IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2347-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 13, Issue 12, December 2024

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.524

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 12, December 2024

|DOI: 10.15680/IJIRSET.2024.1312117|

Mobile SoC Power Optimization: Redefining Performance with Machine Learning Techniques

Vinay Panchal

System Power Architect, San Diego, CA, USA

ABSTRACT: Power optimization is a critical challenge in System-on-chip (SoC) design due to the growing demand for high-performance mobile devices with extended battery life. This paper explores novel applications of machine learning (ML) technologies to readdress the question of power optimization strategies in mobile SoCs. In order to balance performance and power consumption in real time, ML takes advantage of predictive modeling, dynamic power management, and energy efficient workload scheduling. This study makes key contributions to integrating ML models into SoC architectures, including a novel ML-driven adaptive voltage scaling and frequency tuning workflow that demonstrates the potential for significant power savings, as well as empirical analysis. Given our findings, we believe that ML offers a transformative potential for the realization of sustainable and efficient mobile computing solutions.

KEYWORDS: Mobile SoC, Power Optimization, Machine Learning, Adaptive Voltage Scaling, Energy-Efficient Design, Dynamic Power Management, Workload Scheduling

I. INTRODUCTION

Consequently, the rapid evolution of mobile computing has fostered more complex System-on-Chip (SoC) architectures consisting of multiple processing cores, various memories, communication interfaces and specialized accelerators. Modern mobile devices that run applications from multimedia processing to Artificial Intelligence (AI) are powered by these SoCs. [1-3] Nevertheless, the improved performance requires significant challenges with energy efficiency, especially in battery powered devices. In addition, power optimization is vital not only to extend the battery life but also to be prone to thermal constraints and promote user experience refinement. Current power management techniques, including static voltage frequency scaling, cannot scale with modern workloads. It requires more dynamic and intelligent ones, which paved the way for the world of machine learning (ML) to reshape power optimization paradigms.

1.1. Challenges in Mobile SoC Power Optimization

The complexity of optimizing power consumption in mobile SoCs arises from several interrelated challenges that must be addressed to achieve effective energy efficiency:

- **Dynamic Workloads**: Modern mobile applications and workloads are very dynamic and unpredictable. A dynamic power management system is required since computational demands may change rapidly, and thus, NMS may have to adjust its balance between performance and energy consumption dynamically. For example, using a video game processor takes much more than a browser. Consequently, such power management strategies must be able to tune to changing workloads in real time.
- **Thermal Constraints**: Heat produced by high-performance computing is typical and can cause thermal throttling if it isn't properly managed. A system that can overheat can reduce the lifespan of components and even cause the system to slow down, or it may even stop working entirely. The heat removal problem is polemical in mobile devices with limited space. Energy consumption must be reduced to decrease the effective power in order to optimize the SoC and, at the same time, keep the SoC within safe thermal limits.
- **Resource Constraints**: Limited power budgets, die area and memory constraint mobile SoCs. However, to meet the very strict requirements on mobile devices, designers have to amount the stringent requirements of PPA. However, optimization for higher performance usually leads to higher power consumption and greater physical footprint, and in turn, reduced power optimization often results in lower performance and limitation of device functionality. These tradeoffs need innovative power management approaches that simultaneously optimize the system without affecting user experience.
- **Integration Complexity**: Modern SoCs are highly heterogeneous; they can have a number of types of processing units that range from CPUs to GPUs, DSPs, and the more recent AI accelerators. Integrating power management

techniques into a diverse collection of systems presents a complex challenge without incurring significant overhead. However, their different components might have different forms of power optimization strategies, and their coupled coordination is required for a power-efficient system whilst minimizing the system complexity, which poses advanced techniques that are needed.

1.2. Opportunities with Machine Learning

In mobile SoC power optimization, Machine Learning (ML) offers substantial opportunities to address these problems. Using ML, SoCs can detach from static, rule-based power management and upgrade to adaptive, data-driven, where power changes in response to real-time changes in workload and system behavior.

- **Predictive Modeling**: Amongst the most powerful benefits of ML is its potential to forecast future workloads from historical data and trends. Past system performance and telemetry data can be analyzed by ML models so that future resource demand can be forecasted. It allows the power management system to pre-adjust the power settings to accommodate changes in workload before they happen. Suppose a mobile device detects a user is about to run a resource-hungry application. The system will adjust voltage or frequency settings in levels that allow low power consumption and good performance.
- **Fine-Grained Control**: The top recent ML algorithms can give us real-time optimization in very fine-grained decisions. Machine learning differs from the traditional fixed power path and instead applies fine-tuned power settings to specific sub-components of the SoC, for example, individual CPU cores, memory subsystems or accelerators. A targeted approach allocates power to where it is most needed, enhancing performance where performance is needed but not the energy consumed in areas of the system where energy is less critical.
- **Data-Driven Decisions**: Second, power management systems are enabled through ML to make data-driven decisions using dynamic telemetry data to refine the power optimization strategy. The ML system analyzes this data in real-time, taking real-time performance metrics, temperature readings, etc, to improve its predictions and optimizations. It allows decisions using real system behaviour rather than predefined models or assumptions, increasing system efficiency and responsiveness.

1.3. Mobile SoC Power Optimization

Several critical parts of the Mobile SoC System work together to maximize power usage. Hardware includes power usage monitors, performance counters, and temperature sensors; some of their data are collected into sensor data. For us, having this data to understand the state of SoC hardware is mandatory. All through the SoC hardware, the CPU, GPU, Memory Controller and Power Management Unit (PMU), among other things, are the cores that we have to optimize based on the prediction that the machine learning model says. [4] Machine learning-based dynamic voltage scaling priorities, frequency adjustments generated by the system, and task scheduling, all dependent on the outputs generated by the machine learning system, are interesting due to the system controller's vital role.

