INNOVARE JOURNAL OF ENGINEERING & TECHNOLOGY

Vol 13, 2025

ISSN - 2347-1573

Review Article

MACHINE LEARNING TECHNIQUES FOR POWER AND PERFORMANCE OPTIMIZATION IN MOBILE SYSTEM-ON-CHIPS: A SURVEY

SACHIN MANEKAR*

Department of Computer Sciences and Applications, Mandsaur University, Mandsaur, Madhya Pradesh, India.
Email: sachin.manekar@meu.edu.in

Received: 17 September 2025, Revised and Accepted: 22 November 2025

ABSTRACT

The design of the system-on-chip (SoC) must be able to balance between power and performance since the demands of more individuals to have high-performance phones with extended battery lives are growing. Machine learning (ML) methods have become important to optimizing power and performance in handheld SoCs, which need to put a tradeoff between computational performance and energy efficiency in thermally limited conditions. ML can be used to make smart choices to scale in real-time to dynamic voltage and frequency, optimize schedules on the fly, and predict workloads, which results in a better battery life and responsiveness in real-time in mobile devices. ML has emerged as a groundbreaking solution to the power–performance dilemma. Intelligent decisions related to scheduling of tasks, scaling of hardware, and allocation of resources in real time can be made based on patterns of user behavior, workload, and thermal conditions by learning with ML algorithms. The survey discusses the principles of mobile SoC design, identifies the backgrounds of ML that are applied to optimization, and summarizes recent works on security, thermal management, memory layout, and clock power minimization. A comparative analysis identifies the research gaps that exist at the moment, underlining the fact that the current solutions are fragmented, and holistic and cross-layer structures are required.

Keywords: Mobile system-on-chips, machine learning, Power optimization, Performance enhancement, Workload prediction, Energy efficiency, Dynamic voltage and frequency scaling.

 DOI: http://dx.doi.org/10.22159/ijet.2025v13.57904. Journal homepage: https://innovareacademics.in/journals/index.php/ijms

INTRODUCTION

Mobile devices have quickly developed beyond mere communication devices into inseparable computing environments that define contemporary life. In the last 10 years, smartphones have been in the spotlight, especially smartphones, which provide services otherwise offered by personal computers, such as web browsing, calculation, and electronic payment. This has been eased by the development of the Mobile System-on-Chip (SoC) technology, which integrates various functions such as central processing units (CPUs) [1], graphics processing units (GPUs), neural processing units (NPUs), and modems in a single chip. By integrating all these units, SoCs can offer both high computing and be deployed at the same time in more difficult applications such as artificial intelligence (AI), 5G connectivity, and high-resolution imaging [2]. There is an underlying problem, however, in that the increasing complexity of mobile SoCs is presenting a challenge: to find a balance between performance and energy economy. The power capacity and thermal limit of mobile devices are far worse than those of desktop machines, where power consumption can be unreasonably high, leading to a worse user experience and shorter device life. To address this, the trade-offs between the power demands and the computational throughput of SoCs have been taken care of over the long-term by SoC designers. Some of the attempts that are currently being made to ensure performance remains high and battery life stays long include adaptive resource scheduling, workload estimation, and dynamic voltage and frequency (DVFS).

The mobile workloads are becoming diverse and unpredictable; therefore, it is no longer possible to work with the traditional optimization techniques. Machine learning (ML) is a solution that comes in at this stage. The ML algorithms can reason in real time with intelligent decisions regarding how tasks are scheduled, hardware is scaled, and resources are provided by learning patterns of user behavior, workload dynamics, and thermal conditions. By so doing, not only does ML make the performance of the already established mechanisms, such as the DVFS and the adaptive scheduling, more efficient, but also allows novel types of optimization to be performed, specific to heterogeneous SoC elements [3]. The intersection of mobile computing and ML-based optimization, therefore, is one of the most important frontiers in SoC design. ML-enabled solutions to the power-performance dilemma offer scalable, adaptive, and context-aware solutions to the growing complexity of chips and the increasing data-rich content. The present survey tries to explore some of the current developments in ML techniques in mobile SoCs, including the way workload prediction, DVFS, and adaptive scheduling are being reconsidered with intelligent algorithms. It is through the synthesis of these developments that we hope to offer a holistic view of how ML is transforming energy-saving yet high-performance mobile computing environments.

Structure of the paper

The structure of this paper is as follows: Section II reviews the fundamentals of mobile SoC design, Section III introduces ML foundations for SoC optimization, Section IV presents a literature review of recent ML-based techniques, and Section V concludes with future directions.

