capture the properties of the clustering outcome: mdh_{a^*} and s_{a^*} are the factors that measure the pure clustering performance; e_{a^*} and c_{a^*} quantify the desired properties of preliminary THUPs (i.e., balanced and small number of clusters). Weights of the four components are determined in a heuristic way: equal weight is the initial setting, and the stopping criterion is to obtain in q_{a^*}–K_{a^*} curve that does not have an overall upward or downward tendency; otherwise, K_{a^*} might be determined either too large or too small. Experiments show that equal weight fails because the factor of clustering complexity c_{a^*} is too sensitive to K_{a^*}. Therefore, less weight is assigned to c_{a^*} to avoid the tendency of q_{a^*} being dominated by the single factor. Further experiments validate the feasibility of the weight assignment of (0.3, 0.3, 0.3, 0.1), which is able to return a q_{a^*}-K_{a^*} curve without a significant monotonous tendency but a peak of q_{a^*} falling on a suitable K_{a^*} for most typical apps.

Hierarchical Agglomerative Clustering (HAC). While WKM analyzes the typical apps individually, HAC considers the preliminary THUPs derived from different typical apps jointly. In fact, different apps are likely to show similar preliminary THUPs, especially for those apps belonging to the same category. The pair of QQ Music and NetEase Cloud Music as shown below is a good

Table 5. Indices of Hardware Components

Index	1	2	3	4	5	6	7	8	9	10	11	12
Hardware	Audio	Front Camera	Flashlight	Bluetooth	GNSS	Sensor	GPU	Modem	CPU	WiFi	Rear Camera	Display

example (Figure 6(d) and 6(e)). Therefore, the goal of HAC is to group similar preliminary THUPs of different typical apps together to reduce the redundancy.

Concretely, HAC is implemented across all of the preliminary THUPs obtained from WKM. These preliminary THUPs are merged in a bottom-up manner according to their pairwise Euclidean distances. The merging distance increases as the merging process proceeds. Such a process stops once the distance threshold δ is met. To better adapt the HAC algorithm to the data, the value of δ is set statistically instead of, with an absolute distance. We set $\delta = \mu - 1.5\sigma$, where μ and σ are the mean and standard deviation of pairwise Euclidean distances of all preliminary THUPs, respectively.

THUP filtering. The HAC algorithm guarantees that the preliminary THUPs within the same THUP group are closely located and thus considered to be similar. Since our further analysis is based on THUPs instead of typical apps to provide more details of app usage, similar preliminary THUPs (even of different apps) are interchangeable and can replace each other. To further reduce the data dispersion caused by various vector directions, only one THUP is selected from each THUP group as the ultimate THUP. The rest of the preliminary THUPs are treated as atypical HUPs. Specifically, the ultimate THUP is identified as the one that results in the minimum energy discrepancy when using this THUP to represent all HUPs in the HUP clusters associated with the preliminary THUPs in the THUP group (except for HUPs that originally belong to the HUP cluster of this THUP). Denote the jth THUP group as $\mathbb{P}_1^{(j)}$, then the ultimate THUP of group j is selected as

$$\arg\min_{p \in \mathbb{P}_1^{(j)}} \sum_{i \in \{\mathbb{P}_1^{(j)} \backslash p\}} \sum_{u \in \mathbb{U}} d_{u,i} E_{u,i} \| \mathbf{R}_{u,i} - \mathbf{P}_{1,p} \|_2, \tag{11}$$

where $\mathbf{P}_{1,p}$ is the feature vector of the selected THUP p from the jth THUP group.

4.2.2 Intermediate Results.

WKM. The clusters obtained by WKM are called HUP clusters, and their cluster centroids are called preliminary THUPs. We denote the set of preliminary THUPs derived from all typical apps as $\mathbb{P}_1$. Based on the 76 typical apps, 220 preliminary THUPs are identified with interpretable meanings about app functionalities. Due to space constraints, only preliminary THUPs of several typical apps are shown in detail in Figure 6 and some interesting conclusions are demonstrated accordingly.

- **QQ** is an instant messaging app. THUP 1 (65.9%) represents using QQ mainly under mobile data and THUP 2 (34.1%) represents using QQ mainly under WiFi. Through averaging $E_{u,a^*,h}$ over all in-group users, we compare these two preliminary THUPs on Modem (109.4 mAh; 58.9 mAh), WiFi (2.3 mAh; 13.2 mAh), and the remaining 10 hardware components (53.1 mAh; 135.8 mAh). Two interesting facts are observed. First, it turns out that less energy consumption on the WiFi component (THUP 2) can support the intensive app usage reflected by the energy consumption on the other 10 hardware components. This indicates that WiFi is more energy-efficient than Modem in data transferring. Our finding quantitatively coincides with the conclusion made in [5]. Second, QQ users are generally sensitive to mobile data usage. Under the limited WiFi connection, users tend to use QQ less intensively, which is reflected by lower energy consumption on the other 10 hardware components for THUP 1.

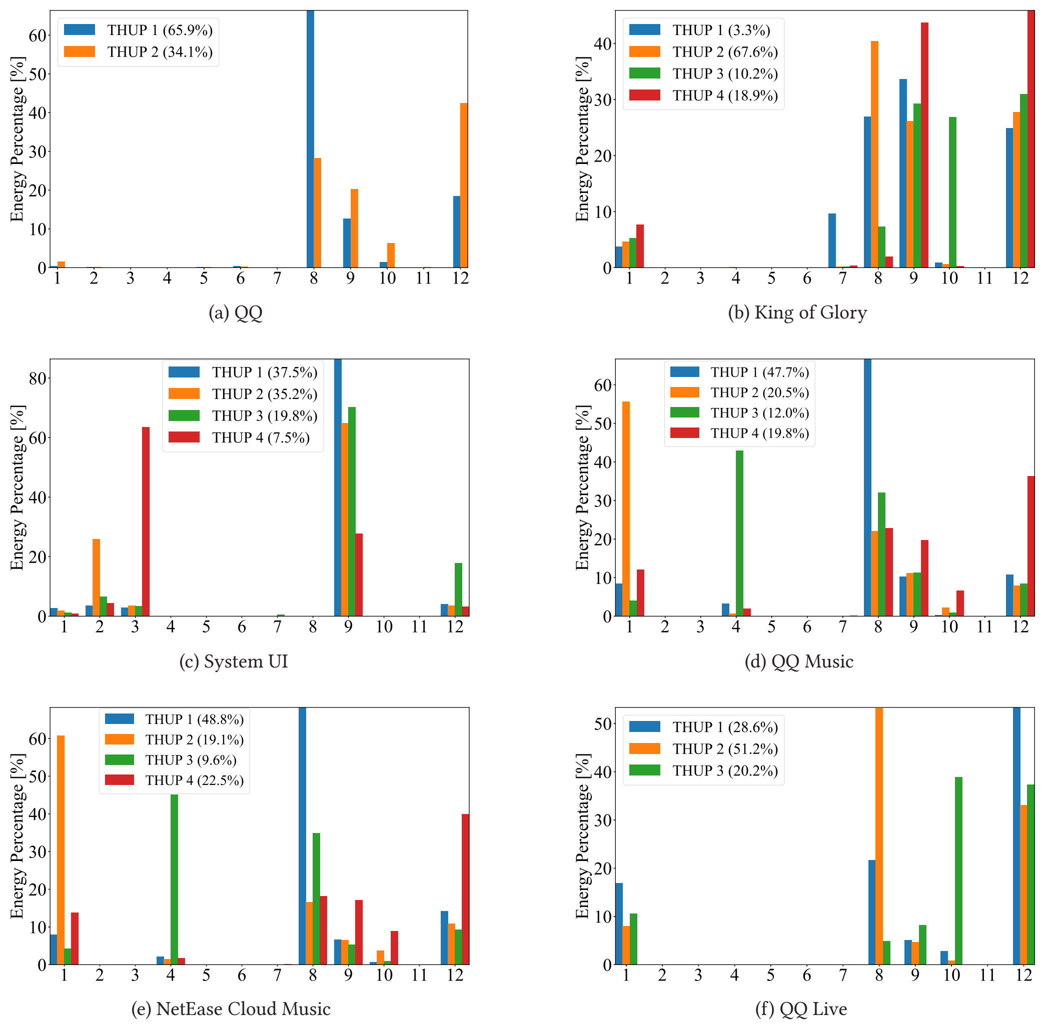


Fig. 6. Preliminary THUPs of several typical apps. The bar chart reflects app energy percentage breakdown on 12 hardware components whose indices are listed in Table 5. The percentage number denoted in the legend reflects how the population is distributed among preliminary THUPs of certain apps.

- **King of Glory** is a multiplayer mobile game. THUP 1 (3.3%) represents activating GPU rendering for better visual experiences, which is reflected by a large overhead on both GPU (73.5 mAh) and CPU (256.0 mAh). THUP 2 (67.6%) represents the usage under mobile data. THUP 3 (10.2%) represents usage under WiFi connection possibly to download large files or updates. THUP 4 (18.9%) represents the offline usage mode. Through averaging $E_{u,a^*,h}$ over in-group users, we compare THUP 2 and 3 on Modem (260.0 mAh; 42.7 mAh), WiFi (3.9 mAh; 156.4 mAh), and the other 10 hardware components (377.8 mAh; 382.7 mAh) which indicates that game players are generally less sensitive to mobile data usage considering THUP 2 and 3 have similar energy consumption on the other 10 hardware components.
- **System UI** is a persistent process for providing UI. THUP 1 (37.5%) represents limited active interaction with UI. It is reflected by the least average app energy consumption (21.4 mAh)

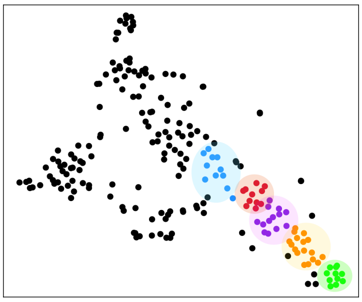


Fig. 7. t-SNE of 220 preliminary THUPs. Five largest THUP groups are colored.

Table 6. Some Preliminary THUP Examples in the Five Largest THUP Groups

Group	In-group preliminary THUP examples
Orange	**QQ1**, QQMusic1, NetEaseCloudMusic1, Kugou2
Purple	**TikTok2**, BaiduBrowser1, AndroidBrowser1
Blue	**WeChat1**, WeChat2, DingTalk1, Weibo6
Red	**QQLive2**, iQiYi1, YouKu1, QQSports1
Green	**BaiduInput1**, AppMarket1, QQDownloader1

In each group, the one marked in bold is chosen to be the ultimate THUP.

and a large overhead on the CPU component (18.5 mAh). THUP 2 (35.2%) represents activating face recognition, with high energy consumption on the Front Camera component (7.7 mAh; < 2.8 mAh for the other preliminary THUPs). THUP 3 (19.9%) represents the preference of checking news from the Notification Panel, featuring a high energy consumption on the Display component (5.4mAh; < 2.1mAh for the other preliminary THUPs). THUP 4 (7.5%) represents activating LED notification and has the highest average app energy consumption (62.5 mAh) mainly on the Flashlight component (39.8 mAh).

— **QQ Music and NetEase Cloud Music** are two music apps and preliminary THUPs discovered for them are almost identical. THUP 1 (around 50%) represents usage under mobile data. THUP 2 (around 20%) represents the preference for sodcasting, with a large app energy percentage on the Audio component (55.7% and 60.8%). THUP 3 (around 10%) represents using Bluetooth earphones and has a large app energy percentage on the Bluetooth component (43.0% and 45.1%). THUP 4 (around 20%) represents browsing contents that leads to a large app energy percentage on the Display component (36.4% and 40.0%).

— **QQ Live** is a live streaming video app. THUP 1 (28.6%) represents the offline usage mode with the highest average energy consumption on the Display component (241.2 mAh; < 137.7 mAh for the other preliminary THUPs). THUP 2 (51.2%) represents watching videos under mobile data. THUP 3 (20.1%) represents watching or downloading videos under WiFi connection.

HAC. The clusters obtained through HAC are called THUP groups. Specifically, on the training part of our dataset, HAC results in 66 THUP groups gathering from 220 preliminary THUPs. Figure 7 and Table 6 illustrate the five largest THUP groups. To visualize the relative distances among the 220 preliminary THUPs, the technique of t-SNE [33] is applied to transform high-dimensional data into two-dimensional data. Table 6 lists several representative in-group preliminary THUPs, where the number at the end of the app name is the THUP index.

It is noticed that preliminary THUPs in these groups share a common feature of high energy percentage on the Modem component, which indicates that mobile data remains a major way to access the Internet for most app usage. Besides the common feature, these THUP groups also have distinct characteristics.

— Many preliminary THUPs in the Orange group come from music apps. Besides the Modem component, their energy consumption is almost equally distributed (10%) on the CPU, Display, and Audio components.

— The Purple group is dominated by the browser category that features high energy consumption on the Modem (60%), Display (30%), and CPU (10%) components. The reason is that browsing naturally leads to a large portion of energy on displaying and loading contents.
— The Blue group is dominated by the instant message category whose energy consumption is mainly distributed on the Modem (40%), Display (40%), CPU (10%), and WiFi (5%) components. As these apps tend to be frequently used at WiFi-stable places, the WiFi component gets a visible share of the energy percentage.
— The Red group is dominated by the video category whose energy consumption is mainly distributed on the Modem (50%), Display (30%), Audio (10%), and CPU (10%) components. Compared with the Purple group, the Modem component shows a lower percentage as a result of the rise in the Audio component.
— The Green group is dominated by the utility category whose energy consumption mainly involves the Modem (80%) and Display (10%) components. Less energy consumption on the other components due to utility apps' limited functionalities explains the exclusively important role of the Modem component.

THUP filtering. An ultimate THUP is denoted as p^* and the set of ultimate THUPs is denoted as $\mathbb{P}_1^*$. The previous 66 THUP groups generate 66 ultimate THUPs. Figure 8 is an overview of the 66 ultimate THUPs. Each cell in the heatmap represents the app energy percentage on a specific hardware component and deeper color indicates a higher percentage. It is noticed that energy consumption is concentrated mainly on the Modem, CPU, WiFi, and Display components for most ultimate THUPs. Moreover, these ultimate THUPs are distinguished from each other with visibly distinct color patterns. This validates the effectiveness of HAC and THUP filtering to some degrees. In Table 6, the preliminary THUPs marked bold.

4.3 Step 3: Atypical-to-Typical Allocation

There are atypical apps and atypical HUPs of typical apps, which result from app filtering in Section 4.1 and THUP filtering in Section 4.2, respectively. Although we employ the representation with those ultimate THUPs to reduce the dimension and dispersion of user data, the energy consumption of atypical apps and atypical HUPs is not negligible. Therefore, in the meantime, we should also maintain the completeness of energy consumption for each user. To maximally maintain the original HUPs of users, we allocate both atypical apps and atypical HUPs to their most similar ultimate THUPs in terms of energy percentage breakdown on hardware components. In other words, we redistribute the energy consumption of atypical apps and atypical HUPs to be as if it is produced by ultimate THUPs. After such allocation, users can then be fully represented only with ultimate THUPs.

4.3.1 Method Description.

Atypical app allocation. The allocation of atypical apps to their most similar ultimate THUPs is conducted on a per-user basis rather than a per-app basis, as how an app is used differs from user to user (i.e., an atypical app can be allocated to distinct ultimate THUPs for different users). For example, for two users preferring different app functionalities of Facebook, one user's Facebook might be allocated to WeChat while the other user's to Baidu Browser.

Concretely, atypical app allocation proceeds as follows:

(1) For the HUP cluster whose cluster centroid is ultimate THUP $p^* \in \mathbb{P}_1^*$, we define the cluster radius $\mathrm{rad}(p^*)$ as 0.8 times of the maximum distance between in-group data points and the ultimate THUP,

$$\mathrm{rad}(p^*) = 0.8 \cdot \max_{u \in \mathbb{U}} d_{u,p^*} \cdot \|\mathbf{R}_{u,a} - \mathbf{P}_{1,p^*}\|_2. \tag{12}$$

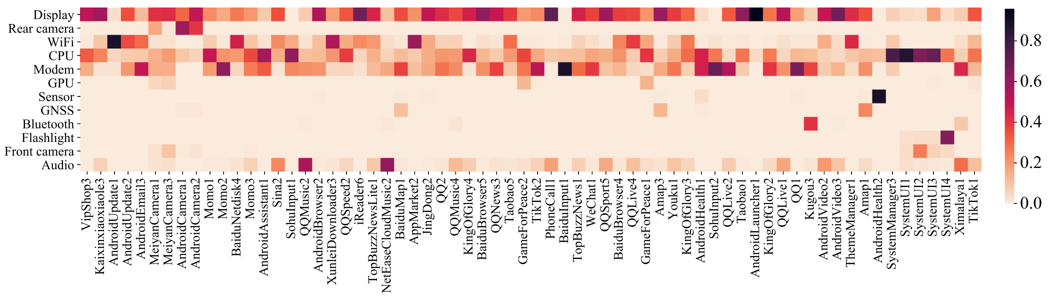


Fig. 8. The heatmap of 66 ultimate THUPs represented by energy percentage breakdown on 12 hardware components.

(2) For each record of atypical app a used by user u, we calculate the Euclidean distance $d(\mathbf{R}_{u,a}, \mathbf{P}_{1,p^*})$ between the HUP reflected by this record and each ultimate THUP $p^* \in \mathbb{P}_1^*$. We denote the closest ultimate THUP as $p_{u,a}^*$.

(3) If $d(\mathbf{R}_{u,a}, \mathbf{P}_{1,p_{u,a}^*}) \leq \text{rad}(p_{u,a}^*)$, we allocate the energy consumption of this record to the ultimate THUP $p_{u,a}^*$ (i.e., the record will be replaced by this ultimate THUP with the same energy consumption). Moreover, the usage time of this record is also adjusted according to the average energy drain speed of the ultimate THUP,

$$T_{u,a} = \frac{E_{u,a}}{\overline{E}_{p_{u,a}^*}} \overline{T}_{p_{u,a}^*}, \tag{13}$$

where $\overline{E}_{p_{u,a}^*}$ and $\overline{T}_{p_{u,a}^*}$ are the average energy consumption and usage time of the HUP cluster whose centroid is THUP $p_{u,a}^*$, respectively.

(4) Otherwise, if $d(\mathbf{R}_{u,a}, \mathbf{P}_{1,p_{u,a}^*}) > \text{rad}(p_{u,a}^*)$, we allocate the energy consumption of this record to an artificially designed THUPs: Virtual. Virtual does not have the attribute of usage time, as it only serves as the container to collect the energy consumption of those records that are similar to none of the ultimate THUPs.

Atypical HUP allocation. The main idea of atypical HUP allocation is the same as that of atypical app allocation. However, it can be implemented with a lower computational cost because the HAC guarantees that ultimate THUPs and atypical HUPs within the same THUP group are closely located and considered as similar. To this end, the energy consumption of those atypical HUPs can be allocated to their corresponding ultimate THUPs in the same THUP group directly without calculating and comparing the Euclidean distance with all ultimate THUPs. In the meantime, the usage time of atypical HUPs is also adjusted according to the average energy drain speed of their corresponding ultimate THUPs.

After the atypical-to-typical allocation, Virtual is added to the set of ultimate THUPs $\mathbb{P}_1^*$, which now consists of 67 ultimate THUPs. We will then represent the smartphone usage of all users with these 67 ultimate THUPs, i.e., the feature vector of each user is represented by energy (or percentage) breakdown on these ultimate THUPs.

4.4 Step 4: TAUP Identification

After the three steps above, the energy consumption data boils down to the feature vectors of users represented with a small number of ultimate THUPs, and all energy consumption is reserved.

While ultimate THUPs reveal app functionalities (i.e., app's energy percentage breakdown on hardware components), this step TAUP Identification aims at exploring the pattern of user preference on these identified ultimate THUPs (i.e., smartphone energy percentage breakdown on ultimate THUPs). This step also constitutes the lower layer of the pattern layering mechanism, as shown in Figure 2.

4.4.1 Method Description. Mathematically, user u's App Usage Pattern (AUP) is denoted as a $|\mathbb{P}_1^*|$-dimensional vector

$$\mathbf{R}_u = \left[r_{u,p^*}\right]_{1\times|\mathbb{P}_1^*|}, \tag{14}$$

whose element $r_{u,p^*} = \frac{E_{u,p^*}}{E_u}$ is the smartphone energy percentage on the THUP $p^* \in \mathbb{P}_1^*$. Notably, $\mathbf{R}_u$ also has unit ℓ_1 norm, as the atypical-to-typical allocation step ensures that user u's smartphone energy consumption is fully assigned to the ultimate THUPs.

Similar to the THUP identification, WKM is employed across all users, while the weight of each user u is defined as its smartphone energy consumption E_u. The number of cluster K is determined by the clustering performance score q,

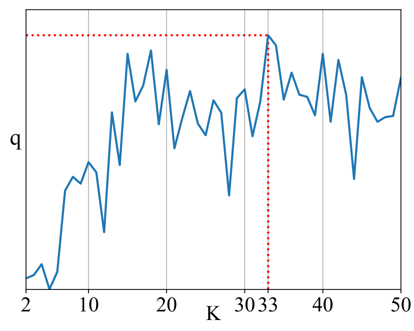
$$q = 0.3 \times (1 - \mathrm{mdp}) + 0.3 \times s + 0.25 \times e + 0.15 \times c, \tag{15}$$

where mdp is the Mean Discrepancy on ultimate THUPs, defined as the weighted root mean squared error between in-group AUPs and the corresponding TAUPs,

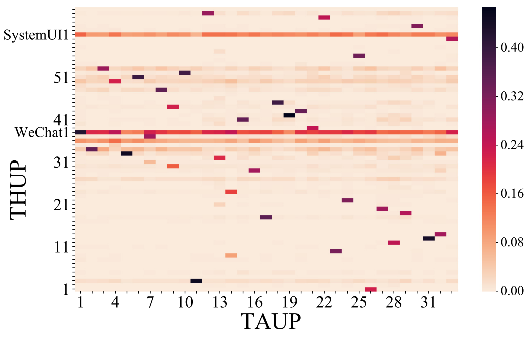
$$\mathrm{mdp} = \frac{\sum_{k=1}^{K}\sum_{u\in\mathbb{U}} l_{u,k}E_u\|\mathbf{R}_u - \mathbf{P}_2^{(k)}\|_2}{\sum_{k=1}^{K}\sum_{u\in\mathbb{U}} l_{u,k}E_u}, \tag{16}$$

where $\mathbf{P}_2^{(k)}$ is the feature vector of the kth TAUP; s and e stay the same as those in Weighted K-Means in THUP identification (Section 4.2.1); c is also defined the same as Equation (10) but with $K_{\max} = 50$ by default in this step. Weights of the four components are also determined in a heuristic way: equal weight is the initial setting, and the stopping criterion is to obtain a q–K curve that does not have an overall monotonous tendency but a peak of q falling on a suitable K within the range of $[2, 50]$. Comparedwith the weight fine-tuning process in THUP identification, as we raise the upper limit $K_{\max}$ from 6 to 50, we allocate 0.05 more weight on c in this step to match its change rate with other factors so as to result in a non-monotone q–K curve.

4.4.2 Intermediate Results. After conducting WKM in this step, TAUPs are identified as the cluster centroids of AUP clusters. At the same time, users are divided into different user groups corresponding to these AUP clusters. The results of TAUP identification are demonstrated in Figure 9. The values of q with respect to different numbers of clusters K's are illustrated in Figure 9(a) where K ranges from 2 to 50. It is observed that the largest value of q is achieved when $K = 33$ (mdp $= 0.25$, $s = 0.18$, $e = 3.64$, $c = 0.34$ before normalization). Correspondingly, users are thus divided into 33 user groups. Figure 9(b) shows in the form of heatmap the identified 33 TAUPs represented by smartphone energy percentage breakdown on 67 ultimate THUPs. Along the horizontal axis, TAUPs are indexed from 1 to 33 according to the size of the user group (i.e., TAUP 1 corresponds to the largest user group). Along the vertical axis, THUPs follow the same order as shown in Figure 8. Each column in the heatmap corresponds to a TAUP represented by the energy percentage breakdown on ultimate THUPs. It is observed that almost all TAUPs feature at least one THUP whose energy percentage is much higher than that of the other TAUPs. This implies that TAUPs are well separated on at least one dimension. The



(a) The *q*-*K* curve with the peak at $K = 33$.



(b) The heatmap of 33 TAUPs.

TAUP index	**1**	**2**	**3**	**4**	**5**	**6**	**7**
THUP	WeChat1	TikTok2	QQ2	Launcher1	GameForPeace2	KingOfGlory2	TopBuzzNews1 1
TAUP index	**8**	**9**	**10**	**11**	**12**	**13**	**14**
THUP	QQLive2	Youku	QQLive1	AndroidUpdate1	TikTok1	Taobao5	BaiduMap1
TAUP index	**15**	**16**	**17**	**18**	**19**	**20**	**21**
THUP	QQLive4	KingOfGlory4	AndroidBrowser2	KingOfGlory3	GameForPeace1	Amap3	QQSport5
TAUP index	**22**	**23**	**24**	**25**	**26**	**27**	**28**
THUP	Ximalaya1	Momo1	TopBuzzNewsLite1	AndroidVideo3	VipShop3	QQSpeed2	BaiduNetdisk4
TAUP index	**29**	**30**	**31**	**32**	**33**		
THUP	XunleiDownloader3	SystemUI3	Momo3	AndroidAssistant1	SystemManager3		

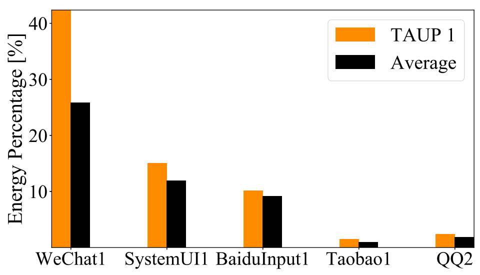
(c) THUPs with the most outstanding energy percentage in each TAUP.

Fig. 9. Results of TAUP identification.

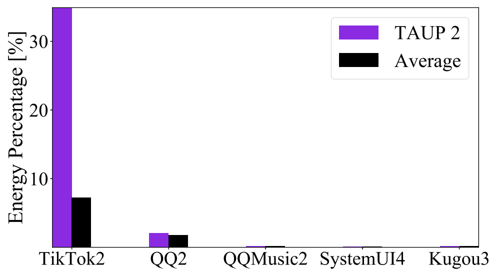
ultimate THUPs that have the most outstanding energy percentage within each TAUP are listed in Table 9(c).

To have a better understanding of the characteristics of the identified TAUPs, the heatmap (Figure 9(b)) is further analyzed in detail. As it is noticed that WeChat1 and SystemUI1 have high energy percentages for almost all TAUPs, we thus further investigate these two THUPs in terms of the different effects of allocation on them. Specifically, WeChat1 holds a large portion of original energy (97.38%) (i.e., few atypical apps or atypical HUPs are allocated to WeChat1). Therefore, WeChat1's overall high energy percentages demonstrate the exclusive role of WeChat in smartphone normal daily use. This finding coincides with WeChat's high app typicality showed in Section 4.1. In contrast, SystemUI1 has the majority of its energy consumption from allocation (98.36%), especially from system processes such as *system_server_HeapTaskDaemon*, *system_server_Binder*, and *kernel_kworker*. This indicates that system processes consume a non-negligible amount of energy in smartphone normal daily use for all user groups.

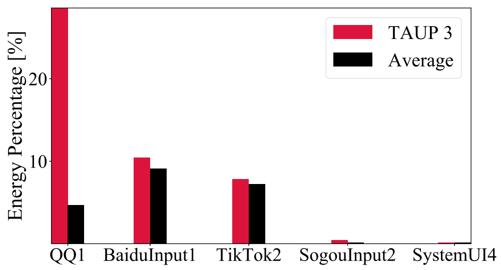
Due to space constraints, as shown in Figure 10, only TAUPs of the five largest user groups are illustrated. To better capture the most significant characteristics of these TAUPs, they are, respectively compared with the average AUP (i.e., averaged energy percentage breakdown on ultimate THUPs among entire users regardless of user groups). For each of these TAUPs, top five ultimate THUPs with the largest energy percentage difference from the average one are shown. We defer the detailed discussion about these TAUPs to Section 4.5.2, presenting together with the intermediate results of UID fitting.



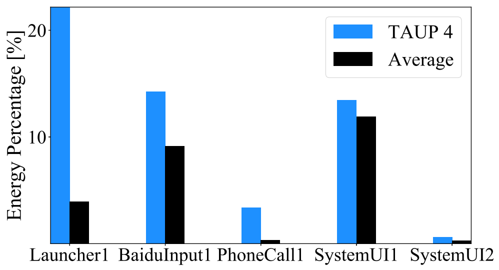
(a) TAUP 1 (49,035 users).



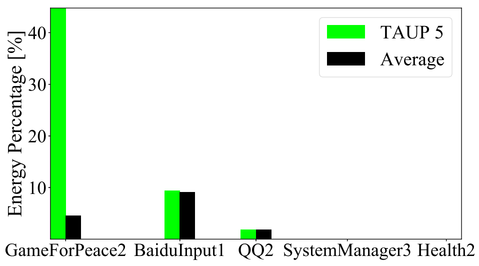
(b) TAUP 2 (12,091 users).



(c) TAUP 3 (9,016 users).



(d) TAUP 4 (8,733 users).



(e) TAUP 5 (6,704 users).

Fig. 10. TAUPs of the five largest user groups. Black bars represent energy percentage of the average AUP whereas the colored bars represent energy percentage of the corresponding user group.

4.5 Step 5: UID Fitting

After the investigation of users' usage pattern that results in ultimate THUPs (Section 4.2) and TAUPs (Section 4.4), users with similar usage patterns are finally grouped together into user groups. However, usage intensity still varies among different users even within the same user group. To this end, The usage intensity is explored independently in this step.

4.5.1 Method Description. Usage intensity of user u is a scalar represented by the smartphone energy consumption $E_u = \sum_{a \in \mathbb{A}} \sum_{h \in \mathbb{H}} E_{u,a,h}$, which is actually the ℓ_1 norm of users' original feature vector $\mathbf{E}_u = [E_{u,a,h}]_{1\times|\mathbb{A}||\mathbb{H}|}$. UID is studied through fitting a **Probability Density Function** (**PDF**) in terms of these usage intensities for each user group that is identified in the TAUP identification step.

Concretely, for anytwo users u_1 and u_2 within the same kth user group, we denote their usage intensities as E_{u_1} and E_{u_2}, respectively. Then the following relationship can be obtained if their

HUPs and AUP are represented by corresponding THUPs and TAUP:

$$\frac{E_{u_1}}{E_{u_2}} = \frac{r_{u_1,p^*} E_{u_1}}{r_{u_2,p^*} E_{u_2}} = \frac{E_{u_1,p^*}}{E_{u_2,p^*}} = \frac{T_{u_1,p^*}}{T_{u_2,p^*}}, \tag{17}$$

where $r_{u_1,p^*} = r_{u_2,p^*} = r_{p^*}^{(k)}$ since u_1 and u_2 are represented with the same TAUP; $E_{u_1,p^*}/E_{u_2,p^*} = T_{u_1,p^*}/T_{u_2,p^*}$ since energy drain speed is considered to be constant for each ultimate THUP, given in Equation (13). Moreover, we also have

$$\frac{E_{u_1}}{E_{u_2}} = \frac{r_{u_1,p^*,h} E_{u_1,p^*}}{r_{u_2,p^*,h} E_{u_2,p^*}} = \frac{E_{u_1,p^*,h}}{E_{u_2,p^*,h}}, \tag{18}$$

where $r_{u_1,p^*,h} = r_{u_2,p^*,h} = r_{p^*,h}$, since u_1 and u_2 are represented with the same ultimate THUP p^*. Besides, we have

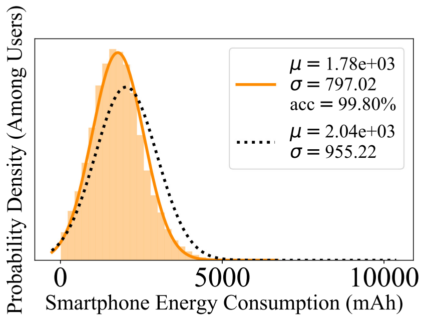
$$\frac{E_{u_1}}{E_{u_2}} = \frac{\sum_{p^* \in \mathbb{P}_1^*} E_{u_1,p^*,h}}{\sum_{p^* \in \mathbb{P}_1^*} E_{u_2,p^*,h}} = \frac{E_{u_1,h}}{E_{u_2,h}}. \tag{19}$$

Therefore, when representing different users in the same user group with the corresponding TAUP and ultimate THUPs, the usage intensity is proportionally linear to: (1) the energy consumption on any of the ultimate THUPs (E_{u,p^*}); (2) the energy consumption on any of the hardware components for any of the ultimate THUPs ($E_{u,p^*,h}$); (3) the energy consumption on any of the hardware components ($E_{u,h}$); and (4) the usage time on any of the ultimate THUPs (T_{u,p^*}). This implies that the distribution of each of these four quantities can be derived from the UID fitted with the usage intensity. Moreover, such a linear relationship guarantees that any value of the usage intensity on the UID curve can be reproduced through proportionally scaling the usage time on ultimate THUPs.

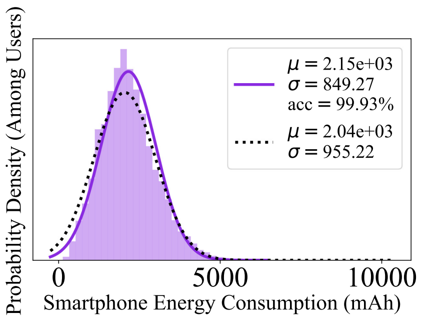
4.5.2 Intermediate Results. In our scenario, normal distribution is employed for fitting the UIDs. We define the fitting accuracy *acc* for each user group as the ratio of the area under the positive part (i.e., positive value along the *x*-axis) of the fitted PDF curve to the sum of smartphone energy consumption of users in that user group. Corresponding to the TAUPs in Figure 10, UIDs of the five largest user groups are illustrated in Figure 11. The UID for each user group is also compared with the UID fitted with entire users regardless of user groups to better capture their respective characteristics.

Jointly investigating the TAUPs and their corresponding UIDs can lead to some interesting observations and further insights about the user groups. Recall that average AUP represents the averaged energy percentage breakdown on ultimate THUPs among entire users regardless of user groups. It can be observed that the five largest user groups share several common characteristics: (1) the fitting accuracy *acc* of their UIDs all exceeds 98%, which indicates that normal distribution can properly capture the usage intensity distribution among users in these user groups; (2) TAUP of each group has at least one distinctive ultimate THUP whose energy percentage significantly exceeds that of the average AUP; this quantitatively reveals that the identified TAUPs and user groups are well distinguished from each other; and (3) Modem-intensive ultimate THUPs still hold the leading position (refer to Figure 8), which coincides with the insight obtained in Section 4.2. Besides these similarities, each of these five user groups possesses some distinctive features.

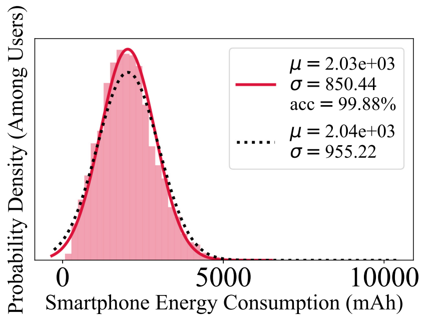
— **TAUP 1** associated with user group 1 is the pattern covering the largest population (35.74%). We label this TAUP as **WeChat1-intensive usage pattern**. The result is consistent with the conclusion in Section 4.1 that WeChat is the most widely used app. Concretely, in terms of WeChat1 (i.e., WeChat usage pattern with Modem), its energy percentage is 16.50% higher



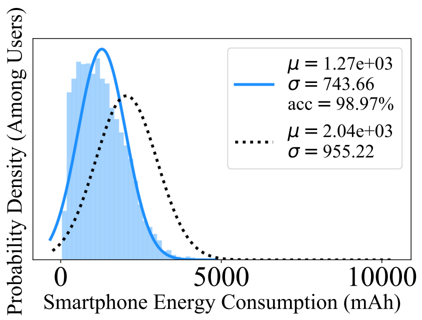
(a) UID 1 (49,035 users).



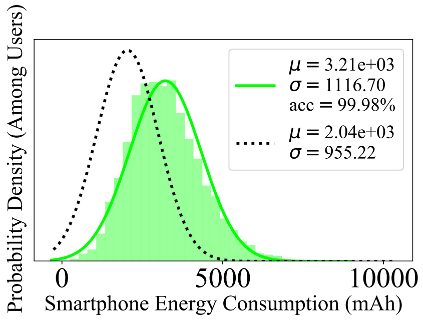
(b) UID 2 (12,091 users).



(c) UID 3 (9,016 users).



(d) UID 4 (8,733 users).



(e) UID 5 (6,704 users).

Fig. 11. UIDs of the five largest user groups. The black dash line represents the probability density function (PDF) of the average AUP whereas the colored solid line represents the fitted PDF of the corresponding user group. The shadowed area represents the true probability distribution of the corresponding user group.

and its energy consumption is 227.06 mAh higher compared with the average AUP. Moreover, SystemUI1 and BaiduInput1 follow right behind WeChat1 probably because the intensive WeChat usage is usually accompanied by intensive message notification (SystemUI1) and intensive text input (BaiduInput1). It is also noticed that the average smartphone energy consumption of user group 1 (1,779.71 mAh) is lower than that of all users (2,036.67 mAh). Therefore, the profile of this user group is likely to be those young people who live in cities and are active in social networking and online shopping. They tend to spend their most time on socializing through WeChat and online shopping through Taobao, which are not very energy-consuming compared with the usage of video or game apps.

— **TAUP 2** associated with user group 2 is the pattern covering the second largest population (8.81%), which can be labeled as **TikTok2-intensive usage pattern**. TikTok is a popular social media short video app ranking third worldwide by November 2018 according to Google Play download statistics [14]. In terms of TikTok2, TAUP 2 has 27.67% higher on energy percentage and 603.68 mAh higher on energy consumption than the average AUP. In the meantime, the average smartphone energy consumption of user group 2 (2,151.25 mAh) is 114.58 mAh higher than that of the average among all users, since video apps are usually energy-consuming. Apart from the high dependence on TikTok2 (the most significant characteristic of this TAUP), QQ2, QQMusic2, and Kugou3 also rank top. Therefore, the user profile of this user group can be described as those people who are interested in multimedia players and especially active in the community of TikTok.

— **TAUP 3** is the pattern associated with user group 3 covering the third largest population (6.75%), which can be labeled as **QQ1-intensive usage pattern**. In terms of QQ1, TAUP 3 has 23.91% higher on energy percentage and 485.36 mAh higher on energy consumption than the average AUP. Compared with QQ2 (42.53% on Display; 28.32% on Modem), QQ1 (18.48% on Display; 66.38% on Modem) represents a more popular usage pattern featuring higher energy percentage on the Modem component. Moreover, the average smartphone energy consumption of user group 3 (2,031.35 mAh) is close to the average among all users. Interestingly, both TAUP 2 (TikTok2-intensive) and TAUP 3 (QQ1-intensive) share the appearance of SystemUI4 (i.e., LED message notification) as top five. Therefore, the profile of users in this group is likely to be those teenagers who rely on QQ to socialize and love playing TikTok. They are supposed to be younger than the users in user group 1 because (1) the target customers of QQ are mainly teenagers; and (2) they demonstrate lower purchase power compared with the users in user group 1.

— **TAUP 4** is the pattern associated with user group 4 covering the fourth largest population (6.36%), which can be labeled as **Traditional usage pattern**. It is noticed that top five ultimate THUPs of TAUP 4 all come from apps in system category: Android Launcher, Baidu Input, Phone Call, and System UI. The relatively high energy percentage on system-related ultimate THUPs implies lower energy percentage on other ultimate THUPs from third party apps. It is also observed that compared with the average AUP, TAUP 4's energy consumption is 202.59 mAh higher on AndroidLauncher1, 36.00 mAh higher on PhoneCall1, and 71.19 mAh lower on SystemUI1. This may be caused by more foreground display time on Android Launcher, more usage of Phone Call for communication in the traditional way, and less notification on System UI due to relatively less usage of third party apps. Moreover, the average smartphone energy consumption of user group 4 (1,274.58 mAh) is 762.09 mAh lower than the average among all users. All the above evidence indicates that users in this group interact with their smartphones less frequently and tend to utilize only the basic functionalities of smartphones instead of popular third party apps. Aged people are probably more likely to stay in this user group.

— **TAUP 5** is the pattern associated with user group 5 covering the fifth largest population (4.89%), which can be labeled as **GameForPeace2-intensive usage pattern**. Game for Peace is a worldwide prevailing multiplayer mobile game. In terms of GameForPeace2, TAUP 5 has 40.21% higher on energy percentage and 1346.46 mAh higher on energy consumption than the average AUP. As game apps are extremely energy-consuming especially when being used under Modem connection, the average smartphone energy consumption of user group 5 (3,214.42 mAh) is 1,177.75 mAh higher than that of the average among all users. The

Table 7. Part of the UPM 1

	Total	Audio	FrontCam	Flashlight	Bluetooth	GNSS	Sensor	GPU	Modem	CPU	WiFi	RearCam	Display
WeChat1	42.36%	0.61%	0.11%	0.00%	0.01%	0.03%	0.07%	0.02%	16.64%	7.42%	1.35%	0.10%	16.01%
SystemUI1	15.01%	0.41%	0.54%	0.44%	0.00%	0.00%	0.00%	0.02%	0.00%	12.97%	0.00%	0.00%	0.62%
BaiduInput1	10.15%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	9.00%	1.06%	0.07%	0.00%	0.01%
Taobao1	1.50%	0.01%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.13%	0.41%	0.02%	0.00%	0.93%
QQ2	2.33%	0.04%	0.00%	0.00%	0.00%	0.00%	0.01%	0.00%	0.66%	0.47%	0.15%	0.00%	0.99%

Table 8. Part of a Concretized UBM 1 with the Sampled Smartphone Energy Consumption of 1,779.71 mAh

	T_{p^*} (min)	E_{p^*} (mAh)	Audio	FrontCam	Flashlight	Bluetooth	GNSS	Sensor	GPU	Modem	CPU	WiFi	RearCam	Display
WeChat1	132.43	753.93	10.93	1.95	0.00	0.13	0.46	1.24	0.36	296.20	132.07	23.95	1.70	284.95
SystemUI1	5.16	267.17	7.37	9.62	7.89	0.07	0.00	0.00	0.31	0.00	230.86	0.00	0.00	11.05
BaiduInput1	0.06	180.59	0.07	0.00	0.00	0.00	0.00	0.00	0.00	160.17	18.92	1.30	0.00	0.13
Taobao1	7.82	26.77	0.17	0.00	0.00	0.00	0.00	0.00	0.05	2.26	7.35	0.34	0.01	16.59
QQ2	8.72	41.48	0.66	0.08	0.00	0.00	0.08	0.13	0.03	11.75	8.43	2.64	0.04	17.64

user profile of this group is likely to be those faithful mobile game lovers who may also have the most urgent demand for the improvement on smartphone energy performance.

4.6 Final results: UBM Establishment

After completing all the five steps above, UBMs can finally be established. Concretely, each UBM is a union of the following:

(1) **Usage Intensity Distribution (UID)**: for each user group, UID is obtained directly from the UID fitting step (Section 4.5).

(2) **Usage Pattern Model (UPM)**: for each user group, UPM is established by consolidating the feature vectors of all ultimate THUPs with the corresponding TAUP of this user group. The consolidated outcome for the kth user group is represented in the feature vector

$$\overline{\mathbf{P}}^{(k)} = \left[r_{p^*}^{(k)} \cdot r_{p^*,h}\right]_{1\times|\mathbb{P}_1^*||\mathbb{H}|}, \tag{20}$$

where $r_{p^*}^{(k)}$ represents the smartphone energy percentage on THUP p^* for the kth user group and $r_{p^*,h}$ represents THUP p^*'s energy percentage on hardware component h. Table 7 takes UPM 1 as an example to show how a UPM looks like. The column "Total" represents the feature vector of TAUP ($\mathbf{P}_2^{(k)} = [r_{p^*}^{(k)}]_{1\times|\mathbb{P}_1^*|}$), and other columns distribute the energy percentage of $r_{p^*}^{(k)}$ to 12 hardware components according to the feature vector of the corresponding ultimate THUP p^* ($\mathbf{P}_{1,p^*} = [r_{p^*,h}]_{1\times|\mathbb{H}|}$).

For each user group, the union of its UID and UPM generates its UBM. Through sampling a certain value of usage intensity from the UID and multiplying it to each entry of the corresponding UPM, we can obtain a concretized UBM (i.e., a concrete UBM instance under certain smartphone energy consumption) represented by the smartphone energy breakdown on hardware components of different ultimate THUPs. Therefore, an infinite number of concretized UBMs can be extended from a single pair of UPM and UID. Such an extension requires little cost of modeling, just by simply proportionally scaling the usage time (thus also energy consumption) of the ultimate THUPs as given in Equation (17).

In Table 8, one of the concretized UBM 1's is taken as an example to show how a concretized UBM looks like. Specifically, it provides values of usage time and energy consumption on each of the ultimate THUPs, along with its energy breakdown on hardware components. In this example, UBM 1 represents the WeChat1-intensive user group. And with usage intensity of 1,779.71 mAh, it is found that WeChat1's usage time reaches 132.43 min and its energy consumption reaches 753.93 mAh (only 22.91 mAh comes from allocation). Such a concretized UBM is thus capable of providing quantitative evidence about smartphone normal daily use.

5 PERFORMANCE EVALUATION

According to Section 4.6, we establish 33 UBMs to represent smartphone usage of all users. In this section, we first evaluate the performance of the 33 established UBMs, and then compare our approach E-Sub with other approaches.

5.1 UBM Performance Evaluation

As each UBM being a union of the corresponding UID and UPM, the performance of UBM is evaluated on the test set in terms of UID and UPM, respectively. The evaluation is conducted from two aspects: (1) behavior coverage; and (2) energy discrepancy.

5.1.1 Behavior Coverage. Behavior coverage measures UPM's ability to cover the diversity of user behavior (i.e., successfully representing the arbitrary usage pattern). It is determined by UPM because UID can involve any value of usage intensity. Specifically, given a new user's data record of smartphone energy consumption, this user can be represented with one of the established UPMs only under certain conditions. For each user in the test set, a successful representation is considered to satisfy the following:

(1) A successful THUP allocation: following the process of atypical app allocation (Section 4.3.1), more than 60% of apps can be allocated to ultimate THUPs with promising similarity instead of Virtual;
(2) A successful TAUP representation: the user's AUP can perfectly fall into one of the user groups (i.e., within 0.8 times of the maximum distance between in-group AUP and corresponding TAUP);

The behavior coverage is defined as the percentage of new users that can be successfully represented according to the two criteria above. Experiments on the test set shows that our established 33 UPMs are able to reach **93.97%** in terms of behavior coverage. This implies that our small number of UPMs can well represent the original highly diverse usage patterns.

5.1.2 Energy Discrepancy. Energy discrepancy involves the perspectives of both UID and UPM. The UID side is measured by KL divergence, which indicates whether the UIDs obtained from the training set can be generalized to the test set. The UPM side is quantified by the mean energy discrepancy under different granularities (i.e., hardware components with / without distinguishing ultimate THUPs). Notably, the energy discrepancy of UID and UPM is calculated based on those users that are successfully represented (i.e., 93.97% new users of the test set in our experiment).

KL Divergence of UID. KLD is the divergence between UIDs obtained from the training part and those obtained from the test part. It evaluates the UIDs' generalization ability. The smaller the KLD, the better the generalization ability of UIDs. Denote the kth UID on the training part as $f^{(k)}$, and the UID of users in the test part who are successfully represented by kth UPM as $\hat{f}^{(k)}$. Then the KLD is calculated as the weighted sum of KL divergence for all K user groups:

$$\mathrm{KLD} = \sum_{k=1}^{K} \rho_k \mathrm{KLD}_k, \quad \mathrm{KLD}_k = \int_0^{\infty} f^{(k)}(x) \log\left(\frac{f^{(k)}(x)}{\hat{f}^{(k)}(x)}\right) \mathrm{d}x, \tag{21}$$

where ρ_k is the normalized number of users in the kth user groups established on the training set.

Mean Energy Discrepancy of UPM. It is defined as the weighted root mean squared error in terms of energy consumption between the representation of usage patterns with UPMs and the usage patterns manifested in reality. It is measured under different granularities.

— MDPH measures under the granularity of hardware components with respect to different ultimate THUPs:

$$\text{MDPH} = \frac{\sum_{k=1}^{K} \sum_{u \in \mathbb{U}} l_{u,k} E_u \| \overline{\mathbf{R}}_u - \overline{\mathbf{P}}^{(k)} \|_2}{\sum_{k=1}^{K} \sum_{u \in \mathbb{U}} l_{u,k} E_u}, \tag{22}$$

where

$$\overline{\mathbf{R}}_u = \left[r_{u,p^*} \cdot r_{u,p^*,h} \right]_{1 \times |\mathbb{P}_1^*||\mathbb{H}|}, \quad \overline{\mathbf{P}}^{(k)} = \left[r_{p^*}^{(k)} \cdot r_{p^*,h} \right]_{1 \times |\mathbb{P}_1^*||\mathbb{H}|}$$

are the user u's and kth UPM's smartphone energy percentage breakdown on hardware components with respect to different ultimate THUPs, respectively.

— MDH measures under the granularity of the hardware components themselves:

$$\text{MDH} = \frac{\sum_{k=1}^{K} \sum_{u \in \mathbb{U}} l_{u,k} E_u \| \widehat{\mathbf{R}}_u - \widehat{\mathbf{P}}^{(k)} \|_2}{\sum_{k=1}^{K} \sum_{u \in \mathbb{U}} l_{u,k} E_u}, \tag{23}$$

where $\widehat{\mathbf{R}}_u$ and $\widehat{\mathbf{P}}^{(k)}$ are the user u's and kth UPM's smartphone energy percentage breakdown on hardware components. They are $|\mathbb{H}|$-dimensional vectors whose elements are $\sum_{p^* \in \mathbb{P}_1^*} (r_{u,p^*} \cdot r_{u,p^*,h})$ for $\widehat{\mathbf{R}}_u$ and $\sum_{p^* \in \mathbb{P}_1^*} (r_{p^*}^{(k)} \cdot r_{p^*,h})$ for $\widehat{\mathbf{P}}^{(k)}$. It is obvious that MDH is less fine-grained than MDPH.

5.2 Performance Comparison

To the best of our knowledge, there is no related work that has reported similar results so far. Therefore, we compare the performance of UBM established by our proposed approach E-Sub with that established by the four simplified versions of E-Sub and the baseline method (i.e., PCA). Even though PCA is one of the most basic dimension reduction techniques, it is more suitable in our scenario, compared with other more sophisticated nonlinear dimension reduction techniques (e.g., Isomap [32]). First, PCA perfectly fits in our dataset, which is simply defined in the traditional Euclidean space and is not supposed to possess an obvious low-dimensional manifold structure. Second, PCA also meets our requirements of Euclidean distance-preserving for dimension reduction (i.e., two points with large Euclidean distance should also have a large distance in the space after dimension reduction). It is because the following user clustering steps are based on Euclidean distance. Therefore, PCA is an appropriate baseline to reduce user behavior diversity directly while maintaining the most important information. To make the results of E-Sub and PCA comparable, it is noted that we set PCA to reduce the original data records to the same size of dimension as E-Sub (i.e., 67 dimensions). Table 9 describes the approaches for comparison in detail.

The results of three energy discrepancy metrics (KLD, MDPH, and MDH) are given in Table 10. It is known that these metrics are dependent on the number of clusters. Thus, to ensure fairness of performance comparison, we fix $K = 33$ for all approaches. It turns out that E-Sub outperforms the rest of the approaches in terms of all metrics. Figure 12 shows the CDF curves (i.e., Cumulative Distribution Function) of the six approaches with respect to KLD, MDPH, and MDH, respectively, when K ranges from 2 to 50.

— **PCA** serves as the baseline method. Table 10 shows that when $K = 33$, E-Sub and all of its variants outperform PCA except for MDPH of NoHAC. Compared with E-Sub, the performance of PCA on KLD, MDPH, and MDH is degraded by 0.1497, 19.51%, and 9.59%, respectively. Based on Figure 12, though PCA can achieve small KLD initially, the performance becomes

Table 9. Description of the Approaches for Comparison

		Our E-Sub	**NoAUP**	**NoLayer**	**NoHAC**	**NoAllo**	**PCA** (baseline)
App Filtering		✓	✓	✓	✓	✓	×
THUP Identification	WKM	✓	✓	×	✓	✓	×
	HAC	✓	✓	×	×	✓	×
	THUP filtering	✓	✓	×	×	✓	×
Atypical-to-Typical Allocation		✓	✓	✓	✓	×	×
TAUP Identification		✓	partially	✓	✓	✓	partially
UID Fitting		✓	✓	✓	✓	✓	✓

Note: "Partially" in terms of TAUP identification means that users are clustered without their feature vectors being decomposed into usage pattern and usage intensity.

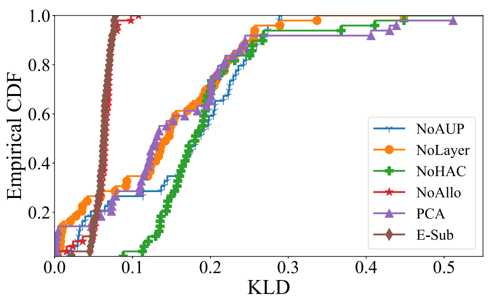
Table 10. Performance Comparison in Terms of Energy Discrepancy When $K = 33$

	KLD	MDPH	MDH
NoAUP	0.1369	26.02%	27.43%
NoLayer	0.1477	28.15%	17.11%
NoHAC	0.1355	33.93%	28.24%
NoAllo	0.0666	24.92%	29.77%
PCA (baseline)	0.1951	32.12%	33.93%
E-Sub	**0.0454**	**22.53%**	**14.42%**

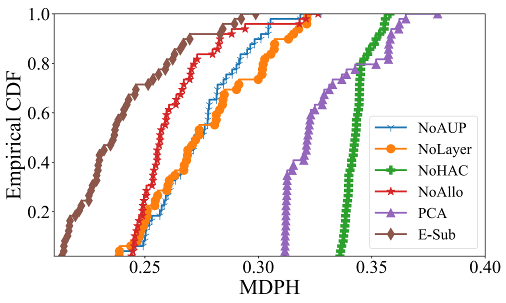
unreliable considering the long tail its CDF curve (i.e., reaching very large values of KLD later). In terms of MDPH, the value of PCA is more than 30% larger than E-Sub under any K from 2 to 50. For MDH, PCA performs the worst in that the smallest value is larger than all values of the other approaches. Moreover, results obtained by PCA also suffer from poor interpretability due to the destroyed physical meaning of dimensions after reduction. Such an outcome indicates that our proposed mechanisms for addressing the challenge of high data diversity are far more effective than directly reducing the dimension of data.

— For **NoAUP**, TAUPs are identified without explicitly decomposing data records into usage pattern and usage intensity (i.e., Weighted K-Means is implemented upon feature vectors not scaled to unit ℓ_1 norm). From Table 10 and Figure 12, it is found that NoAUP's KLD and MDH degrade more significantly than MDPH. Even for MDPH, its CDF curve is situated entirely below that of E-Sub. Therefore, we can conclude that the data dispersion caused by various ℓ_1 norm of vectors (i.e., usage intensities) indeed leads to some degrees of inefficiency in discovering similar user behaviors.

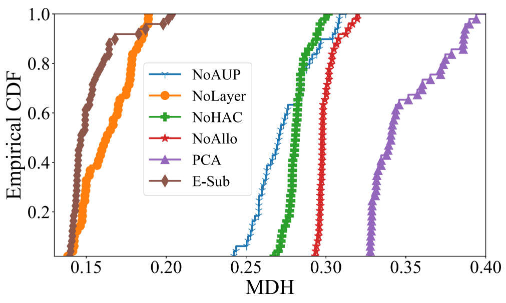
— **NoLayer**'s KLD and MDPH perform far worse than those of E-Sub. For MDH, NoLayer performs most closely to E-Sub than the other approaches. The reason may be that NoLayer conducts clustering directly on hardware components. However, the majority of NoLayer's MDH CDF curve is still situated below that of E-Sub. This indicates that it is useful to investigate the usage pattern hierarchically instead of directly on hardware components. Moreover, NoLayer can hardly provide complete insights and interpretability about how various typical apps are used, as it fails to dive into each THUP of these apps.



(a) KL divergence of UID.



(b) MDPH of UPM.



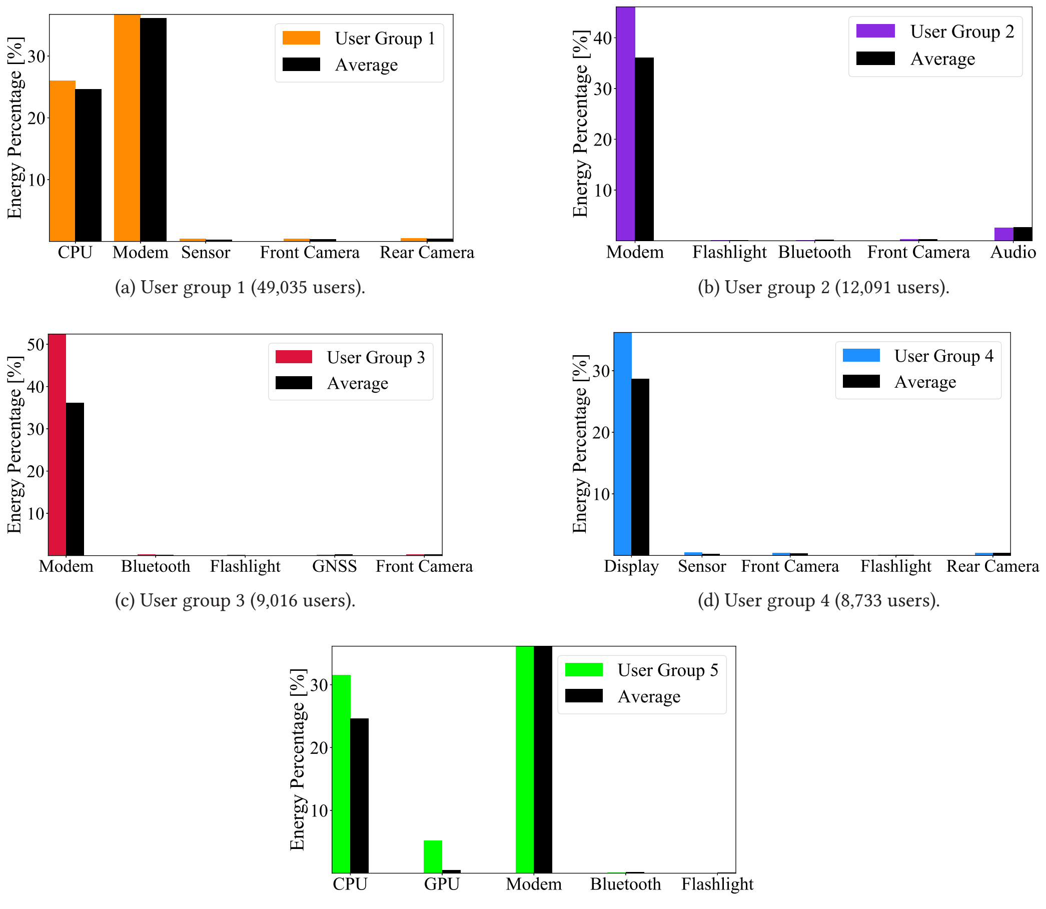
(c) MDH of UPM.

Fig. 12. Performance comparison in terms of energy discrepancy when $K = [2, 50]$.

- **NoHAC**'s MDPH performs even worse than that of PCA. NoHAC's overall bad performance demonstrates that redundant features (221 preliminary THUPs in NoHAC *v.s.* 67 ultimate THUPs in E-Sub) have a significant negative effect on discovering user groups with similar usage patterns.
- For **NoAllo**, the energy consumption of atypical apps and atypical AUPs of typical apps is excluded in establishing UPMs but included in UID fitting. Its similar performance on KLD compared with E-Sub indicates that atypical-to-typical allocation does not influence much the identification of user groups with similar AUPs. However, the performance on MDPH and MDH is significantly affected due to the considerable deviation of the smartphone energy breakdown on ultimate THUPs from the reality (i.e., significant loss of smartphone energy consumption).

6 POTENTIAL APPLICATIONS

In this section, we demonstrate several potential applications that can be realized by leveraging the established UBMs by our E-Sub approach. Smartphone manufacturers can benefit from analyzing our UBMs to (1) build a map between dependence on hardware components and user groups, so that the importance of each hardware component can be identified (Section 6.1); (2) estimate the room for improvement of hardware components, under the assumption that within each user group, individual users' excessive proportion of energy percentage on some hardware components (compared with the corresponding UPM) can be cut off (Section 6.2); and (3) predict the energy performance of the next-generation smartphone, under the assumption that user behaviors do not change significantly between successive smartphone generations (Section 6.3).



(a) User group 1 (49,035 users).

(b) User group 2 (12,091 users).

(c) User group 3 (9,016 users).

(d) User group 4 (8,733 users).

(e) User group 5 (6,704 users).

Fig. 13. Energy percentage breakdown on top five hardware components with the largest degree of dependence for the five largest user group. Black bar represents the average value among all user groups.

6.1 Analyzing the Importance of Hardware Components

To design the next-generation smartphone, it is essential for manufacturers to identify important hardware components for upgrades. The hardware components that are considered to be more demanding for upgrades are those that users depend on more in their smartphone normal daily use.

By leveraging our results of UPM, we define the degree of dependence $d_h^{(k)}$ for hardware component h with respect to the kth user group. It is calculated as the difference between the average hardware energy percentage breakdown of the kth user group $r_h^{(k)}$ and that of all users $\overline{r}_h$, as given in Equation (24):

$$d_h^{(k)} = r_h^{(k)} - \overline{r}_h, \tag{24}$$

where $r_h^{(k)} = \sum_{p^* \in \mathbb{P}_1^*} r_{p^*}^{(k)} \cdot r_{p^*,h}$ can be calculated from the kth UPM, as given in Equation (20).

A higher value of $d_h^{(k)}$ indicates that users in the kth group depend more on the hardware component h during their smartphone usage. In other words, the hardware component h is of higher demand for upgrades for this user group. In Figure 13, it illustrates the top five hardware

Table 11. User Groups and Population Percentage with High Dependence for 12 Hardware Components

	User Groups	Population Percentage
Audio	7, 8, 9, 10, 15, 21, 22, 25, 30	16.98%
Front Camera	/	0
Flashlight	2	8.81%
Bluetooth	3	6.57%
GNSS	20	0.75%
Sensor	4	6.36%
GPU	5, 32	4.92%
Modem	1, 2, 3, 6, 8, 22	**60.25%**
CPU	1, 5, 6, 12, 16, 18, 19, 23, 26, 27, 28, 29, 30, 31, 32, 33	**52.71%**
WiFi	11, 12, 13, 15, 17, 18, 19, 24, 27, 28, 29, 31, 33	12.85%
Rear Camera	14	2.00%
Display	4, 7, 9, 10, 13, 14, 16, 17, 20, 21, 24, 25, 26	**25.13%**

Note: for each user group, top two hardware components with the highest $d_h^{(k)}$ are included into this table. Top 3 most important hardware components are marked in bold.

components with the largest degree of dependence for the five largest user groups. It is shown that WeChat-intensive users have slightly higher dependence on CPU and Modem than the average of all users. TikTok- and QQ-intensive users have a significantly higher dependence on Modem. Traditional users depend more on Display. And GameForPeace-intensive users depend more on CPU and GPU.

Besides measuring hardware importance with respect to user groups, it can also be a measure for all users. We measure it by calculating the total number of the population who have a high dependence on the hardware component. In Table 11, the second and third column list the user group index and the total population with high dependence on a certain hardware component, respectively. It is concluded that Modem, CPU, and Display are the top three most important hardware components. Upgrades of these important hardware components will influence the largest proportion of users.

6.2 Estimating the Room for Improvement of Hardware Components

For smartphone manufacturers, after some iterations of upgrades for some specific hardware components, they usually conduct tests to quantify the improved energy performance. The obtained outcomes need to be assessed by comparing the improved energy consumption with some pre-established targets. Therefore, it is valuable to estimate quantitatively the room for improvement in advance.

Furthermore, for each user group, there are users whose energy percentage breakdown on certain hardware components is higher than that of UPM. These users are likely to manifest excessive dependence on the corresponding hardware components. To this end, focusing on the test outcomes of these users can render the assessment more efficient and specific. Ideally, the upgrades should alleviate the excessive dependence so that these users get more satisfied with the energy performance while remaining the same physical behavior.

Specifically, we define user u's (in user group k) room for improvement $\mathrm{IE}_{u,h}^{(k)}$ with respect to hardware component h to be the reduced amount of energy consumption if the excessive energy percentage on h is cut down. Notably, $\mathrm{IE}_{u,h}^{(k)}$ is always positive. For users whose energy percentage breakdown is lower than that of UPM, we set $\text{IE}_{u,h}^{(k)} = 0$. The calculation of $\text{IE}_{u,h}^{(k)}$ is given in Equation (25), signifying the energy difference when maintaining the energy consumption of all the other 11 hardware components except for h to be constant, and then forcing $r_{u,h}$ to be equal to $r_h^{(k)}$ (referring to Equation (24)).

$$\text{IE}_{u,h}^{(k)} = E_u - \frac{E_u - E_{u,h}}{1 - r_h^{(k)}}. \tag{25}$$

In Table 12, we calculate for each hardware component: (1) the total room for improvement for all users; (2) the percentage of total room for improvement over total energy consumption of all users; and (3) the average room for improvement per user. It is found that Modem has the largest room for improvement among these hardware components (reduce energy consumption by 9.23% and 187.92 mAh). Display ranks the second (reduce energy consumption by 5.06% and 103.01 mAh) and then CPU follows (reduce energy consumption by 4.91% and 100.01 mAh). The finding is consistent with the top three most important hardware components (Table 11). Due to space constraints, in Figure 14, we plot the room for improvement of Modem, CPU, and Display with respect to different user groups. Top five user groups with the highest total room for improvement ($\sum_{u \in \mathbb{U}^{(k)}} \text{IE}_{u,h}^{(k)}$) are included in this figure. Interestingly, for user group 1, even though the room for improvement per user is relatively low, the total room for improvement is considerable due to its large population.

6.3 Predicting the Energy Performance for Next-Generation Smartphones

It is known that UBMs are established to discover user behavior patterns (i.e., app preference, usage time). Even though these UBMs are established from a certain smartphone generation (namely, "current-generation smartphone"), we claim that they can be easily transferred to a new smartphone generation (namely, "next-generation smartphone"), based on the assumption that user behaviors do not change much between successive smartphone generations. In other words, as assuming user behavior unchanged (i.e., app preference, usage time), we can use UBMs established by current-generation smartphones to simulate those of the next-generation smartphones, and then predict the energy performance of the new smartphones even before getting a large amount of data from its users.

Estimating UBMs for the next-generation smartphone is based on implementing existing UBMs on the next-generation smartphone. Suppose a concretized UBM k with a sampled value $E^{(k)}$ in UID is implemented on the new smartphone with a machine (e.g., robotic arms), then we can generate smartphone energy consumption of $\widetilde{E}^{(k)}$, as well as energy breakdown on TAUPs $\widetilde{E}_{p^*}^{(k)}$ and that of hardware components with respect to TAUPs $\widetilde{E}_{p^*,h}^{(k)}$. We define η_k as the ratio of $\widetilde{E}^{(k)}$ to $E^{(k)}$, which is given in Equation (26), signifying the degree of energy performance improvement for the next-generation smartphone with respect to UBM k.

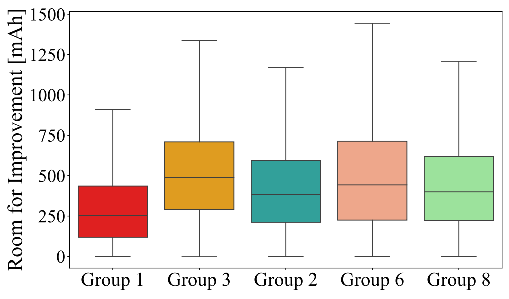
$$\eta_k = \frac{\widetilde{E}^{(k)}}{E^{(k)}}. \tag{26}$$

As the linear relationship for users belonging to the same UBM can be extended to the next-generation smartphone, it is enough to esitimate the UBMs for the next-generation smartphone only by implementing one concretized UBM for each UBM:

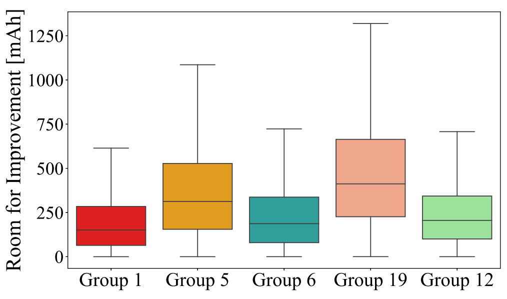
Table 12. Total Room for Improvement (mAh), Ratio of Total Room for Improvement to Total Energy Consumption (%), and Average Room for Improvement (mAh) for 12 Hardware Components

	Total (mAh)	Total Percentage	Average (mAh)
Audio	2,849,818.32	1.02%	20.77
Front Camera	596,147.30	0.21%	4.34
Flashlight	229,609.98	0.08%	1.67
Bluetooth	399,229.40	0.14%	2.91
GNSS	542,827.12	0.19%	3.96
Sensor	105,699.68	0.04%	0.77
GPU	1,560,870.32	0.56%	11.38
Modem	**25,783,989.71**	**9.23%**	**187.92**
CPU	**13,722,264.73**	**4.91%**	**100.01**
WiFi	9,661,447.09	3.46%	70.42
Rear Camera	803,897.96	0.29%	5.86
Display	**14,132,891.06**	**5.06%**	**103.01**

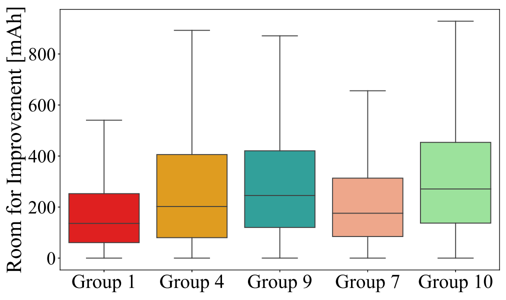
Top 3 hardware components with the largest room for improvement are marked in bold.



(a) Modem.



(b) CPU.



(c) Display.

Fig. 14. Room for improvement of Modem, CPU, and Display with respect to the top-5 user groups that are with the highest total room for improvement.

(1) **Usage Intensity Distribution (UID)**: Denote the PDF of kth UBM's UID as $f^{(k)}$, and that of estimated UBM as $\widetilde{f}^{(k)}$, we have

$$\widetilde{f}^{(k)}(x) = f^{(k)}\left(\frac{x}{\eta_k}\right). \tag{27}$$

(2) **Usage Pattern Model (UPM)**: UPM of the next-generation smartphone can be estimated according to the simulation results of smartphone energy breakdown on hardware components with respect to ultimate THUPs $\widetilde{E}^{(k)}_{p^*,h}$. The process of estimating UPM (as well as estimating TAUPs and ultimate THUPs) is opposite to that of establishing UPM.

— Smartphone energy percentage breakdown on hardware components with respect to ultimate THUPs (i.e., each entry of UPM) is obtained by calculating

$$\frac{\widetilde{E}^{(k)}_{p^*,h}}{\widetilde{E}^{(k)}} = \widetilde{r}^{(k)}_{p^*} \cdot \widetilde{r}_{p^*,h}.$$

— Smartphone energy percentage breakdown on ultimate THUPs is calculated as

$$\widetilde{r}^{(k)}_{p^*} = \frac{\widetilde{E}^{(k)}_{p^*}}{\widetilde{E}^{(k)}};$$

— Ultimate THUP energy percentage breakdown on hardware components is calculated as

$$\widetilde{r}_{p^*,h} = \frac{\widetilde{E}^{(k)}_{p^*,h}}{\widetilde{E}^{(k)}_{p^*}}.$$

7 IMPLICATIONS

Besides the above potential benefits for smartphone manufacturers, many interesting implications can also be drawn from findings during the process of implementing E-Sub, which can be leveraged by multiple key players of smartphone eco-system, e.g., app developers, network providers, users. Implications in this section are organized according to the implementation process of E-Sub, so as to make the presenting logic consistent with the previous sections.

7.1 Perspective on Smartphone Apps

In Section 4.1 app filtering, we quantify how "typical" an app a is with our defined indicator app typicality I_a. The calculation of I_a involves three indicators: app popularity λ_a, app energy intensity μ_a, and app time intensity ν_a.

Through investigating I_a among all apps on the market, we can discern the social trend and values regarding the importance of those apps. We find that WeChat is the most "typical" third-party app, with nearly 100% user coverage, intense energy consumption, and long-term usage time. Since WeChat plays an essential role in smartphone users' daily life, its developers should take on more social responsibilities. For example, the developers should attempt to build a harmonious and healthy social environment considering WeChat is one of the most influential entities that shape people's social behavior.

In the meantime, μ_a's and ν_a's indicate the energy and time aspects of app a during the usage period. Among all categories of apps, game and video apps are the most energy-intensive ones while E-book apps are the most time-intensive apps. Hence, energy performance of game and video apps will to large degrees affect user experience. And developers of these two app categories are supposed to pay more attention to improve app energy efficiency, compared with developing other new app functionalities. For E-book apps, users tend to stay focused on reading for a relatively longer time period. The developers of E-book apps can consider more humanized and energy-efficient design for app users (e.g., automatically switching to the eye-care mode with reduced blue light emission from the screen). Moreover, smartphone manufacturers can also develop some services to adapt to the long-term usage time (e.g., cleaning up background programs during the usage of such apps).

7.2 Perspective on THUPs of Apps

Based on our experiment results in Section 4.2 regarding THUP identification, it is observed that for most apps, their cluster boundaries (i.e., the boundaries of different preliminary THUPs identified through clustering) are well separated in terms of energy percentage breakdown on WiFi and Modem. In other words, for most apps, there are preliminary THUPs that differ in the way of connection to the Internet: through WiFi or through Modem. It also turns out that the majority of app users establish a connection through Modem, which indicates that a stable WiFi connection is not available during most app usage periods. Therefore, app developers can consider adding richer offline functionalities to increase app usage without a stable WiFi connection. Network providers can also leverage this finding to work out more suitable and economical smartphone or app data plans.

Besides, we find that WiFi is more efficient than Modem in terms of energy consumption on data transferring. It coincides with the finding in [5] that WiFi is a more efficient tool to transfer data. To this end, smartphone manufacturers can pay more attention to upgrade the Modem component during the design of new-generation smartphones. In terms of app developers, they may provide relevant services to reduce the amount of data transferred by Modem, such as pre-fetching necessary and important contents under a WiFi-stable environment. For smartphone users, they can also develop the habit of caching contents with WiFi connection especially when the smartphone battery is low.

Furthermore, it is observed that users' sensibility to data traffic consumption differs with the apps they use. For instance, QQ users are sensitive to mobile data traffic consumption, which is reflected on the fact that the energy consumption on the rest 11 hardware components increases a lot with WiFi connection than that with Modem connection. In contrast, King of Glory users are less sensitive, since regardless of mobile data traffic, WiFi, or offline mode, their usage intensity is hardly affected. In general, apps can be divided into data-sensitive apps and data-insensitive apps. For the app developers of data-sensitive apps, they can attempt to reduce the traffic data under cellular network, e.g., sacrificing some resolution of loaded figures. Offline mode can also be designed and refined to attract users who have to worry about data traffic. For network providers, they can launch specific data-plan contracts with those data-sensitive apps so that interested users can enjoy those apps with whatever Internet connection.

7.3 Perspective on Smartphone User Groups

In Section 4.4 on TAUP identification, we report 33 user groups corresponding to 33 TAUPs. The largest 5 user groups, respectively represent 35.73% social lovers who use WeChat with Modem connection intensively; 8.81% short video lovers who use TikTok with Modem connection intensively; 6.75% social lovers who use QQ with Modem connection intensively; 6.36% traditional users with limited smartphone usage; and 4.89% game lovers who use Game For Peace with Modem and CPU intensively. This finding can be referred to as clues to infer various user attributes, e.g., age, gender, job. Notably, instead of trying to infer attributes for individual users, we do so on a group basis, according to distinct cluster characteristics. For instance, both group 1 (WeChat-intensive) and group 3 (QQ-intensive) are classified as social lovers. However, group 1 has a special feature compared with group 3: an excessive use of Taobao (an online shopping app). With common sense, it is reasonable to infer that group 1 mainly consists of young people who have incomes, while group 3 mainly consists of teenagers. It coincides with the fact that the target user of QQ is in general younger than that of WeChat. Unfortunately, we cannot provide a quantitative evaluation for the inference because of the inaccessibility of attribute labels after data desensitization. Though such a group-based inference method is far from enough when it comes to inferring individual attributes accurately, as our intend is to bring up a possible way of inferring user attributes with desensitized data only, it is still plausible for smartphone manufacturers, app developers, or network providers, under the circumstances of limited information and low accuracy requirement.

8 CONCLUSION AND FUTURE WORK

In this work, we propose E-Sub, an energy-based smartphone user behavior modeling approach, to extract a small number of UBMs from highly diverse (i.e., high dimension and high dispersion) energy consumption data of smartphone users. To deal with the challenge of high diversity, three mechanisms are designed: app unification, usage decomposition, and pattern layering. The implementation of E-Sub consists of five steps in which the three mechanisms are integrated across different steps. By carrying out experiments on the training part of our large-scale dataset, the five steps proceed as follows: (1) from 20,964 distinct apps, 76 typical apps are selected; (2) 220 preliminary THUPs are identified for the 76 typical apps, among which 66 distinct ultimate THUPs are reserved; (3) energy consumption of atypical apps and atypical HUPs are allocated to these ultimate THUPs as well as Virtual; (4) 33 TAUPs are identified based on the representation with 67 ultimate THUPs (including Virtual); and (5) 33 UIDs are fitted. Eventually, 33 UPMs are established by consolidating the 33 TAUPs and the 67 ultimate THUPs. Then 33 UBMs are accordingly established as a union of UPMs and UIDs.

On the test part of our dataset, the performance of our established UBMs achieves 94% on behavior coverage and up to 20% improvement on energy discrepancy compared with the baseline approach PCA. It is also proved that E-Sub outperforms all of its variants (i.e., by removing or modifying certain mechanisms). Manifold results of typical apps, THUPs, TAUPs, and UIDs provide significant implications and quantitative evidence about smartphone usage for different stakeholders in the smartphone eco-system. Moreover, we demonstrate in detail several potential applications enabled by our established UBMs, specifically for smartphone manufacturers.

In terms of future work, it is valuable to implement E-Sub on a more complete dataset (i.e., only top 20 most energy-consuming apps are recorded per day in the current dataset). It is expected that the accuracy of UBMs of energy representation will increase by approaching the true value of smartphone total energy consumption. Moreover, it is also valuable to extend E-Sub to time-series data (a denser time stamp is needed rather than per 24 hours in the current dataset) to explore temporal user behavior patterns. Involving time information when conducting user behavior modeling enables established UBMs to distinguish time periods as well as relevant properties of the day (e.g., workday, holiday), which helps to generate more refined models and insights of user behavior.

REFERENCES

[1] Margareta Ackerman, Shai Ben-David, Simina Brânzei, and David Loker. 2012. Weighted clustering. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*. AAAI.

[2] Raja Wasim Ahmad, Abdullah Gani, Siti Hafizah A. b. Hamid, Mohammad Shojafar, Abdelmuttlib Ibrahim Abdalla Ahmed, Sajjad A. Madani, Kashif Saleem, and Joel J. P. C. Rodrigues. 2017. A survey on energy estimation and power modeling schemes for smartphone applications. *International Journal of Communication Systems* 30, 11 (2017), e3234.

[3] App Annie. 2017. Spotlight on Consumer App Usage, Part 1. Retrieved from https://www.insidemarketing.it/wp-content/uploads/2017/06/1705_Report_Consumer_App_Usage_EN.pdf.

[4] Charles Arthur. 2014. Your smartphone's best app? Battery life, say 89% of Britons. *The Guardian*. Retrieved from https://www.theguardian.com/technology/2014/may/21/your-smartphones-best-app-battery-life-say-89-of-britons.

[5] Niranjan Balasubramanian, Aruna Balasubramanian, and Arun Venkataramani. 2009. Energy consumption in mobile phones: *A Measurement Study and Implications for Network Applications*. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Proceedings of the 9th ACM SIGCOMM conference on Internet measurement*. ACM, 280–293.

[6] Huanhuan Cao, Tengfei Bao, Qiang Yang, Enhong Chen, and Jilei Tian. 2010. An effective approach for mining mobile user habits. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. ACM, 1677–1680.

[7] Aaron Carroll and Gernot Heiser. 2010. An analysis of power consumption in a smartphone. In *Proceedings of the the USENIX Annual Techinical Conference*. 1–14.

[8] Xiaomeng Chen, Ning Ding, Abhilash Jindal, Y. Charlie Hu, Maruti Gupta, and Rath Vannithamby. 2015. Smartphone energy drain in the wild: Analysis and implications. *ACM SIGMETRICS Performance Evaluation Review*. 43, 1 (2015), 151–164.

[9] Xiaomeng Chen, Jiayi Meng, Y. Charlie Hu, Maruti Gupta, Ralph Hasholzner, Venkatesan Nallampatti Ekambaram, Ashish Singh, and Srikathyayani Srikanteswara. 2017. A fine-grained event-based modem power model for enabling in-depth modem energy drain analysis. In *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 1, 2 (2017), 45:1–45:28.

[10] Karen Church, Denzil Ferreira, Nikola Banovic, and Kent Lyons. 2015. Understanding the challenges of mobile phone usage data. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*. ACM, 504–514.

[11] J. Clement. 2019. Number of available apps at Google Play from 2nd quarter 2015 to 2nd quarter 2019. *Statista*. Retrieved from https://www.statista.com/statistics/289418/number-of-available-apps-in-the-google-play-store-quarter/.

[12] Trinh-Minh-Tri Do and Daniel Gatica-Perez. 2010. By their apps you shall understand them: Mining large-scale patterns of mobile phone usage. In *Proceedings of the 9th International Conference on Mobile and Ubiquitous Multimedia*. ACM, 27:1–27:10.

[13] Hossein Falaki, Ratul Mahajan, Srikanth Kandula, Dimitrios Lymberopoulos, Ramesh Govindan, and Deborah Estrin. 2010. Diversity in smartphone usage. In *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*. ACM, 179–194.

[14] Mansoor Iqbal. 2019. TikTok Revenue and Usage Statistics . Retrieved from https://www.businessofapps.com/data/tik-tok-statistics/.

[15] Yifei Jiang, Abhishek Jaiantilal, Xin Pan, Mohammad A. A. H. Al-Mutawa, Shivakant Mishra, and Larry Shi. 2013. Personalized energy consumption modeling on smartphones. In *Mobile Computing, Applications, and Services*. Springer, Berlin, 343–354.

[16] Ian Jolliffe. 2011. *Principal Component Analysis*. Springer, 1094–1096.

[17] Simon L. Jones, Denzil Ferreira, Simo Hosio, Jorge Goncalves, and Vassilis Kostakos. 2015. Revisitation analysis of smartphone app use. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 1197–1208.

[18] Philipp Krieter. 2020. Looking inside-mobile screen recordings as a privacy friendly long-term data source to analyze user behavior. Universität Bremen

[19] Jemin Lee, Hyunwoo Joe, and Hyungshin Kim. 2014. Automated power model generation method for smartphones. *IEEE Transactions on Consumer Electronics* 60, 2 (2014), 190–197.

[20] Ding Li, Shuai Hao, William G. J. Halfond, and Ramesh Govindan. 2013. Calculating source line level energy information for android applications. In *Proceedings of the 2013 International Symposium on Software Testing and Analysis*. ACM, 78–89.

[21] Huoran Li, Xuan Lu, Xuanzhe Liu, Tao Xie, Kaigui Bian, Felix Xiaozhu Lin, Qiaozhu Mei, and Feng Feng. 2015. Characterizing smartphone usage patterns from millions of android users. In *Proceedings of the 2015 Internet Measurement Conference*. ACM, 459–472.

[22] Zhung-Xun Liao, Yi-Chin Pan, Wen-Chih Peng, and Po-Ruey Lei. 2013. On mining mobile apps usage behavior for predicting apps usage in smartphones. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*. ACM, 609–618.

[23] R. Lleti, M. C. Ortiz, L. A. Sarabia, and M. S. Sanchez. 2004. Selecting variables for k-means cluster analysis by using a genetic algorithm that optimises the silhouettes. *Analytica Chimica Acta* 515, 1 (2004), 87–100.

[24] Radhika Mittal, Aman Kansal, and Ranveer Chandra. 2012. Empowering developers to estimate app energy consumption. In *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*. ACM, 317–328.

[25] Dalal Navneet and Triggs Bill. 2005. Histograms of oriented gradients for human detection. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*.

[26] Abhinav Pathak, Y. Charlie Hu, and Ming Zhang. 2012. Where is the energy spent inside my app? Fine grained energy accounting on smartphones with eprof. In *Proceedings of the 7th ACM European Conference on Computer Systems*. ACM, 29–42.

[27] S. M. Pincus. 1991. Approximate entropy as a measure of system complexity. In *Proceedings of the National Academy of Sciences*, 2297–2301.

[28] Ahmar Rashid, Muhammad Amir Zeb, Amad Rashid, Sajid Anwar, Fernando Joaquim, and Zahid Halim. 2020. Conceptualization of smartphone usage and feature preferences among various demographics. *Cluster Computing* (2020), 1855–1873.

[29] Ekarat Rattagan, Ying-Dar Lin, Yuan-Cheng Lai, Edward T.-H. Chu, and Kate Ching-Ju Lin. 2018. Clustering and symbolic regression for power consumption estimation on smartphone hardware subsystems. *IEEE Transactions on Sustainable Computing* 3, 4 (2018), 306–317.

[30] Lior Rokach and Oded Maimon. 2005. *Data Mining and Knowledge Discovery Handbook*. Springer, Boston, MA, 321–352.

[31] Fabricio A. Silva, Augusto C. S. A Domingues, and Thais R. M. Braga Silva. 2018. Discovering mobile application usage patterns from a large-scale dataset. *ACM Transactions on Knowledge Discovery from Data* 12, 5 (2018), 59.1–59.36.

[32] Joshua B. Tenenbaum, Vin De Silva, and John C. Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290, 5500 (2000), 2319–2323.

[33] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605.

[34] Gang Wang, Xinyi Zhang, Shiliang Tang, Haitao Zheng, and Ben Y. Zhao. 2016. Unsupervised clickstream clustering for user behavior analysis. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 225–236.

[35] Lide Zhang, Birjodh Tiwana, Zhiyun Qian, Zhaoguang Wang, Robert P. Dick, Zhuoqing Morley Mao, and Lei Yang. 2010. Accurate online power estimation and automatic battery behavior based power model generation for smartphones. In *Proceedings of the 8th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*. ACM, 105–114.

[36] Sha Zhao, Julian Ramos, Jianrong Tao, Ziwen Jiang, Shijian Li, Zhaohui Wu, Gang Pan, and Anind K. Dey. 2016. Discovering different kinds of smartphone users through their application usage behaviors. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 498–509.

Received May 2020; revised March 2021; accepted April 2021