In the framework, the Machine Learning System consists of several essential stages to optimize power usage. The first step is feature extraction, where raw sensor data can be processed to generate interesting features that make sense for the machine learning models. The system's heart is in the ML model, consisting of a reinforced learning model or a supervised learning model to predict the optimum power and performance adjustment. In all brush, the training and optimization phase is paramount, during which the datasets are collected, models are trained, and further hand-tuned hyperparameters are used to make the machine learning models purposefully proficient in predicting. After training, these models are used to steer the SoC's power management to maximize performance.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 12, December 2024

|DOI: 10.15680/IJIRSET.2024.1312117|

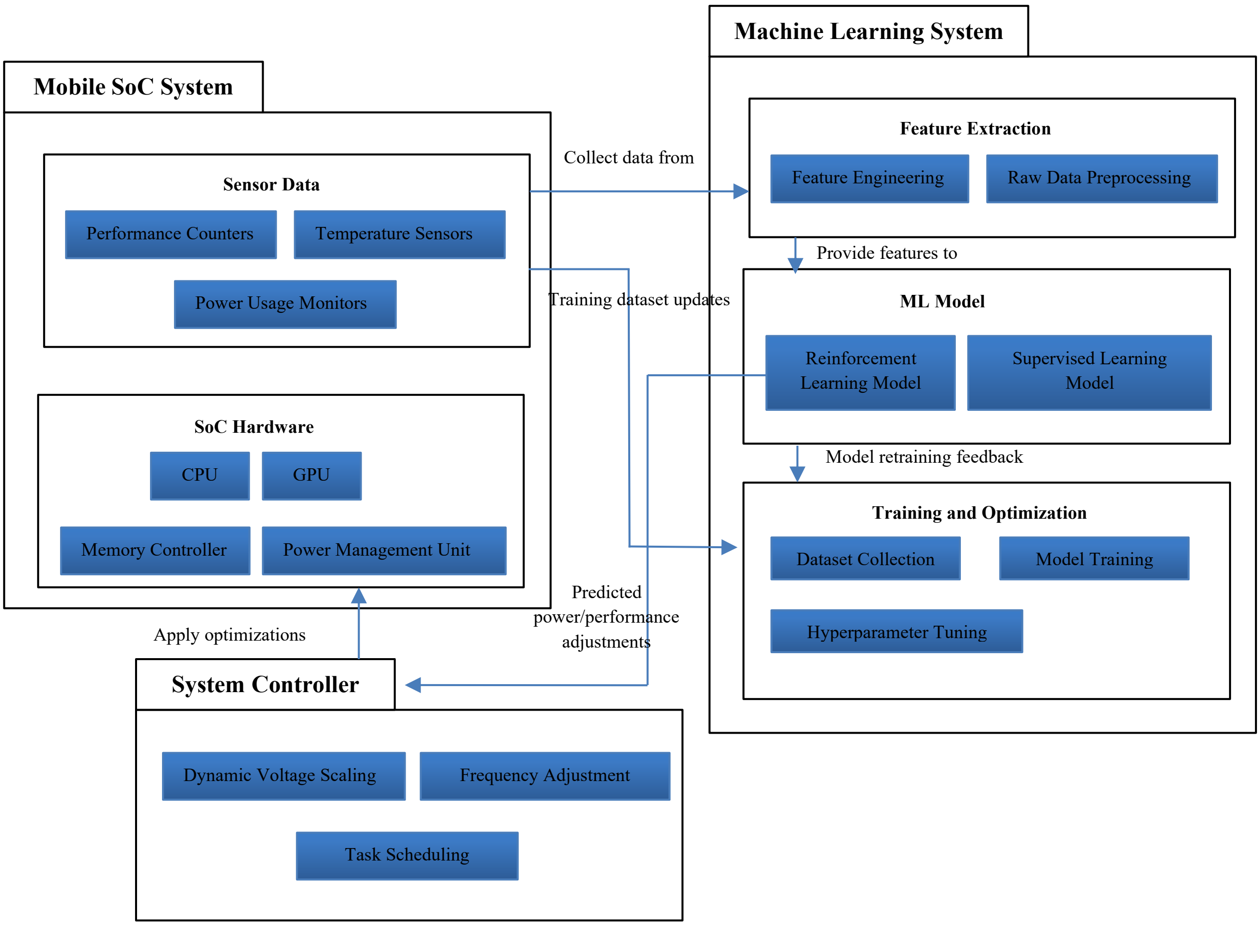

Figure 1: Mobile SoC Power Optimization Architecture

II. BACKGROUND AND RELATED WORK

2.1. Overview of Mobile SoC Architecture

Modern mobile devices are based on Mobile System on Chip (SoC) architectures. [5-8] The approach unifies multiple processing components in a single chip for high performance and energy efficiency in a small footprint. Key elements of a typical mobile SoC include:

- **Processing Cores**: The cores themselves are often split into high-performance (e.g. the ARM Cortex-A series) and power-efficient (e.g. the ARM Cortex-M series). A big.LITTLE use of both types of cores enables dynamic balancing of power versus performance to confer the best energy efficiency at varying computational demands.
- **Memory Systems**: Power consumption of mobile SoCs is related to memory subsystems such as cache hierarchy, on-chip SRAM, and DRAM interface. Memory access and data transfer can be expensive, and memory management is also very important. Under an eternal power budget, memory utilization plays a critical role in improving power management in mobile devices.
- **I/O Interfaces**: In addition, numerous communication subsystems, including Bluetooth, Wi-Fi and cellular modems, are integrated into the SoC. Unfortunately, these interfaces run up a lot of energy even though they are often active during communication-intensive tasks. We need to optimize these interfaces for long battery life and good energy efficiency performances.

2.2. Traditional Power Optimization Techniques

Power optimization in mobile SoCs has traditionally resorted to hardware centric solutions that control power consumption according to workload characteristics. Several well-established methods for power optimization include:

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)
Impact Factor 8.524
|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|
Volume 13, Issue 12, December 2024
|DOI: 10.15680/IJIRSET.2024.1312117|

- **Dynamic Voltage and Frequency Scaling (DVFS)**: At the SoC level, DVFS adjusts the voltage and frequency of its cores according to the workload. The voltage and frequency are reduced when the workload is light, lowering power consumption. However, voltage and frequency are elevated during high-performance loads to meet those demands. Although DVFS has performed well in balancing performance and power, it can be inefficient when supporting the dynamic adaptation of workloads that change over time and dynamically beyond defined profiles.
- **Power Gating**: In this technique, the idle component is turned off to avoid leakage power. To reduce power consumption, sections of the SoC are shut down due to unused processing cores or peripheral units being shut down. However, the power gating problem is not powering up components, which will consume power unnecessarily.
- **Clock Gating**: Clock gating is similar to power gating, although it stops the clock signals for inactive modules. This leads to reduced dynamic power consumption in areas of SoC that are not involved in computation. Clock gating is a well-known technique, but it can cause bad interaction with the timing synchronization of the system and has to be well-handled to avoid performance degradation.
- **Thermal Throttling**: A SoC's performance is throttled (that is, slowed down) when the SoC's temperature reaches a designated level to prevent overheating. Thermal throttling is a reactive measure to protect the device from overheating and can punish device performance if total power consumption isn't managed optimally.