FUNDAMENTALS OF MOBILE SOC DESIGN

A System-on-a-Programmable-Chip (SoPC) is constructed out of Programmable Logic Devices (PLDs); it has all the benefits of simple development, quick prototyping, and the ability to change easily [4]. SoPCs are most often based on Field Programmable Gate Arrays (FPGAs) and use VHSIC Hardware Description Language (VHDL) to define the hardware and are targeted at prototyping and low and medium volume manufacturing. FPGAs, as a major development in Very-Large-Scale Integration (VLSI) design, are more advantageous in terms of efficiency, shorter time of development, and convenience in future changes. Alternatively, the other way of developing a mobile SoC is the Application-Specific Integrated Circuits (ASICs) that offer superior performance and power consumption when keeping the fixed-function applications, although with less flexibility. Fig. 1 shows one generic design flow of mobile SoCs.

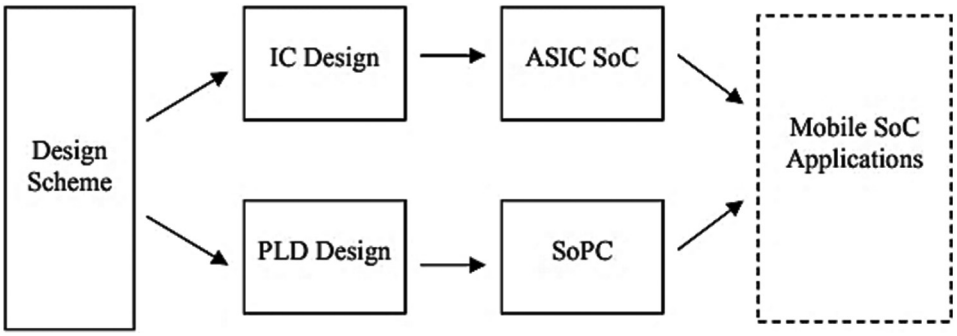

Fig. 1: The design procedure of a mobile system–on–chip

Two directions in the design of mobile SoC applications are a single route that is based on IC design that leads into high-end ASIC SoC, and the other route is based on PLD design that leads into SoPCs in easy-updatable solutions that meet the demands of the mobile application solutions [5].

Multi-processor system-on-chip (MPSoC)

MPSoC is a type of SoC that can have different kinds of computing units, memories, input/output devices (I/O), and other parts, like in [6]. Three sorts of components make up the MPSoC architecture: software subsystems, hardware subsystems, and inter-subsystem communication [7]. Hardware subsystems, software subsystems, and communication between subsystems make up the MPSoC architecture. The Hardware–Software Co-Simulation/Hardware–Software System (HW-SS) stands for the specialized hardware subsystems that carry out the operation of an application or the global memory subsystems. The HW-SS includes both general hardware components and those that provide communication inside subsystems, as shown in Fig. 2. A computer system's hardware either represents global memories that other systems can access or performs the application's intended tasks.

A single software stack arranged in a multithreaded application is executed by the MPSoC architecture in this situation. Whether the software stacks are identical or not, the message-passing organization runs them all [9]. Most of the time, message forwarding is how subsystems communicate with one another. An essential feature of this class is the fact that all of the processors can explicitly communicate with each other through I/O operations. CPUs exchange messages with one another through the use of primitive operations like send and receive.

Strategies for energy-efficiency in SoCs

Modern microarchitecture has made energy efficiency in SoC design a primary goal in response to the rising performance demands of mobile and embedded systems [10,11]. In order to lower power consumption while keeping performance levels sufficient, many methods have been devised. Considering the paramount relevance of mobile platform energy efficiency, this section offers a synopsis of important design concepts that strive to optimize power utilization.

Despite their generalizability, these tactics shine when used in the development of on-chip communication networks that minimize power consumption. The following foundational principles serve as a basis for developing power-conscious SoC architectures (Fig. 3) and SoC strategies:

Involve all layers

The physical layer, the application itself, the system architecture, the operating system, and the communication protocol stack are all affected by the problem of energy efficiency. A system must be adjusted for energy in every part of its design to be energy-efficient. Using parts that have been tuned for low power consumption does not guarantee the most energy-efficient system [12].

