TYPE Original Research
PUBLISHED 19 December 2024
DOI 10.3389/fcomp.2024.1525382

Check for updates

OPEN ACCESS

EDITED BY
Luca Davoli,
University of Parma, Italy

REVIEWED BY
Zainab Al-Waisi,
IMT School for Advanced Studies Lucca, Italy
Syed Usman J.,
Charles Sturt University, Australia

*CORRESPONDENCE
Shariq Hussain
✉ shariq@fui.edu.pk

RECEIVED 09 November 2024
ACCEPTED 09 December 2024
PUBLISHED 19 December 2024

CITATION
Mahmud U, Hussain S and Karamat T (2024) Towards an intelligent energy conservation approach for context-aware systems in smart environments.
Front. Comput. Sci. 6:1525382.
doi: 10.3389/fcomp.2024.1525382

Towards an intelligent energy conservation approach for context-aware systems in smart environments

Umar Mahmud, Shariq Hussain* and Tehmina Karamat

Department of Software Engineering, Foundation University Islamabad, Islamabad, Pakistan

A smart personal space is a context-aware system that recognizes situations using contextual data. A user interacts within the personal space using smart devices that are mobile, and run-on batteries that have limited power. This paper proposes a Power-Constrained Context-Aware System (PCCA) that uses Markov Chain-based pre-classification to predict context change and defer context processing to conserve energy in an intelligent way. A new Markov Chain Module is added that creates a Markov Chain using history information. This enables PCCA to predict context change for the next observation. The results show that PCCA consumes 37% less power than a context-aware system.

KEYWORDS

power awareness, context awareness, energy conservation, context inference engine, smart personal space, IoE

1 Introduction

Smart environments facilitate the Internet of Everything (IoE) by offering users smarter services and seamless adaption while communicating using wireless links (Alaa et al., 2017; Mahmud et al., 2020; Kirsch-Pinheiro, 2023). A smart environment is made up of a user's wearable and handheld devices, and several smart personal spaces (Mahmud et al., 2022a). Figure 1 shows the concept of a Smart Environment composed of several Smart Personal Spaces and interacting with applications and services remotely.

Among the challenges of Smart Environments are heterogeneity, scalability, seamless integration, privacy, trust, context awareness, and power awareness (Malik et al., 2007; Mahmud and Malik, 2014). Context awareness allows a smart personal space to gather data from sensors present on the devices, space, and environment, and deduce situations so that appliances and services can be invoked (Mahmud and Javed, 2012). While the appliances are based on the main power supply, the smart mobile devices and gadgets are battery-operated, thus making power a constraint (Daponte et al., 2013). Power awareness conserves energy by reducing activity or shutting down applications in the event of low power (Cioara et al., 2011). There is a need to monitor the power consumption characteristics of a software module so that conservation mechanisms can be devised (Mahmud et al., 2018; Schaarschmidt et al., 2022).

At the heart of smart personal space is a smart mechanism that gathers data from sensors present in the environment, classifies it based on history, and discovers as well as adapts the services present in the environment (Heyn et al., 2022; Mahmud et al., 2007b). This mechanism is essentially a classification-based algorithm termed context awareness (Kiani et al., 2013; Knappmeyer et al., 2013).

The information captured in a smart environment is leveraged to expedite the discovery, provision, and adaptation of smart services. (Schilit et al., 1994; Mahmud et al., 2020). This information is extensive and can be used to distinguish an activity which is termed context awareness (Abowd et al., 1999; Mahmud and Javed, 2012). The context is gathered, tweaked,

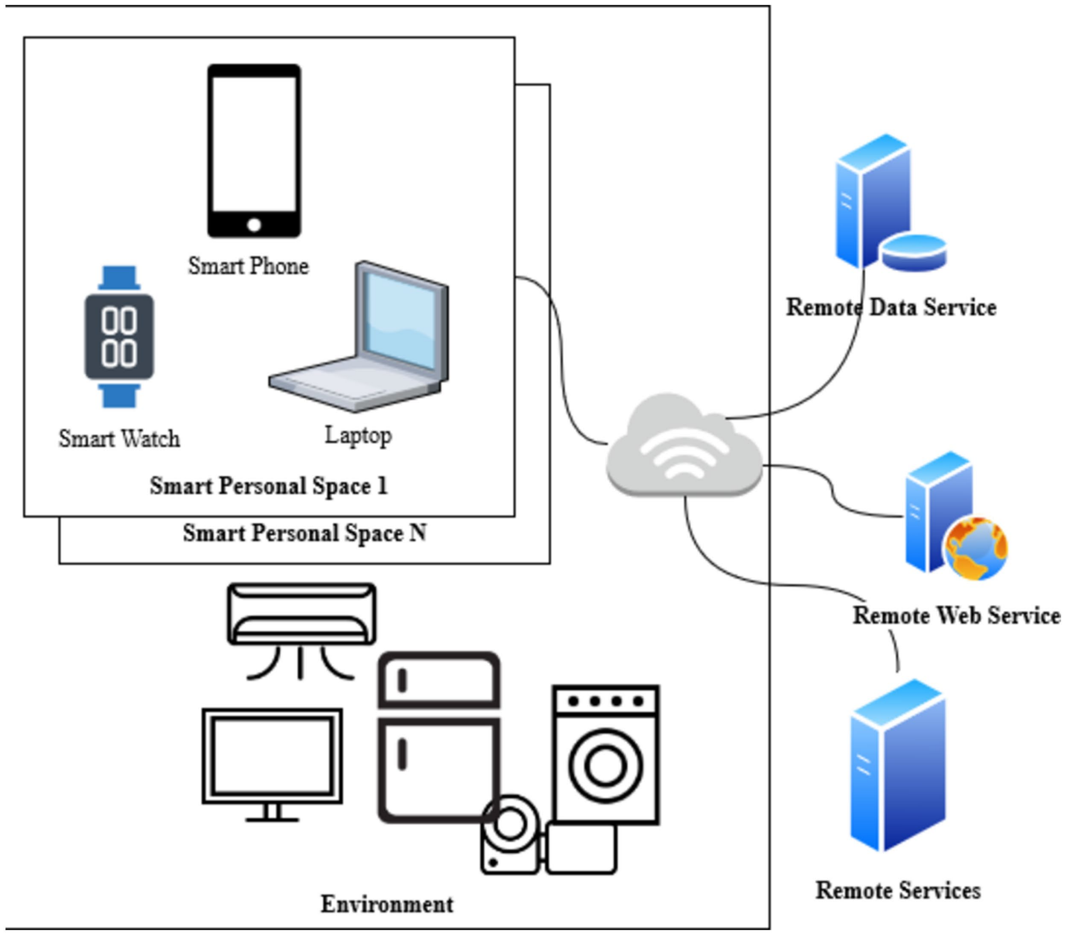

FIGURE 1
A smart environment.

recorded, and then deduced. With the use of machine learning algorithms, the context is classified into activities (Mahmud et al., 2007a; Mahmud and Javed, 2014; Mahmud, 2016). This is termed Context Processing (Mahmud et al., 2022b). Context Processing entails appropriate activities to adapt the services (Augusto et al., 2022). For instance, a smartphone may adjust its orientation according to the user's orientation or adjust the screen brightness based on the surrounding environment. An assortment of attribute-value pairs has been used to model the context. These are XML, RDF, and OWL, which are examples of open web standards.

The capacity of a computer system to provide the same service with a lower performance level while preserving power is known as power awareness (Mahmud and Hussain, 2024). Since power conservation is not considered in efficiency metrics, an efficient system may not be power-aware. In a smart environment, power awareness can also be seen as a cost-of-context (CoC) indicator (Jagarlamudi et al., 2021). Power is a cost attribute that can be used to negotiate Service Level Agreements (SLA) and should be included as a CoC (Mahmud and Hussain, 2024). Power Awareness has two distinct phases Power Profiling and Power Conservation. In Power Profiling, power consumption is monitored which can be used for Power Conservation. Power consumption is a leading issue in reducing the carbon footprints of cryptocurrencies (Kohli et al., 2023). A single sensor that consumes 1 mW of energy can take up to 1,370 mW of energy once the data sensed by the sensor is processed (Yuryur, 2013). This highlights the power inefficiency of battery-operated devices. Each device has multiple embedded sensors that are energy-inefficient once the sensed data is processed. Furthermore, the interaction of these battery-operated devices with each other as well as remote services within a smart environment consequently makes a completely smart environment as energy inefficient. In addition to energy inefficiency during the processing of sensor data, energy is also consumed during communication. Battery-operated smart devices communicate wirelessly and suffer from multiple retransmissions in inherently noisy wireless networks (Sangeetha et al., 2023; Goudarzi, 2022). This further highlights the need for energy conservation in smart environments that gather sensor data and process it. Specifically, in the healthcare sector where sensors as part of Internet-of-Medical-Things (IoMT) monitor patient health and process it to generate alerts and raise alarms. This need is also evident due to the deployment of IoT-based quarantined and remotely located patient monitoring systems (Loke et al., 2023).

While context-aware systems are deployed on battery-operated devices and utilize state-of-the-art machine learning algorithms to classify the context, it is still power inefficient. The sensor data and its subsequent processing make a context-aware system energy inefficient. A context-aware system does not consider power as a constraint. PCCA aims to enhance a context-aware system to enable energy conservation while enabling context awareness. For this purpose, it is necessary to identify the module that consumes more power in a standard context-aware system. Since energy consumption is a function of time, it follows that the module that processes data would have a higher energy consumption (Mahmud and Hussain, 2022a). Given that the processing duration is reduced, the energy consumption of the system can be reduced albeit reduced accuracy.

Since context processing involves running a classification algorithm and assigning a classification label to the current context, it is natural to consider if the classification outcome can be predicted, there would be no need to execute the classification processing phase of a context-aware system. PCCA aims to enhance a context-aware system to enable power conservation while enabling context awareness in smart environments by utilizing pre-classification using Markov Chains. The contributions of this work are listed below:

- The authors present a Power-Constrained Context-Aware System (PCCA) that uses the Markov Chain to predict the activity label of the current context. The paper does not depend on the actual classification algorithm used by a context-aware system but focuses on how to classify the context while conserving power.
- An algorithm to implement PCCA establishes pre-classification using Markov Chains. The evaluation is carried out using two datasets and shows promising results.

The rest of the paper is organized as follows. Section 2 presents the related work, Section 3 presents the model of a context-aware system, which is then improved as an energy-conserved context-aware system in Section 4. The results and discussion are presented in Section 5 and the paper concludes in Section 6.