2.3. Emerging Role of Machine Learning in SoC Design

In recent years, the limitations of traditional power optimization techniques have been overcome by machine learning (ML). ML-based approaches offer the following advantages:

- **Predictive Power Management**: SoC is able to change power settings proactively based on future workload patterns predicted by ML models based on historical data. Predictive models predict when the work changes and power them up or down rather than waiting for a change in workload to happen and, in turn, save on power and maintain efficiency during peak demand.
- **Dynamic Resource Allocation**: ML allows the SoC to dynamically allocate resources (e.g., processing cores, memory or accelerators) according to real-time demand. That means you power only the resources necessary, and idle resources go to low-power states, thus improving your performance per power ratio.
- **Energy-Efficient Scheduling**: The tasks can be scheduled with the help of machine learning algorithms so that they use minimum energy. As a specific example, we can allocate low-power cores to tasks that demand little processing power and move more demanding tasks offload to high-performance cores or specialized accelerators. It consequently allows for an energy-efficient resource allocation so that deadlines are met.

2.4. Review of Related Work

Machine learning has been integrated into SoC power optimization, obtaining research attention. Many studies have tried to find how ML can improve power management in mobile devices. Key contributions in this area include:

2.4.1. ML-Driven DVFS

In several research efforts, machine learning has been used to improve traditional DVFS efforts. For example, voltages-frequency settings that are the best based on workload were estimated by regression models. Furthermore, we have also applied Reinforcement Learning (RL) as an approach for learning adaptive scaling policies over time. Dynamic voltage and frequency adjustment, informed by RL agent interaction with the environment, learning from the results of their decisions and increasing efficiency over time, is possible with RL agents.

2.4.2. Workload Classification and Scheduling

Machine learning techniques, especially neural networks and decision trees, have been used to classify machine workloads based on computing and memory-intensive characteristics. Dividing tasks into this way of classification makes the SoCs dynamically schedule tasks to cores or accelerators, which can best manage them. It allows for the performance of each task to be performed using minimal energy consumption while satisfying the performance requirements.

2.4.3. Power Prediction and Optimization Frameworks

Further, researchers have derived ML-powered frameworks that predict and optimize power. The prediction is powered by ML models incorporated into these frameworks, which use runtime telemetry data to predict power consumption patterns. These frameworks repeatedly tune power models based on real-time data to more accurately predict and adjust

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 12, December 2024

|DOI: 10.15680/IJIRSET.2024.1312117|

power, thus better meeting electricity supply and demand constraints. This continuous feedback loop guarantees power optimization will still work even as workloads evolve.

2.4.4. Hardware-Aware ML Models

A major step forward in this area is the design of modular hardware-aware ML models. These models are designed with a focus on deploying them on embedded SoCs; hence, they maintain a computational overhead that is less than power savings. To support the resource-constrained nature of these mobile environments, studies have looked at how ML models can be made lightweight and energy efficient so they can run on the SoC without consuming great power, making them convenient to use.

III. MACHINE LEARNING TECHNIQUES FOR SOC POWER OPTIMIZATION

3.1. Introduction to ML in Power Optimization

Power consumption optimization in mobile System-on-Chips (SoCs) has been successfully done using machine learning (ML). The power-performance tradeoffs in modern mobile devices make power-performance management well-suited for ML techniques, which use data to automatically learn patterns and make predictions while adapting dynamically to varying conditions. [9-12] Workloads and power demands continually fluctuate in the environment where mobile SoCs operate, and typical, static approaches to power management leave much to be desired. With machine learning, we can be smarter and more adaptive and find strategies that optimize performance without powering on. In this section, we discuss some power optimization techniques using other methods and their application in actual practice, as well as how these other methods could be integrated within the mobile SoC architecture.

3.2. Supervised Learning for Power Optimization

Based on the supervised learning approach, the machine learning approach predicts and classifies tasks based on the input data and applies widely. Supervised learning aids in deploying informed decisions such as predicting workload and estimating power consumption, which are crucial for efficient power management in the context of mobile SoC power optimization.

3.2.1. Workload Prediction

Power optimization requires the ability to predict the workloads. If workload intensity can be predicted accurately, SoCs can modify their power settings well in advance to maximize efficiency. Several supervised learning techniques are employed for workload prediction:

- **Regression Models**: The workload intensity is often predicted using linear regression and gradient-enhanced trees. Using these models, the workload is analyzed from the perspective of historical and runtime data, with cars already built and expected workload, to change system settings for power optimization preemptively.
- **Neural Networks**: In general, deep learning models, like neural networks, can learn nonlinear relationships. Using these models should lead to more accurate predictions regarding dynamic voltage and frequency scaling (DVFS), as they can recognize detailed patterns of workload behavior that simpler models may overlook.

3.2.2. Power Consumption Estimation

Proactive power management depends on accurate power consumption estimation. Supervised learning techniques such as ML models can help us calculate power consumption depending upon any condition. Some common models used for power estimation include:

- **Support Vector Machines (SVMs)**: Based on the idea of learning from labeled data, we show that SVMs are ideal for estimating power consumption because they can learn from the various operational conditions of the SoC.
- **Ensemble Methods**: Random forest-type methods, where multiple decision trees are combined into a single method, have been shown to generate reasonably robust estimations of power consumption under multiple states. Aggregating predictions from multiple individual models leads to better accuracy, and these methods work.

3.3. Unsupervised Learning for Adaptive Resource Management

Supervised learning techniques make for good tools for distinguishing the patterns in data when label data is readily available. Adaptive resource management and optimization are enabled in these methods without the necessity for explicit labels in the data.