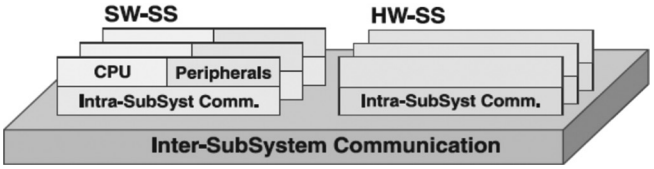

Fig. 2: Architecture of multi-processor system–on–chip [8]

Applying locality of reference

The term "locality of reference" describes the practice of keeping frequently accessed data in proximity to the components that do the processing. It is significantly more energy efficient to access a tiny, local memory rather than a large, distant one.

Avoid useless activity

This serves as the primary impetus for dynamic power management, link-layer protocols, and adaptive error correction. Superfluous activity may arise from multiple sources throughout the system, including redundancy in a high-power operational mode, implementing error control on inherently error-resilient data, or endeavoring to transmit a video frame that is already obsolete.

Dynamic voltage and frequency scaling (DVFS)

DVFS is a popular method that optimizes processor power and frequency in response to changing workloads. Reducing voltage and frequency during periods of low computing demand can result in considerable power savings due to the quadratic relationship between voltage and dynamic power usage. Various DVFS policies leverage workload prediction, thermal thresholds, and application-specific requirements to optimize energy use while preventing performance degradation.

Power gating and clock gating

Power gating minimizes the amount of power lost to the idle functional blocks by shutting them off completely, thus the leakage currents are suppressed. Conversely, clock gating de-asserts the clock signal to certain circuits that are not used actively, avoiding unneeded switching and thus lowering dynamic power requirements. Hardware-based state monitors or software-defined policies that determine idle resources and turn them off in a selective manner tend to control these gating mechanisms [13].

Low-power states and idle modes

Recent SoCs add numerous power states, which are specified by the ACPI or related standards. A big reduction in the amount of power used can be made by transitioning to low-power or sleep modes when the system is idle, for example, standby, deep sleep, or hibernation. Hardware mechanisms and firmware policies that contribute to these modes deal with entry and exit latencies so as to offer seamless performance recovery with little energy expenditure.

Approximate and opportunistic computing

Approximate computing leverages the error tolerance of certain applications, such as multimedia processing or ML inference, to relax computational accuracy in exchange for energy savings. Using reduced-precision arithmetic, skipping of tasks, or over-scaling voltage, SoCs may reduce their energy usage without significant impacts on output quality. Opportunistic computing also uses the differences in the characteristics of workloads and environmental conditions to optimally change the fidelity of computation or the use of resources.

Challenges in power modeling in smartphone SoCs

The complexity of the modeling of the interaction between many heterogeneous components is demonstrated in Table 1. Smartphone SoCs combine many functional blocks with intricate power relationships and share resources [14]. Conventional methods that use independent-component models, do not reflect the interactions in the system that cause a strong influence on power consumption.

ML FOUNDATIONS FOR SOC OPTIMIZATION

The algorithmic framework of ML primarily involves intensive utilization of historical, huge quantities of training data to prepare learning algorithms that consequently produce models to be utilized in prediction and classification [16,17]. ML algorithms have proven to be remarkably general and flexible in a broad variety of areas of application, allowing some promising performance [18]. Fig. 4 demonstrates the four key stages of a common ML workflow of hardware vulnerability analysis.

The first stage entails the gathering of the sets of data related to the specified hardware weaknesses. The second step includes training the ML model using this dataset. The third stage involves evaluation of the trained model. Finally, the proven model can be used to identify the unknown vulnerabilities. As the use of complex and heterogeneous SoC technologies has increased, ML is now a vital device in the optimization of power and performance [19]. ML can dynamically respond to the change of workloads, usage patterns, and thermal conditions, which is crucial in real-time, resource-constrained systems such as smartphones, wearables, and Internet of Things forms of ML Techniques Used [20]:

Supervised learning

Supervised learning is important in modern multi-processor SoC where intelligent and efficient power management is the objective. The typical method is to train a power manager that explains system performance state in terms of easily observed features, for example, the occupancy level of a global service queue [21]. This supervised learning model, when applied using such models as the Bayesian classifier, greatly decreases the time lag in its running.