2 Related works

The advent of Industry 4.0, LTE, 5G, and Web 3.0, and an exponential increase in the production and availability of smart devices have paved the way for smart environments. While the increase has enabled flexible functionality, it comes with a cost of power conservation and increased carbon emissions. In 2013, in the IT industry, approximately 3 billion personal devices consumed up to 1–1.5% of the total power, generated by the world (Stauffer, 2013). In 2020, the number of smartphones was 6 billion, which is a fraction of the total number of devices in use (Li et al., 2021). The consumption is forecasted to be 20% of the total power generation with 50 billion devices by 2030 (Enerdata, 2018; Cisco, 2016). Smart devices are primarily battery-operated and battery life is among the leading concerns of a potential buyer (Gupta, 2011). Souri et al. (2019) have compared different IoT-based communication mechanisms and identified that the power consumed by communication is up to 55% energy in smart environments.

At the core of a smart environment is a context-aware system, This system accesses sensors and services and gathers contextual information, processes it, and classifies its activity (Hashemi and Sadeghi-Niaraki, 2016; Alegre et al., 2016). A context-aware system communicates via wireless links and processes the context using machine learning methods (Ogbuabor et al., 2022; Mahmud et al., 2018; Rana et al., 2020).

Power-aware computing is the ability of a computing machine to operate and provide services while conserving power and reducing carbon emissions. This includes measuring power consumption, predicting battery life, conserving power to provide acceptable service, and increasing battery life. Power awareness is a concept that governs battery-operated devices including smartphones, Battery Electric Vehicles (BEV), mobile traffic signals, quadcopters as well as drones, and missiles (Cao and Yang, 2019; Martyushev et al., 2023). The emergence of smart spaces characterized by the recurrent interaction of small, inexpensive, mobile, battery-operated devices with remote services has led to the urgency of power awareness.

The related work has been organized into different categories as shown in Sections 2.1–2.7.

2.1 Hardware-supported power conservation

Power awareness is realized at all layers of a computer. Starting at the architectural layer to the OS level as well as the application level. Energy-efficient hardware facilitates power conservation during operation. Esquicha-Tejada and Pineda have recently identified NodeMCU and SinricPro as energy-efficient hardware for home automation. This hardware can conserve up to 30% of power yearly (Esquicha-Tejada and Copa-Pineda, 2022). Castillo-Atoche et al. (2022) have designed and developed an Energy harvesting (EH) circuit that uses Convolutional Neural Networks (CNN) to detect cardiac arrhythmia using wearable devices in competitive sports.

Hao et al. (2013) have implemented a power conservation model in mobile devices at the instruction level. Tyagi et al. (2016) have developed an Energy-Efficient Language (Eel), that carries out a multistep compilation to conserve power. This system uses reversible algorithms that have near-zero power waste. Qasim et al. (2021) have compared different design patterns and have found the observer pattern to be power efficient. Pereira et al. (2017) have compared 27 different programming languages for power conservation and have found C-type languages to be the most energy-efficient.

2.2 Threshold or activity based power conservation

Within the kernel, the OS can shift to power-saving mode as soon as a threshold is achieved. Among other methods is to put the communication modules into sleep mode and reduce the brightness of the screen. Dai et al. (2021) have proposed the use of dynamic scheduling using a knapsack algorithm for power efficiency in vehicular networks. Safara et al. (2020) have proposed an energy-efficient routing protocol to reduce power consumption during communication.

Galeana-Zapién et al. (2014) implement an adaptable sensor sampling rate to gather contextual data using internal sensors and sensor services. The sample rate is based on how a mobile user moves around. For instance, if a user is driving to work, their context is likely to remain that way until they get to their destination safely.

Diwan (2022) has developed a threshold-based underwater system for charging remote devices in the Internet of Underwater Things (IoUT). This technique provides a threshold for power conservation and no adaptability is provided.

Elmalaki et al. (2014) propose a battery monitoring mechanism that adjusts the device power settings based on device activity, (Elmalaki et al., 2015). A similar approach has been selected by Anastasi et al. (2015) and Flinn and Satyanarayanan (1999) as well. Zhuang et al. (2010) suggest substituting direct sensor access with network-based sensor access can reduce power. The onboard sensors can then be utilized if the network location mechanism is used, albeit accurate location. Kim et al. (2011) have developed WiFiSense for power consumption by intermittent sensing of Wi-Fi modules. This mechanism uses user activity as a basis for decisions.

2.3 Context change-based power conservation

A proactive monitoring system based on context change has been presented by Kang et al. (2010). Each sensed data is processed continuously to classify context, which might be the same as the last well-known classification. For example, if a person is traveling to work, then the context is not likely to change unless the person arrives at the destination or meets an untoward incident. Kang et al. (2010) propose that while sensing may be carried out continuously, context processing may be suspended unless a change in context is expected.

2.4 Statistics-based power conservation

Sathan et al. (2009) have implemented statistics to poll sensors. According to Sathan, sensors can be called upon when needed and in a non-contiguous manner if the change in the sensor can be predicted. A prediction engine will be added to the sensor layer, which sits between the sensors and the context layer, as an innovation. Due to context processing, handling sensor polling based on statistics at the bottom layer decreases power usage.

2.5 Machine learning supported power conservation

Makhadmeh et al. (2021) have used machine learning and used Swarm Intelligence to optimize battery usage. Tong et al. (2021) have implemented a hierarchical service model for energy efficiency. Chen et al. (2020) have used multilevel task offloading, i.e., to Edge Fog, and then to Cloud in Unmanned Aerial Vehicle (UAV) systems.

Zappi et al. (2008) implemented the Hidden Markov Model (HMM) to classify activity and recommend that power scavenging by using minimal sensors is advantageous. However, this can result in a decrease in accuracy. Hermann et al. (2012) propose a minimal sensor pool for current context classification to conserve power. If there are mismatches in classification, more sensors are added. This mechanism, while power efficient, takes more time in the classification phase.

Mansouri et al. (2018), have used Ant Colony Optimization to reduce the energy consumed by cloud-based applications. This system does not view the smart environment perspective and uses a complex technique to reduce energy consumption.

Moghimi et al. (2012) have used fuzzy logic to implement a Power Conserving Manager for smartphones. This manager infers the sampling rate of the sensors to conserve power.

Yuryur (2013) has developed a system to classify the situation by developing a Reward Process. This mechanism also monitors data continuously and triggers when a context change is detected.

2.6 QoS-based power conservation

Fei et al. (2004) implemented a software-controlled dynamic power management system. This method reduces QoS effectiveness to save power. Flinn has proposed battery optimization using software methods that can increase battery life (Flinn, 2001). Çiçek and Gören (2021) have used the ConvLSTM model for battery life enhancement using time-variant power information.

Reffad and Alti (2023) have proposed a cooperative energy-saving mechanism that considers energy as a QoS parameter, in Industrial IoT (IIoT) environments. The energy-saving process is a result of QoS parameters that can be configured prior to communication.

2.7 Task offloading-based power conservation

Alti et al. (2016), have included the battery level as part of the context of a smartphone, and have enabled task offloading to the cloud as a means to achieve energy conservation. Xia et al. (2021), have energy harvesting in mobile-edge computing environments through distributed task offloading.

Kiani et al. (2014) and Chen et al. (2020) have both proposed the use of the cloud as an alternative to conserve power. Kök and Özdemir (2022) have used Deep Reinforcement Learning (DRL) to support task offloading and save energy consequently. The use of the cloud increases the load on communication (Shahryari et al., 2021). Simoens et al. (2016) have introduced the Internet of Robotic Things (IoRT) where the computation is offloaded to fog rather than the cloud to increase the battery life of robots.

2.8 Comparison of evidence in literature

Power consumption is directly dependent on the execution time of an algorithm (Mahmud and Hussain, 2022b). This means that an algorithm with higher time complexity would consume more power. Algorithms having linear and sublinear time complexity are greener in general. A comparison of the related work is shown in Table 1. This comparison is based on context awareness, time efficiency, use of complete sensors for rich context, high accuracy of context classification, continuous polling of sensors, use of historical information for pre-classification, and task offloading to the cloud. A context-aware system that is power conserving, time efficient, uses rich context, considers all sensor data, does not poll sensors, continuously, uses history information for pre-classification, and does not offload tasks to the cloud is envisaged.

As reported by Alsharif et al. (2024), there is a lack of work on energy efficiency at all three levels, i.e., edge, fog, and cloud of a smart environment. Coupled with context-aware systems, evidence of energy-efficiency is hardly available.

3 Components of a context-aware system

The brain of a smart environment is a context-aware system (Malik et al., 2007). The system receives contextual information, analyses it, and assigns it an activity label (Hashemi and

TABLE 1 Comparison of related works.

Related work	Context-aware system	Power conserving technique	Time efficiency	Use of complete sensors for rich context	High accuracy of context classification	Continuous polling of sensors	Use of history information for pre-classification	Task offloading to the cloud
Reffad and Alti (2023)	✗	Multi-objective QoS and Energy-Saving	✗	✗	✓	✓	✗	✗
Esquicha-Tejada and Copa-Pineda (2022) and Castillo-Atoche et al. (2022)	✗	Power conserving hardware	✓	✗	✗	✗	✗	✗
Dai et al. (2021)	✗	Dynamic Scheduling	✗	✗	✗	✗	✗	✗
Xia et al. (2021)	✗	Distributed Task Offloading	✓	✗	✗	✗	✗	✓
Safara et al. (2020)	✗	Efficient Routing	✗	✗	✗	✗	✗	✗
Mansouri et al. (2018)	✗	Ant Colony Optimization	✗	✗	✓	✗	✗	✗
Alti et al. (2016)	✓	Threshold Based	✓	✗	✗	✓	✗	✓
Kang et al. (2010) and Yuryur (2013)	✓	Context-change detection	✓	✓	✓	✓	✗	✗
Sathan et al. (2009)	✓	Statistics-based sensor polling	✗	✓	✓	✗	✗	✗
Zappi et al. (2008), Galeana-Zapién et al. (2014), and Hermann et al. (2012)	✓	Minimal sensor polling	✗	✗	✗	✗	✗	✗
Diwan (2022)	✗	Threshold-based	✓	✗	✗	✗	✗	✗
Elmalaki et al. (2014), Elmalaki et al. (2015), Anastasi et al. (2015), Flinn and Satyanarayanan (1999), Zhuang et al. (2010), and Kim et al. (2011)	✗	Device-activity detection	✓	✗	✗	✗	✗	✗
Moghimi et al. (2012)	✗	Fuzzy logic	✓	✗	✗	✗	✗	✗
Fei et al. (2004), Flinn (2001), Çiçek and Gören (2021), and Makhadmeh et al. (2021)	✗	Battery-optimization	✓	✗	✗	✗	✗	✗
Kiani et al. (2014), Chen and Venkataramani (2016), Kök and Özdemir (2022), Shahryari et al. (2021), Simoens, et al. (2016), Tong et al. (2021), Chen et al. (2020)	✓	Task-offloading	✗	✓	✓	✗	✗	✓
Hao et al. (2013), Tyagi et al. (2016), Qasim et al. (2021), Pereira et al. (2017)	✗	Course-code optimization	✓	✗	✗	✗	✗	✗
Proposed work (PCCA)	✓	Markov Chain based Pre-classification	✓	✓	✓	✗	✓	✗

Sadeghi-Niaraki, 2016; Alegre et al., 2016). A context-aware system communicates with sensors and services in a smart environment, and the context process contains five components, as shown in Figure 2. The modules are Service Adaptation, Context Aggregation, Context Representation, Context History, and Context Inference. These modules are described in Sections 3.1–3.5.