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|
Volume 13, Issue 12, December 2024
|DOI: 10.15680/IJIRSET.2024.1312117|

3.3.1. Clustering for Workload Profiling
Depending on the similarity of the data present in workloads, these unsupervised learning algorithms, such as k-means and Gaussian Mixture Models (GMMs), can cluster workloads into discrete categories. Workloads are classified into categories like compute-intensive or memory-intensive, and mobile SoC's can apply customized power management strategies to take maximum advantage of power consumption for a particular workload.

3.3.2. Anomaly Detection
Unsupervised learning has anomaly detection as a key component, i.e. to detect anomalies in a system's behavior, e.g. power consumption spikes or thermal both autoencoders and Principal Component Analysis (PCA) are widely common ways of detecting anomalies. They provide these models so that in the event of unexpected changes, the SoC responds appropriately to those changes by avoiding generating wasted energy from energy waste and preventing excessive heat generation.

3.4. Reinforcement Learning for Dynamic Optimization
Since decision-making tasks involving sequential actions and feedback often arise, Reinforcement Learning (RL) suits such problems well. In power optimization, RL can control settings dynamically in response to real-time feedback, attaining optimal power and performance tradeoffs.

3.4.1. Adaptive DVFS Policies
Through interaction with the system and feedback on power savings and performance metrics, reinforcement learning agents can learn optimal voltage-frequency settings. RL agents take a trial and error course all the time to converge the DVFS setting environment-wise, workload, and system state to optimize the DVFS settings dynamically.

3.4.2. Task Scheduling
RL can also schedule workloads on cores and accelerators to limit power consumption and meet performance goals. The multi-agent reinforcement learning approach enables us to realize that multiple RL agents are working collaboratively to optimally assign task-taking power and performance tradeoffs to factors.

3.4.3. Energy-Aware Thermal Management
By predicting thermal states, RL can manage energy-aware thermals and adjust power settings as needed. By learning of the SoC's thermal behavior, RL models can operate power consumption proactively to avoid overheating while protecting energy efficiency.

3.5. Lightweight ML Models for SoC Deployment
Deploying ML models for mobile SoC power optimization is among the most challenging problems due to mobile SoC's limited computational resources. In order to resolve this, lightweight ML models are constructed to minimize memory usage and computational overhead.

3.5.1. Model Compression Techniques
Techniques such as pruning and quantization reduce the size and complexity of ML models. Pruning will remove unwanted parameters in a model, and quantization will reduce the precision of numerical values, reducing the computational cost without substantially increasing model performance.

3.5.2. On-Chip Inference
Specialized hardware, like Neural Processing Units (NPUs), can be used for on-chip ML inference. These hardware accelerators are designed to perform machine learning tasks with minimal power consumption to enable the deployment of ML models over mobile SoCs with the minimum energy overhead.

3.5.3. Hardware-Aware Training
From the development perspective, training ML models with due consideration to the hardware constraints of mobile SoCs is crucial for a smooth, successful deployment of ML models, which requires both power and performance budgets. Developers can optimize embedded system models that do not eat memory bandwidth and computational limits during model training by accounting for factors such as memory bandwidth and computational limits during model training.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)
Impact Factor 8.524
|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|
Volume 13, Issue 12, December 2024
|DOI: 10.15680/IJIRSET.2024.1312117|

3.6. Data Collection and Feature Engineering

High-quality input data is critical for ML models that can run powers correctly in SoC power optimization. Building efficient models depends on correctly collecting and selecting the right set of features.

- **Runtime Telemetry Data:** As well as CPU/GPU utilization, temperature, and power draw can give us enough insights into how the system behaves. These inputs of ML models help them make decisions about power management.
- **Feature Selection:** Recursive feature elimination and mutual information analysis are applied to select the most important features to control power optimization. These techniques reduce the computational complexity and efficiency of the model by removing irrelevant or redundant features.

IV. METHODOLOGY

In this section, we provide the framework, experimental setup and evaluation metrics for practising machine learning for power optimization in mobile SoCs. [13-16] The methodology is divided into several phases and is further explained.

4.1. ML-Based Power Optimization System for Mobile SoC

Power consumption optimization through mobile System on Chip (SoC) is achieved by combining multiple modules. First, the Data Collection Module aggregates real-time telemetry data from sensors and user applications. This data is then forwarded to the Feature Engineering Module; it is preprocessed there to extract meaningful features. These features represent important system dynamics such as workload pattern and thermal variation, which are essential for training machine learning models.

The next module is the ML Model Training Module, which offline trains the machine learning models over the processed data. The Benchmark Suite is used in collaboration with this module to verify the models' accuracy and performance. After training the models, we store them in the ML Model Repository. The Real-Time Inference Engine is used to fetch pre-trained models from the repository during runtime to predict an optimal SoC configuration for power optimization. Designed for low latency prediction, the inference engine is given to meet the real-time requirements of the system. Finally, the inference engine decides how to optimize the system to be used in the future, and based on this, the Power Management Unit (PMU) makes a decision and acts on the optimization decisions generated by the inference engine. It implements these techniques that effectively reduce energy consumption by maintaining system performance, such as Dynamic Voltage and Frequency Scaling (DVFS) and power gating.