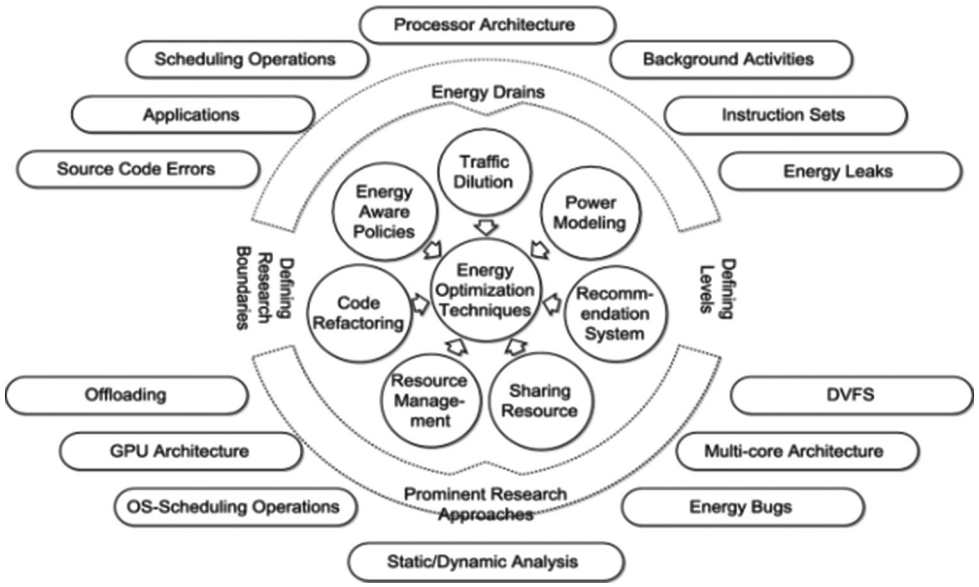

Fig. 3: Energy-efficient core design in mobile processors

Table 1: Key components and power modeling challenges in smartphone SoCs [15]

SoC component	Power consumption characteristics	Modeling challenges
CPU clusters	Heterogeneous architectures with big.LITTLE or DynamIQ configurations consuming different power levels based on workload intensity	Accurately capturing dynamic frequency scaling effects and core-specific power variations across different utilization patterns
GPU subsystem	Power consumption varies significantly with rendering complexity and utilization of specialized hardware features such as tile-based deferred rendering	Modeling the correlation between shader complexity, memory bandwidth utilization, and power consumption across diverse visual workloads
Memory subsystem	Power consumption varies with bandwidth utilization, access patterns, and active power management features	Accounting for complex interactions between memory controller states, DRAM power modes, and application memory access patterns
Specialized accelerators	Significantly better energy efficiency compared to general-purpose processors for domain-specific tasks such as AI inference and image processing	Developing accurate models for novel hardware architectures with limited historical data and rapidly evolving designs
Modem and connectivity	Variable power profiles depending on signal strength, data transmission rates, and active protocol (5G/4G/Wi-Fi/BT)	Capturing the relationship between signal conditions, transmission power, and baseband processing requirements

SoCs: System-on-chips, CPU: Central processing units, GPU: Graphics processing units, NPUs: Neural processing units, DRAM: Dynamic random access memory

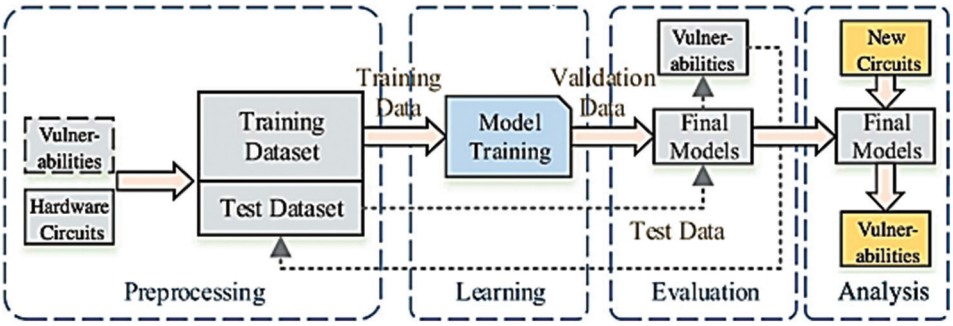

Fig. 4: The general flow of machine learning techniques

Unsupervised learning

Unsupervised approaches (e.g., clustering or regression) extract latent workload patterns or predict system parameters in network-on-chip environments. For example, the LEAD framework applies supervised regression on buffer-traffic data and also uses reinforcement learning (RL) for optimal voltage-frequency (VF) decisions.