3.1 Context aggregation module

The context is collected and aggregated from remote services and intrinsic sensors as attribute-value pairs (Mahmud et al., 2007a). The sensors provide sensing data that can be used to deduce information through remote services. For instance, a GPS sensor might deliver latitude, longitude, and altitude as sensed data. This data can then be utilized to check the weather and location using remote services, resulting in the deduction of further data. The entire set of information is then regarded as contextual data. To distinguish between identical context values with differing attribute labels, a corresponding controlled vocabulary can also be offered. This module uses both I/O and communication modules to gather data and noisy networks preclude its performance.

3.2 Context representation module

The gathered contextual data needs to be stored in a way that allows quick access as well as interoperability if it needs to be shared (Mahmud et al., 2012). This when represented in a suitable format, is termed the current context. Various mechanisms have been proposed in the literature, however, using ontologies as an extension of eXtensible Markup Language (XML), supports both quick access and interoperability as well as provides a means to develop rich context representations (Mahmud, 2015; Mahmud, 2016).

3.3 Context history module

This module interacts with the Context History which has the history of contextual data and associated activity labels stored using SQL or no-SQL mechanisms (Malik et al., 2009). This history is accessible via a POD and can be stored remotely. The main purpose of history is to provide support to machine learning-based context processing that is carried out in the Context Inference Module. History information can also be used for user pattern prediction as well as user preference measurements. This module needs to be secured with attacks on privacy and ownership should be ensured (Malik et al., 2008; Mahmud and Malik, 2014).

3.4 Context inference module

This module is the brain of a context-aware system, it processes context and classifies activity. This module uses context history in either supervised or unsupervised ways and can also use optimization techniques to provide accuracy. This module can also be used to infer situations. This module is computation-heavy and takes more time depending on the size of the history and the performance of the machine learning algorithm used.

3.5 Service adaptation module

This module is used to interact with surrounding appliances as well as remote services for service negotiation or adapted services. This module aids in service discovery as well as delivery to the user based on QoS requirements (Hussain et al., 2014; Hussain et al., 2013). This ensures that services are delivered based on the Service Level Agreements (SLA) (Kouamé et al., 2020).

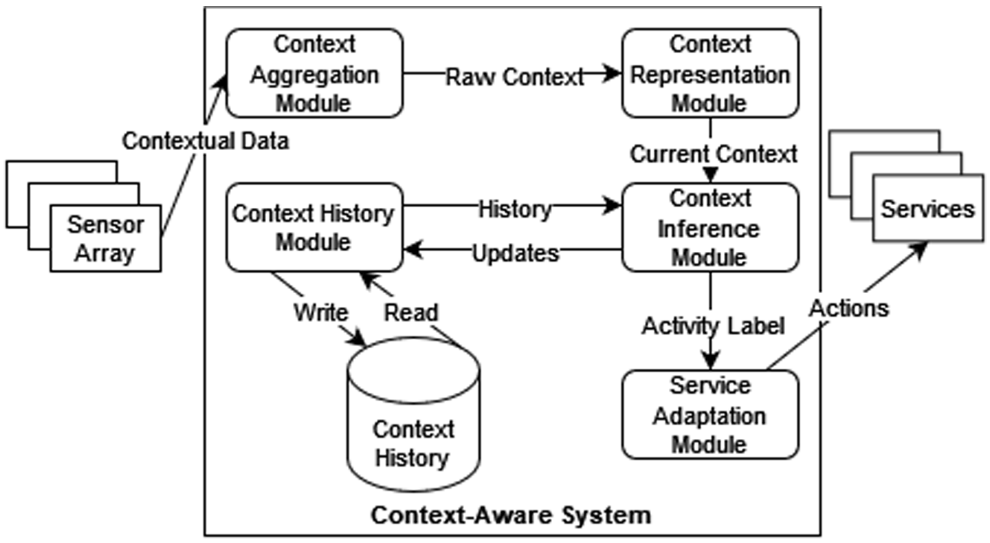

FIGURE 2
Components of a context-aware system.

4 Design of PCCA

The PCCA introduces a new module termed the Markov Chain Module, which is responsible for finding the probability transition matrix for the context history within the context-aware system.

The context process is viewed as a stochastic process in this paper. The fact that the context changes over time and has a probability makes it a stochastic process. Since the context outcomes are in discrete states, the use of a Discrete Time Markov Chain (DTMC) as a realization of a stochastic process is suitable. The DTMC-based perspective is the simplest explanation as required by Occam's Razor. Furthermore, it can be seen in the literature that while context change-based detection mechanisms have been used, they could be implemented much better using a DTMC-based mechanism. Furthermore, the mechanism proposed in this paper does not continually poll sensors, does not offload tasks to the cloud, and considers the history as well as a rich context for power conservation.

At first glance, it appears that the system is subject to more processing as a new module is added, however, this new module helps in reducing the power by reducing the use of the Context Processing Module. Figure 3 shows the design of PCCA as a block, as an improvement of the Context-aware system shown in Figure 2. The tasks of these components are discussed in Sections 4.1–4.5. Figure 4 shows a flow model of the working of PCCA. The flow model shows the generalized view of PCCA, where the context processing is carried out once the predicted state is not the desired state.

In that case, the context is gathered and then processed normally. Table 2 presents the algorithm of PCCA. The algorithm presented in Table 2 is shown visually as a flow model in Figure 4. The algorithm starts with acquiring the context history and constructing the Markov Chain. All the records in the context history are used to construct the Markov Chain. This is carried out by sorting the data time wise and calculating the change in context state for each context state, and recording in a contingency matrix, which is then converted to a probability transition matrix representing the Markov Chain. The algorithm then gathers the latest context and pre-classifies using the Markov Chain, in the event of a miss-classification, the process context is processed much like a standard context-aware system.

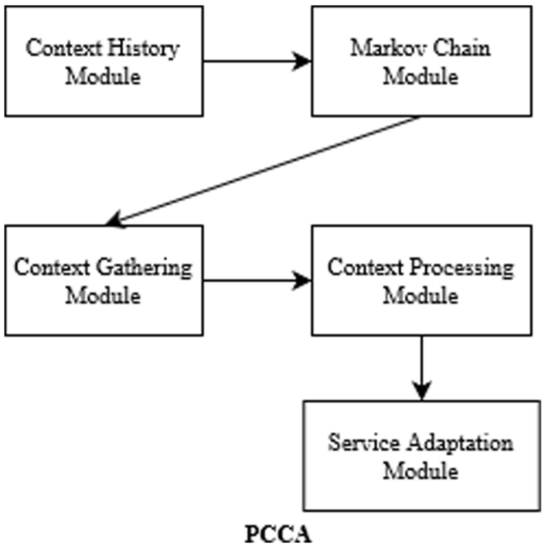

FIGURE 3
Design and components of PCCA.

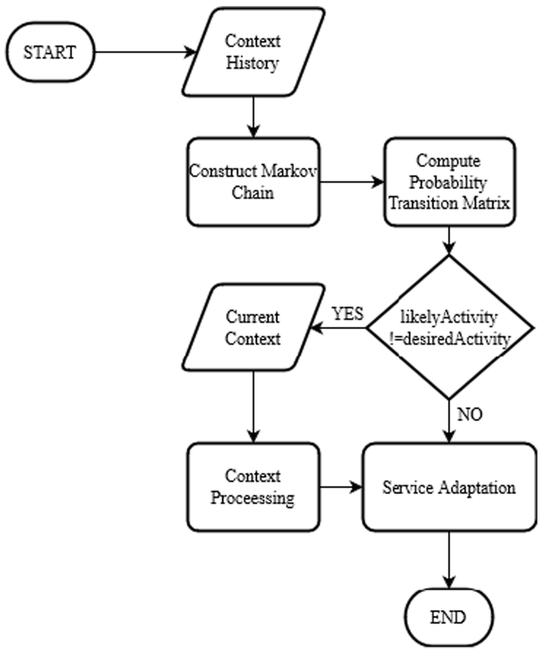

FIGURE 4
Flow model of PCCA.

4.1 Context history module (CHM)

This module contains the history of contextual data and the activity labels associated with them. It is stored as database tables; however, open standards can also be used to store history information. This module is used for two purposes. First, the Markov Chain is constructed using the history information. Secondly, historical information can be used in Context Processing, a technique that uses prior information is employed. For the evaluation of PCCA, sample data is used as the history information in this module. This Module corresponds to the Context History Module and Context Representation Module of a context-aware system as shown in Figure 2.

4.2 Markov chain module (MCM)

This module constructs the Markov Chain and an associated probability transition matrix based on the history information. This module executes before gathering the current context and context processing. A probability transition matrix is computed using the Markov Chain. The Markov Chain is constructed using the algorithm presented in Table 2. For evaluation, the Markov Chain uses the sample data as provided by the Context History Module. The inclusion of a new layer though seems to take more time as a new phase is introduced, but a correct pre-classification due to MCM bypasses CPM and SAM and reduces the overall time elapsed as well as overall energy consumption during context processing. The MCM calculates the probability transition matrix, based on the training data. Given noise

TABLE 2 Algorithm to implement PCCA.

Step #	Pseudocode
1	File SampleDataFile;
2	double[] currentContext;
3	double[][] historyOfContext;
4	int[] activityLabels;
5	int lastCol; //the column that holds the activity labels in history
6	int states[]; //stores the unique activities as states
7	int probmatrix[][];
8	int main(){
9	for (int i = 0; i < SampleDataFile.getRows(); i++){
10	historyOfContext[i][0] = SampleDataFile;
11	}
12	for (int i = 0; i < historyOfContext[][].length(); i++){
13	activityLabels[] = (int) historyOfContext[i][*lastCol*];
14	}
15	constructMarkovChain();
16	executePPCA();
17	}
18	private int. getRows(){
19	int count = 0;
20	While (!EOF){
21	if (SampleDataFile.readLine()! = NULL)}
22	count++;
23	}
24	}
25	return count;
26	}
27	private void constructMarkovChain(){
28	int stateCount[states.lnegth()][states.length()];
29	int initialState = −1;
30	for (int i = 0; I < activityLabels.length(); i++){
31	for (int k = 0; I < activityLabels.length(); ik++){
32	if (activityLables[i] == states[0]){
33	stateCount[i][k]++;
34	}
35	else if(activityLabels[i] = states[1]){
36	stateCount[i][k]++;
37	}
38	else {
39	error.log(error);
40	}
41	}
42	}
43	int totalCount = −1;
44	for (int i = 0; i < stateCount.length(); i++){
45	totalCount+ = stateCount[i];
46	}
47	probMatrix[stateCount.length()][stateCount.length()];
48	for (int i = 0; i < probMatrix.length(); i++){
49	for (int k = 0; k < probMatrix.length(); k++){
50	probMatrix[i][k] = stateCount[i][k]/totalCount;
51	}
52	}
53	}
54	private void executePCCA(){
55	testState = currentContext[][*lastCOl*];
56	previousState; //last known state, could be random for testing
57	int maxProb = −1.0;
58	int likelyState;
59	for (int i = 0; i < states.length(); i++){
60	if(probMatrix[previousState][i] > maxProb){
61	maxProb = probMatrix[previousState][i];
62	likelyState = states[i];
63	}
64	}
65	if (likelyState == testState){
66	Print("Success")
67	}
68	}

in data, the MCM would suffer from overfitting. This can be overcome by recomputing the probability transition matrix once the effectiveness is reduced.

4.3 Context gathering module (CGM)

This module accesses both internal sensors as well as remote services to construct the current context. The current context is the query that needs to be assigned an activity label after context processing in a context-aware system. To evaluate PCCA, a sample query is used as the current context to simulate the context-gathering process. This module corresponds to the Context Aggregator Module as shown in Figure 2. This module consumes power because of accessing remote services, however, not using remote services may lead to missing contextual data and subsequently reduce the accuracy of the context processing technique.

4.4 Context processing module (CPM)

The Context Processing Module is the heart of a context-aware system that corresponds to the Context Inference Module of a context-aware system as shown in Figure 2. This module uses machine learning algorithms that can utilize supervised, unsupervised, or reinforcement learning to deduce an activity label to the current context. Most implementations use prior information which is provided through the

Context History Module. This module performs processing and depends on the size of the prior information as well as floating point calculations as part of the algorithm that is implemented. This module consumes the most power due to computation and reducing power consumption is necessary here.

4.5 Service adaptation module (SAM)

This module corresponds to the Service Adaptation Module in Figure 2 and is responsible for performing recommended actions based on the activity classification by the Context Processing Module. This module can interact with remote services which can lead to higher power consumption. In addition, based on the context, the context-aware system can be put to sleep until the next observation is due.

5 Results and discussion

To check the effectiveness of PCCA, multiple datasets have been selected for context processing in addition to the dataset collected by the app PowerIpsum (Mahmud and Hussain, 2022b). No special machine learning algorithm is preferred for context awareness in this work since the purpose is not to establish a green algorithm but to ensure greenness while any algorithm is used. This means that the technique presented in Table 1 is appropriate for all context-aware systems. The algorithm presented in Table 1 is implemented using Java language. The code is executed for two different datasets. The algorithm assumes that the states or activity labels are in integer form in the history of data and the test data is converted to unique numbers if required.

Evaluation metrics: A major issue exists that the datasets available in the UCI Machine Learning Repository lack power information. Based on the sensor data and deduced data, the available datasets are used to classify activities. This makes it difficult to measure the performance of PCCA against an established benchmark. Hence, there is a need to set up a mechanism that shows the effectiveness of power conservation of context-aware systems. A ratio of correct context classification with and without power conservation can be used as a yardstick. The ratio is termed effectiveness (η) and is shown in Equation 1.

$$\eta = \frac{Power\ consumed\ in\ correct\ context\ classification\ by\ PCCA}{Power\ consumed\ in\ correct\ context\ classification} \quad (1)$$

It is natural to observe that the power consumed by using a conservation mechanism would always be lower than the power consumed without using a conservation mechanism. A smaller value of η, shows that the power conservation mechanism has a higher efficiency.

The choice of metrics is inspired by the concept of speedup when justifying parallel processing over scalar processing. The effectiveness metrics identify how much power has been conserved as a ratio. The mechanism essentially calculates the power conserved for a single classification and can be extended to reflect throughput. In contrast with other metrics including accuracy and precision, energy will be consumed more in the event of inaccurate pre-classification, and thus effectiveness becomes a logical choice. The metric given in Equation 1 does not address energy conservation as throughput, however, measurements overtime can quantify effectiveness as throughput.

Datasets for experimentation: The tests are carried out for each dataset when using PCCA as well as without PCCA. The results are recorded and include time elapsed, power, and energy consumed. The results are then subject to the effectiveness measure given in Equation 1. For a single experiment, the size of the sample data is fixed, and the time elapsed as well as power consumption is measured for all modules of a context-aware system as shown in Figure 2. Furthermore, the memory utilized by each phase is also recorded. The test is run for increasing sample set size, and a query instance is classified. Two tests are done for each sample set, one for correct and one for incorrect query classification. The query is considered as the current context.

The HARUS dataset contains mundane activities carried out while wearing a sensor-mounted watch (Anguita et al., 2012a; Anguita et al., 2012b). The dataset contains six activities with 30 volunteer subjects, such as walking, climbing, laying down, etc. Over 10,000 instances of 561 attributes make up the dataset. The values in the data set are constrained within the range $[-1, 1]$ and are normalized to six decimal places.

The tests are re-executed on the dataset collected by the app PowerIpsum (Mahmud and Hussain, 2022b; Mahmud et al., 2022a). This dataset gathers contextual data using smartphones. The contextual data pertains to daily mundane activities. Table 3 shows the test outcomes. This dataset has 35 attributes and multiple tests including correct and incorrect classification have been executed.

Experimental setup: The test machine is a 32-bit HP Mini 110, with Intel Atom N2600 and 1 GB DDR, running Ubuntu 16.04.6 LTS. This machine has Java 10 installed. This machine is selected because it takes more time to compute and consumes more power and is suitable for measuring power consumption through software. A 32-bit machine is slower than a 64-bit machine, and it is necessary to use a machine for which measurement could easily be profiled. The major issue with 64-bit machines is the low sensitivity of software-based energy consumption estimation. This could only be overcome through the use of external hardware support, which would increase the footprint of smart devices. While the results can easily be shown using a 32-bit machine, the profile would remain the same for 64-bit machines. 64-bit machines would take less time to compute for the same training and test set, but the construction of the Markov chain would remain the same, thus reducing the effect of architecture on the measurement of effectiveness. The variability of hardware, including the type of architecture has not been studied in this paper however, different architectures would lead to the same outcome for the same training set and Markov chain, albeit the values of energy consumption and time elapsed would be different.

In each experiment, the size of the sample size is kept constant as the effect of size has been discussed in Section 5.4. Each experiment has 3 tests, 1st when the non-PCCA system correctly classifies the query, 2nd in which PCCA correctly classifies the query, and 3rd when PCCA is unable to correctly classify the query. There is no need for incorrect classification of non-PCCA systems as the total time elapsed and the energy spent remains the same. The test results and corresponding graphs are discussed below. The training set has been selected and inspected to remove missing attributes and is thus not affected by sparse data. For the construction of the Markov chain, no normalization is required as the Markov chain only uses the outcome labels to construct the probability transition matrix. Furthermore, the size of the Markov chain is dependent on the number of distinct outcome labels in the dataset. A larger Markov chain would take more time to construct. In addition, the size of the training data also affects the calculation of the probability transition matrix and the construction of the Markov chain. A large size of training data as part of the context history would entail more time in constructing the chain. However, the chain is only re-computed once the effectiveness is reduced.

5.1 Test results using HARUS dataset

Table 3 shows the outcome of each experiment conducted on the HARUS dataset. For each experiment it is observed that the power is consumed at the time a phase executes, hence it is more useful to compute the energy consumed rather than power. The energy is computed in Wsec for ease of calculation. Figure 5 shows the Markov Chain for the uniformly selected sample set used for the experiment belonging to the HARUS dataset (Feng, 2017). Equation 6 shows the steady state vector for the sample set uniformly selected from the PowerIpsum dataset. The steady-state vector shows that there is a higher probability of Working in the next observation.

It can be seen that there are six states with W denoting Walking, W-D denoting Walking Downstairs, W-U denoting Walking Upstairs, L denoting Laying, Sit denoting Sitting, and Std denoting Standing. Equation 2 shows the probability transition matrix (P) for the.

Markov Chain is shown in Figure 5. Figure 5 has been constructed using an HTML-based Markov Chain Visualizer developed by Feng (2017). This tool takes the transition matrix as an input and plots the Markov Chain in different colors. The purpose is to visually see the Markov Chain; however, all calculations are carried out using the transition matrix. Equation 3 shows the steady state vector for the sample set uniformly selected from the HARUS dataset. The steady-state vector shows that there is a higher probability of laying or standing in the next observation.

Figure 6 shows the plot of the time elapsed in each phase of a context-aware system for multiple tests, while Figure 7 shows the plot of the energy consumed over time for each phase of a context-aware system corresponding to the tests performed. It can be observed that the most time is spent in CPM and CHM. While the power consumption is measured during each phase, the energy spent in each phase shows that the CPM is the energy center of a context-aware system, and a reduction in time elapsed here would lead to power conservation. The effect of correct or incorrect classification on the time elapsed and the energy consumption is the same. It is also pertinent to see that the increase in sample size results in more energy consumption.

It is evident from Table 3, that when a non-PCCA based system is used the MCM is absent and no power is consumed in MCM, however, when a PCCA based system is used, MCM utilizes power, but if the outcome of MCM is a correct classification, the PCCA based system avoids three modules, i.e., CGM, CHM, and CPM. The overall power consumption is reduced as 3 modules are not executed which reduces the elapsed time. In Table 3, for a correct classification by using PCCA results in 27.61 Wsecs that are much <74.13 Wsecs, that are consumed by a non-PCCA system. It is pertinent to see that during an incorrect classification, the system consumes more energy than a non-PCCA system, i.e., 79.72 Wsecs.

$$P = \begin{bmatrix} 0.21 & 0.14 & 0.18 & 0.24 & 0.13 & 0.10 \\ 0.18 & 0.22 & 0.10 & 0.21 & 0.20 & 0.09 \\ 0.15 & 0.07 & 0.29 & 0.20 & 0.21 & 0.08 \\ 0.05 & 0.02 & 0.03 & 0.29 & 0.19 & 0.42 \\ 0.04 & 0.04 & 0.02 & 0.32 & 0.41 & 0.17 \\ 0.04 & 0.02 & 0.03 & 0.46 & 0.03 & 0.39 \end{bmatrix} \tag{2}$$

$$\begin{bmatrix} W \\ W-D \\ W-U \\ L \\ Sit \\ Std \end{bmatrix} = \begin{bmatrix} 0.07 \\ 0.04 \\ 0.06 \\ 0.33 \\ 0.19 \\ 0.31 \end{bmatrix} \tag{3}$$

Figure 6 shows that using PCCA not only reduces the time elapsed but also reduces the power consumption if PCCA is successful. However, the time elapsed and the energy spent is slightly higher if PCCA results in an unsuccessful case, which is also evident from Table 2. In Table 2, PCCA has reduced the time elapsed, energy spent by 2.6 times, and memory utilization by 2.1 times when compared with a non-PCCA system. The effectiveness of PCCA as given in Equation 4, can be measured as shown in Equation 1.

$$\eta = \frac{27.61}{74.13} = 0.37 \tag{4}$$

TABLE 3 Test results for PCCA and non-PCCA systems using HARUS dataset.

Test	MCM		CGM		CHM		CPM		SAM		Total Time Elapsed (sec)	Total Energy Consumed (Wsec)
	Time (Sec)	Energy (WSec)	Time (Sec)	Energy (WSec)	Time (Sec)	Energy (WSec)	Time (Sec)	Energy (WSec)	Time (Sec)	Energy (WSec)		
Correct classification non-PCCA	0	0	0.50	5.44	2.01	21.88	2.88	31.35	1.42	15.46	6.81	74.13
Correct classification PCCA	1.12	5.39	0	0	0	0	0	0	1.41	15.51	2.51	27.61
Incorrect classification PCCA	1.21	4.57	0.49	5.33	2.09	22.72	2.93	31.85	1.43	15.25	8.15	79.72

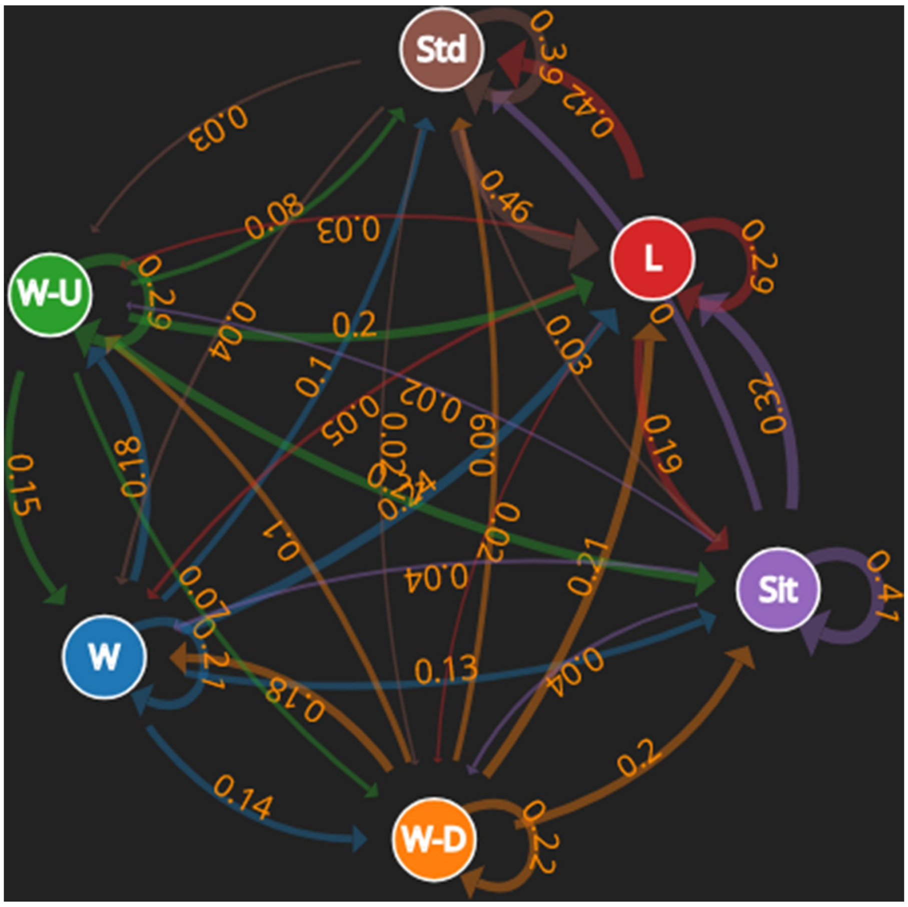

FIGURE 5
Markov chain for HARUS dataset.

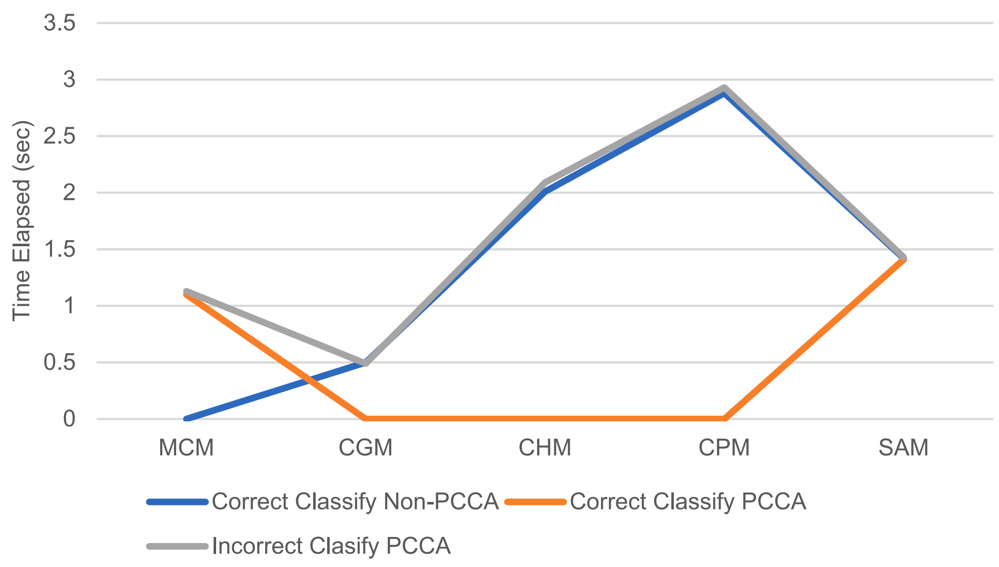

FIGURE 6
Time elapsed for PCCA and non-PCCA executions in HARUS Dataset.

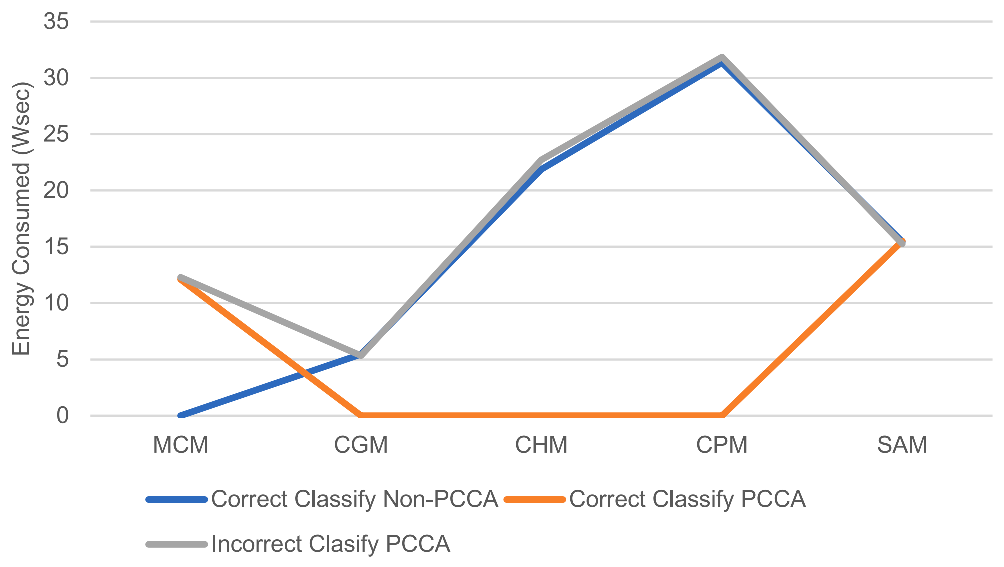

FIGURE 7
Energy consumed for PCCA and non-PCCA executions in HARUS dataset.

TABLE 4 Test results for PCCA and non-PCCA systems using Poweripsum dataset.

Test	MCM		CGM		CHM		CPM		SAM		Total time elapsed (sec)	Total energy consumed (Wsec)
	Time (Sec)	Energy (WSec)	Time (Sec)	Energy (WSec)	Time (Sec)	Energy (WSec)	Time (Sec)	Energy (WSec)	Time (Sec)	Energy (WSec)		
Correct classification non-PCCA	0	0	0.5	5.44	2.01	21.88	2.88	31.35	1.42	15.46	6.81	72.23
Correct classification PCCA	0.49	5.39	0	0	0	0	0	0	1.41	15.51	1.97	20.9
Incorrect classification PCCA	0.42	4.57	0.49	5.33	2.09	22.72	2.93	31.85	1.43	15.25	7.36	79.72

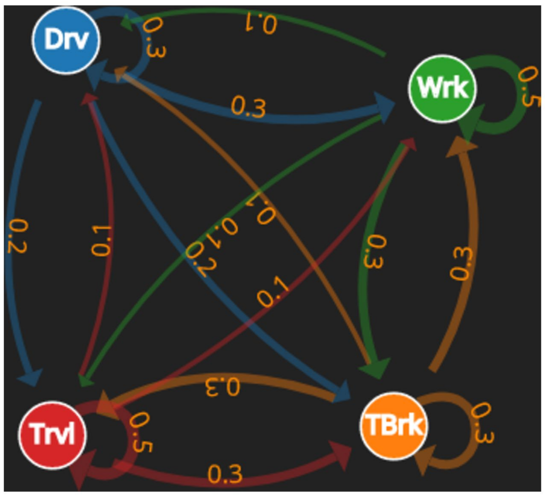

FIGURE 8
Markov chain for PowerIpsum dataset.

The effectiveness of PCCA for the HARUS dataset can be expressed as 37% for the case of correct classification by PCCA as well as non-PCCA systems.

5.2 Test results using PowerIpsum dataset

The test is repeated for the PowerIpsum test set and the test results are shown in Table 4. Figure 8 shows the Markov Chain for the uniformly selected sample set used for the experiments belonging to this dataset. There are four states with Drv denoting Driving, TBrk denoting Tea-break, Wrk denoting Working, and Trvl denoting traveling. Equation 5 shows the probability transition matrix (P) for the Markov Chain shown in Figure 8.

$$P = \begin{bmatrix} 0.3 & 0.2 & 0.3 & 0.2 \\ 0.1 & 0.3 & 0.3 & 0.3 \\ 0.1 & 0.3 & 0.5 & 0.1 \\ 0.1 & 0.3 & 0.1 & 0.5 \end{bmatrix} \tag{5}$$

$$\begin{bmatrix} Drv \\ TBrk \\ Wrk \\ Trvl \end{bmatrix} = \begin{bmatrix} 0.13 \\ 0.3 \\ 0.31 \\ 0.2 \end{bmatrix} \tag{6}$$

Figure 9 shows the plot of the time elapsed in each phase of a context-aware system for multiple tests, while Figure 10

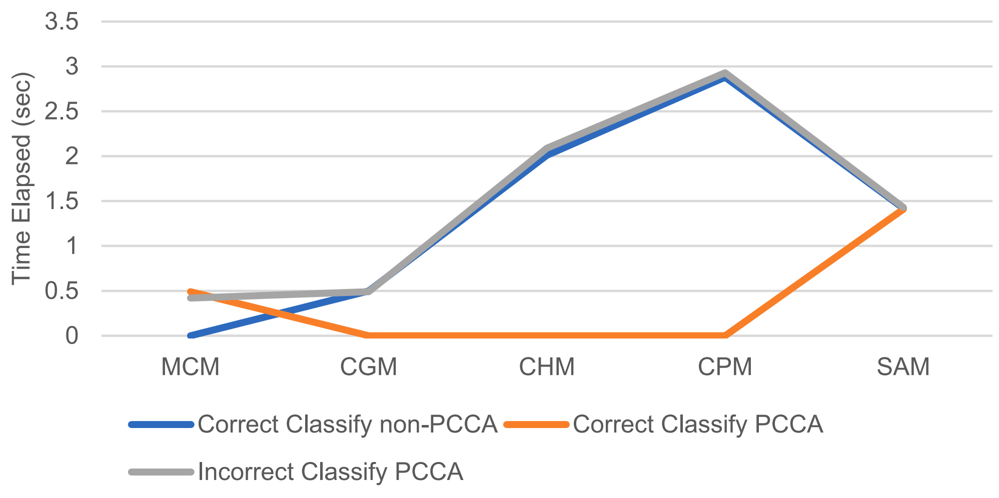

FIGURE 9
Time elapsed for PCCA and non-PCCA executions in PowerIpsum dataset.

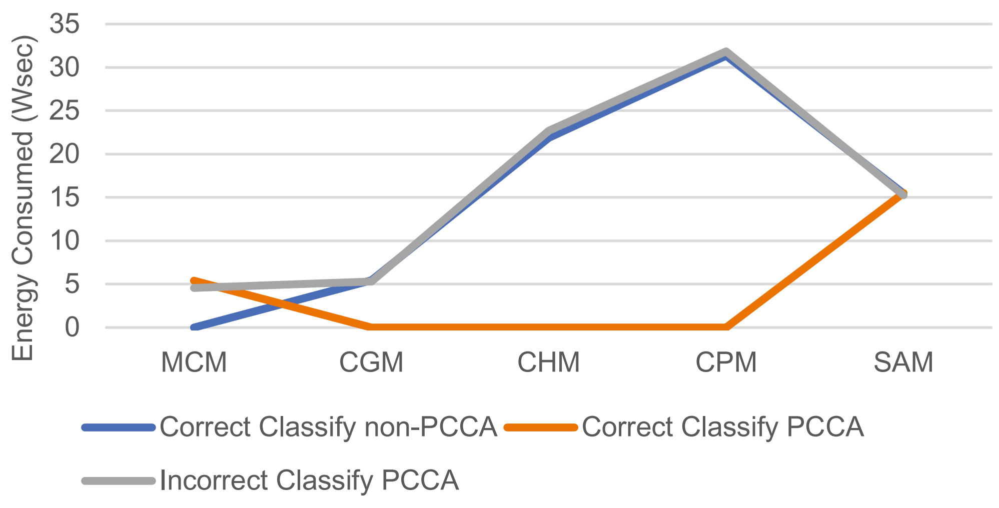

FIGURE 10
Energy consumed for PCCA and non-PCCA executions in PowerIpsum dataset.

shows the plot of the energy consumed over time for each phase of a context-aware system corresponding to the tests performed.

Like the results in Section 5.1, in Table 4, for a correct classification by using PCCA results in 20.09 Wsecs that are much <72.23 Wsecs, that are consumed by a non-PCCA system. In the case of incorrect classification by PCCA, the energy used is higher, i.e., 79.72 Wsecs. This shows that the PCCA system has similar performance across multiple datasets.

Figure 9 shows that using PCCA not only reduces the time elapsed but also reduces the power consumption if PCCA is successful. However, the time elapsed and the energy spent is slightly higher if PCCA results in an unsuccessful case.

In the results shown in Table 3, PCCA has reduced the time elapsed, energy spent by 3.45 times, and memory utilization by 2.2 times when compared with a non-PCCA system. The effectiveness of PCCA as given in Equation 1, can be measured as shown in Equation 7.

$$\eta = \frac{20.9}{72.23} = 0.29 \tag{7}$$

The effectiveness of PCCA for the PowerIpsum dataset can be expressed as 29% for the case of correct classification by PCCA as well as non-PCCA systems.

In both datasets, correct classification in a PCCA based system reduces elapsed time and consequently, the energy consumption. However, PCCA performs a pre-classification stage where a Markov Chain is constructed to predict the likely outcome. With the availability of large amounts of data and the repeating behavior of a person, the success of MCM would increase. This would decrease energy consumption over a period of time.

6 Conclusion

A context-aware system gathers the context using sensors and services via a smart device. This context is then processed, and the activity is classified. This activity can be used to discover, deliver, and adapt remote services. A context-aware system runs on battery-operated smart devices and is constrained by power usage. This opens an avenue to design and develop energy-conserving context-aware systems. The authors of this work present PCCA which views context awareness as a stochastic process and includes a Markov Chain Module in a context-aware system. This new module constructs a Markov Chain and predicts the activity label at the next observation. This prediction, if correct, can allow a context-aware system to defer context processing and deliver the same service while conserving energy. The algorithm for PCCA is tested using two different datasets. Each dataset is used to create test sets and plots of time elapsed, energy consumed, and memory utilized for both non-PCCA and PCCA systems. The effectiveness (η) is measured for each experiment and both datasets show promising results. With up to 3.5 times power conservation when using PCCA, the mechanism appears encouraging. The effectiveness is the percentage decrease in power consumption which is up to 37% based on the experiments performed.

The use of PCCA as a power-conserving context-aware system shows that for any application that facilitates human activity a pre-classification stage based on Markov Chains can reduce the processing time and hence energy would be consumed less. With a 3-fold reduction in power consumption by context-aware applications, the development of energy-conserving apps that run on battery-operated systems can be realized. When viewing a world of more than 8 billion people and up to 15.14 battery-operated smart devices, the decrease in power conservation of context-aware applications would lead to a substantially cleaner and more efficient future. Furthermore, since energy conservation is a consequence of the reduction in processing time, the context-aware system would become faster by the same factor.

However, the downside is that when PCCA makes an incorrect classification, it may consume more power than a non-PCCA system. The effectiveness depends on the Markov Chain constructed using the available sample set. The size of the sample set, and its uniformity can lead to higher success in classification using PCCA. In addition, PCCA is evaluated using a single-user perspective and does not cater to multiple personal spaces. The adoption framework of PCCA is also not explored in this paper.

7 Future work

The future work includes and is not limited to the deployment of PCCA on domain-specific smart environments including the Internet of Medical Things (IoMT), IIoT, and Internet of Social Things (IoST). The data can be recorded, and insights can be drawn based on the contextual data (Munoz-Arcentales et al., 2021).

The evaluation of the framework presented can be subject to hardware-based power consumption measurements. Power consumption and time elapsed can be sensed using sensitive onboard power chips as well as external measurement devices. Furthermore, future work includes the evaluation of PCCA under noise and for multiple personal spaces.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

UM: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. SH: Conceptualization, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. TK: Funding acquisition, Project administration, Resources, Software, Supervision, Validation, Writing – review & editing.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The author(s) declare that no Generative AI was used in the creation of this manuscript.

Publisher's note

References

Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M., and Steggles, P. (1999). "Towards a better understanding of context and context-awareness," in *Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing*, 304–307.

Alaa, M., Zaidan, A. A., Zaidan, B. B., Talal, M., and Kiah, M. L. (2017). A review of smart home applications based on internet of things. *J. Netw. Comput. Appl.* 97, 48–65. doi: 10.1016/j.jnca.2017.08.017

Alegre, U., Augusto, J. C., and Clark, T. (2016). Engineering context-aware systems and applications: a survey. *J. Syst. Softw.* 117, 55–83. doi: 10.1016/j.jss.2016.02.010

Alsharif, M. H., Kelechi, A. H., Jahid, A., Kannadasan, R., Singla, M. K., Gupta, J., et al. (2024). A comprehensive survey of energy-efficient computing to enable sustainable massive IoT networks. *Alex. Eng. J.* 91, 12–29. doi: 10.1016/j.aej.2024.01.067

Alti, A., Lakehal, A., Laborie, S., and Roose, P. (2016). Autonomic semantic-based context-aware platform for Mobile applications in pervasive environments. *Future Internet* 8, 1–26. doi: 10.3390/fi8040048

Anastasi, G., Brienza, S., Re, G. L., and Ortolani, M. (2015). "Energy-efficient protocol design" in Green communications: Principles, concepts and practice. eds. K. Samdanis, P. Rost, A. Maeder, M. Meo and C. Verikoukis (Chichester, UK: John Wiley & Sons, Ltd.), 339–360.

Anguita, D., Ghio, A., Oneto, L., Parra, X., and Reyes-Ortiz, J. L. (2012a). Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. *Ambient assisted living and home care* 7657, 216–223. doi: 10.1007/978-3-642-35395-6_30

Anguita, D., Ghio, A., Oneto, L., Parra, X., and Reyes-Ortiz, J. L. (2012b). UCI machine learning repository. Retrieved from UCI Machine Learning Repository. Available at: http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones (Accessed January 11, 2024).

Augusto, J. C., Quinde, M. J., Oguego, C. L., and Manuel, J. G. (2022). Context-aware systems architecture (CaSA). *Cybernetis and Systems* 53, 319–345. doi: 10.1080/01969722.2021.1985226

Cao, X., and Yang, W. (2019). Establishment of residual value assessment model for electric vehicle based on AHP. *IOP* 688, 033001–033011. doi: 10.1088/1757-899X/688/3/033001

Castillo-Atoche, A., Caamal-Herrera, K., Atoche-Enseñat, R., Estrada-López, J. J., Vázquez-Castillo, J., Castillo-Atoche, A. C., et al. (2022). Energy efficient framework for a AIoT cardiac arrhythmia detection system wearable during sport. *Appl. Sci.* 12, 1–14. doi: 10.3390/app12052716

Chen, J., and Venkataramani, G. (2016). enDebug: a hardware–software framework for automated energy debugging. *J Parallel Distrib Comput* 96, 121–133. doi: 10.1016/j.jpdc.2016.05.005

Chen, Z., Xiao, N., and Han, D. (2020). Multilevel task offloading and resource optimization of edge computing networks considering UAV relay and green energy. *Appl. Sci.* 10:592. doi: 10.3390/app10072592

Çiçek, E., and Gören, S. (2021). Smartphone power management based on ConvLSTM model. *Neural Comput. & Applic.* 33, 8017–8029. doi: 10.1007/s00521-020-05544-9

Cioara, T., Anghel, I., Salomie, I., Copil, G., Pernici, B., and Moldovan, D. (2011). A context aware self-adapting algorithm for managing the energy efficiency of IT service centers. *UbiCC0* 6, 619–630.

Cisco. (2016). Internet of things. Available at: https://www.cisco.com/c/dam/en/us/products/collateral/se/internet-of-things/at-a-glance-c45-731471.pdf (Accessed September 14, 2019).

Dai, L., Chen, T., Chai, Y., and Wang, G. (2021). Energy-efficient distributed packet scheduling optimization strategy in cooperative vehicle infrastructure systems. *Wireless Commun Mobile Comput* 2021:6661623. doi: 10.1155/2021/6661623

Daponte, P., Vito, L. D., Picariello, F., and Riccio, M. (2013). State of the art and future developments of measurement applications on smartphones. *Measurement* 46, 3291–3307. doi: 10.1016/j.measurement.2013.05.006

Diwan, S. A. (2022). A multi-layered energy efficient approach for performance aware internet of ocean things. *Int. J. Int. Mobile Technol.* 16, 88–100. doi: 10.3991/ijim.v16i17.34405

Elmalaki, S., Gottscho, M., Gupta, P., and Srivastava, M. (2014). "A case for battery charging-aware power management and deferrable task scheduling in smartphones," in *6th USENIX Conference on Power-Aware Computing and Systems HotPower '14*. Broomfield, CO: USENIX Association, 1–4.

Elmalaki, S., Wanner, L., and Srivastava, M. (2015). "CAreDroid: adaptation framework for android context-aware applications," in *21st Annual International Conference on Mobile Computing and Networking MobiCom '15,* 386–399. Paris: ACM.

Enerdata. (2018). Information & Communication. Available at: https://www.enerdata.net/publications/executive-briefing/between-10-and-20-electricity-consumption-ict-sector-2030.html (Accessed August 20, 2020).

Esquicha-Tejada, J. D., and Copa-Pineda, J. C. (2022). Low-cost and energy-efficient alternatives for home automation using IoT. *Int. J. Int. Mobile Technol.* 16, 153–168. doi: 10.3991/ijim.v16i05.25575

Fei, Y., Zhong, L., and Jha, N. K. (2004). "An energy-aware framework for coordinated dynamic software management in mobile computers," in *The IEEE Computer Society's 12th Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems, 2004*. Volendam: IEEE, 306–317.

Feng, Y. (2017). Markov chain visualization. San Francisco, CA: GitHub.

Flinn, J. (2001). Extending Mobile computer battery life through energy-aware adaptation. Pittsburg, CA: Carnegie-Mellon University.

Flinn, J., and Satyanarayanan, M. (1999). "Energy-aware adaptation for mobile applications," in *Seventeenth ACM symposium on operating systems principles (SOSP '99)*, 48–53.

Galeana-Zapién, H., Torres-Huitzil, C., and Rubio-Loyola, J. (2014). Mobile phone middleware architecture for energy and context awareness in location-based services. *Sensors* 14, 23673–23696. doi: 10.3390/s141223673

Goudarzi, M. (2022). Energy and time aware scheduling of applications in edge and fog. Melbourne: University of Melbourne.

Gupta, R. (2011). The Mobile Indian. Available at: https://www.themobileindian.com/news/consumers-want-good-battery-life-in-nokia-phones-1148 (Accessed August 20, 2021).

Hao, S., Li, D., Halfond, W. G., and Govindan, R. (2013). "Estimating mobile application energy consumption using program analysis," in *Proceedings of the 2013 International Conference on Software Engineering* (San Francisco: IEEE), 92–101.

Hashemi, M., and Sadeghi-Niaraki, A. (2016). A theoretical framework for ubiquitous computing. *IJAPUC* 8, 1–15. doi: 10.4018/IJAPUC.2016040101

Hermann, R., Zappi, P., and Rosing, T. S. (2012). Context aware power Management of Mobile Systems for sensing applications. Yu, Y., Krishnamachari, B., and Kumar, V. P. Information processing in sensor networks (pp. 1–5). Seattle, WA: Semantic Scholar.

Heyn, H.-M., Subbiah, P., Linder, J., Knauss, E., and Eriksson, O. (2022). "Setting AI in context: a case study on defining the context and operational design domain for automated driving" in REFSQ 2022: Requirements engineering: Foundation for Software Quality (Birmingham: Springer), 199–215.

Hussain, S., Wang, Z., and Toure, I. K. (2014). Performance analysis of web Services in Different Types of internet technologies. *Appl. Mech. Mater.* 513, 1431–1436. doi: 10.4028/www.scientific.net/AMM.513-517.1431

Hussain, S., Zhaoshun, W., and Toure, I. K. (2013). An approach for QoS measurement and web service selectness sureness. *High Technol. Lett.* 19, 283–289. doi: 10.3772/j.issn.1006-6748.2013.03.010

Jagarlamudi, K. S., Zaslavsky, A., Loke, S. W., Hassani, A., and Medvedev, A. (2021). Towards measurable efficient and effective metrics for quality and cost of context. *Modeling Using Context* 4, 1–6. doi: 10.21494/ISTE.OP.2021.0684

Kang, S., Lee, J., Jang, H., Lee, Y., Park, S., and Song, J. (2010). A scalable and energy-efficient context monitoring framework for Mobile personal sensor networks. *EEE Trans. Mobile Computing* 9, 686–702. doi: 10.1109/TMC.2009.154

Kiani, S. L., Anjum, A., Antonopoulos, N., and Knappmeyer, M. (2014). Context-aware service utilisation in the clouds and energy conservation. *J. Ambient. Intell. Humaniz. Comput.* 5, 111–131. doi: 10.1007/s12652-012-0131-1

Kiani, S. L., Anjum, A., Knappmeyer, M., Bessis, N., and Antonopoulos, N. (2013). Federated broker system for pervasive context provisioning. *J. Syst. Softw.* 86, 1107–1123. doi: 10.1016/j.jss.2012.11.050

Kim, K.-H., Min, A. W., Gupta, D., Mohapatra, P., and Singh, J. P. (2011). "Improving energy efficiency of Wi-fi sensing on smartphones," in *2011 Proceedings IEEE INFOCOM*. (Shanghai: IEEE), 2930–2938.

Kirsch-Pinheiro, M. (2023). "The context awareness challenges for PIS" in In the evolution of pervasive information systems. eds. M. K. Pinheiro, C. Souveyet, P. Roose and L. A. Steffenel (Cham: Springer), 43–63.

Knappmeyer, M., Kiani, S. L., Reetz, E. S., Baker, N., and Tonjes, R. (2013). Survey of context provisioning middleware. *IEEE Commun Surv Tutor* 15, 1492–1519. doi: 10.1109/SURV.2013.010413.00207

Kohli, V., Chakravarty, S., Chamola, V., Sangwan, K. S., and Zeadally, S. (2023). An analysis of energy consumption and carbon footprints of cryptocurrencies and possible solutions. *Digit Commun Netw* 9, 79–89. doi: 10.1016/j.dcan.2022.06.017

Kök, İ., and Özdemir, S. (2022). Content-centric data and computation offloading in AI-supported fog networks for next generation IoT. *Pervasive Mobile Comput.* 85:101654. doi: 10.1016/j.pmcj.2022.101654

Kouamé, K.-M., Mcheick, H., and Ajami, H. (2020). Adaptive mechanism model for the prevention of SLA violation in the context of COPD patient monitoring. *Symmetry* 12, 1–16. doi: 10.3390/sym12091575

Li, D., Cui, Z., Bai, C., He, Q., and Yan, X. (2021). A tool for energy consumption monitoring and analysis of the android terminal. *J. Electr. Comput. Eng.* 2021, 1–11. doi: 10.1155/2021/5599055

Loke, E. L., Yusof, R., Mohd, O., Hamid, E., Nahar, H., Arif, F., et al. (2023). IOT based integrated COVID-19 self-monitoring tool (COV-SMT) for quarantine. *Int. J. Int. Mobile Technol.* 17, 141–149. doi: 10.3991/ijim.v17i09.35505

Mahmud, U. (2015). UML based model of a context aware system. *IJAPUC* 7, 1–16. doi: 10.4018/IJAPUC.2015010101

Mahmud, U. (2016). "Organizing contextual data in context aware systems: a review" in Handbook of research on human-computer interfaces, developments, and applications. eds. J. Rodrigues, P. Cardoso, J. Monteiro and M. Figueiredo (Hershey, PA: IGI Global), 273–303.