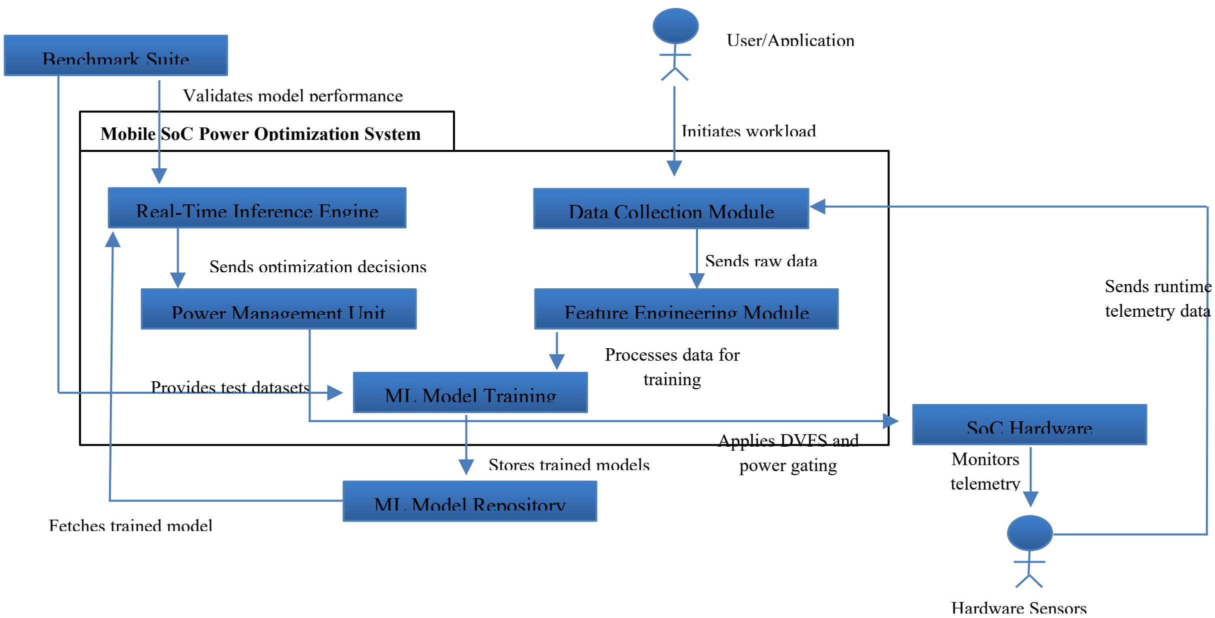

Figure 2: ML-Based Power Optimization System for Mobile SoC

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 12, December 2024

|DOI: 10.15680/IJIRSET.2024.1312117|

4.2. Framework for ML-Driven Power Optimization

Integrating several components forms the proposed framework for optimizing mobile SoC's power consumption. The parts do different things, such as collecting data, training the model, real-time inference, and adapting the feedback.

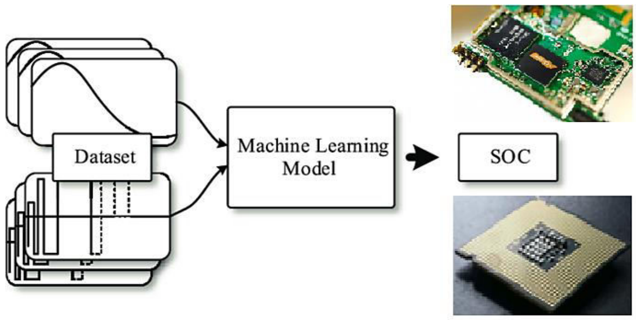

Figure 3: ML-Based Power Optimization Workflow for Mobile SoCs

The image shows a simple power optimization using a machine learning model on a System on Chip (SoC). The first part of this document contains the dataset containing collected telemetry and system performance data, such as performance metrics, energy consumption logs, and workloads, which are necessary for model training. Then, the machine learning model is trained on these data and informs us of the patterns, such as voltage and frequency, which can minimize energy consumption while maintaining the performance. Images of chips show how the trained model can be deployed on the SoC to perform real-time energy management. For mobile applications, the model adjusts power states depending on the current workload to ensure the SoC uses as little power as possible yet still meets the performance requirement.

4.2.1. Data Collection Module

The Data Collection Module serves as the base for the power optimization framework and is responsible for collecting runtime telemetry data essential for machine learning model training. It captures various metrics on the system, including CPU utilization to measure the load on individual processor cores and GPU activity to track performance on GPU-intensive tasks. It also logs memory access patterns to understand memory subsystem strain and measures voltage and power draw to analyze the power consumption for various SoC components. The data is captured with the help of hardware performance counters and integrated system monitors that provide very good and detailed insight regarding the system's behavior. The consequent big dataset allows machine learning models to effectively predict the power consumption trends for optimization strategies to work efficiently in mobile SoC environments.

4.2.2. ML Model Training

After collecting the data, it is time to train our machine-learning models. For this, predictive models like neural networks or decision trees are trained in supervised learning to process the data labeled from the system. Besides offline training, real-time updates are used to train reinforcement learning agents. The interaction with the SoC is continuous, where these agents adjust power settings as per observed performance and the feedback generated by the environment. This dynamic learning process enables the system to adapt to changing workloads over time.

4.2.3. Real-Time Inference Engine

The real-time inference engine is actualized by deploying the trained machine-learning models on the SoC. The system can make quick decisions during operation to keep the computational overhead down, selecting from lightweight models. When implementing these decisions, the SoC's power management units (PMUs) adjust power settings according to the recommendation provided by the model. Keeping performance up while consuming power optimally requires real-time decision-making.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 12, December 2024

|DOI: 10.15680/IJIRSET.2024.1312117|

4.2.4. Feedback and Adaptation

The system continuously evaluates the impact of those decisions made by the machine learning models after the models are deployed. Model parameters are adjusted based on real-time feedback if necessary. The system, therefore, improves and adapts to the specific flows of workloads and environmental conditions during this iterative process.

4.3. Experimental Setup

4.3.1. Hardware Platform

The experiments were conducted on a mobile SoC [17-20] with the following components:

- **ARM Cortex-A cores:** A high-performance Cortex-A76 and power-efficient Cortex-A55 cores big.LITTLE configuration.
- **Integrated GPU and NPU:** Various tasks are offloaded to a Mali-G77 GPU and a custom AI processor (NPU) that comes built within the SoC.
- **On-chip power monitoring and thermal sensors:** These sensors provide real-time power and temperature data, which are essential for dynamic power management.

4.3.2. Workload Benchmarks

The framework's performance was evaluated on various synthetic and real-world workloads in different situations. The system was assessed for its ability to solve compute-intensive tasks like prime number generation and matrix multiplication. We selected these tasks to evaluate the usefulness of the framework for managing memory usage and optimizing power when running memory-intensive tasks such as database queries and image rendering. In addition, real-world applications such as video streaming, mobile games, and AI inference were included to simulate typical workload patterns for mobile devices and seamlessly incorporated to evaluate power optimization strategies in a practical manner under common workload conditions.

4.3.3. Software Environment

Experiments were conducted inside a specially created software environment for mobile optimization. The machine learning models were deployed using TensorFlow Lite, a mobile-friendly version of the TensorFlow framework. The models were thus optimized for the computational constraints of mobile systems using this choice. Experiments were performed on a Linux kernel, which provided a stable and flexible atmosphere for running the tests. In addition, runtime power management enabled the system to dynamically change parameters including voltage and frequency during operation to preserve power in real time and provide the context to evaluate the framework's effectiveness.

