SERENUS: Alleviating Low-Battery Anxiety Through Real-time, Accurate, and User-Friendly Energy Consumption Prediction of Mobile Applications

Sera Lee*
School of Computing, KAIST
Daejeon, Republic of Korea

Dae R. Jeong*
School of Computing, KAIST
Daejeon, Republic of Korea

Junyoung Choi
School of Computing, KAIST
Daejeon, Republic of Korea

Jaeheon Kwak
School of Computing, KAIST
Daejeon, Republic of Korea

Seoyun Son
School of Computing, KAIST
Daejeon, Republic of Korea

Jean Y. Song
Department of Electrical Engineering
and Computer Science, DGIST
Daegu, Republic of Korea

Insik Shin
School of Computing, KAIST
Daejeon, Republic of Korea

ABSTRACT

Low-battery anxiety has emerged as a result of growing dependence on mobile devices, where the anxiety arises when the battery level runs low. While battery life can be extended through power-efficient hardware and software optimization techniques, low-battery anxiety will still remain a phenomenon as long as mobile devices rely on batteries. In this paper, we investigate how an accurate real-time energy consumption prediction at the application-level can improve the user experience in low-battery situations. We present SERENUS, a mobile system framework specifically tailored to predict the energy consumption of each mobile application and present the prediction in a user-friendly manner. We conducted user studies using SERENUS to verify that highly accurate energy consumption predictions can effectively alleviate low-battery anxiety by assisting users in planning their application usage based on the remaining battery life. We summarize requirements to mitigate users' anxiety, guiding the design of future mobile system frameworks.

CCS CONCEPTS

• **Human-centered computing → Interactive systems and tools**.

KEYWORDS

Mobile System, Low-battery Anxiety, Energy Consumption Prediction

*Both authors equally contributed to this paper.

UIST '24, October 13–16, 2024, Pittsburgh, PA, USA

ACM ISBN 979-8-4007-0628-8/24/10
https://doi.org/10.1145/3654777.3676437

ACM Reference Format:
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 *The 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24), October 13–16, 2024, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 20 pages. https://doi.org/10.1145/3654777.3676437

1 INTRODUCTION

While mobile devices are making our daily lives more convenient, they also bring inherent discomfort behind that convenience. As most mobile devices operate on batteries, *low-battery anxiety* [57, 58], the anxiety that people feel when the batteries of their devices are low, is a common displeasing experience to mobile users. In particular, a survey from the industry [33] reports that 90% of mobile users have experienced anxiety when the remaining energy of their batteries drops to 20% or lower. The consequences of low-battery anxiety may affect users' daily lives, distracting them from important events and reducing their productivity at work.

As low-battery anxiety has been recognized as a critical issue in the mobile environment, various attempts have been made to alleviate it. Several works have strived to reduce the likelihood of experiencing low-battery anxiety by optimizing the energy consumption of applications (*e.g.*, video streaming [41, 65], image sensing [36], and web browsers [8]) or hardware usage (*e.g.*, GPU [32, 64], GPS [10, 14], WiFi [25], and display [17]). In addition, there have been efforts to estimate or predict the energy consumption of applications [26, 29, 47, 61, 63] and utilize the estimation to schedule threads energy-efficiently [42, 62] or to identify battery-draining applications [37]. These studies have achieved meaningful results such as extending battery life through efficient resource management.

However, there is still a lack of understanding regarding how to alleviate low-battery anxiety *during the moment while mobile users are actively experiencing it.* When users are unsure whether their battery has enough power, even if their battery does, they have no

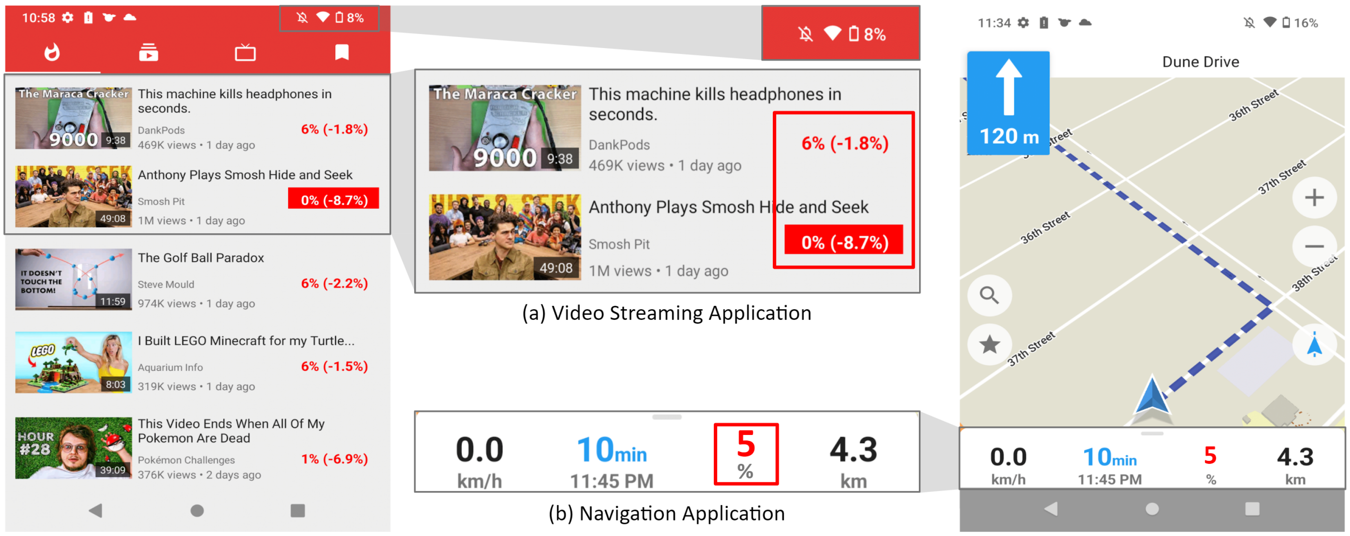


Figure 1: SERENUS provides estimates of how much battery will remain after executing specific use case scenarios by predicting energy consumption. In (a), the video streaming application displays the amount of battery consumed when watching each video and also indicates whether the current battery level is sufficient to watch a given video (as shown by the red-boxed alert in the second video). This information helps users plan their battery usage in low-battery situations, such as deciding not to watch the second video. In (b), the navigation application shows how much battery will be left when the user arrives at their destination. If it indicates that 5% of the battery will remain upon arrival, the user can be assured that their device will stay powered until arrival.

choice but to refrain from using their mobile devices to conserve battery life. This results in a deteriorated user experience. In this work, we aim to aid mobile users to relieve low-battery anxiety in situations where their battery is low.

The key insight to this end is that *eliminating uncertainty about future events is crucial in alleviating anxiety*. In psychology, uncertainty is commonly recognized as the *root cause* of anxiety [11, 20, 21, 23]. In the context of low-battery anxiety, users become anxious because of the uncertainty about how long the battery will last and/or whether they have enough battery until their next charge. We envision that if mobile systems assist users in accurately determining how much energy their future application use will consume, they will feel less anxiety and can plan their application use based on solid information.

We design and implement SERENUS [1], a mobile system framework that predicts the *accurate* energy consumption of applications in *real-time*. Using SERENUS, application developers can easily implement *user-friendly* interfaces displaying energy consumption predictions in their applications, as represented in Figure 1. With minimal modifications to an application, SERENUS automatically collects training data that reflect the characteristics of the device (*e.g.*, screen size). Using this data, SERENUS constructs a device-specific model to predict the application's energy consumption on the device. This allows users to obtain clear information about how much energy is being consumed depending on their actions. As a result, users will experience less low-battery anxiety because they can be confident about whether they have enough battery or can plan future battery usage based on the prediction to preserve enough battery.

Using SERENUS, we conducted two independent controlled experiments and a field study to evaluate its effects in low-battery situations. The first controlled experiment (n=40) was to quantitatively and qualitatively confirm that mobile users feel less low-battery anxiety with accurate energy consumption prediction. The second experiment (n=88) was to observe the relationship between the level of accuracy of energy consumption prediction and users' behavior. During a 7-day field study (n=7), we validated that SERENUS is truly helpful to users in real-world low-battery situations. Throughout these experiments, we verify that SERENUS's energy consumption prediction can not only effectively alleviate low-battery anxiety, but also lead to users' behavioral changes, such as using mobile devices longer instead of stopping using it. We also evaluate various performance characteristics of SERENUS including the prediction accuracy and latency as well as the usability of our system. As a result, we show that SERENUS is sufficiently practical for real-world applications. Finally, we summarize our findings from our experiments, including diverse forms of uncertainty causing low-battery anxiety, and pros and cons of different user interface types. We provide the summary in Table 1.

The main contributions of this paper are as follows:

- We design and implement SERENUS, a mobile system framework to predict the energy consumption of mobile applications. Using SERENUS, developers can easily augment their applications with user-friendly interfaces that display real-time and accurate energy consumption predictions.

[1] Serenus is Latin for serenity.

Benefits of energy consumption prediction (§5)	
Alleviating low battery anxiety	The uncertainty regarding energy consumption is reduced.
Allowing optimization of battery usage	Users gain confidence in the energy consumption of applications.
Types of uncertainty (§7.1.1)	**How predictions reduce the uncertainty (§7.1.2, §7.2)**
Uncertainty about being able to use desired applications with the current battery level	Users adjust settings (*e.g.*, brightness) based on energy consumption predictions to ensure sufficient battery life.
Uncertain battery lifetime	Users feel confident about future battery levels.
Unexpected future event (*e.g.*, sudden call)	
UI types to display predictions (§5.6, see §3) & Suitable usage (§7.3)	
Remaining battery level (Type A)	Proper to applications with fixed-length contents. Easy to compare contents in an application (*e.g.*, different YouTube videos).
Battery lifetime (Type B)	Proper to applications with variable-length contents. Easy to plan overall usage of the application (*e.g.*, Twitch).
Energy consumption per min. (Type C)	Proper to applications where individuals consume contents at different rates. (*e.g.*, Ebook Reader).

Table 1: Summary of findings regarding energy consumption prediction and low-battery anxiety.

- Using SERENUS, we conducted two rounds of user studies (n=40 and 88) in controlled environments and a 7-day field study (n=7) in a real-world environment. As a result, we show that real-time, accurate, and user-friendly energy consumption prediction can not only assist users in relieving low-battery anxiety, but also change users' behavioral patterns.
- We provide various observations from our experiments, which provide a deep understanding and guidelines for alleviating users' low-battery anxiety.

In the rest of this paper, we first provide the background and motivation of this work (§2). Then, we present SERENUS, a mobile system framework that predicts the energy consumption of mobile applications (§3). Using SERENUS, we conducted two rounds of user studies (§4), where results are described in §5. We also present various performance characteristics of SERENUS (§6). Lastly, we discuss our detailed observations from our studies (§7).

2 BACKGROUND AND MOTIVATION

In this section, we explain what low-battery anxiety is (§2.1), followed by previous approaches to mitigate low-battery anxiety (§2.2). Then, we clarify the motivation of this paper (§2.3).

2.1 Low-Battery Anxiety

As mobile devices have become pervasive in our daily lives, the anxiety known as *low-battery anxiety*, which mobile users experience when their battery level is low, has become a serious concern [24, 30, 33, 48, 58]. Low-battery anxiety was first reported in a survey conducted by LG in 2016, which found that 90% of mobile users in the United States experienced panic when the battery level of their smartphones dropped to 20% or lower [33]. This survey also reveals that low-battery anxiety not only imposes psychological pressure on mobile users, but it also leads them to exhibit *irrational behaviors.* For instance, mobile users may drop everything and rush home immediately, ask a stranger to charge their smartphones, or even "secretly borrow" someone else's charger.

2.2 Related Work

Due to its negative impacts on the user experience in mobile environments, alleviating low-battery anxiety has become an important issue in designing mobile systems.

2.2.1 Battery Usage Optimization. Intuitively, optimizing the energy consumption of mobile devices can be effective in addressing low-battery anxiety as it extends the battery life and reduces the likelihood of experiencing low-battery anxiety.

Researchers have proposed various techniques to reduce the energy consumption of a wide range of software (*e.g.*, video streaming [41], downloading files [22], web interactions [34, 49], and image sensing [36]) as well as hardware (*e.g.*, CPU [5], GPU [32, 64], GPS [10, 14], Wi-Fi/4G [8], and display [17, 35]). In addition, researchers have proposed strategies to extend battery life, particularly when devices enter low-battery states. For example, LPVS [57] allows users to watch streaming video for a longer duration when the battery is low. BPM [31] addresses unexpected shutoffs during low-battery situations, and MixMax [30] and SDB [3] adopt heterogeneous batteries to prolong battery life.

Although these studies have achieved meaningful results in extending battery life, they cannot completely prevent low-battery situations (as well as low-battery anxiety) as long as mobile devices operate on batteries. Consequently, mobile users are still suffering from low-battery anxiety [58].

2.2.2 Energy Consumption Estimation and Prediction. Besides optimizing battery usage, several works aim to estimate [13, 26, 29, 44, 47, 50, 60, 61, 63] or predict [17, 43] the energy consumption of applications to better utilize the available battery.