Mahmud, U., Farooq, U., Javed, M. Y., and Malik, N. A. (2012). Representing and organizing contextual data in context aware environments. *J. Comput.* 4, 61–67.

Mahmud, U., and Hussain, S. (2022a). PowerIpsum. San Francisco, CA: GitHub.

Mahmud, U., and Hussain, U. (2022b). "Measuring energy consumption of insertion Sort to establish energy consumption dependency on execution time and memory allocation," in *2022 International Conference on Future Trends in Smart Communities (ICFTSC)* (Kuching: IEEE), 247–250.

Mahmud, U., and Hussain, S. (2024). Augmenting context with power information for green context-awareness in smart environments. *Front. Comput. Sci.* 6:1365500. doi: 10.3389/fcomp.2024.1365500

Mahmud, U., Hussain, S., Malik, A. J., Farooqui, S., and Malik, N. A. (2020). "Realizing IoE for smart service delivery: case of museum tour guide" in Smart systems design, applications, and challenges. eds. J. M. In, P. J. Rodrigues, J. M. Cardoso and C. M. Ramos (Hershey, PA: IGI Global).

Mahmud, U., Hussain, S., Sarwar, A., and Toure, I. K. (2022b). A distributed emergency vehicle transit system using artificial intelligence of things (DEVeTS-AIoT). *Wireless Commun. Mobile Comput.* 2022, 1–12. doi: 10.1155/2022/9654858

Mahmud, U., Hussain, S., and Toure, I. K. (2022a). Gathering contextual data with power information using smartphones in internet of everything. *Wireless Commun. Mobile Comput.* 2022, 1–14. doi: 10.1155/2022/4445751

Mahmud, U., Hussain, S., and Yang, S. (2018). Power profiling of context aware systems: a contemporary analysis and framework for power conservation. *Wirel. Commun. Mob. Comput.* 2018, 1–15. doi: 10.1155/2018/1347967

Mahmud, U., Iltaf, N., and Kamran, F. (2007a). Context congregator: Gathering contextual information in CAPP. Islamabad: Frontiers of Information Technology.

Mahmud, U., Iltaf, N., Rehman, A., and Kamran, F. (2007b). Context-aware paradigm for a pervasive computing environment (CAPP). Villa Real: WWW\Internet 2007, 337–346.

Mahmud, U., and Javed, M. Y. (2012). Context inference engine (CiE): inferring context. *IJAPUC* 4, 13–41. doi: 10.4018/japuc.2012070102

Mahmud, U., and Javed, M. Y. (2014). "Context inference engine (CiE): classifying activity of context using Minkowski distance and standard deviation-based ranks," in *Systems and Software Development, Modeling, and Analysis: New Perspectives and Methodologies* (Hershey, PA: IGI Global), 65–112.

Mahmud, U., and Malik, N. A. (2014). Flow and threat modelling of a context aware system. *IJAPUC* 6, 58–70. doi: 10.4018/ijapuc.2014040105

Makhadmeh, S. N., Al-Betar, M. A., Alyasseri, Z. A., Abasi, A. K., Khader, A. T., Damaševičius, R., et al. (2021). Smart home battery for the multi-objective power scheduling problem in a smart home using Grey wolf optimizer. *Electronics* 10:447. doi: 10.3390/electronics10040447

Malik, N. A., Javed, M. Y., and Mahmud, U. (2008). "Threat modeling in pervasive computing paradigm," in *2008 New Technologies, Mobility and Security (NTMS)*. Tangier: IEEE.

Malik, N. A., Mahmud, U., and Javed, M. Y. (2007). Future challenges in context aware computing. Villa Real: WWW/Internet, 306–310.

Malik, N. A., Mahmud, U., and Javed, M. Y. (2009). Estimating user preferences by managing contextual history in context aware systems. *J. Software* 4, 571–576. doi: 10.4304/jsw.4.6.571-576

Mansouri, K., Alti, A., Roose, P., and Laborie, S. (2018). Dynamic semantic-based green bio-inspired approach for optimizing energy and cloud services qualities. *Emerg. Telecommun. Technol.* 29:e3305. doi: 10.1002/ett.3305

Martyushev, N. V., Malozyomov, B. V., Khalikov, I. H., Kukartsev, V. A., Kukartsev, V. V., Tynchenko, V. S., et al. (2023). Review of methods for improving the energy efficiency of electrified ground transport by optimizing battery consumption. *Energies* 16:729. doi: 10.3390/en16020729

Moghimi, M., Venkatesh, J., Zappi, P., and Rosing, T. (2012). "Context-aware Mobile power management using fuzzy inference as a service," in *International Conference on Mobile Computing, Applications, and Services MobiCASE* 2012. Seattle, WA: Springer, Berlin, Heidelberg, 314–327.

Munoz-Arcentales, A., López-Pernas, S., Conde, J., Alonso, Á., Salvachúa, J., and Hierro, J. J. (2021). Enabling context-aware data analytics in smart environments: an open source reference implementation. *Sensors* 21, 1–29. doi: 10.3390/s21217095

Ogbuabor, G. O., Augusto, J. C., Moseley, R., and van Wyk, A. (2022). Context-aware system for cardiac condition monitoring and management: a survey. *Behav. Inform. Technol.* 41, 759–776. doi: 10.1080/0144929X.2020.1836255

Pereira, R., Couto, M., Ribeiro, F., Rua, R., Cunha, J., and Fernandes, J. P. (2017). "Energy efficiency across programming languages: How do energy, time, and memory relate?," in *SLE 2017: Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering.* (Vancouver, BC: ACM), 256–267.

Qasim, A., Munawar, A., Hassan, J., and Khalid, A. (2021). Evaluating the impact of design pattern usage on energy consumption of applications for Mobile platform. *Appl. Comput. Syst.* 26, 1–11. doi: 10.2478/acss-2021-0001

Rana, B., Singh, Y., and Singh, P. K. (2020). A systematic survey on internet of things: energy efficiency and interoperability perspective. *Emerging Telecommun. Technol.* 32, 1–41. doi: 10.1002/ett.4166

Reffad, H., and Alti, A. (2023). Semantic-based multi-objective optimization for QoS and energy efficiency in IoT, fog, and cloud ERP using dynamic cooperative NSGA-II. *Appl. Sci.* 13, 1–30. doi: 10.3390/app13085218

Safara, F., Souri, A., Baker, T., Ridhawi, I. A., and Aloqaily, M. (2020). Pri Nergy: a priority-based energy efficient routing method for IoT systems. *J. Supercomput.* 76, 8609–8626. doi: 10.1007/s11227-020-03147-8

Sangeetha, M., Erna, C., Kannan, S. R., Salameh, A. A., and Kishore, K. H. (2023). Energy efficient routing scheme for performance enhancement of MANET through dominating set. *Int. J. Int. Mobile Technol.* 17, 89–100. doi: 10.3991/ijim.v17i04.37803

Sathan, D., Meetoo, A., and Subramaniam, R. K. (2009). Context aware lightweight energy efficient framework. *World Acad. Sci. Eng. Technol.* 52, 64–70.

Schaarschmidt, M., Uelschen, M., and Pulvermüller, E. (2022). Hunting energy bugs in embedded systems: a software-model-In-the-loop approach. *Electronics* 11, 1–39. doi: 10.3390/electronics11131937

Schilit, B. N., Adams, N., and Want, R. (1994). Context-aware computing applications. Proceedings of the 1994 first workshop on Mobile computing systems and applications. Washington, DC: ACM, 85–90.

Shahryari, O.-K., Pedram, H., Khajehvand, V., and TakhtFooladi, M. D. (2021). Energy and task completion time trade-off for task offloading in fog-enabled IoT networks. *Pervasive Mobile Comput.* 74:101395. doi: 10.1016/j.pmcj.2021.101395

Simoens, P., Mahieu, C., Ongenae, F., Backere, F. D., Pestel, S. D., and Nelis, J., … Jacobs, A. (2016). "Internet of robotic things: context-aware and personalized interventions of assistive social robots," in *5th IEEE International Conference on Cloud Networking (Cloudnet)* (Pisa: IEEE), 204–207.

Souri, A., Hussien, A., Hoseyninezhad, M., and Norouzi, M. (2019). A systematic review of IoT communication strategies for an efficient smart environment. *Trans. Emerg. Telecommun. Technol.* 33, 1–37. doi: 10.1002/ett.3736

Stauffer, N. W. (2013). Energy-efficient computing. Available at: https://energy.mit.edu/research/energy-efficient-computing/ (Accessed August 25, 2020).

Tong, E., Niu, W., Tian, Y., Liu, J., Baker, T., Verma, S., et al. (2021). A hierarchical energy-efficient service selection approach with QoS constraints for internet of things. *IEEE Trans. Green Commun. Netw.* 5, 645–657. doi: 10.1109/TGCN.2021.3069121

Tyagi, N., Lynch, J., and Demaine, E. D. (2016). "Toward an energy efficient language and compiler for (partially) reversible algorithms," in *Conference: International Conference on Reversible Computation* (Bologna: Springer), 121–136.

Xia, S., Yao, Z., Li, Y., and Mao, S. (2021). Online distributed offloading and computing. *IEEE Trans. Wirel. Commun.* 20, 6743–6757. doi: 10.1109/TWC.2021.3076201

Yuryur, O. (2013). Energy efficient context-aware framework in Mobile sensing. University of South Florida, electrical engineering. Florida: University of South Florida.

Zappi, P., Lombriser, C., Stiefmeier, T., Farella, E., Roggen, D., Benini, L., et al. (2008). "Activity recognition from on-body sensors: accuracy-power trade-off by dynamic sensor selection" in Wireless sensor networks. ed. R. Verdone (Cham: Springer), 17–33.

Zhuang, Z., Kim, K.-H., and Sing, J. P. (2010). "Improving energy efficiency of location sensing on smartphones. MobiSys '10," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*. New York, NY: ACM, 315–330.