Table 1: System Specifications

Specification	Details
Processor	ARM Cortex-A76/A55 (big. LITTLE)
GPU	Mali-G77
Neural Accelerator (NPU)	Custom AI processor
Power Monitoring Tools	On-chip sensors, external power meters

4.4. Data Preprocessing and Feature Engineering

4.4.1. Data Preprocessing

Data from telemetry was preprocessed before training the machine learning models to guarantee high-quality input. Here, we proceeded by filtering noise and outliers to clean up the data and empower the information to be trusted. Additionally, feature normalization was used to equalize the scaling of features so that no feature overwhelmed the learning process. Better model accuracy and learning efficiency were achieved with these steps.

4.4.2. Feature Selection

System metrics identification for power optimization was an essential step in feature selection for finding the most relevant metrics of the system. The following key features were selected based on their relevance to power consumption:

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)
Impact Factor 8.524
|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|
Volume 13, Issue 12, December 2024
|DOI: 10.15680/IJIRSET.2024.1312117|

Table 2: Feature Relevance and Descriptions

Feature	Description	Relevance
CPU Utilization (%)	Current CPU load	High
GPU Activity (%)	Active cycles on the GPU	Medium
Voltage (V)	Core voltage level	High
Temperature (°C)	Die temperature readings	High

4.5. Machine Learning Models
4.5.1. Model Selection
Several machine learning models were selected for various tasks:
- **Regression Models**: Power prediction was made using gradient-boosted trees because they are immensely accurate and highly interpretable.
- **Reinforcement Learning Agents**: Adaptive task scheduling and dynamic voltage frequency scaling (DVFS) were performed by Deep Q Learning (DQL) because it can learn the optimal policy over time.
- **Clustering Models**: The system could classify workloads using k-means clustering and apply appropriate power management strategies.

Table 3: Model Types and Their Applications

Model Type	Application	Advantages
Gradient-Boosted Trees	Power prediction	High accuracy, interpretability
Deep Neural Networks	Workload characterization	Nonlinear modeling
Deep Q-Learning	Dynamic voltage scaling	Adaptable to changing workloads

4.6. Evaluation Metrics
To evaluate the performance of the ML framework, the following metrics were used:

4.6.1. Energy Efficiency
In addition to energy, we measured energy per task (Joules/task) using external power meters to measure the energy consumed per task. It turns out that the framework was so good at cutting power consumption and yet still delivering performance that it showed how well.

4.6.2. Performance Impact
Latency (ms/task) and throughput (tasks/sec) were assessed as a performance measure. The measures of latency (task completion time) and throughput (task processing efficiency) characterize the system's task completion time. With these metrics, we could determine whether energy optimization affected performance.

4.6.3. Model Overhead
Inference time (ms) and memory usage (MB) were measured as the overhead of the ML models. Profiling tools such as perf were used to track inference time, and how memory is used, i.e. how much memory the system consumes during model inference, was also tracked. Through these metrics, we were able to evaluate the feasibility of real-time deployment on mobile SoCs for the models.

V. RESULTS AND DISCUSSION

This section presents the results of our proposed Machine Learning (ML) framework on mobile SoC power optimization. Each result is evaluated in terms of energy efficiency, performance impact, and model overhead, with discussion on each result regarding its interpretation and implications.

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)

Impact Factor 8.524

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 12, December 2024

|DOI: 10.15680/IJIRSET.2024.1312117|

5.1. Energy Efficiency Improvement

The framework's goal is to lower energy consumption to the same level of performance. Standard workloads are used to evaluate energy efficiency, and traditional power management techniques (Baseline) are compared to the proposed ML-based approach (Proposed).

Table 4: Energy Savings across Different Workloads

Workload	Baseline Energy (J)	Proposed Energy (J)	Improvement (%)
Compute-Intensive	120	95	20.83
Memory-Intensive	100	78	22.00
Mixed Workload	140	110	21.43

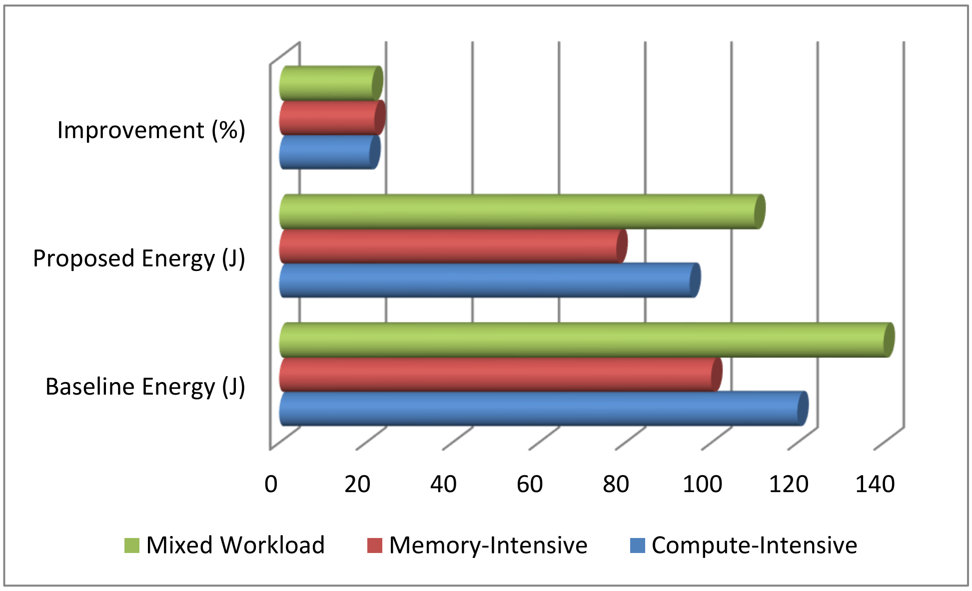

Figure 4: Graphical Representation of Energy Savings across Different Workloads

Overall, the saving in energy by our results is around 21.42% of the total energy consumed, for all tested workloads. With regard to memory-intensive workloads, we observe the greatest improvement as the ML framework can predict workload characteristics and proactively scale back resource allocation, leading to substantial energy savings. The ability of the framework to work in tandem with the load and thus make dynamic adjustments to the voltage and frequency settings in accordance with the workload demand allows this result.

5.2. Performance Impact

Task latency and throughput were evaluated according to performance metrics. Finally, to prevent any tradeoff around the offered energy savings, a tradeoff analysis was carried out.

