Visualizing Power Mode: The Impact of Battery-Saving Indicators on User Behavior During Intensive Mobile Interactions

CHENHAO HONG, City University of Hong Kong, China
XI ZHENG, City University of Hong Kong, China
MINHUI LIANG, City University of Hong Kong, China
JUNQIAO QIU, City University of Hong Kong, China
YUHAN LUO*, City University of Hong Kong, China

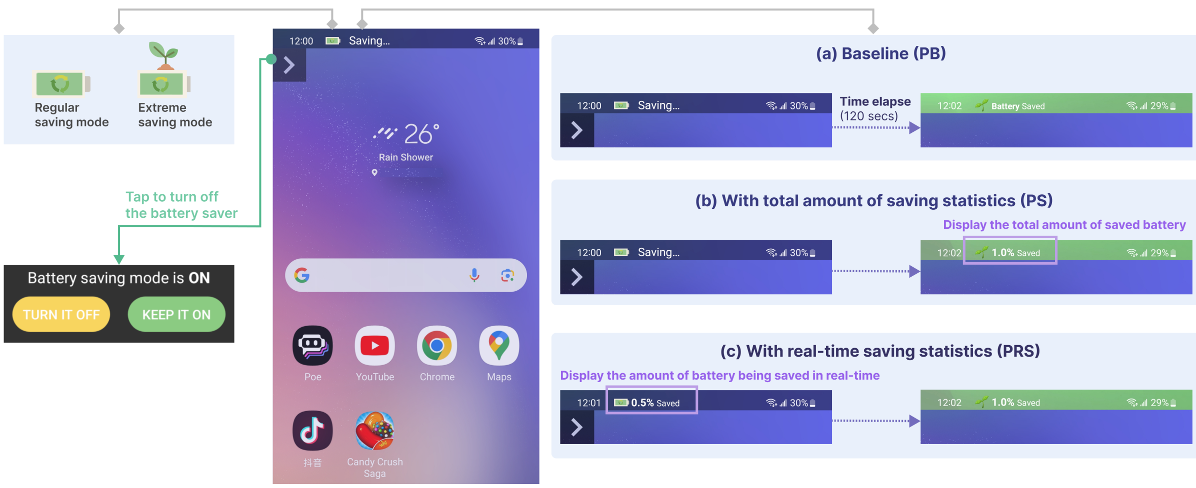

Fig. 1. The illustration of three designs for a battery-saving indicator on the status bar of a mobile phone, including: (a) a baseline version (PB) that displays only the saving status during the process and at the end; (b) a version built upon the baseline showing the total amount of battery saved at the end (PS); and (c) a real-time statistics version (PRS) that continuously displays the amount of battery being saved. When the saver is on, an arrow button appears at the top left, allowing participants to turn off the saver. Additionally, two battery-saving modes—regular and extreme modes—are randomly applied to the saver. They are indicated by different battery icons.

*Corresponding author.

Authors' Contact Information: Chenhao Hong, Department of Computer Science, City University of Hong Kong, Hong Kong, China, chenhhong2-c@my.cityu.edu.hk; Xi Zheng, Department of Computer Science, City University of Hong Kong, Hong Kong, China, zheng.xi@my.cityu.edu.hk; Minhui Liang, Department of Computer Science, City University of Hong Kong, Hong Kong, China, mhliang4-c@my.cityu.edu.hk; Junqiao Qiu, Department of Computer Science, City University of Hong Kong, Hong Kong, China, junqiqiu@cityu.edu.hk; Yuhan Luo, Department of Computer Science, City University of Hong Kong, Hong Kong, China, yuhanluo@cityu.edu.hk.

ACM 2474-9567/2026/3-ART3
https://doi.org/10.1145/3789684

The ubiquitous mobile devices have made balancing battery efficiency and user experience (UX) a critical issue. While prior work has extensively researched battery-saving strategies by managing computational resources, the design of user interface (UI) that communicates battery-saving status with users remains underexplored. In this work, we investigate how different visual representations of battery-saving indicators influence user behavior during intensive mobile interactions. We designed three versions of UI indicators for battery saving, with varying levels of statistical details. Through a between-subjects user study with 36 participants completing a series of intensive tasks on a mobile phone with limited battery, we examined behavioral and perceptual patterns across study conditions. Our findings showed that real-time saving statistics improved battery-saving efficiency and mitigated negative user experience under moderate performance degradation. However, when performance drops sharply, the same indicators may bring frustration and lead users to quickly turn off the saver. Our post-study interviews further revealed the strategic choices made by participants to optimize task completion while saving batteries. With the lessons learned, we discuss the implications of designing visual indicators for battery-saving mode in different interaction scenarios, and propose future directions for building sustainable battery-saving interfaces.

CCS Concepts: • **Human-centered computing → HCI design and evaluation methods**; **Interaction design theory, concepts and paradigms**.

Additional Key Words and Phrases: Mobile Systems, Sustainable Computing, Human-Computer Interaction, Computational Resource Management

ACM Reference Format:
Chenhao Hong, Xi Zheng, Minhui Liang, Junqiao Qiu, and Yuhan Luo. 2026. Visualizing Power Mode: The Impact of Battery-Saving Indicators on User Behavior During Intensive Mobile Interactions. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 10, 1, Article 3 (March 2026), 28 pages. https://doi.org/10.1145/3789684

1 Introduction

Mobile phones have become an integral part of our daily life. Over the past few years, they have rapidly evolved from basic devices for sending text messages and making phone calls to advanced systems capable of performing high-demanding and intensive tasks such as playing videos, gaming, real-time navigation, etc [22, 25, 32]. To ensure a smooth user experience, developers often over-provision performance by utilizing extensive computational resources including CPUs, GPUs, and memory [4, 29]. The utilization of these computational resources, however, is often at the expense of battery drain [6, 7]. With over 5 billion active mobile device users worldwide [58], the total battery energy consumption of these devices was estimated to reach 22.5 TWh in 2021, equivalent to 123% of electricity usage of major technology companies such as Google [46].

The growing energy crisis in the mobile industry has made battery saving an important issue at the societal level, but battery technology has not kept pace with the needs of computational resources in mobile devices. According to a survey of more than 2000 smartphone users, nearly 92% of them reported having low-battery anxiety [64]. To address this challenge, existing research primarily focused on configurations of the back-end/architecture or investigating user tolerance on different configurations, such as the Dynamic Voltage and Frequency Scaling (DVFS) [54]. While these efforts have shown to be effective in saving battery cost while maintaining a relatively good user experience, little exploration has been done at the user interface level. That is, how interface design can contribute to energy savings by mitigating the perceived performance degradation, optimizing user satisfaction, and even encouraging them to adopt more energy-conscious usage behaviors.

In the field of human-computer interaction (HCI), there has been rich empirical evidence showing that informing users about the system status can improve their experiences and even tolerance when facing system performance drop [30, 31, 43]. For example, even simple feedback mechanisms such as loading icons or progress bars have been shown to reduce users' perceived waiting time and frustration during system delays [30], while transparency in AI systems can foster greater user trust and acceptance [61]. However, few studies have applied these design concepts in battery-saving contexts, and little is known about whether and how users may perceive performance drops differently when informed of battery-saving status through the interface. In this light, we aim to explore user interface (UI) design opportunities for “battery-saving modes” where device performance typically decreases, with the goal of improving user experience and encouraging them to engage in more sustainable phone use behaviors. Specifically, we focused on scenarios in which users need to perform interaction-intensive tasks while the battery is running low, and seek to answer two research questions:

- **RQ1.** Whether and how do UI designs that present different levels of details about battery saving status influence users’ perceptions and reactions to the battery saving mode?
- **RQ2.** What are users’ attitudes and expectations regarding the trade-offs between battery saving and system performance?

To investigate these questions, we designed three versions of battery-saving indicators placed at the status bar on the top of the screen. These versions share the same battery-saving modes at the backend, with one moderately decreasing device performance and the other doing so more aggressively (see Figure 1):

- The *baseline* version (PB), which only shows whether the battery saver is on or off, and when it ends.
- The *total saving* version (PS), which is built upon the baseline version and further shows the total battery saved at the end of the saving session.
- The *real-time saving* version (PRS), which is built upon the total saving version and shows the real-time battery saved during the saving session, in addition to the total saving.

To examine the effects of the three versions of designs on user perception and behaviors, we conducted a between-subjects lab study with 36 mobile phone users. During the study, participants were randomly assigned to one of the three interaction conditions to complete a set of predefined tasks. At the end of the study, we interviewed participants to further understand their experiences and thoughts about the battery-saving indicator.

We choose a controlled lab study to ensure consistent measurement of participants’ interaction behaviors, while recognizing that such control may limit ecological validity compared to studies conducted in real-world environments. To mitigate this limitation, we simulated a constrained battery scenario by setting the initial charge to 30% and employed a performance-based compensation, which prompted participants to strategically manage the trade-off between task completion and battery usage.

We found that PRS group exhibited a significantly lower turn-off rate (5.97% for PRS vs. 16.64% for PB) and higher saving efficiency (97.66% for PRS vs. 90.35% for PB) compared to PB group under the regular saving mode. PRS group tended to remain in saving sessions longer before disabling the saver, spending an average of 75.17 seconds compared to the PB group’s 50.78 seconds. Interview data further highlighted diverse user perceptions and preferences, informing design implications for sustainable and user-friendly battery-saving UIs.

Our findings contribute to the IMWUT and HCI communities in three key areas: (1) an empirical understanding of how varying indicator designs in battery-saving modes affect user perceptions and behavior; (2) insights into individual preferences and expectations of battery saving in their own lived experiences; and (3) design implications for interfaces that encourage sustainable mobile phone use.

2 Related Work

In this section, we first cover related research on improving energy efficiency in mobile devices, including hardware-level optimizations and adaptive energy management, and then explore the opportunities for UI-driven, human-centered approaches aimed at enhancing user experience through UI design within and beyond battery-saving applications.

2.1 System-Level Optimization via Hardware Control

Research on improving battery efficiency in mobile devices has evolved through several stages. Initial efforts mainly focused on hardware-level strategies, including innovations in CPU/GPU architecture, display technologies, and integrated power management systems [1, 9, 55, 63]. For instance, Kadjo et al. [27] proposed a Multi-Input

Multi-Output (MIMO) state-space controller to dynamically coordinate CPU-GPU frequency scaling, achieving a 17.4% reduction in energy use on Intel Baytrail-based Android devices with only 0.9% performance loss. Similarly, Park et al. [45] developed a CPU and GPU Dynamic Voltage Frequency Scaling (DVFS) strategy named Co-Cap to limit frequency provisioning through coordinated frequency capping, improving energy per frame by 10% on average on over 70 mobile gaming workloads. Prakash et al. [51] targeted thermal regulation to jointly manage energy and performance in various mobile workloads, achieving up to a 90% reduction in temperature variance while improving the Frames Per Second (FPS). These hardware-level solutions have shown promising results in reducing power consumption and have been applied to various industry products, however, they are limited in their responsiveness to user context and preferences [23, 50], as researchers found later [36, 56], users vary widely in their tolerance for performance degradation and expectations regarding battery life, depending on both personal preferences and usage scenarios.