RL

SoC elasticity, the use of RL is advantageous for DVFS management. In contrast to supervised learning, RL agents (including the Q-learning ones) determine the optimal choices of VF values based on multiple interactions with the environment using the feedback, namely the consumption of energy levels and performance indicators [22].

Emerging role of ML in SoC design

ML has recently expanded on the capabilities of conventional power optimization methods. The following benefits are offered by ML-based methods [23]:

- Predictive power management: SoC can proactively adjust power settings in response to workload patterns anticipated by ML models using past data. Instead of waiting for a shift in workload to occur, predictive models may power them up or down as the work changes, saving electricity and maintaining efficiency even during peak demand
- Dynamic resource allocation: A SoC can use ML to dynamically assign resources (such as processor cores, memory, or accelerators) in response to actual demand. That way, can improve performance per power ratio by just powering the resources that are actually needed, while letting idle resources go into low-power states
- Energy-efficient scheduling: ML algorithms can be used to plan jobs in a way that minimizes energy use. To provide one concrete example, and can assign low-power cores to low-power jobs and offload high-performance or specialized accelerators to workloads that demand more processing power. The result is that resources can be allocated in an energy-efficient manner, allowing for the meeting of deadlines.

SOC testing

The term "System-on-Chip" describes the recent trend of consolidating various ICs into a single, more complicated unit [24]. The number of transistors per chip has increased as a result of technological advancements in silicon shrinkage. As a result, test time is significantly increased due to an increase in both the number of defects and test vectors. Research on ways to shorten test duration is an area of interest within the SOC design paradigm [25]. This SOC Testing focuses on the following primary areas [26]:

- Managing all aspects of SOC testing
- Integrity and user-defined logic testing.
- Various vendor-provided testing cores with varying functionality
- Integrating the main inputs and outputs of the system with the cores.

LITERATURE REVIEW

This section presents a literature review of power and performance optimization techniques for mobile SoCs, emphasizing thermal management, memory allocation, and security enhancements. It presents other modern innovations and developments based on ML and architecture. To provide a concise overview, a summary of the reviewed studies, their main approaches, findings, challenges, and directions is provided in Table 2.

Lee *et al*. examine security risks unique to mobile SoCs built on heterogeneous chiplet technology and suggest a robust security architecture to mitigate the risks. Based on case studies on Android mobile systems, suggest an optimized architecture, which is founded on threat modeling and vulnerability classification in different security situations. The suggested architecture ensures the current level of security in mobile SoCs and effectively partitions IPs with the aim of reducing security threats without incurring high costs [27].

Cho *et al*. explore the incorporation of sophisticated features into the compact physical size of a mobile SoC. Their analysis sheds light on the design of the application processor (AP) to support more system blocks and features on a smaller silicon surface, and underlines the issues of space-efficiency in current SoC architectures. The temperature control may impair the performance that can be realized in circumstances of thermal runaway, which can occur when heat from various system blocks on the same chip interacts. Based on estimating the amount of heat emitted by each system in the system when it is introduced to the chip, the floor-planning can be done in a manner that minimizes the maximum temperature of a scenario operation. When a block of the system turns out to be a hot spot, its position is also altered after visiting the temperature map, which is created as a result of thermal simulation. The final temperature-optimized floorplan is arrived at by replicating the steps of waiting till the outcome is retrieved and asking the simulation to be performed again [28].

Im *et al*. present a fresh implementation of the linear parameter varying (LPV) reduced-order model (ROM) that is able to consider nonlinear factors. The proposed modified LPV ROM approach, which was adopted instead of flow velocity, employed the parameter of the heat transfer coefficient (HTC) as the parameter of scheduling. The concept was demonstrated using a prototype smartphone that has an AP. Before that, the simulated transient thermal performance of the smartphone was used to obtain responses of the HTC to different levels of conduction. Furthermore, the LTI ROM was created at different HTCs. In addition, to establish the connection between the surface temperature and the HTC, a transient thermal simulation of the computational fluid dynamics (CFD) was employed to determine the relationship. Finally, the instant LTI ROM was updated after each time step based on the surface temperature to make it possible to run fast thermal simulations with the modified LPV ROM. Analyzed the short-time behavior of modified LPV ROM, linear time-invariant (LTI) ROM, and overall CFD simulation to make sure that the modified ROM was correct. The modified LPV ROM was found to be over 88,000 times faster than the current CFD simulation at solving time to allow the allowable temperature error to be reached in a dramatically shorter period of time [29].

Table 2: Comparative analysis of recent studies on ml techniques for power and performance optimization in mobile SoCs

Reference	Study on	Approach	Key findings	Challenges	Future direction
Lee *et al*., (2025)	Security in Mobile SoCs with Chiplets	Threat modeling, vulnerability categorization, secure IP partitioning	Designed a low-cost security architecture for heterogeneous chiplet-based mobile SoCs	Difficulty in balancing performance, security, and cost	Use of ML for real-time threat detection and adaptive hardware trust zoning
Cho *et al*., (2023)	Thermal-aware floorplanning in SoCs	Iterative simulation and real-time temperature mapping	Optimized block placement to reduce hotspot formation and improve performance	Long design cycles due to repeated thermal simulations	Apply ML models for fast thermal prediction and layout optimization
Im *et al*., (2023)	ROM modeling for thermal prediction	Modified LPV ROM using HTC as scheduling parameter	Achieved 88,000×faster thermal simulation versus CFD with acceptable error	Handling nonlinearities in heat transfer via HTC correlation	ML-based real-time thermal modeling using surrogate models
Mishra *et al*., (2023)	Thermal/power co-optimization at advanced nodes	System-level thermal resistance breakdown	Identified co-optimization paths in chip-package-cooling systems	Thermal limits constrain SoC scaling in angstrom nodes	AI-driven thermal-aware scheduling and dynamic cooling strategies
Agrawal *et al*., (2023)	DRAM memory optimization for DNNs	Best-fit heuristic for buffer allocation, concat-aware strategy	Achieved average 15% DRAM footprint reduction, up to 35% in some models	Limited DRAM and high memory demands of DNNs	Apply reinforcement learning to memory scheduling and allocation
Lee *et al*., (2022)	Clock power optimization in mobile SoCs	Clock selection optimization, recursive config, PLL on/off automation	Reduced power via aggressive clock gating with micro-architectural support	Managing hundreds of PLLs and thousands of MUX configurations	ML-based clock usage prediction and dynamic control
Hort *et al*., (2022)	Android application performance on SoCs	Survey of 156 studies focusing on responsiveness, energy, memory	Non-functional optimization critical for user experience and power efficiency	Trade-off between responsiveness and resource consumption	ML-guided profiling and dynamic tuning of application behavior

CFD: Computational fluid dynamics, DNNs: Deep neural networks, DRAM: Dynamic random access memory, HTC: Heat transfer coefficient, ML: Machine learning, PLLs: Phase-locked loops, SoCs: System-on-chips, LPV: Linear parameter varying, ROM: Reduced-order model

Mishra *et al*. emphasize the fact that the growing computational needs in consumer electronics, including mobile phones and automobiles, and data centers to support HPC applications, have posed major thermal problems. The main problems are that these issues are caused by the ever-growing transistor density that results in power density. Moreover, further complicating the thermal constraints is an enhanced packaging technology, such as 2.5D and 3D integration. Determine the temperature of hot objects and the amount of cold air needed in future thermo-limited SOCs in the A14 and A5 advanced Angstrom nodes. To explore the possibility of a co-optimization of the chip-package-cooling system, it must be divided into its contributions, and that is, the thermal resistance breakdown has to be done in different sources. Some of the findings of the analysis could be utilized by the system software to carry out thermally aware work scheduling [30].

Agrawal *et al*. deal with the issue by suggesting two methods for maximizing the distribution of dynamic random access memory (DRAM). To improve the allocation, the first technique iteratively employs best fit and aims to maximize the use of the various buffers utilized during inference. To handle feature map allocation issues pertaining to the concat operation, investigate a best-fit alternative in the second approach. Using an experimental set of proprietary deep neural networks (DNNs), the suggested approach was able to average a 15% reduction in memory footprint. The reduction for DNNs in a certain category exceeds 35% [31].

Lee *et al*. highlight the significant area and power overhead associated with clock sources such as phase-locked loops (PLLs) in mobile SoCs, noting their widespread sharing among numerous clock-consuming components to maximize resource efficiency. Tens of PLLs and hundreds of multiplexers and dividers are built into modern SoCs to send clocks to thousands of users spread out across the die. However, due to the dynamic nature of mobile workloads, only a limited subset of circuits is active at any given time. Mobile SoCs demand more adaptive methods, unlike traditional methods that are targeted at environments where parallelism and homogeneity are high. Although the techniques at the circuit level, such as resonant clocking and adiabatic drivers, can provide a certain power protection, further advantages are obtained by including violent clock gating at the micro-architectural level. Optimization of clock selection, automatic control of PLL on/off, software transparency, and support for recursive setup are some of the advancements introduced by the proposed clock management architecture [32].

Hort *et al*., the success of mobile platforms and the happiness of their users depend on the timely delivery of high-performance mobile applications. This is especially crucial for systems with limited resources, such as mobile devices. User satisfaction is thus heavily influenced by non-functional performance parameters such as energy and memory consumption. The purpose of this research was to compile a thorough literature review on optimizing the non-functional performance of Android apps. For this study, Examined through 156 original articles published between 2008 and 2020 that address the topic of mobile app performance optimization [33].

Research gap

However, regardless of tremendous progress in mobile SoC security, thermal optimization, memory allocation, and performance improvement, the current research is enormously specialized and atomized in the context of isolated problems. Studies in this area have concentrated more on domain-specific solutions, including chiplet-based security architectures, thermal-aware floorplan, ROM-thermal modeling, DNNs-DRAM optimization, and clock-management strategies, but have not provided a combined framework to consider security, thermal efficiency, and resource management as a single domain. In addition to this, the majority of research is based on controlled simulations or proprietary datasets, which do not allow generalization to the real world. Holistic and cross-layer solutions are evident where hardware, software, and optimization of the systems have to work together to achieve the increased complexity of next-generation mobile SoCs.

CONCLUSION AND FUTURE WORK

ML Techniques for Power and Performance Optimization in Mobile SoCs have shown immense promise in addressing critical challenges related to thermal management, energy efficiency, memory utilization, and real-time performance. By leveraging supervised, unsupervised,

and RL models, SoCs can dynamically adapt to fluctuating workloads and environmental conditions, enabling intelligent power control and efficient resource allocation. This survey has explored the increase in the use of ML to optimize power and performance in mobile SoC platforms. Modern SoCs with CPUs, GPUs, NPUs, and modems have a high potential, such as AI, 5G, and high-resolution imaging, but still have to deal with the current challenge of balancing computational throughput and energy efficiency with limited battery and thermal constraints. Traditional approaches such as DVFS, workload forecasting, and adaptive scheduling have not been fully complete but were limited to suit different and unpredictable workloads. ML has emerged as a transformative approach that offers smart and real-time results in task scheduling, hardware scaling, and resource allocation. ML enhances existing processes and introduces novel optimization options, mirrored by heterogeneous SoC components, through the application of supervised, reinforcement, and unsupervised learning. The future research must seek holistic and cross-layer models, real-world data, and light adaptive models, and promising aspects include thermal-aware scheduling, RL-based memory, and context-sensitive optimization to guarantee sustainable, energy-efficient, high-performance mobile computing.

REFERENCES

1. Reddy S.S. Comparative analysis of CPU scheduling algorithms for performance efficiency. 2019.
2. Saba I, Arima E, Liu D, Schulz. Orchestrated co-scheduling, resource partitioning, and power capping on CPU-GPU heterogeneous systems via machine learning. In: Lecture Notes in Computer Science. Vol. 13642. Cham: Springer; 2022. p. 51-67.
3. Mittal S. A survey of techniques for improving energy efficiency in embedded computing systems. Int J Comput Aided Eng Technol 2014;6:440.
4. Liu XW, Liu LM. A mobile computing SoC design. Adv Mater Res 2012;605-607:2049-2052.
5. Kapadia HP. Cross-platform UI/UX adaptions engine for hybrid mobile apps. Int J Nov Res Dev 2020;5:30-37.
6. Popovici K, Rousseau F, Jerraya AA, Wolf M. Embedded Software Design and Programming of Multiprocessor System-on-Chip. New York, NY: Springer; 2010.
7. Duggasani AR. The integration of java-based applications in software development and AI: A conceptual review. Int J Adv Res Sci Commun Technol 2025;5:11.
8. Kumar ND, Mohan G. Design and optimization of system-on-chip (SOC). J Emerg Technol Innov Res 2015;2:234-9.
9. Kalava SP. Enhancing software development with AI-driven code reviews. North Am J Eng Res 2024;5:1-7.
10. Kumar S, Singh R, Kumar S, Gupta S. Light Weight Resnet for Detection of Wheat Yellow Rust Over Mobile Captured Images from Wheat Fields. In: 2023 3rd Asian Conference on Innovation in Technology (ASIANCON). IEEE; 2023.p. 1-4.
11. Patel R. Sustainability and energy management: Trends and technologies for a greener industrial future. Int J Adv Res Sci Commun Technol 2024;4:886-98.
12. Gupta S. Design strategies for mobile SoCs: Balancing battery life and computational demand. J Glob Res Math Arch 2025;12:12-20.
13. Maddali G. Efficient machine learning approach based bug prediction for enhancing reliability of software and estimation. Int J Res Eng Sci Manag 2025;8:1-7.
14. Vangaru KK. Accurate power modeling and estimation for smartphone SoCs using postsilicon correlation. Sarcouncil J Multidiscip 2025;5:514-23.
15. Bhat G, Mandal SK, Manchukonda ST, Vadlamudi SV, Agarwal A, Wang J, *et al*.Per-core power modeling for heterogenous SoCs. Electronics 2021;10:2428.
16. Pan Z, Mishra P. A survey on hardware vulnerability analysis using machine learning. IEEE Access 2022;10:49508-27.
17. Prajapati N. The role of machine learning in big data analytics: Tools, techniques, and applications. ESP J Eng Technol Adv 2025;5:16-22.
18. Dattangire R, Vaidya R, Biradar D, Joon A. Exploring the Tangible Impact of Artificial Intelligence and Machine Learning: Bridging the Gap between Hype and Reality. In: 2024 1st International Conference on Advanced Computing and Emerging Technologies (ACET). IEEE; 2024. p. 1-6.
19. Majumder RQ. Machine learning for predictive analytics: Trends and future directions. Int J Innov Sci Res Technol 2025;10:3557-64.
20. Patel R. Optimizing communication protocols in industrial iot edge networks: A review of state-of-the-art techniques. Int J Adv Res Sci Commun Technol 2023;4:1-12.
21. Jung H, Pedram M. Supervised learning based power management for multicore processors. IEEE Trans Comput Des Integr Circuits Syst 2010;29:1395-408.
22. Molnos A, Lesecq S, Mottin J, Puschini D. Investigation of Q-learning applied to DVFS management of a System-on-Chip. IFAC Pap Online 2016;49:278-84.
23. Panchal V. Mobile SoC power optimization: Redefining performance with machine learning techniques. Int J Innov Res Sci Eng Technol 2024;13:1-17.
24. Sakthivel P, Narayanasamy P. Test design and optimization for multiple core systems-on-a-chip using genetic algorithm. Int J Comput Sci Netw Secur 2006;6:121-9.
25. Efthymiou A, Bainbridge J, Edwards, DA. Adding Testability to an Asynchronous Interconnect for GALS SoC. In: Proceedings of the Asian Test Symposium; 2004.
26. Ravi S, Lakshminarayana G, Jha NK. Testing of core-based systems-on-a-chip. IEEE Trans Comput Des Integr Circuits Syst 2001;20:426-39.
27. Lee E, Park K, Lee J, Park DJ. Security Architecture for Heterogeneous Chiplet-Based Mobile SoC. In: 2025 IEEE International Conference on Consumer Electronics (ICCE); 2025. p. 1-6.
28. ChoY, Kim H, Lee K, Im Y, Lee H, Kim M. Thermal Aware Floorplan Optimization of SoC in Mobile Phone. In: 2023 22nd IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm); 2023. p. 1-7.
29. ImY, Jung G. Lee M, Gangrade A, Kim S. Thermal Modeling and Optimization of Mobile Device using modified LPV ROM. In: 2023 22nd IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm); 2023. p. 1-8.
30. Mishra S, Venkateswarlu S, Vermeersch B, Brunion M, Lofrano M, Abdi DB, *et al*. Towards Chip-Package-System Co-optimization of Thermally-limited System-On-Chips (SOCs). In: 2023 IEEE International Reliability Physics Symposium (IRPS); 2023. p. 1-7.
31. Agrawal S, Ghosh P, Kumar G, Radhika T. Memory Footprint Optimization for Neural Network Inference in Mobile SoCs. In: 2023 IEEE Women in Technology Conference (WINTECHCON); 2023. p. 1-6.
32. Lee JG, Jeon H, Choi Y, Kim A. Fully Automated Hardware-Driven Clock-Gating Architecture with Complete Clock Coverage for 5nm Exynos Mobile SoC. In: 2022 IEEE International Solid-State Circuits Conference (ISSCC); 2022. p. 216-8.
33. Hort M, Kechagia M, Sarro F, Harman M. A survey of performance optimization for mobile applications. IEEE Trans Softw Eng 2022;48:2879-904.