A common method in these approaches is to construct a *power model.* A power model is a mathematical model or equation that takes various system states (*e.g.*, CPU utilization) and/or hardware states (*e.g.*, the brightness level of the display) to estimate the energy consumption of the system or applications. To this end, V-Edge [61] leverages voltage dynamics via the battery interface of mobile devices, Chen *et al.* [13] and Carroll *et al.* [12] rely on CPU utilization and external power meters, and Pathak *et al.* [47] infers hardware usage by tracing I/O operations via system calls. In some recent mobile system framework-specific studies [26, 29, 63], hardware usage and state are more precisely determined through monitoring Android's binder inter-process communication (IPC).

Such power models have the potential to allow users or systems to better manage the remaining battery. For instance, estimating the energy consumption of previously executed applications allow mobile users to identify battery-draining applications [37] or mobile systems to schedule threads energy-efficiently [42, 56, 62]. Furthermore, power models can be used to predict future energy consumption. PowerForecaster [43] predicts how much more energy a mobile device will consume after installing a given application, and Battery+ [17] predicts the power saving of a dark mode of applications.

Power model. Power models are mathematical equations that capture dependencies between resource usage and energy consumption. In this paper, we adopt power models devised in previous works [26, 29, 63] and used in the Android Open Source Platform.

As an example, power models for the audio and screen hardware, $P_{display}$ and P_{audio}, that output estimated energy consumption of these devices can be represented as

$$P_{display}(B,t) = \begin{cases} \beta^{B}_{display} \times t & \text{if the display is turned on,} \\ & (0 \leq B \leq Max\ brightness\ level) \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

$$P_{audio}(V,t) = \begin{cases} \beta^{V}_{audio} \times t & \text{if the audio is playing sound,} \\ & (0 \leq V \leq Max\ volume\ intensity) \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

where t is the time duration that the hardware is used (*i.e.*, hardware usage), $\beta^{B}_{display}$ represents the power coefficient of the display hardware when the display has a brightness level B, and β^{V}_{audio} is the power coefficient of the audio hardware when its volume intensity is V. Power coefficients capture the energy consumption rate depending on the resources' usage, and constructing a power model means determining values of power coefficients.

2.3 Motivation

Although low-battery anxiety has been receiving considerable attention since it was first reported, we have identified two critical issues. First, low-battery anxiety is an inherent (and inevitable) concern when using battery-powered devices. Second, there is still a lack of understanding regarding how to alleviate low-battery anxiety when users are actually experiencing it. Unlike other studies aimed at avoiding anxiety due to low-battery situations, this work is motivated to alleviate the extent of low-battery anxiety that mobile users experience when their batteries are running out of charge.

To achieve this goal, we draw on findings from psychology, which states that anxiety stems from uncertainty [11, 20, 21]. More specifically, low-battery anxiety can be attributed to the uncertainty surrounding a mobile device's remaining battery when running applications. When mobile users realize that their batteries are running out of charge, they become anxious as they are unsure about how much energy applications will consume and whether their batteries will last until the next charge. This uncertainty provokes users to refrain from using their mobile devices to preserve their battery life.

To reduce low-battery anxiety by eliminating such uncertainty, we envision a mobile system that accurately predicts the energy consumption of applications by using a power model. With this system, application developers can easily integrate user interfaces that display the predicted energy consumption into their applications. Consequently, users will feel less low-battery anxiety as they will have a clearer idea of how long they can use the remaining battery. Moreover, we also expect that users will be able to plan their application usage based on the prediction as well as the remaining battery level. This will enable them to keep their devices alive while using applications.

3 SERENUS: MOBILE SYSTEM FRAMEWORK TO ACCURATELY PREDICT ENERGY CONSUMPTION OF APPLICATIONS

In this section, we first provide the overall design of Serenus (§3.1). And then, we explain two phases of Serenus, each of which is to construct the power model (§3.2) and to predict the energy consumption (§3.3).

3.1 Design Overview of Serenus

Here we elicit the design requirements of Serenus and the overall architecture of Serenus to satisfy them.

3.1.1 Design Requirements. While the primary goal of Serenus is to alleviate low-battery anxiety, we design Serenus to be practical for real-world application development by satisfying the following requirements.

- **R1: Accurate real-time prediction**: The foremost factor in making Serenus beneficial for alleviating low-battery anxiety is to provide accurate prediction of energy consumption in real-time, which makes users confident in how much energy applications will consume.
- **R2: User-friendly interface**: When users are anxious due to the low battery level, they will be less inclined to manipulate their phones to explore a new interface. Serenus aims to provide energy consumption prediction with interfaces that are intuitive and require less cognitive load.
- **R3: Ease of integration**: As Serenus introduces new features to mobile applications, application developers need to make modification to their applications. If the effort required to adopt Serenus is significant, developers might not use Serenus at all, regardless of the benefits it offers. Therefore, ensuring the easy integration of Serenus is also crucial.

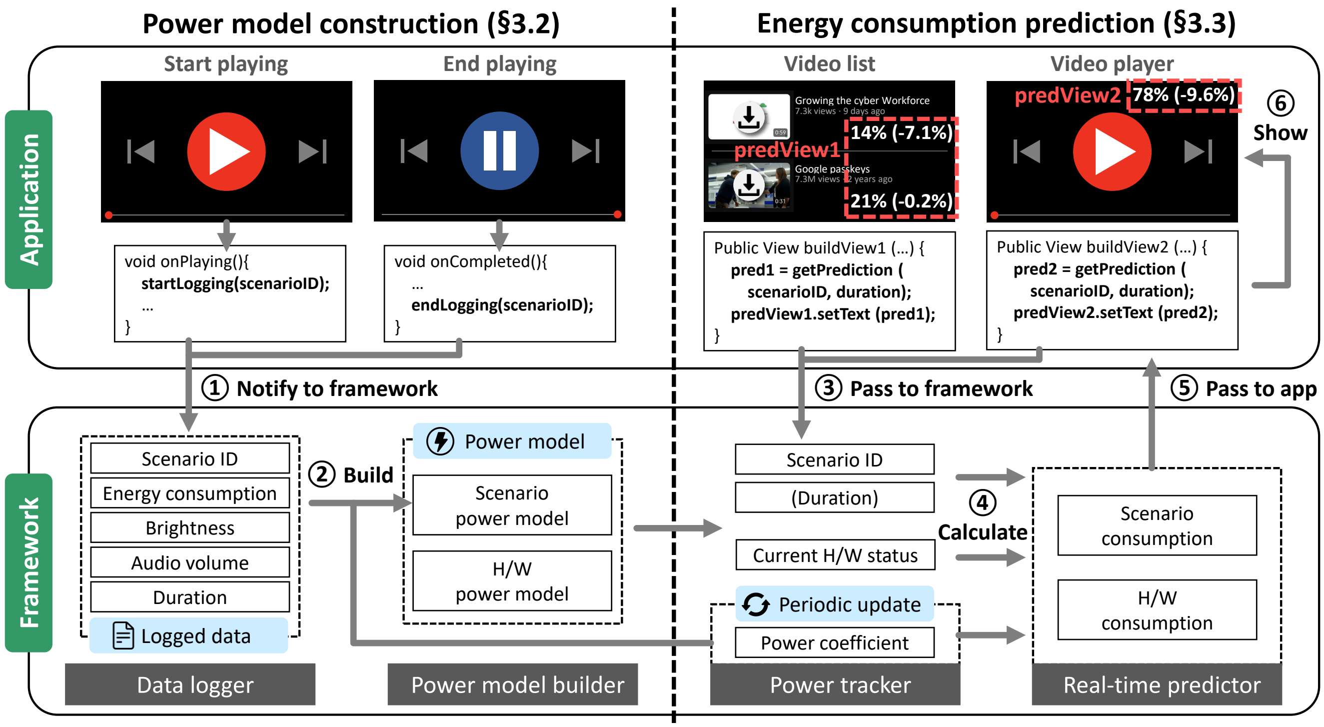


Figure 2: Overall architecture of SERENUS. SERENUS constructs power models on-device through two phases: the power model construction phase (§3.2) and the energy consumption prediction phase (§3.3). In the first phase, SERENUS collects various data while the application is running (①), and constructs power models (②). In the second phase, when a user application requests the amount of energy consumption (③), SERENUS calculates the energy consumption prediction (④) and returns it to the user program (⑤). Lastly, the user application displays the predicted energy consumption in real time (⑥).

Application	Use case scenario	Power consumption (mAh/min)
Twitch [59]	Broadcasting a live stream	25.7
	Watching a live stream	6.7
Ulike [9]	Taking a filtered video	25.3
	Watching a saved video	6.0

Table 2: Various use case scenarios of real-world applications and their per-minute power consumptions. As shown in this table, there can be significant differences in energy consumption between different use case scenarios even for a single application.

3.1.2 Overall Architecture. Figure 2 shows the overall architecture of SERENUS. As shown in this figure, SERENUS consists of two phases. The first phase is to construct a power model, which will be used to predict energy consumption. In this phase, SERENUS collects various execution information (*e.g.*, volume intensity of audio and brightness of the display), and constructs a power model that predicts the energy consumption based on execution information using a light-weight machine learning algorithm (*i.e.*, linear regression). The second phase is to predict the energy consumption of applications using the power model. In this phase, SERENUS collects execution information as done in the first phase, and then returns the predicted energy consumption to the application. Lastly, the application displays the predicted value.

Use case scenario-based energy prediction (for R1): As shown in Table 2, real-world applications may include multiple use case scenarios, where each shows different energy consumption characteristics. For instance, Twitch, a live streaming application, has (at least) two use case scenarios: broadcasting a live stream and watching a live stream. It is clear that these two use case scenarios consume different amounts of energy as broadcasting a live stream uses a camera while watching a live stream does not. SERENUS is designed to differentiate multiple use case scenarios, thereby enabling precise predictions of energy consumption for each scenario.

In-application display (for R2): When conveying the energy consumption prediction to users, we display the predicted energy consumption inside the existing applications' layouts. This approach conveys the predicted energy consumption without additional manipulation, allowing users to obtain information with a low cognitive load.

On-device power model construction (for R3): SERENUS is designed for easy integration into applications, allowing developers to avoid extensive data collection for different devices or significant engineering efforts. With minimal modifications to an application, SERENUS automatically collects training data reflecting the characteristics of the device that impact energy consumption (*e.g.*, screen size) for a short period after the application's installation. Using the collected data, SERENUS builds device-specific power models. Although this approach requires initial data collection, our evaluation shows that SERENUS can accurately predict energy consumption with a small amount of data (§6.1.3).

3.2 Power Model Construction

To construct a power model, SERENUS needs to collect data as training samples. Thus, developers need to make their applications notify the Data logger of the use case scenario's execution. During the execution of a use case scenario, the Data logger records and stores training samples, which will be used to construct prediction models (*e.g.*, the Scenario power model in Figure 2).

3.2.1 Collecting Training Samples. The first step in SERENUS involves data collection to create power models (represented as ① in Figure 2). To differentiate between various use case scenarios within a single application (*e.g.*, watching a video clip or live streaming on YouTube), SERENUS collects distinct sets of training data for each scenario. To achieve this, SERENUS relies on `Scenario ID`, a unique identifier for each use case scenario. Developers select `Scenario IDs` arbitrarily for different use case scenarios. Subsequently, all of SERENUS's APIs accept the `Scenario ID` as an argument. Upon receiving API calls, SERENUS distinguishes various scenarios using `Scenario ID` as well as an additional application ID, which is uniquely assigned by a mobile system (*e.g.*, `uid` in Android) and can be read later (*e.g.*, using `getCallingUid()` in Android).

To collect training data at the start and the end of each use case scenario, developers are required to insert two SERENUS API calls for each use case scenario, one at the start and another at the end of the scenario. In the case of the "watching a video clip" scenario of YouTube (*i.e.*, Figure 2), developers need to tweak the `onPlaying()` and `onCompleted()` callbacks, which are triggered when a user starts and finishes watching a video clip. Within these callbacks, developers need to insert `startLogging()` and `endLogging()` respectively, both with the same associated `Scenario ID`.

When the inserted functions are executed, the Data logger collects training samples and stores them with a label of (Scenario ID, application ID). Each training sample is a 5-tuples of (*duration*, *brightness*, *volume*, *battery*, β^*) where *duration* describes the time duration between the invocation of *startLogging()* and *endLogging()*, *brightness* and *volume* respectively represents the brightness level of the display and the volume intensity of the audio, *battery* indicates the amount of consumed energy in the battery during the duration, and β^* represents the aggregated power coefficient of various hardware resources.

Tracking the aggregated power coefficient β^*. Instead of holding power coefficients for all hardware resources, SERENUS aggregates power coefficients for all hardware resources (*e.g.*, GPS and CPU) except display and audio devices. This is because the display and audio hardware can be manipulated regardless of the use case scenario logic (*e.g.*, one can turn off the display or mute the audio while YouTube is playing a video or a music). SERENUS, therefore, keeps three power coefficients, $\beta_{display}$, β_{audio} (adapted from [26, 63]) and β^*. SERENUS can individually predict the energy consumption of the display, audio, and remaining hardware resources using these three power coefficients. Adding up these three predictions results in energy consumption prediction of the device *before* running the scenario.