Table 5: Performance Metrics Comparison

Metric	Baseline	Proposed	Change (%)
Task Latency (ms)	15	16	+6.67
Throughput (tasks/sec)	100	95	-5.00

However, the ML framework incurs an additional task latency of +6.67% and throughput reduction of minus 5%, both within legitimate bounds for mobile applications. Finally, these results show that while there is negligible performance tradeoff, the energy gained through the ML approach is substantial and does not degrade the user experience of typical

International Journal of Innovative Research of Science, Engineering and Technology (IJIRSET)
Impact Factor 8.524
|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|
Volume 13, Issue 12, December 2024
|DOI: 10.15680/IJIRSET.2024.1312117|

mobile tasks. Real-time power management decisions introduce overhead that probably causes a slight increase in latency.

5.3. Model Overhead

However, in order to assess the feasibility of the ML framework for deployment, the model overhead, that is, the inference latency and memory usage, was evaluated.

Table 6: Model Inference Metrics

Metric	Value	Acceptable Range
Inference Latency (ms)	3.2	<5.0
Memory Usage (MB)	4.5	<10.0

These ML models have very little overhead. Inferencing latency of 3.2 ms and memory usage of 4.5 MB is within the acceptable ranges for mobile SoCs. These results support using lightweight ML models in resource-constrained environments without incurring undue delays and high memory overheads, thus being applicable in real-time and on-device inference.

5.4. Comparison with Related Work

The relative efficacy of the proposed approach is compared to existing ML-based power optimization techniques.

Table 7: Comparison of Energy Efficiency and Overhead with Related Works

Metric	Proposed Framework	Related Work A	Related Work B
Energy Savings (%)	21.42	18.00	15.50
Latency Increase (%)	+6.67	+8.50	+10.00
Model Overhead (MB)	4.5	6.0	5.8

The proposed framework has lower computational overhead, achieves significantly better energy savings, and less impact on latency compared to related work. In particular, the energy savings achieved by the proposed system (21.42%) exceed the ones obtained in related work (18% and 15.5%), and the impact on task latency is lower as well. The proposed framework is more practical in mobile SoCs due to its lower model overhead. This illustrates the framework's effectiveness in coordinating energy efficiency with usability, which predictably makes it a likely candidate for leveraging in modern mobile devices.

5.5. Insights and Implications

Several key insights and implications arise from the results:

- **Predictive Power Management**: Predictive ML models allow better adjustment to the SoC's power settings. The framework can significantly lessen energy consumption without sacrificing performance by anticipating workload changes. One of the advantages of this approach to power management over traditional ones is that it is proactive instead of reactive, reacting through workload changes.
- **Workload Adaptation**: The diversity of workloads that the ML framework can handle is very effective. The system dynamically categorizes workloads and selectively varies the power used in various workloads, from purely compute to pure memory and mixed usage.
- **Scalability**: With the ML models being lightweight, the framework can scale for various SoC architectures. It makes the models small in size with low memory and computational requirements; thus, they can be used on a lot of mobile device systems, including expensive high-end smartphones and cheaper power-constrained devices. Such scalability allows the framework to be used in various platforms without putting significant resource demands.

VI. CHALLENGES AND LIMITATIONS

Although this proposed ML framework for SoC power optimization shows promise, many challenges and limitations still need to be addressed to make the framework applicable and scalable for widespread users. These challenges are categorized as follows:

- **Data Availability and Quality:** Accurate ML models are only possible if we have high-quality runtime telemetry data. Yet there are issues of privacy concerns, lack of standardization over the broad range of various SoC architecture designs, and noisy or missing telemetry data as obstacles to data collection. This poor quality of data can have negative effects on the model's performance, overfitting or poor generalization and, in turn, can demotivate an effective framework.
- **Computational and Memory Overhead:** Even lightweight ML models are indeed difficult to deploy in resource-constrained environments. To support a system's ongoing functionality, it is paramount for their inference latency to remain low and memory use to be minimized. The computational overhead inherent to their processing could negate the power savings, especially in small-scale or legacy SoC designs.
- **Generalization across Workloads:** The framework's performance is very sensitive to the diversity of the training dataset. Specific workloads may not be realistically accommodated by models trained on a particular set of workloads unless they include experiences that suit the particular workloads being faced. However, the use of inconsistent optimization across diverse use cases translates into inefficiency in power management as well as lower performance in real-world applications.
- **Integration Complexity:** There is a gap between the capabilities of present-day ML models and existing SoC architectures, which vary in configuration, such as PMUs and voltage/frequency control mechanisms. Welcoming such diversity can create pitfall compatibility issues, and coordination with hardware levels such as DVFS and thermal management adds an hour of time, resulting in huge development time but also reducing adaptability across different platforms.
- **Training Time and Cost:** The training of ML models, and particularly of deep learning approaches, is time and resource-consuming. Training offline can be expensive, and, in some instances, it is impossible to perform the updates continuously. Furthermore, in budget-constrained environments, it can deter adoption when the workload or usage pattern changes, requiring retraining.

VII. FUTURE DIRECTIONS

7.1. Advanced ML Techniques

Machine learning techniques will play a crucial role in future path power optimization for mobile SoC. The integration of Explainable AI (XAI), where transparent ML models such as decision trees or attention mechanisms will allow developers to know and be confident in power optimization models' decisions, remains one promising direction. In safety-critical applications, this will be important. Finally, meta-learning and auto-ML techniques will contribute significantly to enabling models to change dynamically in different workloads and architectures with no manual intervention, reducing the need to retrain unnecessarily. Instead of pushing sensitive data to centralized servers, federated learning enables models to train locally on edge devices, ensuring that user privacy is maintained and that mobile and IoT-type environments can be better served.

7.2. Real-Time Adaptation and Hardware-Software Co-Design

Future research will focus on real-time adaptation. Real-time power management can be optimized dynamically by using reinforcement learning agents that can design self-learning systems that can continuously optimize their power management with real-time feedback as workloads change. Moreover, deploying dynamic models will be more energy efficient without compromising performance as more intensive tasks are delegated to lightweight models, and more sophisticated tasks are handed over to bigger models at peak demand. The inclusion of ML-friendly hardware architectures, such as Neural Processing Units (NPUs), will drive the evolution of hardware-software co-design in accelerating ML-driven power management tasks. Future optimization strategies will be further refined using more granular on-chip telemetry data to enhance on-chip telemetry support.