2.2 User-Aware Adaptive Energy Optimization

In recent years, researchers have shifted the focus towards exploring adaptive systems that dynamically balance energy savings with user experience (UX). For example, through surveying over 2,500 users, Halpern et al. [22] found that user satisfaction with identical CPU resource scheduling (e.g., core count and frequency) varied significantly in different interaction contexts, such as mobile gaming and video streaming. These findings have prompted research to advance toward more fine-grained resource allocation. On the one hand, researchers utilized self-adaptive techniques to optimize the balance between user satisfaction and energy consumption [16, 24]. For example, Hwang et al. designed RAVEN [24] to track and predict frame similarity based on human visual perception of graphics changes. When succeeding frames were found to be similar, RAVEN lowered the rendering rate to save energy. This kind of approaches focused computational resources on high-interest areas (e.g., faces and moving objects) instead of uniformly rendering the entire scene, enabling perception-aware energy savings. On the other hand, emotion recognition and perception prediction started to serve as an important reference. Poyraz et al [50]. predicted user satisfaction and adjusted CPU core count and frequency accordingly by analyzing users' mobile phone interaction patterns (e.g., high-frequency touch and device shaking) captured through built-in sensors, such as accelerometers and gyroscopes. Li et al. [34, 35] analyzed users' facial expressions (e.g., smiling and frowning) using the front-facing camera. When positive emotions are detected, the system appropriately reduces the resource consumption of the current task. Conversely, if negative emotions are detected, the resource allocation for the current task is immediately increased to ensure a satisfactory user experience.

However, these solutions primarily performed automatic resource allocation on the backend, leaving users outside the gate of how the battery is optimized on their devices. In other words, the implementation of these energy management strategies fully relies on the predefined logic, while users are unable to inform the system regarding their current preferences and needs. For instance, with the rendering framework RAVEN, users do not have the agency to indicate areas of their interests on the display screens [24]. On systems that adjust device performance based on changes in users' facial expressions, continuous camera monitoring may raise privacy concerns, and cultural differences may introduce ways that user emotions are interpreted [34, 35]. These issues highlight a tension in existing battery efficiency optimization: when the system takes full control of resource allocation, users are viewed as "passive recipients", and thus lose the opportunity to become active managers of the resources in their devices [20].

2.3 Enhancing User Experience via UI Design: Opportunities for Battery Saving

The user interface (UI) is the essential part of computational systems that enables user control and directly shape their experience and perception of system performance. In particular, "visibility of the system status" has been one of the golden usability heuristics since 1990s, highlighting that visualizing the activities behind the interface could improve user perceived control of the system and mitigate negative experiences [43]. For instance, several research studies showed that during waiting times (e.g., website loading, file download), dynamic visual indicators such as loading icons, progress bars, and text message can effectively distract users from the negative perception of delay and increase their willingness to wait [21, 30, 40, 42]. More relevantly, Ferreira et al. uncovered that the transparency of battery information, such as the remaining battery time and what is draining the battery, can significantly reduce user frustration [17]. Jung et al. further proposed Powerlet that could display the power consumption of apps in real time, leading users to reduce daily energy consumption by 8.2% [26]. A recent work done by Lee et al. found that predicting future battery usage can significantly mitigate low-battery anxiety. Specifically, they developed Serenus with real-time prediction of battery consumption of different applications and videos, such as "watching this video will reduce battery by 1.8%" [31]. The rich empirical evidence demonstrated the potential of transparent battery and performance visualizations to manage user expectations, reduce frustration, and promote energy-aware behaviors.

Despite the benefits, users' perceptions and preferences for different UI design strategies for battery saving remain unclear. Our work extends this line of research by investigating how the level of detail in system status information affects user perception during intensive mobile interactions, as well as whether and how this information can encourage more battery-conscious behaviors.

3 Battery-Saving Framework

To answer the research questions, we first developed a custom framework that runs on the test device to change the system configuration seamlessly in the background. At the UI level, the framework displays the battery saving statistics on the status bar; in the backend, it alters system configuration and logs user interactions.

Device & Operating Platform. We used Samsung Galaxy A54 as our test device. This device was released in 2023 and represents a typical mid-range Android device. Powered by the Exynos 1380 SoC, the device has an octa-core CPU (4×2.4 GHz Cortex-A78 & 4×2.0 GHz Cortex-A55) and a penta-core GPU (5×950 MHz Mali-G68 MP5) [14]. The octa-core CPU is divided into two clusters: the little cluster with Cortex-A55 cores and the big cluster with Cortex-A78 cores. All CPU cores within the same cluster share the same frequency, while for GPU, its five cores are the same and share the same frequency. The display is a 6.4-inch Super AMOLED panel with a resolution of 1080×2400 pixels. The device runs Android 13. We rooted the device to gain access to system configuration files and to run our framework with the help of Magisk [62], with built-in features such as "Adaptive Brightness" and "Battery Saver" disabled to avoid interference.

App Selection. To capture user interactions that represent a majority of smartphone usage scenarios, we selected target applications to test based on a 2024 report from Statista [59], showing that social media (35.1%), entertainment (32.7%), utility & productivity (13.6%), gaming (9.7%), and web browsers & search engines (5.8%) are the top five categories that add up to over 95% of mobile app usage. Therefore, we selected Douyin (Chinese version of TikTok, social media), YouTube (entertainment), Google Maps (utility), Candy Crush (gaming), and Chrome (web browser) to cover these five categories. In addition, we included Poe to represent the growing popularity of AI chatbot applications. This selection allowed us to evaluate how the UI designs perform under diverse interaction conditions. For each app, we tailored battery-saving strategy to its specific interaction patterns, as described later in Section 3.2.3. In the following, we elaborate on our design of the battery-saving indicator and implementation details.

3.1 Battery-Saving Indicator Design

3.1.1 Leveraging System Visibility to Mitigate Negative Impact on User Experience. Our design of the three versions of the battery-saving indicator represents three levels of details about the saving status (Figure 1): PB (baseline) showing only whether the saving mode is on and when it ends ("*Battery Saved*"); PS (total saving statistics), adding the total amount of battery saved at the end of per session to the PB ("*1% Battery Saved*"); PRS (real-time saving statistics), adding a real-time display of the battery being saved to the PS. This setup was inspired by prior work that leverages system visibility to mitigate negative user experience during performance degradation [21, 30, 40, 42], while enabling us to examine how the granularity (how much information to disclose) and frequency (how often to update this information) of system status affects users' perceptions and behaviors. To minimize distraction and avoid being obtrusive, we adopted a peripheral design concept [37] by placing a small indicator within the phone's status bar.

3.1.2 Examining Experience Variations Under Different Performance Degradation. To investigate how the intensity of performance degradation influences user tolerance, we offered two distinct battery-saving modes: Regular (RM) and Extreme (EM) ones. The former applies moderate hardware constraints to extend battery life, whereas the latter enforces more aggressive strategies to maximize battery saving. The effects of different degradation intensity were also commonly examined in prior work [22, 33]. To help users visually distinguish these two modes, we designed two different icons for them: a standard battery icon for the Regular mode, and a battery with a small growing tree for the Extreme mode, symbolizing its greater environmental and energy-saving benefits (see the icon area in the top-left of Figure 1). Additionally, to streamline a user study later, these two saving modes were designed to impact the system performance more compared to typical built-in battery savers for the purposes of achieving noticeable saving during a short period of time.

3.1.3 Providing Accessible Control of the Saving Mode. To enable control of the saving mode, we placed floating window with a transparent background that allows users to stop the saving mode at any time (see the arrow button in the screenshot in Figure 1). This setup helps us gather user preferences in a lightweight manner: keeping the mode active implies user acceptance of the performance trade-off, while deactivating it signals intolerance.

3.2 Backend Saving Modes

We first defined the target saving rates for our two battery-saving modes, and then created corresponding performance drops that would accompany these saving rates. Previous studies have shown the typical power consumption of different components in a smartphone, with the three major power-hungry components in idle state being the System on Chip (SoC, containing CPU and Graphics), the Global System for Mobile Communications (GSM), and the screen [5]. Of them, SoC and screen brightness are the only two components that can be adjusted by software to save power. Therefore, our battery saver focuses on adjusting the SoC and screen brightness.

3.2.1 Simulating Battery Drain and Defining Saving Rates. We developed a background service to simulate battery drain at a predefined, consistent rate, for all participants, eliminating impacts from other background services. The rate of battery drop is set to two to three times the actual rate, simulating a degraded battery life to make the impact of saving modes more prominent.

We drop the battery at a rate of 0.75%/min in Default, 0.30%/min in RM, and 0.20%/min in EM. These rates were calibrated for our 55-minute study starting at 30% battery (see Section 4.3.2). The Default rate (0.75%/min) is designed to drain the 30% battery in 40 minutes, compelling participants to use a saving mode to complete the session. For instance, considering RM only, a participant would need to spend at least 25 minutes in it to prevent an early shutdown. Additionally, these settings create a noticeable difference in battery consumption between the three modes: RM saves around 60% of power compared to Default, while EM saves around 73% of power compared to Default. Once the simulated battery level reaches 0, the background service will shut down the test device, and we will conclude the user study in advance.

3.2.2 Energy Consumption Measurement. To design tailored SoC configurations for the six apps we selected, we first measured the energy consumption of each app. We obtained the power profile [57], provided by the device manufacturer, from the firmware of the test device [53]. The power profile file contains the energy consumption of each component in the device (see Appendix B for the full power profile). The power profile value is given in milliamps (mA), representing the current consumption of the component at a nominal voltage. It contains the estimated energy consumption of the CPU and GPU cores at different frequencies.

We tested each app on the test device for a period of ten minutes, following normal usage patterns, and utilized Android Debug Bridge (adb) [13] to monitor the CPU and GPU frequencies at a rate of 1 Hz. We noted that GPU cores are not always active, so we also recorded the percentage of active GPU time for each app.

We compute the total SoC power profile value (PPV_{SoC}) by summing up the subtotal values of three components: the little CPU cluster (denoted as L), the big CPU cluster (denoted as B), and the GPU (denoted as G). The calculation is given by

$$\begin{aligned} Sub_{\mathrm{L}} &= n_{\mathrm{L}} \cdot PPV_{\mathrm{L}}(Freq_{\mathrm{L}}) + (N_{\mathrm{L}} - n_{\mathrm{L}}) \cdot PPV_{\mathrm{Susp}}, \\ Sub_{\mathrm{B}} &= n_{\mathrm{B}} \cdot PPV_{\mathrm{B}}(Freq_{\mathrm{B}}) + (N_{\mathrm{B}} - n_{\mathrm{B}}) \cdot PPV_{\mathrm{Susp}}, \\ Sub_{\mathrm{G}} &= N_{\mathrm{G}} \cdot PPV_{\mathrm{G}}(Freq_{\mathrm{G}}) \cdot Act\%, \\ PPV_{\mathrm{SoC}} &= Sub_{\mathrm{L}} + Sub_{\mathrm{B}} + Sub_{\mathrm{G}}. \end{aligned}$$

Here, n_{L} and n_{B} represent the number of enabled cores in the little and big clusters, while N_{L} and N_{B} represent the total number of cores available in those clusters. For the GPU, N_{G} denotes the total number of cores, and $Act\%$ represents the percentage of active GPU time observed during the measurement window. It is not possible to disable GPU cores on our test device.

We also modeled the screen power consumption. As the power consumption of the OLED screen is approximately linear with brightness level [10], we model its power profile value as

$$PPV_{\mathrm{screen}}(B) = (PPV_{\mathrm{full}} - PPV_{\mathrm{min}}) \cdot B + PPV_{\mathrm{min}},$$

where B is the brightness level (normalized 0–1), $PPV_{\mathrm{full}} = 252$ mA is the power at full brightness, and $PPV_{\mathrm{min}} = 81.5$ mA is the power at minimum brightness (see Appendix B).

3.2.3 Saving Mode Configurations. Based on the SoC energy consumption of each app we get, we found that these six apps can be divided into two groups: Low Energy Consumption (LEC) and High Energy Consumption (HEC). The LEC group includes Candy Crush and Poe, while the HEC group includes Chrome, Douyin, Google Maps, and YouTube. The apps within each group have similar frequency settings and power profile values, which allows us to design optimized battery-saving modes for each group.

From the SoC part, we utilized the CPU hotplug feature of Android's underlying Linux kernel [15] to disable CPU cores as part of the battery-saving modes. More specifically, we disabled CPU cores in the big cluster first, as the little cluster is more power-efficient and can handle most of the light tasks. We then disabled CPU cores in the little cluster if it is possible to save more power while maintaining a reasonable performance, but we ensured that at least one core is active in each cluster to maintain system stability. Lastly, we adjusted the CPU and GPU frequencies to approximate the desired battery-saving levels and a create distinguishable performance difference. The final configurations of the battery-saving modes and estimated power savings are shown in Table 1.

Reducing the screen brightness can also bring a substantial amount of power saving. We set the screen brightness to 80% of user's setting in RM and 60% of user's setting in EM to further reduce power consumption. When entering Default setting from RM or EM, we set the brightness level back to the user setting again. The brightness level is adjusted gradually over a period of 10 seconds to prevent sudden changes in brightness that might affect user experience [12]. Let B_{user} be the user's brightness setting. In a saving mode with a reduction factor R_{mode} (where $R_{\mathrm{RM}} = 0.2$ for RM and $R_{\mathrm{EM}} = 0.4$ for EM), the power saved, $\Delta PPV_{\mathrm{screen}}$, is

$$\begin{aligned} \Delta PPV_{\mathrm{screen}} &= PPV_{\mathrm{screen}}(B_{\mathrm{user}}) - PPV_{\mathrm{screen}}(B_{\mathrm{user}} \cdot (1 - R_{\mathrm{mode}})) \\ &= (PPV_{\mathrm{full}} - PPV_{\mathrm{min}}) \cdot B_{\mathrm{user}} \cdot R_{\mathrm{mode}}. \end{aligned}$$

Table 1. SoC configurations and power profile values of the default setting and our two battery-saving modes: regular (RM) and extreme (EM).

Config	CPU Little Cores			CPU Big Cores			GPU Cores				Total	% Default
	Freq	Num	Sub	Freq	Num	Sub	Freq	Num	Act	Sub		
LEC												
Candy Crush												
Default	1186	4	149	1127	4	446	354	5	99.64	501	1096	100
RM	1632	1	68	1632	1	198	221	5	99.64	329	595	54.29
EM	864	1	48	864	1	97	221	5	99.64	329	474	43.25
Poe												
Default	1140	4	150	1452	4	765	367	5	60.71	312	1227	100
RM	1632	1	68	1632	1	198	221	5	60.71	200	466	37.98
EM	864	1	48	864	1	97	221	5	60.71	200	345	28.12
HEC												
Chrome												
Default	1263	4	168	1519	4	949	359	5	55.63	280	1397	100
RM	533	2	62	1248	2	250	221	5	55.63	184	496	35.50
EM	1248	2	88	864	1	97	221	5	55.63	184	369	26.41
Douyin												
Default	1247	4	162	1310	4	710	351	5	100	495	1367	100
RM	533	2	62	1248	2	250	221	5	100	330	642	46.96
EM	1248	2	88	864	1	97	221	5	100	330	515	37.67
Google Maps												
Default	1323	4	178	1772	4	1174	358	5	76.68	388	1740	100
RM	533	2	62	1248	2	250	221	5	76.68	253	565	32.47
EM	1248	2	88	864	1	97	221	5	76.68	253	438	25.17
YouTube												
Default	1470	4	199	1624	4	1135	469	5	56.64	398	1732	100
RM	533	2	62	1248	2	250	221	5	56.64	187	499	28.81
EM	1248	2	88	864	1	97	221	5	56.64	187	372	21.48

Freq: Frequency (MHz)
Num: Number of enabled cores
Act: Percent of GPU active time
Sub: Subtotal power profile value (mA)
Total: Total power profile value (mA)

Therefore, when user's brightness setting is 100%, we estimate that RM reduces up to 34.1 mA of current and EM reduces up to 68.2 mA by lowering the brightness. With the SoC and brightness adjustments, our saving modes are able to create a distinguishable performance difference corresponding to the defined saving rates.

4 User Study

We conducted a between-subjects user study in a lab environment with the test device we used for developing the framework. Like prior work that examined user experience in system performance degradation [30, 42], including those in battery saving contexts [34, 35, 50], we also chose to conduct the study in a lab environment given the research goals and realistic constraints. First, to make the effects of the three battery-saving indicators (PB, PS, PRS) comparable, we needed to measure participants' interaction patterns under a consistent setting. Thus, a controlled lab environment was preferred for minimizing confounding factors that can arise (e.g., various phone models and app usages, inconsistent system performance). Second, in real-life settings, the availability of portable chargers often prevent the use of battery-saving modes, which makes it difficult to observe the behaviors we aimed to study. Third, obtaining root access to participants' phones to adjust CPU/GPU configurations poses safety and privacy concerns.

Realizing that the generalizability of the findings might be limited, we made efforts to improve the ecological validity by simulating an urgent, low-battery scenario: we designed a series of time-limited and goal-oriented tasks with a performance-based compensation mechanism. In addition, we conducted a post-task interview to understand participants' subjective experience on the indicator designs and their experiences of using battery-saving features, gathering qualitative insights that complement the quantitative data.

Before proceeding to the formal study, we tested the framework with two pilot participants to ensure that the experience was smooth without technical glitches. This helped ensure a smooth and reliable experience for participants in the formal study. The study was approved by the university's ethics review committee.

4.1 Participants

We recruited 36 participants (16 male and 20 female) from the university community and through social media. The participants are between 19 and 33 years old (M = 23.47, SD = 3.81); they are all smartphone users and college students. In a between-subjects lab study, we assigned these participants to complete a series of tasks with our battery saving framework, interacting with one of the three battery-saving indicator versions (PB, PS, PRS). Each condition had 12 participants, with a balanced distribution across gender, age, and major of study.

4.2 Experiment Setup & Compensation Mechanism

We designed a set of tasks in each of the six apps for participants to complete (see Table 2), which required intensive interaction with a time limit (9 minutes per app) and were thus challenging enough to create a sense of urgency. Note that these tasks were intentionally designed to be goal-oriented to maintain participants' attentiveness and promote active, rather than passive, device interaction. This approach helped us observe how battery-saving decisions are made when they are under cognitive demand. For example, participants were not simply asked to watch a video, but to find specific information within it.

Moreover, we tied a performance-based bonus to participants' task completion. Specifically, each participant was compensated with a base amount of HKD 70, plus an additional HKD 5 for each task completed, up to a maximum of HKD 30. Therefore, a total compensation varied from HKD 70 to HKD 100. Given the difference between the lowest and highest compensation did not vary significantly (HKD 30, which is approximately USD 3.86), this "micro-incentive" strategy should have not introduced excessive stress to participants [39]. Instead, it served two purposes in our study: simulating the pressure one feels to finish important activities on a device before running out of battery, and motivating participants to complete the tasks efficiently.

4.3 Procedure

The study consists of three parts in a private and spacious lab environment. One researcher was present, seated across the table from the participant to observe and take notes without intrusion. The entire process took around 90 minutes and consisted of a tutorial, six interaction tasks on the testing phone, and a debriefing interview.

4.3.1 Tutorial. Participants were first given a brief introduction to the study and read a consent form regarding the collection of their interaction with the test device and interview audio. We explained the differences of the two

Table 2. Interaction Tasks Assigned to Each App in the Lab Study

App	**Tasks**
Candy Crush	• (Optional) Try some entry levels to get familiar. • Complete level 17 to 20.
Chrome	• Search for three topics we specified and read out one result to us. • Search for three images we specified, zoom in, and take a screenshot.
Douyin	• Visit the homepage of a user we specified, and find the video with the most comments. • Watch the video, tell us some information about it, and read out the top-liked comment. • Given the name of four singers/bands, identify if they have an official Douyin account.
Google Maps	• Design a one-day travel plan by public transportation to your favorite city. • The plan should include at least three attractions, two restaurants, and one hotel. • Take screenshots of the plan, and make sure to depart from and return to the hotel.
Poe	• Pick an official bot and a popular bot from the homepage. • Chat with each bot by sending at least five text, one voice, and one image message. • Create a customized bot, and send the same amount of messages to it.
YouTube	• Watch three videos we specified, and answer one question about each video. • Drag the progress bar and change the playback speed of the video to locate the answer.

saving modes—regular (RM) and extreme modes (EM)—regarding their influence on the phone's performance and battery saving, as well as the difference of the visual icons representing each mode. Depending on the participant's group assignment, we explicitly explained the meanings of the battery-saving indicator, demonstrated the steps to turn off the battery saver (see Figure 1), and reminded them that their tasks would start with 30% battery left. We also showed them a task list for each app and explained the compensating mechanism. After that, we gave the participant some time to interact with the testing device to familiarize themselves with the device, with 31% battery left. Each participant was also asked to set the brightness and sound levels of the device based on their preference. This setup meant to reflect their own preferences and also served as the base for the framework to later adjust the brightness level for battery-saving purposes. The tutorial lasted around 15 minutes.

4.3.2 Interaction Tasks. After the tutorial, we set the battery level of the test device to 30% and handed it over to the participants. Participants were allotted 9 minutes to complete the tasks in each app, following a specific sequence that was randomly generated for each participant to eliminate order effects. We told participants that if the battery ran out during the tasks, the study would end beforehand and they would receive the amount of compensation based on their task completion. During this process, the battery saver was automatically turned on at the 1, 4, and 7-minute marks and remained on for 2 minutes by default, and participants could choose to turn it off (by expanding the top left arrow icon and clicking the "turn it off" button) or leave it on to reduce the chance of the battery running out. Each time it was on, the framework randomly chose a battery-saving mode (RM or EM), while ensuring that each mode was chosen at least once. When each battery-saving session was over, the framework applied the default configuration with a visual note indicating that the saving session had ended (as shown in Figure 1), and gradually adjusted the screen brightness to the initial setting.

In the study lab, the task descriptions for each app and the remaining time were shown on a big screen for participants' reference. When time was up, we reminded the participant to close the app before switching to the next one, ensuring the performance of the phone is not affected by the previous app. If the participant completed all tasks within less than 9 minutes, they could use the remaining time to freely explore the app. The interactive session lasted about 55 minutes, sometimes shorter when the phone ran out of the simulated battery. At the end of the session, we retrieved the interaction data from the test device and stopped the framework.

4.3.3 Debriefing Interview. We interviewed each participant after the interaction tasks. The interview questions focused on participants' overall experiences during the interaction tasks, including their perceived impact of the battery saver on the performance of the testing devices, differences in the two saving modes, and the difficulty of the tasks. We also asked them to share their smartphone charging habits, real-life situations where they often find themselves needing to use their phone on low battery, and their general thoughts on saving battery on smartphones in daily life. Each interview session lasted approximately 15 minutes.

4.4 Data Collection and Analysis

4.4.1 Interaction Data. We collected participants' task completion and their reactions to the battery-saving mode, which were compared between the three groups using Kruskal-Wallis test. Specifically, when the saver is manually turned off by the participants, the framework recorded information including the app they were interacting with, remaining battery level displayed on the status bar, the saving mode applied (regular or extreme), and the current timestamp. We employed a quantitative approach to analyze the data. Note that since the two saving mode—regular (RM) and extreme (EM)—were randomly applied by the framework, the maximal amount of battery that can be saved varied for each participant. Instead of directly comparing the absolute amount of battery saved, we calculated a normalized metric referred to as the "Saving Efficiency". For each two-minute saving session, we established the maximal possible amount of battery saved (0.9% for RM and 1.1% for EM, refer to Section 3.2.1), which represents the total savings if the mode was never turned off. The Saving Efficiency was then calculated by dividing a participant's actual battery saved (based on the time the saver was active) by this maximal amount. In addition to the turning-off behaviors and saving efficiency, we collected the touchscreen interaction logs to analyze participants' engagement with the device. We recorded the touch events in the entire study including the battery-saving and default sessions. We then calculated the "Touch Frequency," the number of touch-down events per minute for each applied configuration.

We used a mixed-effects logistic regression model with the bobyqa optimizer [49] to analyze whether they turned off the battery saver, and linear mixed-effects regression models to analyze when they turned it off, their saving efficiency, and touch frequency. To identify an optimal model, we began with a null model and systematically added the predictors *saving mode* (or *config* in case of touch frequency), *group*, and their interaction terms to it, using likelihood ratio tests to evaluate each predictor's contribution. The predictors *app* and *remaining battery* were also included in the model selection process as they are the contextual factors that might influence participants' decisions. To account for individual differences and repeated measurements, we also included a random intercept for *participant*. Note that we incorporated the *app* predictor as an independent variable rather than considering its interaction with other predictors. As noted in prior research [2, 41], introducing too many variables to the model could lead to overfitting and multicollinearity issues, which may lower the validity of the model. In our preliminary analysis, we also encountered convergence issues when including the interaction terms involving *app*. Therefore, we decided to include only the main effect of *app* to maintain model interpretability. For each model, the app with the largest mean value is chosen as the reference level for the *app* predictor.

As a result of the model selection process, we obtained final models that included the predictors *group*, *mode* (or *config*), their interaction term, and *app*, as well as the random intercept for *participant* for turning-off behavior and saving efficiency analysis. For touch frequency analysis, we further included the *remaining battery* as a predictor (See Appendix A for details).

4.4.2 Interview Data. We recorded the audio of the debriefing interviews and transcribed them into text. We then adopted a bottom-up approach to analyze the data following the steps of thematic analysis [3]. After familiarizing themselves with the transcripts, two researchers first independently highlighted the text segments that were deemed interesting and created initial codes to describe the data, which resulted in a total of 385 codes; next, the two researchers reviewed each other's initial codes and collaboratively refined these codes; then, we organized the refined codes into emerging themes through rounds of discussions with other team members. Our analysis was centered on two aspects: (1) deriving insights from participants' perceptions and experiences of the visual indicators of the battery saver to complement the quantitative results; and (2) understanding participants' expectations and preferences for saving battery on their mobile phones in real life.

5 Findings

We found that making the battery-saving status visible to users generally enhanced their experience and agency during intensive mobile interactions. However, in certain situations, battery-saving indicators can inadvertently introduce stress and frustration. In this section, we describe the findings in two parts: participants' activities during the battery-saving sessions (RQ1) and their expectations toward mobile battery saving modes (RQ2).

Table 3. Participants' task completion, sessions they turned off the battery-saving mode, average timings of turning off the saving mode, and overall battery-saving efficiency. RM and EM refer to regular and extreme saving modes, respectively.

Group	Task Completion Rate	Turning-off Rate		Turning-off Timing		Saving Efficiency	
		RM	EM	RM	EM	RM	EM
PB	61.11 %	16.64 %	11.82 %	50.78 s	48.06 s	90.35 %	92.71 %
PS	70.83 %	8.01 %	18.64 %	29.34 s	51.60 s	94.04 %	88.55 %
PRS	61.11 %	5.97 %	16.34 %	75.17 s	54.90 s	97.66 %	91.52 %

5.1 Activities in Battery-Saving Sessions

A summary of the quantitative results from participants' task completion, reactions to the saving mode (whether and when they turned it off), and overall battery-saving efficiency is shown in Table 3. Below, we elaborate on these results plus participants' screen touch behaviors that indicate their task engagement. Each subsection combines quantitative details together with qualitative insights learned from the interviews.

5.1.1 Task Completion. The average number of completed tasks (out of 6) is 3.67 (SD = 1.11) for PB, 4.25 (SD = .60) for PS, and 3.67 (SD = 1.18) for PRS groups, without significant differences shown in the Kruskal-Wallis test (H = 3.190, p = .203). This result suggested that the battery-saving indicators did not improve or hinder task completion among participants. Nevertheless, participants in all three groups reported that their task progress constantly influenced their tolerance of the performance drop caused by the battery saver, noting that they would turn off the saving mode when they felt stressed about completing the current task (PRS-1, PRS-8, PS-25, PS-26, PRS-28, PB-36), and that they would keep the saver on if they completed the task in advance (PRS-6, PRS-11, PB-19). In addition, participants shared their thoughts on the difficulty of tasks, where they perceived tasks in unfamiliar apps as more difficult (PRS-2, PRS-15).

Table 4. The effect of saving *mode*, *group*, and *app* on participants' turn-off behavior.

Variable	Estimate	Odds Ratio	CI_{lower}	CI_{upper}	p-value
(Intercept)	−1.15	0.32	0.14	0.70	0.005^{**}
Mode = EM	−0.47	0.63	0.28	1.38	0.248
Group = PS	−0.98	0.37	0.13	1.05	$0.061^{\dagger}$
Group = PRS	−1.34	0.26	0.08	0.81	0.020^{*}
App = Candy Crush	−0.62	0.54	0.25	1.17	0.117
App = Chrome	−0.96	0.38	0.17	0.88	0.023^{*}
App = Douyin	−0.05	0.95	0.47	1.93	0.884
App = Google Maps	−0.66	0.51	0.24	1.12	$0.093^{\dagger}$
App = YouTube	−1.51	0.22	0.08	0.58	0.002^{**}
Mode = EM × Group = PS	1.58	4.88	1.50	15.82	0.008^{**}
Mode = EM × Group = PRS	1.72	5.57	1.57	19.77	0.008^{**}

Model: `glmer(Turn Off ~ Mode * Group + App + (1 | Participant))`
References: Group = PB, Mode = RM, App = Poe
Significance: † $p < 0.1$, * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

5.1.2 Reactions to the Saving Mode. To understand participants' reactions to the saving mode, we examined two aspects: whether they turned off the mode while performing the tasks (Turn Off) and when they turned it off (Timing). We built a mixed-effects logistic regression model with participants' group assignment (versions of the saving indicator), the saving mode, and the app they were interacting with as predictors (see Table 4). The results reveal that the effects of the indicator designs on turning-off behaviors are significantly influenced by the interaction between the group and saving mode. To further understand this interaction, we examined the main effects of the indicator designs within each saving mode. In regular saving mode (RM), PS group has 0.37 times the odds compared to PB group to turn off the saver (marginally significant), while PRS group has 0.26 times the odds compared to PB group (significant). However, controlling for app effects, switching to the extreme saving mode (EM) significantly reversed the trend, with PS having 1.81 times the odds compared to PB to turn off the saver (OR = $0.37 \times 4.88 = 1.81$), and PRS having 1.45 times the odds compared to PB group (OR = $0.26 \times 5.57 = 1.45$). EM does not have a strong effect on PB group. Regarding the effect of different apps, three apps by Google (Chrome, Google Maps, YouTube) show a significant or marginally significant decrease in the odds of turning off the saver compared to the reference app (Poe).

To analyze when participants turned off the saver, we built a linear mixed-effects model with participants' group assignment, the saving mode, and the app they were interacting with as predictors (see Table 5). In the model, PRS group showed a marginally significant increase of 23.70 seconds in timing compared with PB. Notably, Douyin shows a significantly decreased timing of −26.16 seconds compared to the reference app (Chrome), indicating participants turn off the saver much earlier in Douyin. This is possibly due to the nature of short videos that not only has a higher demand for performance, but also makes users sensitive to lags. Other apps do not have a substantial influence on when participants turned off the saver.

Participants explained different ways they reacted to the saving mode during the post-study interviews. On the one hand, some participants always kept the saver on to ensure sufficient battery for task completion. Regardless of whether the saving statistics were available, they felt that knowing the battery is being saved provided an assurance for task completion (PB-36, PS-12). Particularly, participants in the PS and PRS group considered the amount of saved battery as a nudge as well as a sense of achievement (PS-7, PRS-11, PRS-28): "*It's necessary to have the saving number to accommodate performance drop*" (PRS-11).

Table 5. The effect of saving *mode*, *group*, and *app* on participants' timing of turning off the battery saver (in seconds).

Variable	Estimate	CI_{lower}	CI_{upper}	p-value
(Intercept)	65.72	43.50	88.05	$<0.001^{***}$
Mode = EM	−7.85	−27.62	11.90	0.433
Group = PS	−14.44	−38.28	9.61	0.234
Group = PRS	23.70	−3.39	51.17	$0.088^{\dagger}$
App = Candy Crush	−18.41	−41.72	4.94	0.120
App = Douyin	−26.16	−47.58	−4.67	0.018^{*}
App = Google Maps	−9.76	−32.53	13.13	0.397
App = Poe	−9.31	−30.78	12.43	0.388
App = YouTube	−10.52	−39.35	18.38	0.472
Mode = EM × Group = PS	16.00	−15.14	46.92	0.308
Mode = EM × Group = PRS	−11.43	−43.90	21.06	0.487

Model: `lmer(Timing ~ Mode * Group + App + (1 | Participant))`
References: Group = PB, Mode = RM, App = Chrome
Significance: † $p < 0.1$, * $p < 0.05$, *** $p < 0.001$

On the other hand, some participants reacted to the saving mode in more strategic ways to optimize the chance for completing as many tasks as possible. First, due to the decreased performance, participants would choose to turn off the saver to speed up task completion (PRS-1, PRS-6, PB-19, PRS-28). Interestingly, three participants reported their behaviors were affected by the remaining battery. When they believed the remaining battery was enough for the rest of the study, they chose to turn off the battery saver as a trade-off to improve performance (PS-3, PS-25, PB-34). In contrast, participants may sacrifice performance when battery was about to run out, as several of them reported being more tolerant of performance decrease when the battery was low or dropping fast (PRS-2, PRS-8, PRS-15, PS-32). In the PB group, low battery generally made participants less tolerant to performance drop (PB-16).

5.1.3 Saving Efficiency. Similar to timing, we used participants' group assignment, the saving mode, and the app they were interacting with as predictors to build the linear mixed-effects model to examine battery-saving efficiency (see Table 6). In this model, the reference condition is RM mode with the PB group. Similar to the turning-off behavior, participants' saving efficiency was significantly influenced by the interaction between the saving mode and group assignment. Under RM mode, PS participants showed 4.06% increase in saving efficiency higher than PB without significance; while PRS participants exhibited a significant 7.30% improvement compared to PB. However, when controlling for app, this trend reversed under EM mode. Compared to PB, there was a decrease in saving efficiency: −8.72 for PS ($\beta = 4.06 - 8.72 \approx -4.66$) and −8.85 for PRS ($\beta = 7.30 - 8.85 \approx -1.55$). Note that switching to the EM mode did not significantly influence the saving efficiency in PB group.

Across the tested apps, Douyin showed a significant reduction (9.88%) in saving efficiency compared to the reference app (YouTube). Poe also showed a significant decrease (7.20%) in saving efficiency compared to YouTube, while others did not show strong effects.

5.1.4 Screen Touch. To further understand how different versions of battery-saving indicator may influence participants' engagement with the device, we examined their screen touch behaviors. Focusing on the average touches per minute, we used participants' group assignment, the hardware configuration (two saving modes configuration plus the default configuration), the app they were interacting with, and the remaining battery level as predictors to build the linear mixed-effects model (see Table 7).

Table 6. The effect of *mode*, *group*, and *app* on participants' battery-saving efficiency (in percent).

Variable	Estimate	CI_{lower}	CI_{upper}	p-value
(Intercept)	94.63	88.81	100.45	<0.001***
Mode = EM	2.58	−2.96	8.12	0.361
Group = PS	4.06	−2.40	10.49	0.215
Group = PRS	7.30	0.77	13.82	0.029*
App = Candy Crush	−4.31	−9.83	1.22	0.127
App = Chrome	−1.42	−6.94	4.11	0.615
App = Douyin	−9.88	−15.40	−4.36	<0.001***
App = Google Maps	−3.37	−8.90	2.17	0.233
App = Poe	−7.20	−12.72	−1.68	0.011*
Mode = EM × Group = PS	−8.72	−16.57	−0.87	0.030*
Mode = EM × Group = PRS	−8.85	−16.70	−1.01	0.027*

Model: `lmer(Efficiency ~ Mode * Group + App + (1 | Participant))`
References: Group = PB, Mode = RM, App = YouTube
Significance: $^{\dagger}\ p < 0.1$, $^{*}\ p < 0.05$, $^{**}\ p < 0.01$, $^{***}\ p < 0.001$

Table 7. The effect of *config*, *group*, *app*, and *remaining battery* on participants' touches per minute.

Variable	Estimate	CI_{lower}	CI_{upper}	p-value
(Intercept)	58.65	53.93	63.35	<0.001***
Config = EM	−5.28	−9.42	−1.13	0.013*
Config = RM	−2.00	−6.20	2.20	0.351
Group = PS	6.16	1.28	11.03	0.014*
Group = PRS	3.26	−1.62	8.13	0.189
App = Candy Crush	−43.08	−46.49	−39.68	<0.001***
App = Chrome	−15.75	−19.15	−12.35	<0.001***
App = Douyin	−33.15	−36.56	−29.74	<0.001***
App = Google Maps	−25.16	−28.57	−21.75	<0.001***
App = YouTube	−38.41	−41.81	−35.00	<0.001***
Battery	0.18	0.04	0.31	0.010*
Config = EM × Group = PS	−2.59	−8.56	3.37	0.394
Config = RM × Group = PS	−2.29	−8.15	3.56	0.443
Config = EM × Group = PRS	−1.57	−7.45	4.32	0.601
Config = RM × Group = PRS	−0.29	−6.22	5.63	0.922

Model: `lmer(Frequency ~ Config * Group + App + Battery + (1 | Participant))`
References: Group = PB, Config = Default, App = Poe
Significance: $^{\dagger}\ p < 0.1$, $^{*}\ p < 0.05$, $^{**}\ p < 0.01$, $^{***}\ p < 0.001$

There are several significant main effects in the results. First, we found the PS group showed a significant increase of 6.16 touches per minute compared to the PB group, suggesting that the battery-saving indicator with saving statistics might have encouraged more screen touches. However, this pattern was not observed in the PRS group, likely because the continuous presentation of real-time saving statistics created a more direct and persistent awareness of battery usage, thereby tempering the impulse to interact unnecessarily with the screen. Second, the remaining battery level showed a small but significant positive effect on touch frequency, indicating that users tended to touch the screen more often when there was more battery left. In other words, they may avoid screen touches when the battery was low. Third, compared to the reference app (Poe), all apps showed a significant decrease in touch frequency, with Candy Crush exhibiting the lowest touch frequency, followed by YouTube, Douyin, Google Maps, and Chrome. This can be possibly due to the text-based interaction on Poe, which requires frequent keyboard input, whereas other applications primarily utilize touch gestures for navigation. Nevertheless, the post-hoc pair-wise comparison further reveals that the touch frequency of any two apps was significantly different from each other (all $p < .05$), with the exception of the comparison between Candy Crush and YouTube being marginally significant ($p = .080$). These results suggested that each app elicited a distinct pattern of touch interaction.

When comparing the saving mode to the default configuration, we found that under EM, there was a significant decrease of -5.28 touches per minute, indicating that severe performance drop led to reduced touch frequency.

5.2 Expectations and Preferences for Battery Saving

By analyzing interview data related to participants' expectations and preferences for saving battery in their daily life, we organized the findings into three parts: (1) the trade-off between device performance and battery saving, (2) how important saving battery is to them, and (3) the features they found helpful for battery saving.

5.2.1 Trade-off Between Device Performance and Battery Saving. When it comes to whether to prioritize performance or the battery saving, participants often took different strategies to navigate this trade-off. Some expressed a strong preference for performance over battery conservation, stating that they could only accept battery savers that had minimal impact on performance (PRS-14, PRS-15, PS-32, PB-36), such as slightly lowering the screen brightness. In contrast, some participants who were less sensitive towards performance change adopted a "battery-first" approach, and they were willing to sacrifice performance in exchange of extended battery life (PB-9, PRS-11). In some extreme cases, they would even shutdown their phone to save battery (PB-19, PB-21). For example, PB-9 said "*although it makes the phone slower, it is more important for me to save battery for urgent situations.*" Additionally, some participants sought to strike a balance between performance and battery, adjusting their strategies depending on the interaction context. The remaining battery is generally considered in the decision-making process (PRS-2, PRS-6, PRS-8, PRS-10, PRS-11). Other factors may also be considered by individuals. For example, PRS-14 would prioritize battery when they are outside, while PRS-1 would prioritize performance in games.

5.2.2 Perceived Importance of Battery Saving. Most participants agreed that battery saving is important, highlighting that saving mode can reduce low-battery anxiety (PRS-1, PRS-6, PRS-14, PB-21, PRS-28, PB-36), benefit long-term battery health (PRS-8, PB-16, PB-18), and eliminate the need for bringing extra chargers or power banks (PRS-6, PB-18, PS-32). Specifically, PRS-10 added that as mobile phone is an essential part of various daily activities such as taking transportation, making payment, and keeping connected with others, saving battery is extremely important, and PB-9 mentioned that saving mode is helpful in emergency. In addition, two participants, PS-7 and PRS-28, acknowledged the positive impact of saving batteries on the environment over time.

On the other hand, other participants thought that battery saving is not important, especially future technologies like superchargers (PS-12, PRS-14) will make saving mode unnecessary. PRS-11 added that "*Mobile phones use limited energy. For other scenarios like data centers and air conditioning, they use a lot of energy.*" At a larger scale, PB-17 thought that advancements in renewable energy generation would make it less important to save energy, and stated that they always purchased a phone with the largest battery so that they would not need to worry about battery saving.

5.2.3 Desired Features for Battery Saving. Reflecting on the study experience, participants shared their ideas of how they would like their phone's battery to be saved in daily life. Many participants wanted an automatic saving mode, where the system could intelligently switch between different strategies based on interaction scenarios and user preferences (PS-7, PS-12, PB-21, PRS-14, PB-36), partially as they do not want to learn about different saving mechanisms due to the technical complexity (PB-17). Some of them expressed a desire for full control over the extent of performance drop (PRS-1, PRS-2, PB-18, PRS-15, PB-20, PRS-28), while ensuring the device can respond fast in urgent situations (PRS-6). They also expressed a need for more battery-saving options that do not affect the performance of the phone. Responding to the brightness-lowering experience in the study, although some participants noticed the change (PS-5, PS-7, PRS-11, PB-20, PS-32), they all felt such changes were acceptable (PRS-1, PRS-2, PRS-14, PB-21, PS-32, PB-36). In addition, participants also mentioned other ways to save battery, such as by disabling the Wi-Fi, Bluetooth and location services (PRS-1, PRS-10, PB-19, PRS-28). In short, participants suggested that an ideal battery-saving system should provide flexible configuration options to satisfy users' needs for battery saving in various contexts, while offering an automatic mode for those who prefer a more hands-off approach.

6 Discussion

Our findings showed that during intensive interactions with mobile phones at low battery levels, visual indicators of battery-saving status can help users alleviate stress and effectively navigate their current tasks. However, the impact of these indicators was closely tied to the extent of performance drop, where users could feel frustrated if the performance degradation is significant. Drawing from these findings, we discuss the implications and opportunities for balancing battery efficiency and user experience through the lens of user interface design. Additionally, we reflect on lessons learned from our lab experiment, regarding what had worked well and what can be improved, to inform future research seeking to capture user behaviors related to battery saving.

6.1 The Effects of the Battery-Saving Indicator

Our study showed that under RM (regular saving mode), the visual indicator of battery-saving status was effective in encouraging participants to stay in the saving mode (i.e., less likely to turn off the mode) as well as increasing battery-saving efficiency. Several factors may contribute to this effectiveness. First, the visual indicator provided clear and immediate feedback about how much battery could be saved just within a few minutes, reinforcing users' awareness of how such a short period of performance downgrade can help with battery saving. This information was likely to motivate participants to keep the saving mode on, as they could see the positive impact of their choices on battery life. Similar findings were also reported in prior work on system visibility [30, 42, 43], which has shown that users are more tolerant for performance limitations when they understand the rationale and perceive tangible benefits.

Second, while the saving indicator helped mitigate potential frustration caused by performance drop under RM, it became less effective and even adversely led participants to turn off the saving mode under EM, where the performance of the device further degraded. Combining the interview findings, we suspect this was due to several reasons. Partly, participants experienced frustration with the noticeable drop in performance, which diminished their willingness to accept the limitations imposed by the extreme saving mode. The degradation in functionality thus may have overshadowed the perceived benefits of battery conservation, leading them to prioritize performance over battery life. For the PRS group, the real-time statistics demonstrated an even more counterproductive effect: by making performance degradation immediately salient, it might make battery saving a costly trade-off, leading users to deactivate the mode. Similar findings have been reported while studying the associations between psychological impacts of delay on commuters and visual indicators of the real-time transportation information [52]. Our analysis of touch frequency further supported this user observation, with

EM causing a significant reduction of over 5 touches per minute compared to the default configuration, suggesting that the EM negatively impacted user engagement. Another possible reason, as PS and PRS groups brought up during the interviews, was that, knowing the amount of battery that has been saved in prior sessions can help them make strategic decisions to turn off the battery saver in later sessions to reclaim better performance: "*It made me braver to turn it off if I have already saved some battery*" (PS-3).

Based on these observations, when applying battery-saving frameworks on mobile devices, designers and developers could consider providing visual indicators of the saving status differently depending on the performance degradation. For example, when the degradation is moderate, the indicators could highlight the concrete benefits of battery saving, such as the amount of time left for different applications (e.g., "saved 5 minutes for browsing or 3 minutes video watching"), similar to the strategies used in the recent study [31]. Conversely, when the degradation is significant, it might be more effective to omit visual indicators altogether. Instead, designers could focus on enhancing user control by allowing users to easily switch between saving modes based on their current needs. This could involve a simple toggle or button that enables users to quickly revert to regular mode when they require full device functionality.

6.2 Toward a Sustainable and User-Friendly Battery-Saving Interface

Drawing on our lab study and interviews, we see two promising directions for building sustainable and user-friendly battery-saving interface: (1) framing battery saving as a reward for strategic interactions, and (2) promoting motivation to save batteries by cultivating ecological awareness.

6.2.1 Battery-Saving as a Reward of Strategic Interaction. Our interviews revealed that many users enjoy making strategic decisions to optimize battery and performance, a finding consistent with Choe et al.'s observation that designers could enhance user satisfaction and efficiency on mobile devices by allowing users to customize their settings according to individual needs [47]. Similarly, Froehlich et al. has employed game elements, such as points, challenges, and leaderboards, to encourage sustainable behavior [20]. Building on these works, we envision "battery-saving kits" that allow users to select combinations of strategies tailored to their interaction needs and preferences. These kits could include options such as adjusting screen brightness, disabling Bluetooth, or limiting background app activity, enabling users to create personalized battery-saving profiles. In addition, the kits could be packed with a smart power analyzer, such as the one proposed by Datta et al. [11] to help users understand their usage patterns and make recommendations for optimizing battery life. By allowing users to customize their settings according to individual needs, designers could enhance user satisfaction and efficiency on mobile devices [47]. This emphasis on personalized experiences aligns with Blom and Monk's theory, which underscores users' broad expectation to receive tailored technology interactions [44].

6.2.2 Cultivating Ecological Awareness. In our study, users are primarily motivated by a utilitarian perspective, seeking to extend their device's battery life for their compensation-related tasks, with only two participants (PS-7, PRS-28) bringing up environmental sustainability as a reason for saving battery in our follow-up interviews. To shift this utilitarian perspective and increase their awareness of the importance of mobile battery saving at a larger scale, designers could consider integrating concrete sustainability-related benefits into the battery-saving application. With the similar approach used by Kjeldskov et al. [28], we could encourage users to adopt battery-saving practices not only for their own benefit, but also for the greater good of the environment and society. In other works [19], researchers have demonstrated the value of environmental sensing feedback in helping users understand how their behaviors impact the world around them. Although the environmental benefits of saving battery power on mobile devices cannot be measured directly, we can adapt existing models to estimate ecological impact, such as equivalent carbon emission reductions or energy savings, and present these to users in relatable terms. This approach has shown effective in some commercial applications. For example, Alipay's

Ant Forest app, which turns low-carbon actions into virtual saplings, combined with multi-dimensional rewards (enjoyment, achievement, emotional attachment) sustain long-term engagement [38, 65]. Likewise, Kjeldskov et al. demonstrated that comparative usage visualizations and social influence features markedly boost awareness and motivation for energy saving [28]. By combining visible environmental impact, peer benchmarking, and continuous mobile feedback, users could be inspired to save battery not just for personal gain but for broader ecological benefit.

6.3 Lessons Learned From Studying Battery-Saving Behaviors in Lab Environments

While controlled laboratory experiments cannot fully capture the complexity of real-world battery-saving behaviors, our study establishes an empirical foundation for subsequent field investigations in two key aspects: reward model and task design.

First, to carry out an experiment in the field, one challenge is the ubiquitous charging resources may prevent people from enabling the saving mode on their phones, especially for the tasks they must complete. Therefore, we need to first convince users the importance of saving batteries on their phones. In addition to cultivating ecological awareness as mentioned previously, our performance-based compensation model has demonstrated effectiveness in motivating participants to adopt strategic thinking when balancing performance needs against battery constraints. Although these strategies may not fully align with real-world decision-making processes, they reveal fundamental behavioral patterns that persist across contexts such as low battery anxiety [64] and loss aversion [48]. For instance, many participants voluntarily chose to persist with battery-saving mode during less time-sensitive tasks, demonstrating similar behavioral patterns to those observed in other field studies [64]. Thus, even in a field study setting, establishing a clear incentive structure (e.g., through social recognition, or environmental impact feedback) could effectively motivate sustained engagement with power-saving features while maintaining ecological validity.

Second, battery consumption patterns vary significantly across different applications, and users may save different levels of batteries depending on their preferred apps and interaction routines. This variability made it challenging for consistent evaluation of saving efficiency across participants. In our study, the use of goal-oriented tasks, such as answering specific questions from videos or navigating predetermined routes, created a standardized framework that enabled straightforward comparison of battery-saving efficiency across all participants. These controlled tasks ensured that each participant engaged in similar types and levels of smartphone interaction, thereby eliminating the confounding effects of different app usage patterns. Thus, future research may focus on either evaluating the saving efficiency for specific apps or develop app-agnostic evaluation metrics that can normalize across diverse usage patterns.

7 Limitations and Future Work

First, before a field testing in real-life situations, the findings of this study cannot be generalized. Although we attempted to simulate real-life scenarios by designing tasks in six apps from different categories and linking task completion with compensation, the lab environment may not fully capture the complexities and nuances of real-world usage. Second, the performance-based compensation could have introduced bias to participants' behavior (as explained in Section 6.3). Third, while we examined most common app categories (See Section 3), they did not cover every possible use case such as shopping and photo taking, due to the limitation of time and lab environment. Relatedly, participants' expectations of the study varied: two (PRS-6, PRS-14) thought that they were in an experiment context, and the battery level would be safe, while another two (PRS-14, PB-36) mentioned that they were too focused on the task to notice the battery saver or the indicator. Moreover, some participants, especially those from PS and PRS groups needed to learn the new user interface of the battery saver. Under the pressure of completing tasks within a limited time, this unfamiliarity could have influenced their experience and performance in the study [8, 18, 60].

As the first step to examine the impact of UI design on user experience in a battery-saving context, our study systematically documented nuanced behavioral patterns in how users respond to visual power-saving indicators and established their tolerance thresholds across varying degrees of performance compromise. These findings not only revealed how UI elements can be designed to shape battery-saving behaviors but also served as an important empirical foundation for designing adaptive battery management systems in the future. Going forward, we plan to introduce a customization system that allows users to set their own preferences for the battery-saving UI, as described in Section 6.2. Furthermore, we aim to extend our study to a larger sample size in a real-life field study to better understand the impact of the battery-saving UI over time.

8 Conclusion

This study explores in battery-saving scenarios where system performance drops, how UI designs that present different levels of details about battery saving status can affect user experience and their reactions to the saving mode. Our findings suggested that under moderate performance degradation, the real-time battery-saving statistics can effectively enhance user experience while encouraging sustainable interaction behaviors compared to the other two versions. But when the performance of the device drops aggressively, the indicator worsened user experience and failed to improve saving efficiency. Through post-study interviews, we highlighted the key aspects that influenced participants' reactions to the saving mode, such as the sense of control and feedback clarity. With the lessons learned, we further discussed how to build sustainable and user-friendly battery-saving interfaces and how to carry this investigation forward through a field study.

Acknowledgments

We thank our participants for their time and contributions to this study. We also thank the anonymous reviewers for their thoughtful suggestions. The project was supported by City University of Hong Kong (#9220150 and #9610597).

References

[1] Bhojan Anand, Karthik Thirugnanam, Jeena Sebastian, Pravein G. Kannan, Akhihebbal L. Ananda, Mun Choon Chan, and Rajesh Krishna Balan. 2011. Adaptive display power management for mobile games. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services* (Bethesda, Maryland, USA) *(MobiSys '11)*. Association for Computing Machinery, New York, NY, USA, 57–70. doi:10.1145/1999995.2000002

[2] Douglas Bates, Reinhold Kliegl, Shravan Vasishth, and Harald Baayen. 2018. Parsimonious Mixed Models. arXiv:1506.04967 [stat.ME] https://arxiv.org/abs/1506.04967

[3] Virginia Braun and Victoria Clarke and. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3, 2 (2006), 77–101. arXiv:https://www.tandfonline.com/doi/pdf/10.1191/1478088706qp063oa doi:10.1191/1478088706qp063oa

[4] Rajeev BV and Kale Bakula. 2015. A developer's insights into performance optimizations for mobile web apps. In *2015 IEEE International Advance Computing Conference (IACC)*. 671–675. doi:10.1109/IADCC.2015.7154791

[5] Aaron Carroll and Gernot Heiser. 2010. An analysis of power consumption in a smartphone. In *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference* (Boston, MA) *(USENIXATC'10)*. USENIX Association, USA, 21.

[6] Xiaomeng Chen, Ning Ding, Abhilash Jindal, Y. Charlie Hu, Maruti Gupta, and Rath Vannithamby. 2015. Smartphone Energy Drain in the Wild: Analysis and Implications. In *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems* (Portland, Oregon, USA) *(SIGMETRICS '15)*. Association for Computing Machinery, New York, NY, USA, 151–164. doi:10.1145/2745844.2745875

[7] Xiaomeng Chen, Abhilash Jindal, Ning Ding, Yu Charlie Hu, Maruti Gupta, and Rath Vannithamby. 2015. Smartphone Background Activities in the Wild: Origin, Energy Drain, and Optimization. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking* (Paris, France) *(MobiCom '15)*. Association for Computing Machinery, New York, NY, USA, 40–52. doi:10.1145/2789168.2790107

[8] Ju-Choel Choi. 2020. User Familiarity and Satisfaction With Food Delivery Mobile Apps. *Sage Open* 10, 4 (2020), 2158244020970563. arXiv:https://doi.org/10.1177/2158244020970563 doi:10.1177/2158244020970563

[9] Ki-Seok Chung and TaeHoon Kim. 2012. Benefits of the big . LITTLE Architecture. https://www.yumpu.com/en/document/view/41694370/benefits-of-the-biglittle-architecture-arm

[10] Pranab Dash and Y. Charlie Hu. 2021. How much battery does dark mode save? an accurate OLED display power profiler for modern smartphones. In *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services* (Virtual Event, Wisconsin) *(MobiSys '21)*. Association for Computing Machinery, New York, NY, USA, 323–335. doi:10.1145/3458864.3467682

[11] Soumya Kanti Datta, Christian Bonnet, and Navid Nikaein. 2014. Personalized power saving profiles generation analyzing smart device usage patterns. In *2014 7th IFIP Wireless and Mobile Networking Conference (WMNC)*. 1–8. doi:10.1109/WMNC.2014.6878858

[12] Thomas Davies and Ashweeni Beeharee. 2012. The case of the missed icon: change blindness on mobile devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 1451–1460. doi:10.1145/2207676.2208606

[13] Android Developers. [n. d.]. *Android Debug Bridge (adb)*. https://developer.android.com/tools/adb

[14] DeviceSpecifications. 2023. *Samsung Galaxy A54 5G - Specifications*. https://www.devicespecifications.com/en/model/d6a85c23

[15] The Linux Kernel documentation. [n. d.]. *CPU hotplug in the Kernel*. https://docs.kernel.org/core-api/cpu_hotplug.html

[16] Begum Egilmez, Matthew Schuchhardt, Gokhan Memik, Raid Ayoub, Niranjan Soundararajan, and Michael Kishinevsky. 2017. User-aware Frame Rate Management in Android Smartphones. *ACM Trans. Embed. Comput. Syst.* 16, 5s, Article 131 (sep 2017), 17 pages. doi:10.1145/3126539

[17] Denzil Ferreira, Eija Ferreira, Jorge Goncalves, Vassilis Kostakos, and Anind K. Dey. 2013. Revisiting human-battery interaction with an interactive battery interface. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Zurich, Switzerland) *(UbiComp '13)*. Association for Computing Machinery, New York, NY, USA, 563–572. doi:10.1145/2493432.2493465

[18] C. Flavián, M. Guinalíu, and R. Gurrea. 2006. The influence of familiarity and usability on loyalty to online journalistic services: The role of user experience. *Journal of Retailing and Consumer Services* 13, 5 (2006), 363–375. doi:10.1016/j.jretconser.2005.11.003

[19] Jon Froehlich. 2009. Sensing and feedback of everyday activities to promote environmentally sustainable behaviors. In *Doctoral Colloquium in the Adjunct Proceedings of UbiComp, Orlando, Florida, USA*. Citeseer.

[20] Jon E Froehlich. 2015. Gamifying green: gamification and environmental sustainability. (2015). https://ieeexplore.ieee.org/document/7040536

[21] Erik Geelhoed, Peter Toft, Suzanne Roberts, and Patrick Hyland. 1995. To influence time perception. In *Conference Companion on Human Factors in Computing Systems*. 272–273. https://dl.acm.org/doi/pdf/10.1145/223355.223670

[22] Matthew Halpern, Yuhao Zhu, and Vijay Janapa Reddi. 2016. Mobile CPU's rise to power: Quantifying the impact of generational mobile CPU design trends on performance, energy, and user satisfaction. In *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 64–76.

[23] Max Hort, Maria Kechagia, Federica Sarro, and Mark Harman. 2022. A Survey of Performance Optimization for Mobile Applications. *IEEE Transactions on Software Engineering* 48, 8 (2022), 2879–2904. doi:10.1109/TSE.2021.3071193

[24] Chanyou Hwang, Saumay Pushp, Changyoung Koh, Jungpil Yoon, Yunxin Liu, Seungpyo Choi, and Junehwa Song. 2017. RAVEN: Perception-aware Optimization of Power Consumption for Mobile Games. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking* (Snowbird, Utah, USA) *(MobiCom '17)*. Association for Computing Machinery, New York, NY, USA, 422–434. doi:10.1145/3117811.3117841

[25] Nayeem Islam and Roy Want. 2014. Smartphones: Past, Present, and Future. *IEEE Pervasive Computing* 13, 4 (2014), 89–92. doi:10.1109/MPRV.2014.74

[26] Wonwoo Jung, Yohan Chon, Dongwon Kim, and Hojung Cha. 2014. Powerlet: an active battery interface for smartphones. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Seattle, Washington) *(UbiComp '14)*. Association for Computing Machinery, New York, NY, USA, 45–56. doi:10.1145/2632048.2636085

[27] David Kadjo, Raid Ayoub, Michael Kishinevsky, and Paul V. Gratz. 2015. A control-theoretic approach for energy efficient CPU-GPU subsystem in mobile platforms. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. 1–6. doi:10.1145/2744769.2744773

[28] Jesper Kjeldskov, Mikael B. Skov, Jeni Paay, and Rahuvaran Pathmanathan. 2012. Using mobile phones to support sustainability: a field study of residential electricity consumption. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 2347–2356. doi:10.1145/2207676.2208395

[29] Jaspreet Kumar. 2024. Mobile App Performance Optimization: A Comprehensive Guide. *IJFMR - International Journal For Multidisciplinary Research* (Dec 2024). doi:10.36948/ijfmr.2024.v06i06.33520

[30] Carine Lallemand and Guillaume Gronier. 2012. Enhancing User eXperience during waiting time in HCI: contributions of cognitive psychology. In *Proceedings of the Designing Interactive Systems Conference* (Newcastle Upon Tyne, United Kingdom) *(DIS '12)*. Association for Computing Machinery, New York, NY, USA, 751–760. doi:10.1145/2317956.2318069

[31] Sera Lee, Dae R Jeong, Junyoung Choi, Jaeheon Kwak, Seoyun Son, Jean Y Song, and Insik Shin. 2024. SERENUS: Alleviating Low-Battery Anxiety Through Real-time, Accurate, and User-Friendly Energy Consumption Prediction of Mobile Applications. In *Proceedings of the*

37th Annual ACM Symposium on User Interface Software and Technology. 1–20. https://doi.org/10.1145/3654777.3676437

[32] Tong Li, Yali Fan, Yong Li, Sasu Tarkoma, and Pan Hui. 2023. Understanding the Long-Term Evolution of Mobile App Usage. *IEEE Transactions on Mobile Computing* 22, 2 (2023), 1213–1230. doi:10.1109/TMC.2021.3098664

[33] Xueliang Li, Junyang Chen, Yepang Liu, Kaishun Wu, and John P Gallagher. 2023. Combatting Energy Issues for Mobile Applications. *ACM Transactions on Software Engineering and Methodology* 32, 1 (2023), 1–44.

[34] Xueliang Li, Shicong Hong, Junyang Chen, Guihai Yan, and Kaishun Wu. 2022. Using Psychophysics to Guide Power Adaptation for Input Methods on Mobile Architectures. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. 514–527. doi:10.1109/HPCA53966.2022.00045

[35] Xueliang Li, Zhuobin Shi, Junyang Chen, and Yepang Liu. 2023. Realizing Emotional Interactions to Learn User Experience and Guide Energy Optimization for Mobile Architectures. In *Proceedings of the 55th Annual IEEE/ACM International Symposium on Microarchitecture* (Chicago, Illinois, USA) *(MICRO '22)*. IEEE Press, 868–884. doi:10.1109/MICRO56248.2022.00064

[36] Chen Ling, Wonil Hwang, and Gavriel Salvendy. 2006. Diversified users' satisfaction with advanced mobile phone features. *Univers. Access Inf. Soc.* 5, 2 (July 2006), 239–249. doi:10.1007/s10209-006-0028-x

[37] Tara Matthews, Tye Rattenbury, and Scott Carter. 2007. Defining, designing, and evaluating peripheral displays: An analysis using activity theory. *Human–Computer Interaction* 22, 1-2 (2007), 221–261.

[38] Lingyun Mi, Ting Xu, Yuhuan Sun, Jingjing Zhao, Tao Lv, Xiaoli Gan, Ke Shang, and Lijie Qiao. 2021. Playing Ant Forest to promote online green behavior: A new perspective on uses and gratifications. *Journal of Environmental Management* 278 (2021), 111544. doi:10.1016/j.jenvman.2020.111544

[39] Mohamed Musthag, Andrew Raij, Deepak Ganesan, Santosh Kumar, and Saul Shiffman. 2011. Exploring micro-incentive strategies for participant compensation in high-burden studies. In *Proceedings of the 13th International Conference on Ubiquitous Computing* (Beijing, China) *(UbiComp '11)*. Association for Computing Machinery, New York, NY, USA, 435–444. doi:10.1145/2030112.2030170

[40] Brad A Myers. 1985. The importance of percent-done progress indicators for computer-human interfaces. *ACM SIGCHI Bulletin* 16, 4 (1985), 11–17. https://doi.org/10.1145/1165385.317459

[41] In Jae Myung. 2000. The Importance of Complexity in Model Selection. *Journal of Mathematical Psychology* 44, 1 (2000), 190–204. doi:10.1006/jmps.1999.1283

[42] Fiona Fui-Hoon Nah. 2004. A study on tolerable waiting time: how long are web users willing to wait? *Behaviour & Information Technology* 23, 3 (2004), 153–163. https://doi.org/10.1080/01449290410001669914

[43] Jakob Nielsen. 1994. *10 Usability Heuristics for User Interface Design*. https://www.nngroup.com/articles/ten-usability-heuristics/

[44] Jan o. Blom and Andrew F. Monk. 2003. Theory of Personalization of Appearance: Why Users Personalize Their PCs and Mobile Phones. *Human-Computer Interaction* 18, 3 (2003), 193–228. arXiv:https://www.tandfonline.com/doi/pdf/10.1207/S15327051HCI1803_1 doi:10.1207/S15327051HCI1803_1

[45] Jurn-Gyu Park, Chen-Ying Hsieh, Nikil Dutt, and Sung-Soo Lim. 2016. Co-Cap: energy-efficient cooperative CPU-GPU frequency capping for mobile games. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing* (Pisa, Italy) *(SAC '16)*. Association for Computing Machinery, New York, NY, USA, 1717–1723. doi:10.1145/2851613.2851671

[46] David Patterson, Jeffrey M. Gilbert, Marco Gruteser, Efren Robles, Krishna Sekar, Yong Wei, and Tenghui Zhu. 2024. Energy and Emissions of Machine Learning on Smartphones vs. the Cloud. *Commun. ACM* 67, 2 (jan 2024), 86–97. doi:10.1145/3624719

[47] Chen Liao Pilsung Choe and Wei Sun. 2012. Providing customisation guidelines of mobile phones for manufacturers. *Behaviour & Information Technology* 31, 10 (2012), 983–994. arXiv:https://doi.org/10.1080/0144929X.2011.644581 doi:10.1080/0144929X.2011.644581

[48] Geoffrey F. Potts. 2011. Impact of reward and punishment motivation on behavior monitoring as indexed by the error-related negativity. *International Journal of Psychophysiology* 81, 3 (2011), 324–331. doi:10.1016/j.ijpsycho.2011.07.020 PROCEEDINGS OF THE 15TH WORLD CONGRESS OF PSYCHOPHYSIOLOGY of the International Organization of Psychophysiology (I.O.P.) Budapest, Hungary September 1-4, 2010.

[49] Michael JD Powell et al. 2009. The BOBYQA algorithm for bound constrained optimization without derivatives. *Cambridge NA Report NA2009/06, University of Cambridge, Cambridge* 26 (2009), 26–46.

[50] Emirhan Poyraz and Gokhan Memik. 2019. Using Built-In Sensors to Predict and Utilize User Satisfaction for CPU Settings on Smartphones. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 1, Article 21 (mar 2019), 25 pages. doi:10.1145/3314408

[51] Alok Prakash, Hussam Amrouch, Muhammad Shafique, Tulika Mitra, and Jörg Henkel. 2016. Improving mobile gaming performance through cooperative CPU-GPU thermal management. In *Proceedings of the 53rd Annual Design Automation Conference* (Austin, Texas) *(DAC '16)*. Association for Computing Machinery, New York, NY, USA, Article 47, 6 pages. doi:10.1145/2897937.2898031

[52] Mahdi Rezapour and F Richard Ferraro. 2022. Investigating the association of psychological impacts of delay on the perceived importance of real-time information: three-stage least square instrumental regression. *Discover Psychology* 2, 1 (2022), 27. https://doi.org/10.1007/s44202-022-00039-1

[53] Rui Rua and João Saraiva. 2023. E-MANAFA: Energy Monitoring and ANAlysis tool For Android. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering* (Rochester, MI, USA) *(ASE '22)*. Association for Computing Machinery, New York, NY, USA, Article 202, 4 pages. doi:10.1145/3551349.3561342

[54] Youngbin Seo, Jeongki Kim, and Euiseong Seo. 2012. Effectiveness analysis of DVFS and DPM in mobile devices. *Journal of Computer Science and Technology* 27 (07 2012). doi:10.1007/s11390-012-1264-6

[55] Munam Ali Shah, Naila Naheed, and Sijing Zhang. 2015. Energy efficiency in smartphones: A survey on modern tools and techniques. In *2015 21st International Conference on Automation and Computing (ICAC)*. 1–6. doi:10.1109/IConAC.2015.7313972

[56] Wei Song, Dian Tjondronegoro, and Michael J. Docherty. 2010. Exploration and Optimization of User Experience in Viewing Videos on a Mobile Phone. *Int. J. Softw. Eng. Knowl. Eng.* 20 (2010), 1045–1075. https://api.semanticscholar.org/CorpusID:4413856

[57] Android Source. [n. d.]. *Mweasure poer values*. https://source.android.com/docs/core/power/values

[58] Statista. 2024. *Number of smartphone users worldwide*. https://www.statista.com/forecasts/1143723/smartphone-users-in-the-world

[59] Statista. 2024. *Distribution of the time spent using mobile apps worldwide in 2024, by category*. https://www.statista.com/statistics/1465726/global-daily-time-spent-mobile-usage/

[60] David Schuff Stella Tomasi and Ozgur Turetken. 2018. Understanding novelty: how task structure and tool familiarity moderate performance. *Behaviour & Information Technology* 37, 4 (2018), 406–418. arXiv:https://doi.org/10.1080/0144929X.2018.1441325 doi:10.1080/0144929X.2018.1441325

[61] Jonas Wanner, Lukas-Valentin Herm, Kai Heinrich, and Christian Janiesch. 2022. The effect of transparency and trust on intelligent system acceptance: Evidence from a user-based study. *Electronic Markets* 32, 4 (2022), 2079–2102.

[62] John Wu. [n. d.]. *Magisk*. https://github.com/topjohnwu/Magisk

[63] Noor Zaman and Fatimah Abdulaziz Almusalli. 2017. Review: Smartphones power consumption & energy saving techniques. In *2017 International Conference on Innovations in Electrical Engineering and Computational Technologies (ICIEECT)*. 1–7. doi:10.1109/ICIEECT.2017.7916593

[64] Yu Zhang, Guoming Tang, Qianyi Huang, Kui Wu, Yangjing Wu, and Yi Wang. 2022. Investigating Low-Battery Anxiety of Mobile Users. In *2022 IEEE International Conferences on Internet of Things (iThings) and IEEE Green Computing & Communications (GreenCom) and IEEE Cyber, Physical & Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*. 272–279. doi:10.1109/iThings-GreenCom-CPSCom-SmartData-Cybermatics55523.2022.00022

[65] Yixiang Zhang, Shiqi Xiao, and Guanghui Zhou. 2020. User continuance of a green behavior mobile application in China: An empirical study of Ant Forest. *Journal of Cleaner Production* 242 (2020), 118497. doi:10.1016/j.jclepro.2019.118497

Appendix

A Model Selection Process

A.1 Turn Off

List of models we built to examine whether participants turned off the saver (Turn Off) using `glmer`:

```
base_model: Turn_Off ~ 1 + (1 | Participant)
model_1: Turn_Off ~ Mode + (1 | Participant)
model_2: Turn_Off ~ Mode + Group + (1 | Participant)
model_3: Turn_Off ~ Mode * Group + (1 | Participant)
model_4: Turn_Off ~ Mode * Group + App + (1 | Participant)
model_5: Turn_Off ~ Mode * Group + App + Rem_Battery + (1 | Participant)
```

ANOVA table for the models:

```
           npar    AIC    BIC  logLik -2*log(L)   Chisq Df Pr(>Chisq)
base_model    2 493.94 502.89 -244.97    489.94
model_1       3 491.31 504.73 -242.66    485.31  4.6261  1   0.031489 *
model_2       5 494.60 516.97 -242.30    484.60  0.7119  2   0.700500
model_3       7 489.36 520.68 -237.68    475.36  9.2380  2   0.009863 **
model_4      12 483.55 537.24 -229.78    459.55 15.8132  5   0.007398 **
model_5      13 485.55 543.71 -229.77    459.55  0.0023  1   0.961666
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Based on the results, we can see that the model with the lowest AIC is model 4, which is significantly better than model 3. A likelihood ratio test also shows that model 3 is significantly better than model 1 ($\chi^2(4) = 9.96$, p = .041), which is also significantly better than the null model. Model 5 does not improve the fit of model 4. Therefore, model 4 is selected for the analysis of *Turn Off*, which includes the interaction between *group* and *mode*, as well as *app* as a fixed effect.

A.2 Timing

List of models we built to examine when participants turned off the saver (Timing) using `lmer`:

```
base_model: Timing ~ 1 + (1 | Participant)
model_1: Timing ~ Mode + (1 | Participant)
model_2: Timing ~ Mode + Group + (1 | Participant)
model_3: Timing ~ Mode * Group + (1 | Participant)
model_4: Timing ~ Mode * Group + App + (1 | Participant)
model_5: Timing ~ Mode * Group + App + Rem_Battery + (1 | Participant)
```

ANOVA table for the models:

```
           npar    AIC    BIC  logLik -2*log(L)  Chisq Df Pr(>Chisq)
base_model    3 804.57 811.83 -399.28    798.57
model_1       4 806.57 816.24 -399.28    798.57 0.0000  1     0.9983
model_2       6 806.87 821.39 -397.44    794.87 3.6958  2     0.1576
model_3       8 807.68 827.03 -395.84    791.68 3.1888  2     0.2030
model_4      13 810.52 841.97 -392.26    784.52 7.1601  5     0.2090
model_5      14 811.89 845.75 -391.94    783.89 0.6360  1     0.4252
```

Based on the results, none of the models significantly improved the fit of the previous model. Nevertheless, for consistency, model 4 is selected for the analysis of *Timing*, which includes the interaction between *group* and *mode*, as well as *app* as a fixed effect.

A.3 Saving Efficiency

List of models we built to examine when participants' saving efficiency (in percent) using `lmer`:

```
base_model: Efficiency ~ 1 + (1 | Participant)
model_1: Efficiency ~ Mode + (1 | Participant)
model_2: Efficiency ~ Mode + Group + (1 | Participant)
model_3: Efficiency ~ Mode * Group + (1 | Participant)
model_4: Efficiency ~ Mode * Group + App + (1 | Participant)
model_5: Efficiency ~ Mode * Group + App + Rem_Battery + (1 | Participant)
```

ANOVA table for the models:

```
           npar    AIC    BIC  logLik -2*log(L)   Chisq Df Pr(>Chisq)
base_model    3 5816.4 5829.8 -2905.2    5810.4
model_1       4 5814.5 5832.4 -2903.3    5806.5  3.8363  1   0.050155 .
model_2       6 5816.9 5843.7 -2902.4    5804.9  1.6538  2   0.437411
model_3       8 5815.1 5850.9 -2899.6    5799.1  5.7788  2   0.055610 .
model_4      13 5808.3 5866.5 -2891.2    5782.3 16.7607  5   0.004977 **
model_5      14 5810.2 5872.8 -2891.1    5782.2  0.1805  1   0.670908
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Based on the results, model 4 is selected for the analysis of *saving efficiency*, as it has the lowest AIC and shows significant improvement over the previous model. Model 4 includes the interaction between *group* and *mode*, as well as *app* as a fixed effect.

A.4 Touch Frequency

List of models we built to examine when participants' touch frequency using `lmer`:

```
base_model: Touch_Freq ~ 1 + (1 | Participant)
model_1: Touch_Freq ~ Config + (1 | Participant)
model_2: Touch_Freq ~ Config + Group + (1 | Participant)
model_3: Touch_Freq ~ Config * Group + (1 | Participant)
model_4: Touch_Freq ~ Config * Group + App + (1 | Participant)
model_5: Touch_Freq ~ Config * Group + App + Rem_Battery + (1 | Participant)
```

Note that the predictor *config* here includes the default config in addition to the two saving mode configs. ANOVA table for the models:

```
            npar   AIC   BIC  logLik -2*log(L)    Chisq Df Pr(>Chisq)
base_model     3 11900 11915 -5946.8     11894
model_1        5 11887 11913 -5938.5     11877  16.7087  2  0.0002354 ***
model_2        7 11886 11922 -5935.9     11872   5.1460  2  0.0763057 .
model_3       11 11893 11950 -5935.6     11871   0.6611  4  0.9560388
model_4       16 11252 11335 -5610.1     11220 651.0263  5  < 2.2e-16 ***
model_5       17 11248 11335 -5606.8     11214   6.5886  1  0.0102634 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Based on the results, model 5 is selected for the analysis of *touch frequency*, as it has the lowest AIC and shows significant improvement over the previous model. Model 5 includes the interaction between *group* and *config*, as well as *app* and *remaining battery* as fixed effects.

B Power Profile Values of Test Device

Below is the power profile file obtained from the device firmware. We added comments to the file to provide corrections and clarifications for some of the values.

```xml
<?xml version="1.0" encoding="utf-8"?>
<device name="Android">
    <item name="none">0</item>
    <item name="ambient.on">24.2</item>
    <item name="screen.on">81.5</item>
    <item name="screen.full">252</item>
    <item name="audio">37</item>
    <item name="video">49</item>
    <item name="camera.avg">192.1</item>
    <item name="camera.flashlight">149</item>
    <item name="radio.scanning">102.1</item>
    <array name="radio.on">
        <value>7.3</value>
        <value>7.3</value>
    </array>
    <item name="modem.controller.sleep">0</item>
    <item name="modem.controller.idle">85</item>
    <item name="modem.controller.rx">98</item>
    <array name="modem.controller.tx">
        <value>128</value>
        <value>140</value>
        <value>197</value>
        <value>266</value>
        <value>345</value>
    </array>
    <item name="modem.controller.voltage">3700</item>
    <item name="wifi.controller.idle">2.1</item>
    <item name="wifi.controller.rx">125</item>
    <item name="wifi.controller.tx">575</item>
    <array name="wifi.controller.tx_levels">
        <value>0</value>
```

```xml
    </array>
    <item name="wifi.controller.voltage">4000</item>
    <array name="wifi.batchedscan">
        <value>.0002</value>
        <value>.002</value>
        <value>.02</value>
        <value>.2</value>
        <value>2</value>
    </array>
    <item name="gps.on">71.5</item>
    <item name="bluetooth.controller.idle">4.6</item>
    <item name="bluetooth.controller.rx">90</item>
    <item name="bluetooth.controller.tx">114</item>
    <item name="bluetooth.controller.voltage">4000</item>
    <item name="cpu.suspend">6</item>
    <item name="cpu.idle">17.5</item>
    <array name="cpu.clusters.cores">
        <value>6</value>    <!-- number of big cores, should be 4 -->
        <value>2</value>    <!-- number of little cores, should be 4 -->
    </array>
    <array name="cpu.core_speeds.cluster0">   <!-- frequencies -->
        <value>533000</value>
        <value>672000</value>
        <value>768000</value>
        <value>864000</value>
        <value>960000</value>
        <value>1056000</value>
        <value>1152000</value>
        <value>1248000</value>
        <value>1344000</value>
        <value>1440000</value>
        <value>1536000</value>
        <value>1632000</value>
        <value>1728000</value>
        <value>1824000</value>
        <value>1920000</value>
        <value>2002000</value>
    </array>
    <array name="cpu.core_power.cluster0">  <!-- current in mA -->
        <value>25</value>
        <value>27</value>
        <value>29</value>
        <value>30</value>
        <value>31</value>
        <value>33</value>
        <value>35</value>
        <value>38</value>
        <value>40</value>
        <value>43</value>
        <value>46</value>
        <value>50</value>
        <value>54</value>
        <value>59</value>
        <value>67</value>
        <value>76</value>
    </array>
    <array name="cpu.core_speeds.cluster1">
        <value>533000</value>
        <value>672000</value>
        <value>768000</value>
        <value>864000</value>
        <value>960000</value>
        <value>1056000</value>
        <value>1152000</value>
        <value>1248000</value>
        <value>1344000</value>
        <value>1440000</value>
        <value>1536000</value>
        <value>1632000</value>
        <value>1728000</value>
        <value>1824000</value>
        <value>1920000</value>
        <value>2016000</value>
        <value>2112000</value>
        <value>2208000</value>
        <value>2304000</value>
        <value>2400000</value>
    </array>
```

```
    <array name="cpu.core_power.cluster1">
        <value>52</value>
        <value>61</value>
        <value>70</value>
        <value>79</value>
        <value>87</value>
        <value>97</value>
        <value>106</value>
        <value>119</value>
        <value>131</value>
        <value>146</value>
        <value>163</value>
        <value>180</value>
        <value>199</value>
        <value>223</value>
        <value>252</value>
        <value>292</value>
        <value>331</value>
        <value>410</value>
        <value>500</value>
        <value>579</value>
    </array>
    <array name="gpu.speeds">
        <value>100000</value>   <!-- actually unsupported -->
        <value>221000</value>
        <value>351000</value>
        <value>455000</value>
        <value>552000</value>
        <value>650000</value>
        <value>754000</value>
        <value>845000</value>
        <value>949000</value>
    </array>
    <array name="gpu.active">
        <value>34</value>   <!-- actually unsupported -->
        <value>66</value>
        <value>99</value>
        <value>123</value>
        <value>160</value>
        <value>206</value>
        <value>259</value>
        <value>318</value>
        <value>377</value>
    </array>
    <item name="battery.capacity">4905</item>
    <item name="battery.typical.capacity">5000</item>
</device>
```