To track β^*, Power tracker monitors the device's battery capacity, and when the capacity changes, Power tracker calculates and updates β^* as follows:

$$\beta^* = (C_{t2} - C_{t1} - P_{display}(B, d) - P_{audio}(V, d))/d \qquad (3)$$

where $t1$ and $t2$ are the time when β^* was lastly updated and when β^* is being updated, and d is $(t2 - t1)$. C_t is the battery capacity at time t, $P_{display}(B, d)$ and $P_{audio}(V, d)$ are power models of the display and audio hardware (see §2.2) where B and V represent the current brightness level and the current volume intensity.

3.2.2 On-Device Power Model Construction. When training samples are sufficiently collected (*i.e.*, five by default), Power model builder constructs a power model that calculates the *additional* energy consumption incurred by running a use case scenario (see ② in Figure 2).

Constructing scenario power model. Power model builder finds out a value of $\beta_{scenario}$ that also aggregates additional hardware usage of running the scenario except the display and the audio devices. Then, the additional energy consumption of running the scenario excluding the display and the audio devices, $P_{scenario}$, during a specific time duration t can be predicted as follows:

$$P_{scenario}(t) = \beta_{scenario} \times t \qquad (4)$$

SERENUS determines the value of $\beta_{scenario}$ through the following two steps.

Step 1: Calculating the additional energy consumption of running the use case scenario from training samples: For each training sample labeled with (Scenario ID, application ID), Power model builder calculates the additional energy consumption of the scenario, $C_{scenario}$ consumed when collecting the training sample (*duration*, *brightness*, *volume*, *battery*, β^*).

$$C_{scenario} = C - P_{display}(B, d) - P_{audio}(V, d) - \beta^* \times d \qquad (5)$$

where C, B, V, and d are *battery*, *brigtness*, *volume*, and *duration* in the training sample respectively.

Step 2: Constructing the power model $P_{scenario}$: After Step 1, Power model builder produces a set of two pairs (*duration*, $C_{scenario}$). Then, as Equation 4 is a linear equation, Power model builder uses linear regression. Specifically, Power model builder considers $C_{scenario}$ as an output of Equation 4 and *duration* as an input t. Using the Ordinary Least Squares (OLS) [19] method, it finds out a value of $\beta_{scenario}$ Lastly, Power model builder stores $\beta_{scenario}$ with a label of (Scenario ID, application ID).

3.3 Energy Consumption Prediction

Once a power model is constructed, the mobile application can display the energy consumption of a use case scenario by requesting the predicted value from SERENUS.

3.3.1 Calculating Predicted Energy Consumption. To predict energy consumption, the application starts the prediction process by calling a SERENUS API function, `getPrediction()`, with two parameters: `Scenario ID` and `duration` (as shown as ③ in Figure 2). `Scenario ID` is used to identify the use case scenario for which energy consumption needs to be predicted. `duration` represents the amount of time during which this use case scenario will be executed. For instance, in the case of YouTube, `duration` could correspond to the (remaining) time duration of a video clip being watched. If developers want to predict power consumption per unit of time, they can specify a time unit (*e.g.*, 1 minute). In this case, SERENUS will periodically estimate energy consumption for the unit of time.

Upon receiving a request, SERENUS looks up a power coefficient that matches the label (Scenario ID, application ID), and if it exists, retrieves $\beta_{scenario}$, a power coefficient of this scenario (*i.e.*, ④). Then, SERENUS takes the current brightness B, current volume intensity V, and β^*. With these parameters, SERENUS predicts the battery prediction, $Pred_{scenario}$, that the use case scenario will consume during the duration d with the following equation.

$$Pred_{scenario}(d) = P_{scenario}(d) + P_{display}(B,d) + P_{audio}(V,d) + \beta^* \times d \tag{6}$$

After predicting the energy consumption, SERENUS lastly returns the predicted value, $Pred_{scneario}$ to the application (⑤ in Figure 2).

3.3.2 Displaying Predicted Energy Consumption. To display the predicted energy consumption, we need to modify the user interface (*i.e.*, ⑥ in Figure 2). This modification mainly involves generating a new visual component (*i.e.*, `View` in Android) that continuously displays the predicted energy consumption in real time. In Figure 2, developers create two visual components, `predView1` and `predView2`, each of which is used to display the energy consumption prediction when listing video clips and when playing a video clip. During runtime, `predView1` and `predView2` display energy consumption predictions by periodically calling the `setText()` function with the values of `pred1` and `pred2`, and energy consumption predictions returned by `getPrediction()`. As a result, the application can display the energy consumption prediction when displaying a list of video clips and playing a video clip.

4 STUDY DESIGN

To understand whether predicting the energy consumption of applications can relieve low-battery anxiety and how it does so, we outline three hypotheses as follows.

H1: If users can perceive the accurate prediction of applications' energy consumption while using them, they will feel less low-battery anxiety.

H2: With energy consumption prediction, users will effectively plan their application usage, allowing them to use their devices longer without needing to refrain from using them.

H3: When the accuracy of energy consumption predictions is higher, users will be more inclined to consciously plan their application usage.

To verify the hypotheses, we conducted two independent studies using SERENUS with realistic scenarios in controlled laboratory environments. In the first study (§4.1), we simulated the situation of using a smartphone when on the move outside and conducted an in-lab controlled study to verify H1 and H2. In the second study (§4.2), we further aimed to understand the relationship between the level of accuracy of energy consumption prediction and users' behavior (H3). Therefore, we conducted a large-scale survey study where a scenario of being on a train with a limited smartphone battery was given. Lastly, a field study (§4.3) was conducted to validate the usefulness of SERENUS in real-world low battery situations.

4.1 Study 1: Simulation of Using a Smartphone While on the Move Outside

Study 1 examines how much energy consumption prediction helps alleviate a user's low-battery anxiety when a user is in a situation where their battery is low (*i.e.*, verifying **H1** and **H2**). To this end, this study assumes that a user is trying to reach a destination using a phone with low battery power, and we observe differences in the user experience depending on the existence of energy consumption prediction.

4.1.1 Participants. We recruited a total of 40 participants through local community platforms. These individuals are regular smartphone users who have experienced low-battery anxiety. To incentivize their participation, each participant was offered a compensation of approximately 15 USD.

4.1.2 Experimental Conditions. Conditions are divided into the system condition and the baseline condition depending on whether they use SERENUS or not. Battery consumption prediction is provided in the system condition. Participants can check the predicted remaining battery while using navigation, video, radio, and Rich Site Summary (RSS) applications. In contrast, participants in the baseline condition used the same applications, without any predictions about battery consumption.

4.1.3 Tasks and Procedures. Each participant experienced the simulation of two sequential scenarios under the given condition. First, they acted as if they went out to eat with a professor from a school. Then, they returned to the school. We provided participants with phones that had 16% battery power at the start of the experiment.

In the first scenario, participants moved to the destination using the navigation application for 15 minutes. They simulated movement through the console without actually moving. To be realistic, participants chose the restaurant themselves and navigated there. In the second scenario, participants came back from the restaurant after eating. They used applications freely among video, radio, and RSS applications while excluding the navigation application since they were assumed to know the way back. The second scenario also took 15 minutes. To simulate a more realistic situation, information about how far they had moved was provided instead of the remaining time. By instructing that participants should use their phones to finish the experiment at the school, participants did not let their phones shut off. Then, they had to scan the barcode we provided using a QR code application at the end of the experiment.

A five-minute tutorial was provided to get participants used to moving with the console and Serenus's UI. During the experiment, participants rated their immediate anxiety each time the battery of their phones reached 15%, 10%, and 5%. Finally, they responded to a post-survey after the experiment asking about their experience and thoughts on using the phones.

4.1.4 Measures and Analysis. We measured how Serenus affected the degree of low-battery anxiety and participants' behaviors. In both the system condition and the baseline condition groups, questions about their anxiety and behaviors were asked equally.

To analyze low-battery anxiety, we measured anxiety during the experiment when the battery level reached 15%, 10%, and 5%. The six-item short-form of the state scale of the Spielberger State-Trait Anxiety Inventory (STAI-6) [40] was adopted due to its brevity.

We collected log data on participants' usage of each application in the second scenario of getting back from the restaurant, as well as the extent to which participants consumed their battery. Then, we analyzed participants' behaviors through the post-survey (7-point Likert Scale and narrative questions) and user behavior logs. Each participant answered the following:

(1) *Why did you feel anxiety during the experiment?*
(2) *How much planning went into using the streaming application regarding battery usage and why?*
(3) *How much planning went into using the radio application regarding battery usage and why?*
(4) *How much planning went into using the RSS application regarding battery usage and why?*

Additionally, to gain insights into the suitable user interface for displaying energy consumption predictions in various applications, we conducted an additional survey within the baseline condition group. For each of the four specific types of applications (streaming, navigation, radio, and RSS), we suggested three user interface candidates: Type A, B, and C, as illustrated in Figure 3. Participants were then asked to rank the three suggested UI types and provide reasons for their rankings. To analyze the result, we computed average ranking assigning a weighted score of 3 points to the most preferred case, 2 points to the second most preferred case, and 1 point to the least preferred case in their responses. Furthermore, to evaluate the usability of Serenus within applications that have UI Type A, participants in the system condition group answered the System Usability Scale [6].

4.2 Study 2: Large-Scale Simulation of a Scenario of Being on a Train

Study 2 further investigates the relationship between the accuracy of energy consumption prediction and user experience through a large-scale survey. In this study, survey participants were assumed to be on a train with a phone with low battery power. With energy consumption predictions provided, we asked how user experience changes depending on their accuracy.

4.2.1 Participants. We collected a total of 88 responses from participants through community platforms. Participants were informed that it was a survey experiment simulating a smartphone-using situation. Since it was conducted as an online survey, participants answered their own survey questions individually.

4.2.2 Experimental Conditions. Each participant went through a single condition among seven, one baseline condition and six system conditions, where the conditions were randomly assigned. As in Study 1, participants in the baseline condition were not provided with the energy consumption prediction. Unlike Study 1, however, Study 2 had six system conditions according to the error rate of energy consumption prediction. These six system conditions are labeled as S-X%, where X denotes the error rate, such as S-0%, S-10%, S-20%, S-30%, S-40%, and S-50%. Participants in these conditions were provided with the predicted energy consumption with corresponding errors during the simulation.

4.2.3 Tasks and Procedures. The simulation was conducted through a survey experiment. Participants acted as if they were on a train to meet friends and chose whether to watch video clips through YouTube or not until they arrived. The scenario assumed that it would take 150 minutes for the participants to reach their destination by train, and they wanted to watch six preferred videos sequentially. When participants decided whether to watch a video or not, the remaining time decreased based on the duration of the video. The battery level started at 48% and participants were instructed not to let their phones die so that they can contact their friends after getting off the train.

Participants decided whether to watch each of the six videos based on the given information: current battery level, remaining time to arrival, and video duration. The predicted battery level after watching a video was only provided in system conditions, with the error rate varying among conditions. Each time participants made a decision on whether to watch a video, and we asked them to provide reasons for their decisions, encouraging them to make deliberate choices from a set of sample options (*e.g., "I thought the battery was enough to watch a video"*, or *"I trusted the prediction of the system"*).

The survey began with a tutorial prior to the experiment. This tutorial helped participants understand how and what information was provided by the system. By going through the system several times during the tutorial, they could notice the inaccuracy of the prediction of battery consumption. After the experiment, participants were asked to rate their experience using the system.

4.2.4 Measures and Analysis. We measured how participants behaved depending on the accuracy of the system through the survey questions and behavior logs. The following question was asked to measure how they felt during the experiment in every condition:

(1) *How much planning went into your application usage regarding battery usage*?

Additionally, the following questions were asked only in system conditions to measure the usability of the system:

(2) *How helpful was the system?*
(3) *Are you willing to use the system?*

To analyze the actual user behavior, we also recorded how many videos participants chose to watch during the experiment.

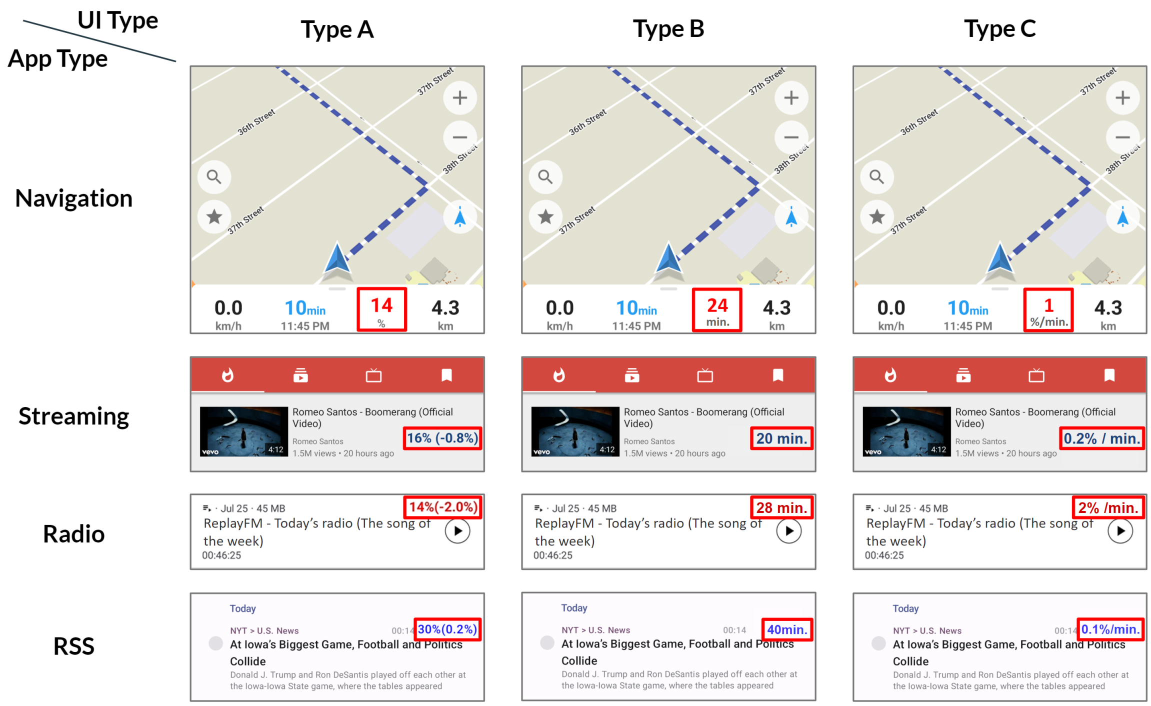


Figure 3: Possible UI designs to display energy consumption predictions. In Type A, Serenus predicts and displays the remaining battery level after the application runs. Type B displays the predicted time one can use the application before the battery runs out. In Type C, the amount of battery that the application will consume per unit of time (*e.g.*, minute) is displayed.

4.3 Field Study: Validating the Usefulness of Serenus in Real-World Low-Battery Situations

As Study 1 (§4.1) and Study 2 (§4.2) are conducted in controlled laboratory environments, we lastly verify the usefulness of Serenus in real-world situations by deploying it to mobile users and surveying user experiences of using Serenus.

4.3.1 Participants. We recruited 7 participants through local community platforms, where all participants have experienced low-battery anxiety before and were using Android-based mobile devices. To incentivize their participation, each participant was offered a compensation of approximately 70 USD (*i.e.*, 10 USD per day).

4.3.2 Task and Procedures. As Serenus constructs power models inside a mobile platform (refer to Figure 2), these participants were required to use mobile phones equipped with the Serenus platform. Thus, we provided Pixel 6 phones equipped with Serenus to the participants for use instead of their own phones during the study. Each participant had three days to get used to their provided phone and then started the experiment with the phone for seven days.

Each participant used a phone equipped with Serenus in their daily lives to experience whether the energy consumption prediction of Serenus helps alleviate low-battery anxiety. Participants were asked to use and charge their phones as usual while Serenus provided energy consumption predictions for four types of applications, video streaming, radio, RSS, and naviation applications. At the end of each day, they answered online survey questions regarding their user experiences that day. After finishing the study, they answered survery questions about the overall user experience of using Serenus.

4.3.3 Measures and Analysis. In this study, we did not use STAI-6 for two main reasons: 1) STAI-6 is suitable for measuring anxiety levels at the moment when participants are experiencing anxiety, and 2) it was not feasible to ask participants to answer survey questions for STAI-6 whenever they felt anxious in their daily lives. Consequently, instead of comparing anxiety levels of two groups based on STAI-6, we chose to measure the improvement of user experience of each participant with Serenus compared to their previous experiences.

In the daily survey, we first asked whether participants experienced low battery situations and low-battery anxiety:

(1) *Did your battery level drop below 20% today?*
(2) *When it happened, did your feel anxious because of the low battery level?*

Then, the following questions (7-point Likert scale) were asked each day to measure whether Serenus was helpful in alleviating low-battery anxiety that day:

(3) *When the battery level is lower than 20%, compared to the existing Android system, was energy consumption prediction helpful or hindering in alleviating low-battery anxiety?*
(4) *When the battery level is lower than 20%, compared to the existing Android system, was energy consumption prediction helpful or hindering in planning application usage?*

(5) *When the battery level is enough (i.e., more than 20%), compared to the existing Android system, was energy consumption prediction helpful or hindering in alleviating low-battery anxiety?*
(6) *When the battery level is enough (i.e., more than 20%), compared to the existing Android system, was energy consumption prediction helpful or hindering in planning application usage?*

Possible answers were: strongly hindering, hindering, somewhat hindering, neither hindering nor helpful (*i.e.*, equal to the existing Android system), somewhat helpful, helpful, and strongly helpful.

At the end of the study, one 7-point Likert scale question and one descriptive question were asked to measure the overall improvement of the user experience and analyze negative impacts of Serenus:

(1) *Overall, compared to the existing Android system, how helpful or hindering was the system that provides energy consumption predictions?*
(2) *Have you had an experience where energy consumption prediction was caused discomfort or inconvenience when using the mobile device?*

Possible answers for the first question were the same as the above questions. We also required users to provide descriptive reasons for their answers.

5 USER STUDY RESULTS

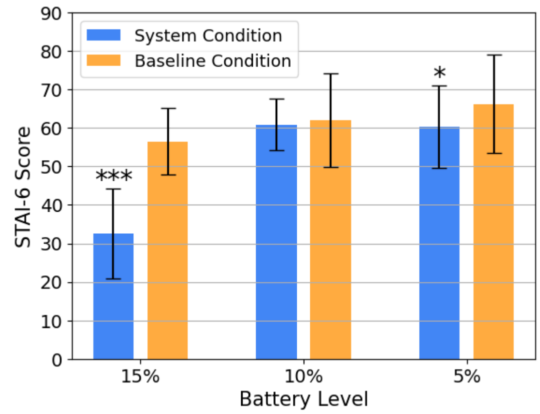
This section provides the details of how our user studies verify the three hypotheses described in §4. Specifically, we first explain how we filter out noisy data in Studies 1 and 2 (§5.1). Then we explain how Study 1 verifies **H1** (§5.2) and **H2** (§5.3), followed by how Study 2 verifies **H3** (§5.4). Additionally, §5.5 demonstrates that users feel Serenus is helpful in real-world low-battery situations, while §5.6 and §5.7 provide additional findings on preferences of different UI types and the usability of Serenus respectively.

5.1 Filtering

Before conducting any analysis on the collected user study results, we reviewed responses in Study 1 and Study 2 and filtered out noisy data based on consistent criteria, such as doing unsolicited behaviors or not performing the given tasks properly.

In Study 1, we excluded five participants, resulting in 18 participants in the system condition and 17 participants in the baseline condition for analysis. Among the participants in the system condition, two used features that were not provided (*e.g.*, flashlight), which could alter the usage pattern compared to other participants. Additionally, three participants in the baseline condition used applications that we did not provide, corrupting the experiment.

In Study 2, 17 participants were excluded out of 88 total participants, resulting in 13 participants for the baseline condition, 13 participants for the S-0% condition, 10 participants for the S-10% condition, 13 participants for the S-20% condition, 14 participants for the S-30% condition, 12 participants for the S-40% condition, and 11 participants for the S-50% condition. Among 17 excluded participants, five participants submitted the responses irrelevantly to the questions such as choosing personal preferences on the video. Additionally, 12 participants had inconsistencies in their responses. For example, they indicated trust in the system's predictions in one answer, but in another response to the question regarding the prediction error rate, they answered it was higher than 80%.



(a) STAI-6 scores when the battery reached each battery level (the lower is better).

	The frequency of the keyword *"uncertainty"* and *"unpredictable"*
SC (N=18)	7
BC (N=17)	15

(b) The frequency of the keywords mentioned as a reason for the anxiety.

Figure 4: Quantitative results from the survey in Study 1 (N=18 for SC, N=17 for BC). Participants felt less anxiety in the system condition. Especially, when the battery reached 15% and 5%, there were significant decreases in anxiety. Additionally, participants in the baseline condition mentioned the keywords "uncertainty" and "unpredictable", which are well-known as a root cause of low-battery anxiety, more than the system condition.

5.2 H1: Users Feel Less Low-Battery Anxiety When Energy Consumption Prediction is Provided in Low-Battery Situations

In Study 1, we analyzed survey responses from the system condition (SC) and baseline condition (BC) groups. By a numerical comparison of anxiety levels (*i.e.*, STAI-6 scores [40]) between the two groups, we verified **H1** and demonstrated the effectiveness of energy consumption prediction in alleviating low-battery anxiety.

We also found that the energy consumption prediction reduced uncertainty, which is the root cause of anxiety [11, 20, 21], by analyzing the frequency of keywords such as *"uncertainty"* and *"unpredictability"* in responses to narrative questions from the two groups.

STAI-6 scores analysis. As shown in Figure 4a, participants in the SC group experienced lower levels of anxiety compared to the BC group when the battery level dropped to 15% (U=286.5 and p<.001) and 5% (U=203.5 and p<.05). Specifically, the average STAI-6 scores at 15% battery level stood at 32.59 (SD=11.7) for the SC group and

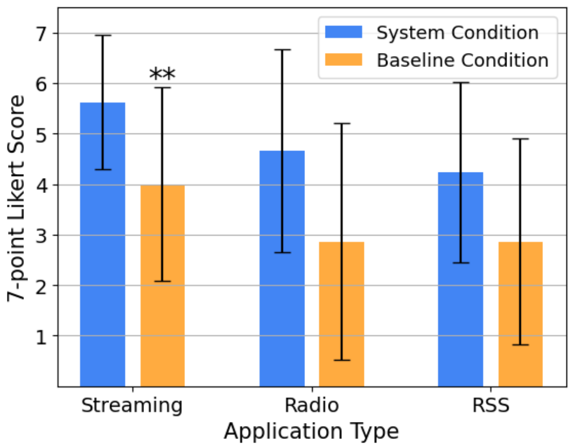


Figure 5: Average scores for answering *"How much planning went into your application usage regarding battery usage?"* for each application (higher is better). Participants answered this question only if they had used that specific application during the experiment. Participants in the system condition rated higher than those in the baseline condition. Notably, there was a significant difference in the streaming application.

56.47 (SD=8.62) for the BC group. At the 5% level, the scores were 60.37 (SD=10.7) for the SC group and 66.27 (SD=12.85) for the BC group.

In contrast, we did not observe significant differences at the 10% battery level (U=165.0 and p>.05), even though the SC group members experienced marginally lower anxiety levels than the BC group. The corresponding STAI-6 averages at 10% were 60.93 (SD=6.65) for the SC group and 61.96 (SD=12.1) for the BC group. We attribute this small difference at the 10% battery level to the absence of a notification alert upon reaching this battery level. Unlike at 15% and 5%, the lack of such system alerts might have made participants less aware of their battery situation, resulting in an insignificant statistical difference in their anxiety levels.

Content analysis regarding anxiety-related keywords. We observed a significantly lower frequency of *"uncertainty"* and *"unpredictability"* keywords in the SC group compared to the BC group. As shown in Figure 4b, only 38% (7 out of 18) of the participants in the SC group mentioned these keywords as a reason for their anxiety. In contrast, 88% (15 out of 17) participants in the BC group mentioned those keywords in their responses. This result aligns with the psychological knowledge that anxiety stems from uncertainty. In other words, *SERENUS effectively addresses the root cause of low-battery anxiety*, leading users to feel less low-battery anxiety.

5.3 H2: Users Can Consciously Plan Their Application Usage Based on Energy Consumption Predictions

In addition to alleviating low-battery anxiety, we expected that SERENUS helps users optimize their application usage so they will use their mobile devices longer instead of stopping using devices prematurely. We verified this **H2** through two analyses with 1) participants' responses to the post-survey and 2) various log data, including total energy consumption and application usage time, collected throughout the experiment.

Post-survey analysis. As shown in Figure 5, we observed that participants in the SC group planned their application usage more than those in the BC group. We statistically compared their responses regarding the degree of planning for each application between the SC and BC groups. For the streaming application, which was the most frequently used by both groups, we found statistical significance (U=137.0 and p<.01). For other applications, we could not establish statistical significance due to the insufficient number of samples, although the averages were noticeably different. In the responses regarding the planning of streaming application usage, the average scores were 5.62 (N=13 and SD=1.33) for the SC group and 4.00 (N=14 and SD=1.92) for the BC group. For the radio application, the average scores were 4.67 (N=6 and SD=2.01) for the SC group and 2.86 (N=7 and SD=2.34) for the BC group. For the RSS application, the average scores were 4.23 (N=13 and SD=1.79) for the SC group and 2.86 (N=7 and SD=2.04) for the BC group.

Log data analysis. Based on the log data from participants, we evaluated how effectively they planned their application use to optimize usage until the end of the experiment. To this end, we compared the total energy consumption of the participants until the end of the experiment. As a result, participants in the SC group consumed significantly more energy than participants in the BC group (U=103.0 and p<.01). Participants in the SC group used 612.86 mAh (SD=32.61) on average, while participants in the BC group used 570.44 mAh (SD=34.71) on average.

Furthermore, to evaluate how effectively the participants planned their application usage, we compared application usage time. We inquired about participants' preferred application through a pre-survey before the experiment and then evenly assigned them to the BC and SC groups based on their responses. Then, we compared the application usage time of the streaming applications between the two groups. We could not compare other applications because most of the participants preferred the streaming application, and very few participants preferred the RSS and radio applications.

In the comparison of application usage time of the streaming application, we observed that participants in the SC group used the streaming application longer than those in the BC group. Specifically, participants in the SC group used the streaming application for 8.25 minutes (SD=4.73) on average, while participants in the BC group used it for 4.88 minutes (SD=4.1) on average.

Overall, we verified **H2** from both users' responses to the post-survey and the log data. We found that users planned and optimized their application usage to use devices longer with the provided prediction of energy consumption.

5.4 H3: The Accuracy of Energy Consumption Prediction Positively Affects Users' Application Usage Planning

Throughout our large simulation study in Study 2, we found that there is a significant difference in users' behaviors when the accuracy of prediction is above and below 15%, which supports our last hypothesis, **H3**.

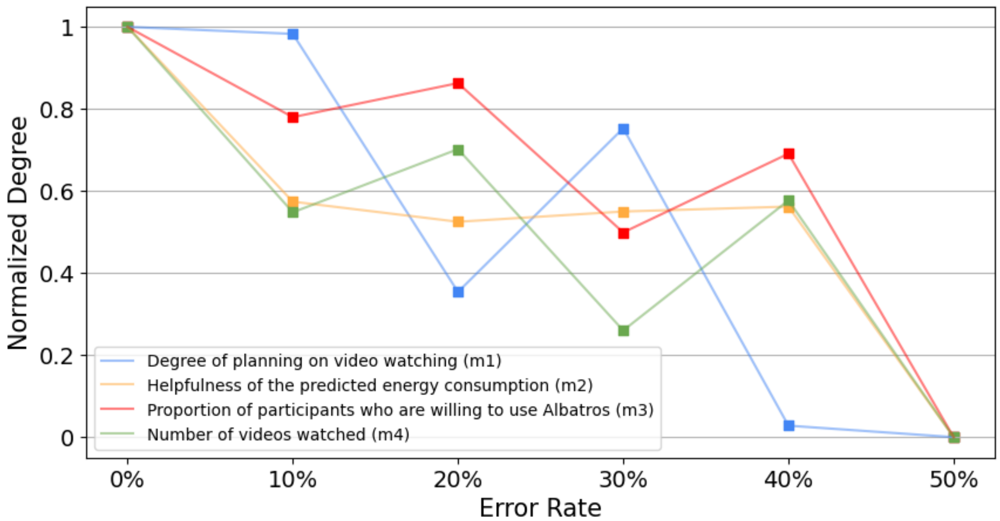


Figure 6: Trends of four metrics, across six system condition groups. We applied min-max normalization to the average values of these four metrics to analyze the trend. Generally, the lower the error rate, the more effective the system. Notably, when the error rate increased from 10% to 20%, the degree of planning on video watching decreased remarkably.

Users' tendency according to error rates. As a reminder of subsubsection 4.2.4, from seven groups, we gathered data of four metrics: **m1**) degree of planning on video watching, **m2**) helpfulness of the predicted energy consumption, **m3**) proportion of participants who are willing to use Serenus, and **m4**) number of videos watched. To analyze tendencies with respect to error rates, we employed min-max normalization on the gathered data. While the raw data is available in the Appendix (Table 5 in §8), our discussion in this section is based on the normalized data.

As shown in Figure 6, we can observe that the four metrics commonly tend to decrease as the error rate increases. In particular, we can observe three notable points: **t1**) the degree of **m2** decreased sharply at 50%, **t2**) the S-10% group exhibited a remarkably higher degree of **m1** compared to the S-20% group, and **t3**) the degree of **m1** significantly increases at the 30% prediction error rate.

Among those notable points, we focused on **t2**, the significant difference in the degree of planned video watching between S-10% and S-20%. This is mainly because, as shown in §5.3, how participants plan application usage is highly affected by low-battery anxiety. Additionally, we did not consider **t3** because the standard deviation at 30% is too large, and **t3** is possibly observed due to this high standard deviation. With **t2**, we realized that only participants in two groups S-0% and S-10% planned their application usage better than the baseline condition group, while participants in the other system condition groups (S-20%, S-30%, S-40% and S-50%) planned their application usage *worse* than the baseline condition group. From this, we deduced an error rate of 15% is meaningful, and we studied more details.

Significant differences between above and below 15%. To further study the accuracy of 15%, we reclassified participants into two groups: a group, called SU-15%, that experienced error rates under 15% (*i.e.*, S-0% and S-10%) and the other group called SO-15%, which experienced error rates over 15% (*i.e.*, from S-20% to S-50%). We found that participants in the SU-15% group benefited significantly more than those in the SO-15% group in all aspects.

Results are described in Figure 7. In particular, regarding **m1**, the SU-15% group planned significantly better than the SO-15% group (U=787.0 and p<.01), with the average scores on the degree of planned video watching of 5.22 (SD=2.15) for the the SU-15% group and 4.00 (SD=1.96) for SO-15% group. For **m2**, the SU-15% group felt the predicted energy consumption was significantly more helpful than the SO-15% group (U=772.0 and p<.01), with the average scores on how helpful the predicted energy consumption of 5.78 (SD=1.24) for SU-15% group and 4.84 (SD=1.58) for SO-15% group. For **m3**, a higher proportion of participants in the SU-15% group were willing to use the system (U=695.5 and p<.05). Specifically, 87% of the SU-15% group (20 out of 23 participants) answered they wanted to use Serenus. In contrast, 66% of the SO-15% group (33 out of 50 participants) answered so. Lastly, regarding **m4**, the SU-15% group watched 3.57 (SD=1.31) videos out of six while the SO-15% group watched 3.14 (SD=1.13), indicating the SU-15% group watched more videos than the SO-15% group (U=722.0 and p<.05).

Considering the significant differences between the accuracy above 15% and under 15%, we conclude that 15% accuracy is notably meaningful, supporting our last hypothesis **H3**.

5.5 Helpfulness of Serenus in Real-World Low-Battery Situations

Frequency of low-battery situations and low-battery anxiety. First, we examined how many times participants experienced low-battery situations (*i.e.*, the battery level dropped below 20%) during the evaluation. According to answers to the daily questions, six out of seven participants (*i.e.*, 86%) experienced low-battery situations

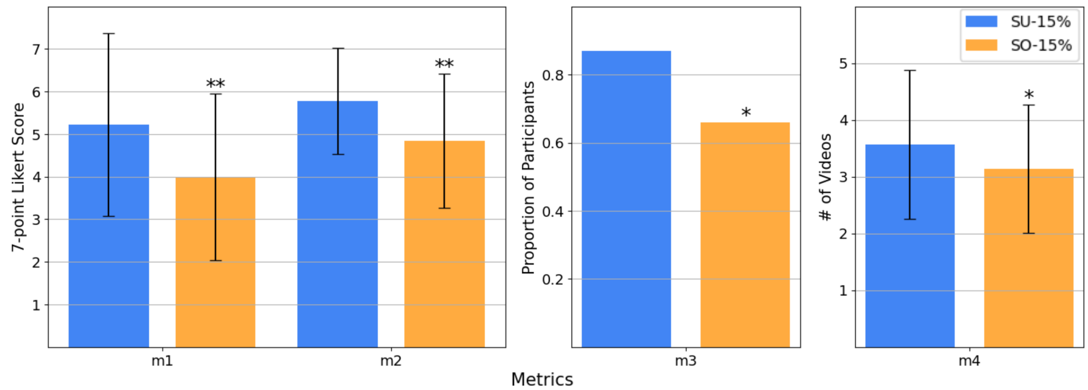


Figure 7: Average values and standard deviations of metrics from the two participant groups regrouped based on the 15% error rate: SU-15% and SO-15% (higher is better). Participants in the SU-15% group showed significantly higher values in all four metrics than those in the SO-15% group, implying energy consumption prediction with an error rate under 15% outstands compared to others.

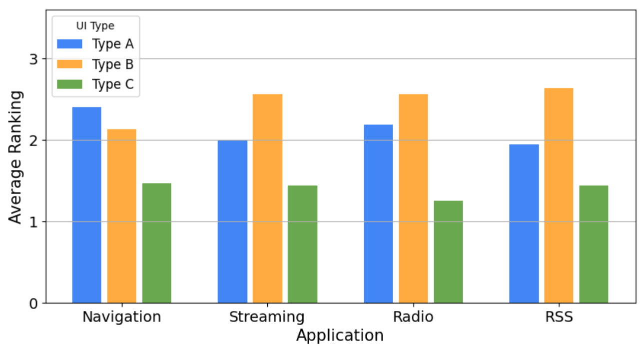


Figure 8: Preferences for UI types in Figure 3 by application among participants in the baseline condition group of Study1. For the navigation application, participants have a similar preference for both Type A and Type B. Meanwhile, participants prefer Type B for the rest of the applications.

during the experiment. In addition, among 45 answers [2] to the daily question, 11 (*i.e.*, 24.4%) reported that they experienced a low battery situation that day. This means that participants underwent a low-battery situation once or twice a week on average. Furthermore, out of those 11 daily answers, 10 reported that participants felt anxious because of the low battery level.

Helpfulness in alleviating low-battery anxiety. Next, we inspected how much Serenus can improve the user experience in low-battery situations. To this end, we analyzed answers to two questions regarding low-battery situations from the daily survey. As a reminder, the two questions are *"When the battery level is lower than 20%, compared to the existing Android system, was energy consumption prediction helpful or hindering in alleviating low-battery anxiety?"* and *"When the battery level is lower than 20%, compared to the existing Android system, was energy consumption prediction helpful or hindering in planning application usage?"*.

As shown in Figure 9, we observe that participants thought that Serenus is helpful in alleviating low-battery anxiety and in planning their application use. In particular, among the 11 daily answers where participants underwent low battery situations, 10 were positive regarding alleviating low-battery anxiety (*i.e.*, somewhat helpful, helpful, and strongly helpful). In other words, participants reported that using Serenus has more advantages in alleviating low-battery anxiety compared to the existing system (*i.e.*, Android). Additionally, participants' answers were close to *"helpful"* on average. This result supports our claim on **H1** (§5.2) by demonstrating participants feel that Serenus is more helpful than a system without energy consumption prediction (*i.e.*, Android) in real-world low-battery situations.

Regarding the improvement of planning for application use, we found that 7 out of the 11 daily answers reported positive reponses, and the average was in the middle of *"somewhat helpful"* and *"helpful."* These answers also support our claim on **H2** (§5.3) to some extent, but fewer participants provided positive feedback than in the case of **H1**. We analyzed descriptive reasons to understand the reason. As a result, we found out that a few participants did not change how they use applications, although they reported energy consumption predictions were somewhat helpful in alleviating low-battery anxiety. For example, one participant mentioned that *"Since I mainly watch videos from channels I subscribe to, I kept watching videos I wanted to see regardless of the prediction. [FS06]"* In such cases, Serenus is of limited help, only giving users confidence about battery consumption.

Additional benefits in situations where the battery is sufficient. Surprisingly, we found that one participant answered that on three days out of a week, Serenus was helpful even when the battery power of the device was enough. As a recall, the question was *When the battery level is enough (i.e., more than 20%), compared to the existing Android system, was energy consumption prediction helpful or hindering in alleviating low-battery anxiety?*

[2]Seven participants performed the experiment over seven days, while four participants did not answer daily survey questions of one day.

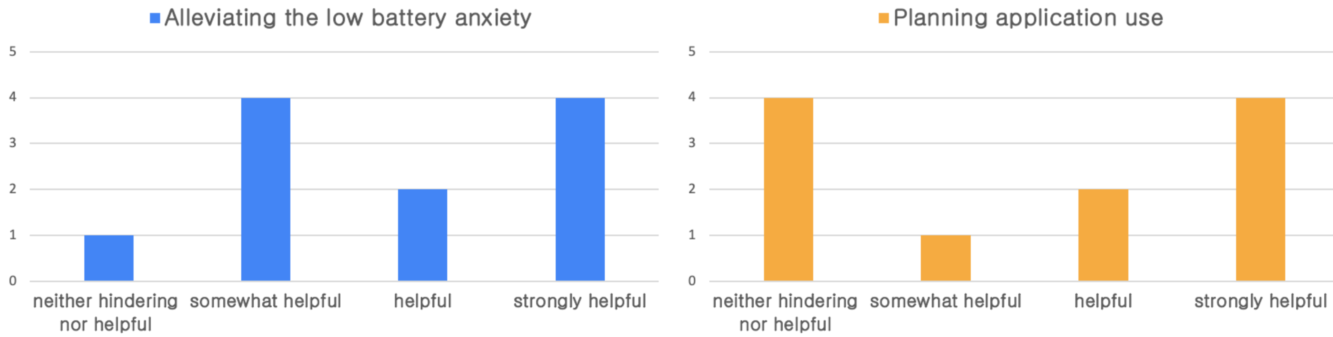


Figure 9: During the field study in §4.3, we surveyed the effects of SERENUS in low-battery situations on the degree of alleviation of anxiety and the planning and usage of applications. The survey was conducted daily using a 7-point Likert scale. The questions within this scale were specifically designed to compare SERENUS against the existing Android system. The responses consistently indicated that SERENUS performs comparably to or surpasses the current Android system.

We analyzed descriptive reasons for the above answers and found that the participant did not even want to experience a low-battery situation. Specifically, the participant gave the reason as the following reason: *"By looking at the current device's battery level (approximately 30-50%) and the predicted value, I took care to ensure that the remaining battery capacity did not fall below the psychological Maginot line (i.e., 20%)."* Based on this participant, We conjecture that SERENUS can be helpful in situations even where the battery level is sufficient if a user is very sensitive to battery draining, and we leave a further study in such situations as future work.

5.6 Preferences for Different Types of UI for Energy Consumption Prediction

Through responses from participants of Study 1 who did not use SERENUS, we analyzed preferences for UI types for energy consumption prediction. As shown in Figure 8, we observed that participants preferred Type A, showing the remaining battery, and Type B, showing available time, across most applications compared to Type C, energy consumption per minute. In the navigation application, the preference for Type A was the highest, while the preference for Type B was the highest in the streaming, radio, and RSS applications. However, the difference in preferences between Type A and Type B was very marginal.

5.7 System Usability

Using the System Usability Scale (SUS) [6] in Study 1, we confirmed that SERENUS offers high usability to users. Since SERENUS provides energy consumption predictions just before users decide on application execution, we surveyed participants in the system condition (SC) group who experienced SERENUS during Study 1, using SUS to validate its usability.

As a result, participants in the SC group gave positive scores regarding the usability of SERENUS. The average usability score of SERENUS was 76.67 (SD=11.3). In general, a SUS usability score of 70 or above is considered to indicate positive usability [4, 6, 7]. Notably, the average score for *"I thought the system was easy to use."* was 4.39 out of 5, and for *"I would imagine that most people would learn to use this system very quickly."*, it was 4.32 out of 5.

6 SYSTEM EVALUATION

In this section, we evaluate various system characteristics of SERENUS. In particular, we show that SERENUS achieves high accuracy in energy consumption prediction (§6.1); SERENUS incurs small runtime overhead and negligible energy consumption (§6.2); and the engineering effort to adopt SERENUS is minimal (§6.3).

Experimental setup. All our experiments were conducted on a Pixel 6 device equipped with Google Tensor (S5P9845) SoC, and we implement a prototype of SERENUS on Android 12.1 (Software version: SP2A.220505.002). The evaluation to measure the accuracy (§6.1) were additionally conducted on Pixel 3 and Pixel 4 XL, both are equipped with Qualcomm Snapdragon 855, to confirm that SERENUS shows high accuracy on various mobile device models.

6.1 Accuracy of Energy Consumption Prediction

We first evaluate the prediction accuracy, an important characteristic of SERENUS. We measure the accuracy across different real-world use case scenarios under diverse execution environments on various devices. In this evaluation, power models assume the use of WiFi and GPS, while §7.5 discusses how SERENUS can handle cases where this assumption does not hold.

For all evaluations for the accuracy, we compare the predicted energy consumption with the actual energy consumption, which can be read from the battery monitoring interface [18, 27], a hardware unit built in smartphones. Additionally, we repeat each measurement 30 times to calculate all average values unless otherwise mentioned.

6.1.1 *Prediction Accuracy for Various Applications Across Different Device Models.*

In order to see the prediction accuracy of SERENUS across various applications and use case scenarios, we evaluate 12 free and open-source software (FOSS) [15] Android applications [1, 2, 28, 38, 39, 45, 46, 51–55] in different categories as shown in Table 3. For this evaluation, we modified the 12 applications to apply SERENUS. To construct power models for these applications on each device, we repeatedly execute each application in our laboratory and collect data samples. For each use case scenario, 20 data samples

Application	Category	Use case scenario	MAPE (%) Pixel 3	Pixel 4XL	Pixel 6
AntennaPod [1]	Podcast & Audio Book Player	Listen radio	6.3	7.4	8.6
ReadYou [2]	RSS readers	Read articles with scrolling	7.6	6.3	7.8
organicmaps [46]	Maps & Navigation	Search & Navigate routes	9.4	10.2	8.2
NewPipe [45]	Media Frontends	Watch a video clip	11.8	8.6	8.5
BinaryEye [39]	Barcode Scanner	Capture a bar-code using camera	8.7	8.2	9.1
FreeDcam [28]	Camera	Take a video	8.4	8.2	9.4
Simple-Voice-Recorder [55]	Voice-Record	Record using a microphone	9.6	7.4	9.5
Simple-Flashlight [53]	Flashlight	Turn on the flashlight	6.7	8.9	9.6
Simple-Gallery [54]	Image Viewer	Watch saved videos	6.8	5.8	8.8
Simple-Calculator [51]	Calculator	Calculate mathematical expression	5.6	6.2	8.5
		Calculate in widget mode	5.3	6.2	7.6
Simple-Clock [52]	Clock & Time	Run stopwatch	7.1	6.2	7.6
		Run timer	7.0	4.2	7.6
simplytranslate_mobile [38]	Translator	Translate English sentences into Korean	7.6	7.1	8.4

Table 3: List of use case scenarios and SERENUS' prediction error (MAPE) for each use case scenario.

are collected, and the data is partitioned into training and testing datasets at an 8:2 ratio. The training dataset (*i.e.*, 16 samples) and testing dataset (*i.e.*, 4 samples) are used to construct SERENUS and measure its average prediction accuracy.

As shown in Table 3, SERENUS demonstrated high accuracy across various applications and their use case scenarios on each device. To evaluate the regression-based power models on each device, we applied the power model to the test data set and calculated the mean absolute percentage errors (MAPE) to verify the accuracy of the power model. As a result, SERENUS shows MAPE of 7.7%, 7.2%, and 8.5% on average in Pixel 3, Pixel 4XL, and Pixel 6, respectively. All these MAPE values are under the 15% accuracy derived from Study 2 in §5.4.

6.1.2 Prediction Accuracy Under Different Environments. We also evaluate whether SERENUS can predict energy consumption accurately even if the environment differs from when data samples are collected. In this evaluation, we change the execution environment by 1) adjusting screen brightness and volume levels, and 2) running different applications in the background.

Different screen & audio settings. When users use their phones, they often change the brightness or audio volume, which significantly impacts energy consumption. To investigate the impact of these changes on prediction, we select the "watch a video clip" scenario of NewPipe because it is affected by both screen brightness and audio volume. We measure the accuracy of predicting the energy consumption of the scenario over a total of four experiments using two screen brightness levels and two audio volume levels. The evaluation results show that the average adjusted R^2 and MAPE values are 0.86 and 9.5%, respectively.

Applications running in the background. Users often use multiple applications simultaneously such as "video calling while watching a video". In this experiment, we measure the accuracy of predicting the energy consumption of a scenario when another application is running in the background. We select "watch a video clip" as the target scenario and measure the accuracy of predicting its energy consumptin when a camera application is operating in the background. On average, the target scenario consumes 0.23% of the battery per minute, while the camera application running in the background uses 0.11% per minute. Under this condition, the accuracy of SERENUS yields an adjusted R^2 value of 0.85 and a MAPE of 10.6%.

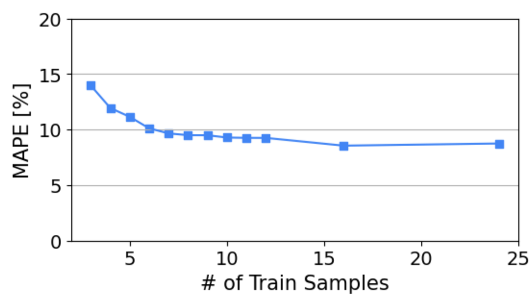


Figure 10: Prediction error (MAPE) of SERENUS when varying the number of training data samples. As the amount of training data increases, the error decreases. The prediction error converges when the number of samples exceeds 16.

Overall results. From the two usage environments, we confirm that while SERENUS's accuracy decreases slightly, it still remains within acceptable ranges. The increased error rates (*i.e.*, MAPE values) for both experiments are 9.5% and 10.6% respectively. These values are still below the threshold of accuracy to be helpful in planning application usage (discussed in §5.4), showing that SERENUS can predict the energy consumption even under environments that are different from when collecting training data samples.

6.1.3 *Prediction Accuracy According to the Number of Training Data.*

Because Serenus constructs power models after installing applications, it undergoes the "cold start" problem, which is characterized by low accuracy when the number of training data samples is small. To determine how much data should be collected to achieve acceptable accuracy, we measure the accuracy of predicting the energy consumption of the "watch a video clip" scenario according to the number of training samples.

As depicted in Figure 10, Serenus shows a small error even when the number of data samples is small. Serenus achieves meaningful prediction accuracy with as few as seven training samples, indicating that Serenus does not require a long time to construct the power models. In the case of video streaming applications, Serenus can provide accurate predictions of energy consumption after users watch seven videos.

6.2 System Overhead of Serenus

We further evaluate the system overhead of Serenus for two primary purposes: 1) to confirm that Serenus ensures low latency for real-time prediction, and 2) to demonstrate that the storage overhead of Serenus is negligible.

6.2.1 *Additional Energy Consumption of Serenus.*

To measure how much energy Serenus consumes, we compare the total energy consumption of two Pixel 6 devices running a video streaming application for 2.5 hours. One of the devices is equipped with Serenus, while the other runs the unmodified Android platform. We choose a video streaming application because it is an application that users may run for a long time and that may degrade the user experience significantly if Serenus consumes a lot of energy.

We repeat the measurement 30 times and obtain the average of energy consumption on each device. As a result, we confirm that Serenus consumes 1.3% more energy compared to the unmodified Android platform. Translating the additional consumption into video-watching time, if a user can watch a video for 2.5 hours on the unmodified Android platform with a given amount of energy, they can watch the same video 115 seconds less (*i.e.*, they can watch the video for 2 hours and 28 minutes) on Serenus. Although the additional energy consumption shortens the video-watching time, we believe that the shortened time is negligible and does not have a significant impact on the user experience.

Furthermore, it is worth noting that the additional energy consumption can be easily reduced. The current implementation of Serenus 's APIs invokes RPC calls excessively frequently, causing extra energy use. During measurements, RPC calls were made whenever a video frame was updated, which is dozens of times per second. To reduce energy consumption, we can modify Serenus 's API to make an RPC call only once per second. One call per second should be sufficient, as the remaining battery power does not change drastically within one second. This modification does not require additional changes to applications.

6.2.2 *API Invocation Latency.*

We measure the latency of API invocations required to integrate Serenus into applications. This is because if the latency is too high, users may notice a delay while using applications, which can degrade the user experience. We chose NewPipe, a video streaming application, to measure the latency and took the average latency of 1,000 measurements.

As a result, API invocations incur a latency of tens of milliseconds, where the main source of the latency is inter-process communication (IPC) between an application and the system framework. In particular, the API invocations of `startLogging()` and `endLogging()` (*i.e.*, APIs to collect data samples described in §3.2) take 3.92ms (SD=1.12) and 3.76ms (SD=1.29) respectively, which is negligible. On the other hand, the API invocation of `getPrediction()` to retrieve the predicted energy consumption takes 10.48ms (SD=1.98). Given that the refresh rate of the most recent smartphone, the iPhone 15, is 60Hz (*i.e.*, 16ms per frame), the prediction API has such low latency that it can update every frame without users noticing any delays in prediction.

6.2.3 *Storage Overhead.*

Serenus incurs storage overhead as it collects data on mobile devices and stores power models in storage. According to our measurement, collecting training data samples consumes 140 bytes per data sample while Serenus needs to collect data samples for each use case scenario. Considering Serenus achieves prediction accuracy with approximately 30 training samples, Serenus needs to collect 30 times the number of use case scenarios, resulting in a total storage overhead of 140 bytes multiplied by 30 times the number of use case scenarios. On the other hand, storing power models consumes around 100 bytes per model. Therefore, the total amount of storage usage is approximated to 100 bytes times the number of use case scenarios that a user is using. In both cases of collecting data samples and storing power models, Serenus consumes less than 1 MB even if a user uses 100 applications.

6.3 Engineering Efforts to Integrate Serenus into Applications

To quantitatively assess developers' efforts in integrating Serenus into their applications, we count the required lines of code (LoC) for applying Serenus in the four open-source applications from User Study 1: Newpipe, Organic Maps, AntennaPod, and ReadYou. As mentioned in §3, developers need to adjust their applications for two reasons: 1) to notify when a scenario begins and ends (§3.2) and 2) to retrieve and display the predicted energy consumption (§3.3). Table 4 shows the LoCs [3] to integrate Serenus into the applications. As shown in this table, integrating Serenus requires small engineering efforts, which are approximated to an average of 5.5 lines of code. This source code modification simply involves calling APIs to communicate with Serenus (see Figure 2) and creating layout components (*i.e.*, Android's `View`), and it does not involve any complicated algorithm or implementation.

7 DISCUSSION AND LIMITATION

In this section, we discuss various observations from our experiments, which provide a deep understanding and guidelines in alleviating low-battery anxiety.

[3] We measure LoCs using cloc [16].

Application name	Programming language	Lines of Code (LoC)	
		Power model construction (§3.2)	Energy consumption prediction (§3.3)
NewPipe	Java & Kotlin	2	4
ReadYou	Kotlin	2	4
organicmaps	Java	2	3
AntennaPod	Java	2	3

Table 4: List of Serenus-integrated open source applications. For each application, we count the lines of code needed for APIs to communicate with Serenus.

7.1 How Energy Consumption Predictions Alleviate User's Low Battery Anxiety

7.1.1 Diverse Forms of Uncertainty. While uncertainty is commonly recognized as the root cause of anxiety, responses from participants in the baseline condition group of Study 1 (§4.1) revealed that diverse forms of uncertainty arise when the battery level is low. We identified these types of uncertainty to provide a deep understanding of low-battery anxiety, which can guide the future development of mobile system framework as follows.

The first is the "uncertainty of the device's remaining battery life." Five out of 17 participants felt anxious because they did not know when their battery would run out: *"I was anxious because I didn't know when my phone would run out of battery [B01, B03, B04, B07, B10]."* The second is the "uncertainty about whether it is okay to use the desired application". Nine out of 17 participants felt anxious about whether they could use their desired applications. Examples include *"I was anxious that I might not be able to scan the final assignment QR code when I return to school"* and *"I was afraid of not reaching my desired destination using the navigation application [B04, B07, B08, B09, B11, B12, B14, B16, B17]."* And the third is "uncertainty about unpredictable future situations". Two out of 17 participants expressed anxiety about not being able to use applications due to unpredictable situations in the future: *"I was worried about missing important calls later on [B06]"* and *"I was concerned about not receiving important phone calls or messages [B02]."*

7.1.2 Detailed Observations on How Prediction Alleviates Uncertainty. To ascertain the reasons for the reduction of low-battery anxiety by Serenus, we thoroughly examined responses related to overall anxiety during the whole experiment and the application-specific anxiety from participants in the system condition group of Study 1.

First and foremost, we observed that confidence in the accuracy of the application's energy consumption prediction alleviated their anxiety. Specifically, 22% (4 out of 18) of the participants explicitly expressed reduced anxiety levels attributed to the accuracy of the battery predictions *"The certainty that the energy consumption prediction is precise alleviated my anxiety [S05, S08, S09, S10]."* Further, 28% (5 out of 18) of the participants diminished their anxiety by managing future battery levels based on the predicted energy consumption of the application, thereby strategizing for potentially unpredictable situations. Examples include *"The ability to intuitively manage the battery based on the application's energy consumption prediction reduced my anxiety"*, and *"Considering the possibility of unpredictable situations, like needing to make a call, I preserved battery by prediction value [S03, S07, S09, S10, S04]."*

7.2 Improvements in User Experience

We observed that Serenus enhanced the user experience by providing energy consumption predictions. Participants of Study 1 commented in the post-survey that their user experience had improved after receiving energy consumption predictions from Serenus, even though Serenus was primarily designed for alleviating low-battery anxiety.

By recognizing the energy consumption predictions of various use case scenarios, participants had more options regarding which application to use and what content to play. Specifically, some participants became more flexible in their decision-making based on each use case scenario's predicted energy consumption. One participant shared, *"I was initially watching a video, but since I checked that the remaining battery was not enough to keep watching videos until arrival, so I switched to reading the news which consumes less battery [S07, S11]."* Another participant commented, *"I was reading a news article because the battery did not have much charge left. But later, I noticed that my phone wouldn't turn off, so I strategically watched the video [S15]."* Some participants used the predicted energy consumption to decide which contents within applications to play. For instance, *"To keep my phone on, I watched short videos because I was reassured by the low energy consumption of short videos [S12]."* Also, some participants adjusted settings such as screen brightness or audio volume in specific use case scenarios to manage the remaining battery. We observed responses such as *"Initially, when watching videos, I viewed them normally. But as the battery decreased, I found myself lowering the screen brightness [S08, S10]."* These observations reveal that Serenus helped users make informed decisions and improved their experience.

7.3 Guidelines for Application Developers in Designing User Interfaces

There is no dominantly preferred UI type for energy consumption prediction among the three types, Type A, B, and C (shown in Figure 3). While Types A and B are generally more favored than Type C, there are still situations where Type C is chosen. After closely examining the reasons for their preferences, we categorized them into the following three categories.

Intuitive user interface. For most applications, participants preferred Type A and Type B more than Type C, because those types are more intuitive to understand and easier to plan their application usage. They reasoned that, unlike Type C, there was no need for additional calculations to determine the available usage time or battery when selecting the desired applications or content. Some participants commented, *"We need to calculate additionally to know total battery consumptions of all contents when using Type C [B14, B01, B02, B03, B16]."*

Application-specific usage patterns. We observed that UI preferences could vary according to application-specific usage patterns. For instance, users typically have a predetermined destination in mind when using a navigation application. In such a scenario, users were more interested in the remaining battery after they arrive (Type A) rather than others. One participant mentioned, *"For map apps, what's important is whether I can reach my destination with the current battery state or not [B05]."* Participants preferred Type A for applications that normally have a fixed-length of scenario (*e.g.*, navigation application, YouTube), while Type B is preferred for applications with a fluctuant length of scenario (*e.g.*, Twitch).

Diversity in individuals' characteristics for each application. In a single application, preferences for UI can vary depending on an individuals' characteristics. For the streaming application, preferences for Type A and Type B varied based on individuals' diverse selection criteria. Some participants who preferred Type B, considered it was more important to decide how long to watch the video than which video to watch: *"Being able to see the available viewing time before watching helps in planning the app usage [B00, B03, B04, B13]."* Other participants who favored Type A mentioned that they selected contents to watch based on a relative comparison of the energy consumption prediction of each content: *"It allows for a relative comparison by content [B05, B06, B10, B16]."* For some applications that can have various characteristics according to users, preference for Type C may increase. For example, for the RSS application, since the reading speed varies from person to person, some participants answered that they preferred Type C: *"Everyone reads at a different pace [B07, B10]."*

7.4 Additional Cognitive Burden due to Energy Consumption Prediction

As SERENUS provides users with more information, mobile users who are robust to low-battery anxiety might perceive the energy consumption prediction as unnecessary or even as an additional cognitive burden. In order to confirm how many users think SERENUS negatively, we analyzed answers to narrative questions from our user studies.

As a result, 2 among 40 participants of Study 1 (§4.1) provides negative feedbacks on SERENUS. Specifically, we observed the following comments: *"It was annoying because energy consumption kept being displayed on the screen." [S01]* and *"It was hard to use the application freely because the amount of battery decreasing is displayed in real time." [S06]*. Despite this negative feedback, we believe that SERENUS holds value from two perspectives. First, based on our user studies, most of users found SERENUS and its energy consumption predictions useful without being disruptive. Second, energy consumption prediction can be easily configured to be invisible or hidden. Since energy consumption predictions are displayed through Android's `View`, they can be easily removed from the dispaly with a few lines of code. With some engineering efforts from developers, we believe that users who prefer not to use SERENUS will not experience significant inconvenience, as they can simply hide the energy consumption prediction.

7.5 Various Factors that Affect the Accuracy of Energy Consumption Prediction

While our evaluation was conducted with the assumption that WiFi and GPS are used, the hardware users employ affects the accuracy of energy consumption predictions. For example, WiFi and LTE modules consume different amounts of energy [26, 63], and a user can decide whether to turn GPS on or off while using Google Maps.

SERENUS can address these factors by recognizing which hardware component is being used and selectively applying power models accordingly. For example, SERENUS may choose a model based on whether LTE or WiFi is used. Power models for various hardware components have been well-studied in previous works [26, 29, 47, 61, 63]. In addition, although the current implementation shows high accuracy, improving power models is orthogonal to the findings of this paper. Future improvements to SERENUS can be achieved by incorporating more precise models as they are developed. We leave the development of more accurate power models as future work.

8 CONCLUSION

In this paper, we conduct in-depth study on how to alleviate low-battery anxiety. The key insight to this end is that low-battery anxiety stems from the uncertainty around the energy consumption of applications. To eliminate the uncertainty, we propose SERENUS, a mobile system framework to accurately predict the energy consumption in real-time. With SERENUS, we conducted two rounds of user studies. From these studies, we demonstrated that 1) the real-time, accurate, and user-friendly energy consumption prediction can significantly alleviate low-battery anxiety, 2) with the prediction, users can plan their application usage to use mobile devices longer, and 3) the accuracy of the prediction affects users' application usage plans. Lastly, we summarize our findings from our experiments, which provides deep understanding and a guideline in alleviating users' low-battery anxiety.

ACKNOWLEDGMENTS

We sincerely appreciate anonymous reviewers for their insightful and valuable comments. This work was supported in part by Samsung Electronics (IO210204-08385-01), NRF (RS-2024-00347516), National Research Foundation of Korea (NRF) grant funded by the Ministry of Science and ICT (NRF-2022R1C1C1003123 and NRF-RS-2024-00353125).

REFERENCES

[1] AntennaPod. 2023. GitHub - AntennaPod/AntennaPod: A podcast manager for Android. https://github.com/AntennaPod/AntennaPod. [Accessed 11-09-2023].

[2] Ashinch. 2023. GitHub - Ashinch/ReadYou: An Android RSS reader presented in Material You style. https://github.com/Ashinch/ReadYou. [Accessed 11-09-2023].

[3] Anirudh Badam, Ranveer Chandra, Jon Dutra, Anthony Ferrese, Steve Hodges, Pan Hu, Julia Meinershagen, Thomas Moscibroda, Bodhi Priyantha, and Evangelia Skiani. 2015. Software defined batteries. In *Proceedings of the 25th ACM Symposium on Operating Systems Principles (SOSP)*. Monterey, CA.

[4] Aaron Bangor, Philip T Kortum, and James T Miller. 2008. An empirical evaluation of the system usability scale. *Intl. Journal of Human–Computer Interaction* 24, 6 (2008), 574–594.

[5] Jing Bi, Haitao Yuan, Shuaifei Duanmu, MengChu Zhou, and Abdullah Abusorrah. 2020. Energy-optimized partial computation offloading in mobile-edge computing with genetic simulated-annealing-based particle swarm optimization. *IEEE Internet of Things Journal* 8, 5 (2020), 3774–3785.

[6] John Brooke. 1996. Sus: a "quick and dirty'usability. *Usability evaluation in industry* 189, 3 (1996), 189–194.

[7] John Brooke. 2013. SUS: a retrospective. *Journal of usability studies* 8, 2 (2013), 29–40.

[8] Duc Hoang Bui, Kilho Lee, Sangeun Oh, Insik Shin, Hyojeong Shin, Honguk Woo, and Daehyun Ban. 2013. Greenbag: Energy-efficient bandwidth aggregation for real-time streaming in heterogeneous mobile wireless networks. In *Proceedings of the 34th IEEE Real-Time Systems Symposium (RTSS)*. Vancouver, Canada.

[9] Bytedance Pte. Ltd. 2023. Ulike - Define your selfie in. https://play.google.com/store/apps/details?id=com.gorgeous.liteinternational. [Accessed 15-09-2023].

[10] Anthony Canino, Yu David Liu, and Hidehiko Masuhara. 2018. Stochastic energy optimization for mobile GPS applications. In *Proceedings of the 26th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE)*. Lake Buena Vista, FL.

[11] R Nicholas Carleton, MA Peter J Norton, and Gordon JG Asmundson. 2007. Fearing the unknown: A short version of the Intolerance of Uncertainty Scale. *Journal of anxiety disorders* 21, 1 (2007), 105–117.

[12] Aaron Carroll and Gernot Heiser. 2010. An analysis of power consumption in a smartphone. In *2010 USENIX Annual Technical Conference (USENIX ATC 10)*.

[13] 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 Conference (SIGMETRICS)*. Portland, OR.

[14] Yohan Chon, Elmurod Talipov, Hyojeong Shin, and Hojung Cha. 2011. Mobility prediction-based smartphone energy optimization for everyday location monitoring. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems (SENSYS)*. Seattle, WA.

[15] F-Droid Contributors. 2023. F-Droid - Free and Open Source Android App Repository. https://f-droid.org/. [Accessed 11-09-2023].

[16] Albert Danial. 2023. cloc. https://github.com/AlDanial/cloc.

[17] 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 ACM International Conference on Mobile Computing Systems (MobiSys)*. Virtual.

[18] Mian Dong and Lin Zhong. 2011. Self-constructive high-rate system energy modeling for battery-powered mobile systems. In *Proceedings of the 9th ACM International Conference on Mobile Computing Systems (MobiSys)*. Washington, DC.

[19] Francis Galton. 1886. Regression towards mediocrity in hereditary stature. *The Journal of the Anthropological Institute of Great Britain and Ireland* 15 (1886), 246–263.

[20] Veronica Greco and Derek Roger. 2001. Coping with uncertainty: The construction and validation of a new measure. *Personality and individual differences* 31, 4 (2001), 519–534.

[21] Dan W Grupe and Jack B Nitschke. 2013. Uncertainty and anticipation in anxiety: an integrated neurobiological and psychological perspective. *Nature Reviews Neuroscience* 14, 7 (2013), 488–501.

[22] Liang He, Guozhu Meng, Yu Gu, Cong Liu, Jun Sun, Ting Zhu, Yang Liu, and Kang G Shin. 2016. Battery-aware mobile data service. *IEEE Transactions on Mobile Computing* 16, 6 (2016), 1544–1558.

[23] Jacob B Hirsh, Raymond A Mar, and Jordan B Peterson. 2012. Psychological entropy: a framework for understanding uncertainty-related anxiety. *Psychological review* 119, 2 (2012), 304.

[24] Samuel Isuwa, David Amos, Amit Kumar Singh, Bashir M Al-Hashimi, and Geoff V Merrett. 2023. Content-and Lighting-Aware Adaptive Brightness Scaling for Improved Mobile User Experience. In *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 1–2.

[25] Kyoung-Hak Jung, Yuepeng Qi, Chansu Yu, and Young-Joo Suh. 2014. Energy efficient wifi tethering on a smartphone. In *Proceedings of the 2014 IEEE International Conference on Computer Communications (INFOCOMM)*. Toronto, Canada.

[26] Wonwoo Jung, Chulkoo Kang, Chanmin Yoon, Donwon Kim, and Hojung Cha. 2012. DevScope: a nonintrusive and online power analysis tool for smartphone hardware components. In *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*.

[27] Wonwoo Jung, Chulkoo Kang, Chanmin Yoon, Dongwon Kim, and Hojung Cha. 2012. Nonintrusive and online power analysis for smartphone hardware components. In *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*.

[28] KillerInk. 2023. GitHub - KillerInk/FreeDcam: FreeDcam is a CameraApp for Android >4.0(ics) wich try to enable stuff that is forgotten by the manufacturs. https://github.com/KillerInk/FreeDcam. [Accessed 11-09-2023].

[29] Dongwon Kim, Yohan Chon, Wonwoo Jung, Yungeun Kim, and Hojung Cha. 2016. Accurate prediction of available battery time for mobile applications. *ACM Transactions on Embedded Computing Systems (TECS)* 15, 3 (2016), 1–17.

[30] Jaeheon Kwak, Sunjae Lee, Dae R Jeong, Arjun Kumar, Dongjae Shin, Ilju Kim, Donghwa Shin, Kilho Lee, Jinkyu Lee, and Insik Shin. 2023. MixMax: Leveraging Heterogeneous Batteries to Alleviate Low Battery Experience for Mobile Users. In *Proceedings of the 21st ACM International Conference on Mobile Computing Systems (MobiSys)*. Helsinki, Finland.

[31] Youngmoon Lee, Liang He, and Kang G Shin. 2020. Causes and fixes of unexpected phone shutoffs. In *Proceedings of the 18th ACM International Conference on Mobile Computing Systems (MobiSys)*. Toronto, Canada.

[32] Jingwen Leng, Tayler Hetherington, Ahmed ElTantawy, Syed Gilani, Nam Sung Kim, Tor M Aamodt, and Vijay Janapa Reddi. 2013. GPUWattch: Enabling energy optimizations in GPGPUs. In *Proceedings of the 40th ACM/IEEE International Symposium on Computer Architecture (ISCA)*. Tel-Aviv, Israel.

[33] LG. 2016. "LOW BATTERY ANXIETY" GRIPS 9 OUT OF TEN PEOPLE. https://www.lg.com/us/PDF/press-release/LG_Mobile_Low_Battery_Anxiety_Press_Release_FINAL_05_19_2016.pdf.

[34] Ding Li, Yingjun Lyu, Jiaping Gui, and William GJ Halfond. 2016. Automated energy optimization of http requests for mobile applications. In *Proceedings of the 38th International Conference on Software Engineering (ICSE)*. Austin, TX.

[35] Ding Li, Angelica Huyen Tran, and William GJ Halfond. 2015. Nyx: A display energy optimizer for mobile web apps. In *Proceedings of the 23rd ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE)*. Bergamo, Italy.

[36] Robert LiKamWa, Bodhi Priyantha, Matthai Philipose, Lin Zhong, and Paramvir Bahl. 2013. Energy characterization and optimization of image sensing toward continuous mobile vision. In *Proceedings of the 11th ACM International Conference on Mobile Computing Systems (MobiSys)*. Taipei, Taiwan.

[37] Xiao Ma, Peng Huang, Xinxin Jin, Pei Wang, Soyeon Park, Dongcai Shen, Yuanyuan Zhou, Lawrence K Saul, and Geoffrey M Voelker. 2013. {eDoctor}: Automatically Diagnosing Abnormal Battery Drain Issues on Smartphones. In *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*. 57–70.

[38] ManeraKai. 2023. GitHub - ManeraKai/simplytranslate_mobile: Privacy friendly frontend to Google Translate. https://github.com/ManeraKai/simplytranslate_mobile. [Accessed 11-09-2023].

[39] markusfisch. 2023. GitHub - markusfisch/BinaryEye: Yet another barcode scanner for Android. https://github.com/markusfisch/BinaryEye. [Accessed 11-09-2023].

[40] Theresa M Marteau and Hilary Bekker. 1992. The development of a six-item short-form of the state scale of the Spielberger State—Trait Anxiety Inventory (STAI). *British journal of clinical Psychology* 31, 3 (1992), 301–306.

[41] Jiayi Meng, Qiang Xu, and Y Charlie Hu. 2021. Proactive energy-aware adaptive video streaming on mobile devices. In *Proceedings of the 2021 USENIX Annual Technical Conference (ATC)*. Virtual.

[42] Grace Metri, Weisong Shi, Monica Brockmeyer, and Abhishek Agrawal. 2014. BatteryExtender: an adaptive user-guided tool for power management of mobile devices. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*.

[43] Chulhong Min, Youngki Lee, Chungkuk Yoo, Seungwoo Kang, Sangwon Choi, Pillsoon Park, Inseok Hwang, Younghyun Ju, Seungpyo Choi, and Junehwa Song. 2015. PowerForecaster: Predicting smartphone power impact of continuous sensing applications at pre-installation time. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems (SENSYS)*. Seoul, South Korea.

[44] Radhika Mittal, Aman Kansal, and Ranveer Chandra. 2012. Empowering developers to estimate app energy consumption. In *Proceedings of the 18th ACM Annual International Conference On Mobile Computing And Networking (MobiCom)*. Istanbul, Turkey.

[45] Team NewPipe. 2023. NewPipe - a free YouTube client. https://newpipe.net/. [Accessed 11-09-2023].

[46] Organic Maps OÜ. 2023. Organic Maps: Offline Hike, Bike, Trails and Navigation — organicmaps.app. https://organicmaps.app/. [Accessed 11-09-2023].

[47] Abhinav Pathak, Y Charlie Hu, Ming Zhang, Paramvir Bahl, and Yi-Min Wang. 2011. Fine-grained power modeling for smartphones using system call tracing. In *Proceedings of the 6th European Conference on Computer Systems (EuroSys)*. Salzburg, Austria.

[48] Rui Pereira, Hugo Matalonga, Marco Couto, Fernando Castor, Bruno Cabral, Pedro Carvalho, Simão Melo de Sousa, and João Paulo Fernandes. 2021. GreenHub: a large-scale collaborative dataset to battery consumption analysis of android devices. *Empirical Software Engineering* 26 (2021), 1–55.

[49] Jie Ren, Lu Yuan, Petteri Nurmi, Xiaoming Wang, Miao Ma, Ling Gao, Zhanyong Tang, Jie Zheng, and Zheng Wang. 2020. Camel: Smart, adaptive energy optimization for mobile web interactions. In *Proceedings of the 2020 IEEE International Conference on Computer Communications (INFOCOMM)*. Virtual.

[50] Alex Shye, Benjamin Scholbrock, and Gokhan Memik. 2009. Into the wild: studying real user activity patterns to guide power optimizations for mobile architectures. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. New York, NY.

[51] SimpleMobileTools. 2023. GitHub - SimpleMobileTools/Simple-Calculator: A calculator for quick simple calculations with a nice user interface and no ads. https://github.com/SimpleMobileTools/Simple-Calculator. [Accessed 11-09-2023].

[52] SimpleMobileTools. 2023. GitHub - SimpleMobileTools/Simple-Clock: Combination of a beautiful clock with widget, alarm, stopwatch & timer, no ads. https://github.com/SimpleMobileTools/Simple-Clock. [Accessed 11-09-2023].

[53] SimpleMobileTools. 2023. GitHub - SimpleMobileTools/Simple-Flashlight: A simple modern flashlight with SOS, stroboscope & bright display, has no ads. https://github.com/SimpleMobileTools/Simple-Flashlight. [Accessed 11-09-2023].

[54] SimpleMobileTools. 2023. GitHub - SimpleMobileTools/Simple-Gallery: A premium app for managing and editing your photos, videos, GIFs without ads. https://github.com/SimpleMobileTools/Simple-Gallery. [Accessed 11-09-2023].

[55] SimpleMobileTools. 2023. GitHub - SimpleMobileTools/Simple-Voice-Recorder: An easy way of recording any discussion or sounds without ads or internet access. https://github.com/SimpleMobileTools/Simple-Voice-Recorder. [Accessed 11-09-2023].

[56] Gaurav Singla, Gurinderjit Kaur, Ali K Unver, and Umit Y Ogras. 2015. Predictive dynamic thermal and power management for heterogeneous mobile platforms. In *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 960–965.

[57] Guoming Tang, Kui Wu, Deke Guo, Yi Wang, and Huan Wang. 2020. Alleviating low-battery anxiety of mobile users via low-power video streaming. In *Proceedings of the 40th International Conference on Distributed Computing Systems (ICDCS)*.

[58] Guoming Tang, Kui Wu, Yangjing Wu, Huan Wang, and Guangwu Qian. 2021. Modeling and Alleviating Low-Battery Anxiety for Mobile Users in Video Streaming Services. *IEEE Internet of Things Journal* 9, 7 (2021), 5065–5079.

[59] Twitch Interactive, Inc. 2023. Twitch: Live Game Streaming. https://play.google.com/store/apps/details?id=tv.twitch.android.app. [Accessed 15-09-2023].

[60] Chengke Wang, Fengrun Yan, Yao Guo, and Xiangqun Chen. 2013. Power estimation for mobile applications with profile-driven battery traces. In *International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE, 120–125.

[61] Fengyuan Xu, Yunxin Liu, Qun Li, and Yongguang Zhang. 2013. V-edge: Fast self-constructive power modeling of smartphones based on battery voltage dynamics. In *Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. Lombard, IL.

[62] Kaige Yan, Xingyao Zhang, and Xin Fu. 2015. Characterizing, modeling, and improving the QoE of mobile devices with low battery level. In *Proceedings of the 48th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. Waikiki, Hawaii.

[63] Chanmin Yoon, Dongwon Kim, Wonwoo Jung, Chulkoo Kang, and Hojung Cha. 2012. {AppScope}: Application Energy Metering Framework for Android Smartphone Using Kernel Activity Monitoring. In *Proceedings of the 2012 USENIX Annual Technical Conference (ATC)*. Boston, MA.

[64] Jie You, Jae-Won Chung, and Mosharaf Chowdhury. 2023. Zeus: Understanding and Optimizing {GPU} Energy Consumption of {DNN} Training. In *Proceedings of the 20th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. Boston, MA.

[65] Haibo Zhang, Prasanna Venkatesh Rengasamy, Shulin Zhao, Nachiappan Chidambaram Nachiappan, Anand Sivasubramaniam, Mahmut T Kandemir, Ravi Iyer, and Chita R Das. 2017. Race-to-sleep+ content caching+ display caching: A recipe for energy-efficient video streaming on handhelds. In *Proceedings of the 44th ACM/IEEE International Symposium on Computer Architecture (ISCA)*. Toronto, Canada.

APPENDIX

Groups	m1	m2	m3	m4
Baseline (N=13)	4.46 (1.94)	-	-	3.08 (1.19)
0% (S-0%) (N=13)	5.23 (2.24)	6.23 (1.01)	12	3.77 (1.17)
10% (S-10%) (N=10)	5.2 (2.15)	5.2 (1.32)	8	3.3 (1.49)
20% (S-20%) (N=13)	4.08 (1.85)	5.08 (0.86)	11	3.46 (1.13)
30% (S-30%) (N=14)	4.79 (2.15)	5.14 (1.79)	9	3.00 (0.88)
40% (S-40%) (N=12)	3.5 (1.73)	5.17 (1.19)	9	3.33 (1.15)
50% (S-50%) (N=11)	3.45 (1.97)	3.81 (2.04)	4	2.73 (1.35)

Table 5: Averages of responses in Study 2 for the baseline group and the six system condition groups. Three columns, m1, m2, and m4, demonstrate averages scores of *the degree of planning on video watching* (m1), *helpfulness of the predicted energy consumption* (m2), and *number of videos watched* (m4) respectively. Numbers in parentheses are standard deviations. The m3 column represents *the number of participants who are willing to use Serenus*. For the baseline group, participants did not answer the questions for m2 and m3, because they did not use Serenus. When analyzing these results, we employed min-max normalization as represented in Figure 6.