7.3. Sustainability, Cross-Domain Applications, and Collaborative Optimization

Following this, future research will explore decarbonizing the power usage and ML training process of SoC. We will also build battery-aware optimization techniques that kindle the battery life as much as possible while keeping the performance good and providing a better user experience. By adapting ML frameworks for IoT and edge devices in cross-domain applications, efficient power management for the resource-constrained environment will be possible. Solutions to these problems will scale these techniques to datacenters to address the increasing energy efficiency requirements of cloud computing. Additionally, collaborative optimization frameworks that facilitate multi-device coordination across interconnected devices like smartphones and IoT systems will be available to obtain network-wide energy savings. So widespread will prevail the adoption of ML-based power optimization across industries, all thanks to standardized optimization protocols that will ensure compatibility for MPSoCs with diverse and heterogeneous SoC architectures.

VIII. CONCLUSION

Finally, the integration of machine learning techniques into mobile SoC power optimization provides a significant enhancement of power efficiency without the need for performance losses. Predictive models trained from real-time telemetry data can then be leveraged to adjust the power settings dynamically, like voltages and frequency, to agree with the current workload. It adapts to ensure that during low-demand cases, power consumption is minimized while enough resources are allocated during peak workload cases. We demonstrate how this framework realizes promising energy savings over different workloads, inducing minimal impact of task latency and throughput on modern mobile applications, and it serves as a viable solution but with no additional performance overhead.

Although the application of machine learning models for power optimization has some inconveniences, such as data quality, computational overhead and adaptation to dynamic workloads, these challenges, notwithstanding inherent model interpretability, real-time adaptation and hierarchical hardware/software design, have been making progress on more efficient and scalable solutions. Future work will also consider maximizing the generalization of the model, decreasing the training costs, and allowing the model to interface seamlessly with next-generation mobile SoCs. In the end, as machine learning becomes more refined, it will be instrumental to delivering more performance and energy management to the next generation of mobile SoC.

REFERENCES

1. Ajani, T. S., Imoize, A. L., & Atayero, A. A. (2021). An overview of machine learning within embedded and mobile devices–optimizations and applications. Sensors, 21(13), 4412.
2. Cai, H., Lin, J., Lin, Y., Liu, Z., Tang, H., Wang, H., ... & Han, S. (2022). Enable deep learning on mobile devices: Methods, systems, and applications. ACM Transactions on Design Automation of Electronic Systems (TODAES), 27(3), 1-50.
3. Qin, S. J., & Chiang, L. H. (2019). Advances and opportunities in machine learning for process data analytics. Computers & Chemical Engineering, 126, 465-473.
4. Zhou, L., Pan, S., Wang, J., & Vasilakos, A. V. (2017). Machine learning on big data: Opportunities and challenges. Neurocomputing, 237, 350-361.
5. Grosse, P., Durand, Y., & Feautrier, P. (2009). Methods for power optimization in SOC-based data flow systems. ACM Transactions on Design Automation of Electronic Systems (TODAES), 14(3), 1-20.
6. Pezzini, P., Gomis-Bellmunt, O., & Sudrià-Andreu, A. (2011). Optimization techniques to improve energy efficiency in power systems. Renewable and Sustainable Energy Reviews, 15(4), 2028-2041.
7. Benini, L., & Micheli, G. D. (2000). System-level power optimization: techniques and tools. ACM Transactions on Design Automation of Electronic Systems (TODAES), 5(2), 115-192.
8. Ayalew, M. F., Hussen, S., & Pasam, G. K. (2018). Optimization techniques in power system. International Journal of Engineering Applied Sciences and Technology, 3(10), 2455-2143.
9. Reich, Y., Konda, S. L., Levy, S. N., Monarch, I. A., & Subrahmanian, E. (1993). New roles for machine learning in design. Artificial intelligence in engineering, 8(3), 165-181.
10. Perera, A. T. D., Wickramasinghe, P. U., Nik, V. M., & Scartezzini, J. L. (2019). Machine learning methods to assist energy system optimization. Applied energy, 243, 191-205.
11. Jung, H., & Pedram, M. (2010). Supervised learning based power management for multicore processors. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 29(9), 1395-1408.
12. Carvalho, L. F., Barbon Jr, S., de Souza Mendes, L., & Proenca Jr, M. L. (2016). Unsupervised learning clustering and self-organized agents applied to help network management. Expert Systems with Applications, 54, 29-47.

|www.ijirset.com |A Monthly, Peer Reviewed & Referred Journal| e-ISSN: 2319-8753| p-ISSN: 2347-6710|

Volume 13, Issue 12, December 2024

|DOI: 10.15680/IJIRSET.2024.1312117|

13. Boulesnane, A., & Meshoul, S. (2021, July). Reinforcement learning for dynamic optimization problems. In Proceedings of the Genetic and Evolutionary Computation Conference Companion (pp. 201-202).
14. Ku, C. Y., & Liu, T. T. (2019). A voltage-scalable low-power all-digital temperature sensor for on-chip thermal monitoring. IEEE Transactions on Circuits and Systems II: Express Briefs, 66(10), 1658-1662.
15. Chung, C. C., & Yang, C. R. (2011). An autocalibrated all-digital temperature sensor for on-chip thermal monitoring. IEEE Transactions on Circuits and Systems II: Express Briefs, 58(2), 105-109.
16. Dong, G., & Liu, H. (Eds.). (2018). Feature engineering for machine learning and data analytics. CRC press.
17. Fachantidis, A., Taylor, M. E., & Vlahavas, I. (2017). Learning to teach reinforcement learning agents. Machine Learning and Knowledge Extraction, 1(1), 21-42.
18. Chen, S. J., Lin, G. H., Hsiung, P. A., & Hu, Y. H. (2009). Hardware software co-design of a multimedia SOC platform. Springer Science & Business Media.
19. Moldovan, A. N., Weibelzahl, S., & Muntean, C. H. (2013). Energy-aware mobile learning: Opportunities and challenges. IEEE Communications Surveys & Tutorials, 16(1), 234-265.
20. Zanella, M. (2015). Energy-aware runtime management of distributed mobile devices.

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY