Center for Management Science Research
ISSN: 3006-5291 3006-5283 Volume 3, Issue 4, 2025

AI FOR OPERATING SYSTEM OPTIMIZATION: A REVIEW OF INTELLIGENT SCHEDULING, RESOURCE ALLOCATION, AND SECURITY

Shayan Abbas[1], Faryal Abbas Abdi[*2], Sumera Mehmood[3]

[1]*Research Assistant & Student, Sir Syed University of Engineering and Technology*
[*2]*Indus University*
[3]*PhD Scholar and Lecturer of Finance, IQRA University, Karachi, Pakistan*

[*2]faryal.abbas1996@gmail.com

DOI: https://doi.org/10.5281/zenodo.16991852

Keywords
Artificial Intelligence in Operating Systems, AI-driven Task Scheduling, Predictive Memory Management, Intelligent Resource Allocation, AI-enhanced OS Security, Reinforcement Learning in OS, Deep Learning for System Optimization, Kernel-Level AI Integration, Real-time AI Inference, OS Performance Benchmarking, Adaptive Operating Systems, Interpretable Machine Learning, Federated Learning for OS, Smart Containers and AI-kernels, Unified AI-OS Architecture

Article History
Received: 27 May, 2025
Accepted: 05 August, 2025
Published: 29 August, 2025

Corresponding Author: *
Faryal Abbas Abdi

Abstract
This paper presents a literature review of the incorporation of Artificial Intelligence (AI) in optimization of operating system (OS), and focused on the efforts of Artificial Intelligence in the area of task scheduling, resource allocation, memory management, and security management. Despite the fact that reinforcement learning, LSTM networks, and autoencoders are AI models that have demonstrated measurable improvements in specific OS subsystems, their application remains isolated and disjointed. The main limitations described in the research are the lack of cross-domain integration, inefficiency in real-time, non-interpretability of models, and the absence of common benchmarking processes. The paper presents the review of the recent literature with the assistance of which it is possible to focus on the need of low-latency, interpretable, and modular AI models that could be applied to the operating system components on the kernel level. It also highlights the importance of having open platform APIs and unhomogenized datasets to increase scalability and reproducibility. The findings can be applied as a guideline to the development of intelligent, adaptive and secure operating systems that are capable of self-optimization and autonomous decision making in a broad range of computing environments.

INTRODUCTION

The evolution of operating systems (OS) has been critical in the evolution of computation and its history can be traced back to the early 1950s when the computer architecture was dominated by the batch processing systems. OS design has changed over the years to develop into complex multitasking

environments but the problem of resource scheduling, system optimization and security still persisted especially in heterogeneous and distributed computing systems. In the last few decades, Artificial Intelligence (AI) has emerged as an effective tool to optimize a broad spectrum of computing paradigms, and it has inspired novel intelligent scheduling, predictive memory management, and adaptive security architectures [Zhang et al., 2024]. As networks shift to high-density cloud, edge, and mobile, the complexity of dynamic workloads is proving to be overwhelming to manage using static and rule-based OS strategies. To give an example, legacy kernel designs might lack real-time flexibility to support large scale concurrent processes or even strong security as threat models evolve [Wang & Xing, 2025].

Statistically, the global AI in IT operations market is projected to surpass USD 20 billion by 2027, which is a sign of the rising rate of automation in infrastructure-level decision-making. This growth is exponential, but the combination of AI models with low-level OS functionality remains in its infancy, and is often restricted to a particular application scenario, e.g., energy-aware scheduling or anomaly detection. Moreover, there exists a methodological fragmentation, in which AI systems that are intended to be run on cloud platforms are rarely ported to embedded OS systems or legacy kernels [Joloudari et al., 2022]. The lack has led to poor cross-platform adaptability and poor reproducibility of AI-optimized OS innovations, and the literature gap is substantial in the area of system-wide incorporation of AI into the entire OS stack [Zhang et al., 2024].

This discrepancy, in its turn, leads to the following problem statement: despite the huge potential of AI to enhance the capabilities of OS, most of the implementations remain siloed, and there is no single image of how they work in the real world regarding kernel optimization, task scheduling, memory allocation, and security enforcement. In this respect, the present study addressed that gap by conducting a secondary qualitative literature review of the integration of AI into the core OS functionality. Through a systematic review of peer-reviewed articles, 2020-2025, this research paper addresses how effective, applicable, and constrained AI models are in the context of operating systems [Ratnayake, 2024]. By doing so, it does not only summarize the current developments but also identifies the challenges, open research questions, and potential directions towards a unified AI-OS architecture [Zhang et al., 2024].

The primary research objective is to survey how the current AI techniques have been implemented into operating systems regarding intelligent scheduling, real-time resource allocation, predictive memory usage, and security monitoring. To achieve this, the study employed the qualitative literature review method that is more inclined towards interpretive research as opposed to empirical experimentation. The rationale of such an approach is that it is capable of deriving more information out of heterogeneous research findings, especially in new areas where quantitative meta-analyses are not possible due to the immaturity and non-standardization of experimental designs [Bikkasani & Yerabolu, 2024]. Thematic synthesis is also possible in the interpretive approach, and this enables thematic synthesis to compare different AI models, such as reinforcement learning and evolutionary optimization, in different system contexts, such as cloud OS, mobile kernels, and IoT operating environments [Joloudari et al., 2022].

The primary research question that the study pursued is as follows: *How have AI models been utilized to optimize operating system performance in the domains of task scheduling, resource allocation, memory prediction, and system security, and what gaps remain unaddressed in current literature?* This question revealed both technical implementations and architectural concerns, such as the interpretability of the models, computational cost, and flexibility to various types of systems [Wang & Xing, 2025]. It also aims to reveal the comparison of these AI-enhanced mechanisms with or superior to the traditional OS modules, especially in different workloads and adversarial conditions [Zhang et al., 2024].

The study is relevant in that it can be applied in the development of intelligent OS architectures of the future. As AI is increasingly used in the application-level development, its further integration into the operating system can lead to a paradigm shift, where the OS components are not only reactive but also proactive in the context of computational and security needs [Ratnayake, 2024]. Moreover, the study results can be helpful to system architects, kernel developers, and cybersecurity professionals who are interested in creating more autonomous, resilient, and efficient

computing platforms. As edge computing and decentralized systems gain popularity, the ability of OS layers to make intelligent decisions without the need to communicate with a centralized server is becoming the most significant aspect, which makes this research a necessity and a technological possibility [Zhang et al., 2024].

Literature Review

Introduction to AI in Operating Systems

Computing environments have been based on operating systems (OS) which manage the resources of the hardware and provide basic services to the software applications. The design of OS is a history of a long-term effort to improve efficiency, scalability, and responsiveness to growing computational requirements. The early computing models were the batch processing systems where the jobs were executed in batches without user interaction and then the time-sharing systems of the 1960s that allowed multiple users to interact with the computer simultaneously. Modern general-purpose operating systems Modern general-purpose operating systems became possible with the introduction of multiprogramming, virtual memory, and multithreading in the 1970s and 1980s [Chakraborty, 2023]. The traditional optimization methods in these systems have relied largely on the static algorithms, e.g., First Come First Serve (FCFS), Round Robin (RR), and Least Recently Used (LRU) which operate on deterministic rules and fixed heuristics. These mechanisms are rigid in terms of unpredictable and complex usage patterns, despite being effective when workloads are managed [Chakraborty, 2023].

The prevalence of cloud infrastructure, edge devices, and IoT networks has led to a high rate of system complexity that has revealed the inefficiencies of rule-based optimization strategies. Resource managers and static schedulers are just not dynamic in response to variable workloads, resource contention and security threats, especially in real-time systems and multi-tenant environments. In this respect, the emergence of Artificial Intelligence (AI) in systems engineering offers a revolutionary alternative. The AI techniques, such as supervised learning, unsupervised clustering, reinforcement learning, and deep neural networks, have been found to be more flexible, predictive, and self-governing in various computing fields. That is why they can be applied to the complexity of low-level system tasks such as kernel task scheduling, memory management, and anomaly detection [McDermott et al., 2021].

The motivation to integrate AI into the functionality of operating systems is not just the inefficiency of the legacy methods, but a more broad-based paradigm shift in systems engineering. AI enables systems to optimize, heal and defend themselves through learning real-time data rather than pre-programmed instructions. According to Adeyeye and Akanbi (2024), AI and machine learning have become one of the primary factors of automation, scalability, and resilience of systems and have made it possible to make decisions at scale that would otherwise be impossible to make using manual or rules-based approaches. In addition, as operating systems become more and more central to resource management in highly decentralized systems, such as edge computing nodes, autonomous vehicles, and intelligent industrial systems, it is not just desirable but inevitable to incorporate intelligent control logic into the operating system [Adeyeye & Akanbi, 2024].

The topicality and topicality of such integration are also emphasized by the increasing popularity of AI-enabled infrastructure in different industries. It can be seen with AI-powered hypervisors or AI-powered resource schedulers in cloud-native operating systems that AI-enhanced operating systems are not only feasible but can be more efficient and responsive than traditional architectures [Bosch et al., 2021]. In addition, the shift to cyber-physical systems and real-time decision-making platforms places an immense burden on the OS to predict, adapt, and defend itself without human intervention. Safarzadeh and Loghmani (2021) state that the real potential of AI lies in the fact that it can operate in low-level conditions where the decision windows are constrained by microseconds latencies, and the error margin is small [Safarzadeh & Loghmani, 2021].

In the contextualization of this literature review, one should consider how AI models are applied in the most significant areas of OS optimization, i.e., task scheduling, resource allocation, memory prediction, and security monitoring. Each of these spheres has technical problems and opportunities of AI interference. A critical literature review will be carried out to understand the effectiveness of the deployed

models, implementation or scalability gaps, and future research directions that will assist in developing a unified AI-OS framework that will strike a balance between performance, interpretability, and system robustness [McDermott et al., 2021].

AI-Driven Task Scheduling

Traditional Scheduling Techniques

Some of the conventional task scheduling algorithms that have been applied as the foundation of process management in general-purpose and real-time operating systems include the First-Come-First-Serve (FCFS), Shortest Job First (SJF), Round Robin (RR), and Multilevel Queue (MLQ). The FCFS is the easiest method, where tasks are allocated according to their arrival order, which is likely to lead to low average waiting times due to the convoy effect [Ismael et al., 2021]. SJF attempts to eliminate this by selecting the task that takes the least amount of time to execute and this reduces the average waiting time but requires proper estimation of the time taken to execute the task which is not possible in dynamic conditions. Round Robin improves interactivity by using fixed time slices of each task but suffers a high context-switching overhead in the case of high load [Chakraborty, 2023]. Multilevel Queue scheduling divides processes into multiple queues based on priority or type and uses different scheduling policies on each queue, but at the expense of task migration between queues, which results in inefficiencies [Ismael et al., 2021]. These conventional approaches are easy to implement and have minimal overheads but are reactive and rule-based in nature. They do not perform well with very dynamic workloads, shared resources in distributed systems, and failure to predict the nature of tasks beforehand [Zhang et al., 2024]. They cannot be used in systems that require adaptive and predictive scheduling mechanisms, particularly in decentralized and multi-tenant systems because they are rigid and deterministic.

Machine Learning and Reinforcement Learning Models for Task Scheduling

The popularity of AI-based scheduling approaches has been due to adaptive learning of task patterns and optimization in uncertain, real-time environments. Supervised learning models have been used to predict task runtimes and schedule priority classification, but the actual novelty is the reinforcement learning (RL) methods, which represent scheduling as a sequential decision-making process [Shyalika et al., 2020]. Q-learning, Deep Q-Networks (DQN) and policy-gradient algorithms have shown a lot of promise in dynamically adapting scheduling policies based on the feedback of the environment. The methods enable autonomous systems to train the optimal sequence of tasks that minimizes latency, maximizes throughput, and equalizes the load [Rjoub et al., 2021]. To illustrate, DQN-based schedulers are more effective than static in cloud environments since they can be retrained on novel task distributions, but policy-gradient approaches are more effective in continuous state spaces, especially in heterogeneous CPU-GPU environments [Maged & Mikhail, 2020]. Cloud-native systems that use RL-based scheduling, e.g., PerOS, are more responsive and consume more resources than traditional OS-level schedulers [H Such methods however require good quality reward functions and large training data, which limits their application to low-power or embedded systems where computational resources are scarce [Ilager et al., 2020]. In addition, the problem of model drift in the shifting workload patterns demands continuous retraining and model update [Zhang et al., 2024].

Comparative Performance Analysis

The comparative analysis of the traditional and AI-based scheduling algorithms reveals that AI models are never worse than legacy algorithms in the task completion time, response latency, and load balancing. Q-learning and DQN are such reinforcement learning algorithms that are more flexible to adapt to changing workloads and offer self-regulating behavior as opposed to the fixed behavior of RR or FCFS [Shyalika et al., 2020]. In multi-tenant cloud systems, AI schedulers have shown 20-30 percent improvement in the rate of CPU and memory utilization and a reduction in SLA violations during peak periods [Rjoub et al., 2021]. Unlike rule-based schedulers, AI models can be trained on the data of the runtime to learn the nuances of scheduling to offer a higher degree of control over the execution of tasks [Ilager et al., 2020]. However, the gain in efficiency is traded off with the complexity of the model, training overhead and inference latency in deep models. Moreover, AI schedulers are resilient in

simulated or controlled environments, yet their resilience in the real world depends on the rate of retraining, generalization of the model, and tuning of the system [Zhang et al., 2024]. This introduces the problem of sustainability in environments where the computational resources are scarce or where the access to historical data is scarce. Nevertheless, AI methods are attractive to the design of future OS due to their scalability and context-awareness, which is critical in flexibility and predictive responsiveness [Hammad & Abu-Zaid, 2024].

Identified Gaps

Despite the great progress, the existing AI-based scheduling models have several critical drawbacks. One of the primary issues is generalization across different workloads because models trained on a specific type of tasks or resource configuration cannot be transferred to other environments efficiently, and retraining and parameters adjustment are needed [Zhang et al., 2024]. This limits their use in systems which require plug-and-play flexibility. In addition, most models cannot be responsive in real-time, especially those based on deep networks, which restricts their application in latency-sensitive systems such as embedded or industrial IoT systems. To illustrate, DQN schedulers may require several milliseconds to make a scheduling decision, which is impractical in microsecond-scale real-time tasks of an OS [Maged & Mikhail, 2020]. The other gap is that there is no explainability and trust in AI decisions, particularly in safety-critical applications where deterministic behavior is preferable over probabilistic inference [Safarzadeh & Loghmani, 2021]. In addition, few models take into account multi-objective trade-offs, e.g. between performance and energy consumption or between fairness of processes. Finally, the coupling of AI schedulers with native OS kernel modules is minimal, resulting in external scheduling layers instead of embedded solutions, thereby causing overhead and reducing system cohesion [Kumar & Priyadarshini, 2024]. These gaps justify the need of hybrid solutions that entail the provision of lightweight heuristics and adaptive AI policies, and the provision of kernel-level APIs to enable seamless integration.

Intelligent Resource Allocation

Resource Allocation Challenges in Modern OS Architectures

Resource allocation in modern operating systems has become highly complicated due to the trend towards heterogeneous computing that has been driven by multi-core CPUs, GPU acceleration, virtualization and distributed systems. Unlike the monolithic architectures of yesteryear, the modern OS must be able to support a broad variety of execution environments, such as hypervisors in data centers and lightweight containers in edge devices, and must find the appropriate balance between performance and efficiency in each of them. The popularity of virtualization such as KVM, Hyper-V, and Xen has introduced a layer of abstraction that introduces additional resource contention and variability in resource scheduling [Djordjevic et al., 2021]. In addition, cloud-native infrastructure means that limited CPU, GPU, and I/O bandwidth is shared with hundreds or thousands of concurrent containers or serverless instances, and static allocation methods are insufficient [Abid et al., 2020]. Multi-core processors are parallel, but introduce new problems to OS schedulers: to make intelligent, context-sensitive choices about thread scheduling, NUMA balancing, and power-state management--issues that are not easily scaled with standard heuristics. Resource interference, neighbor effects, and cache contention play an essential role in such a situation [Awaysheh et al., 2021]. Worse still, real time applications and latency sensitive services need very precise provisioning rules to make sure that QoS requirements are not violated. These dynamic problems need dynamic and independent allocation mechanisms that are able to predict, respond and optimize resource usage in real time.

AI Models for CPU/GPU/IO Resource Distribution

To address these problems, different AI models are applied in dynamic management of system resources. To illustrate, to optimize multi-objective resource scheduling policies, e.g., those of CPU-GPU coordination, evolutionary algorithms such as genetic algorithms (GAs) emulate a natural selection process [Abid et al., 2020]. Swarm intelligence methods that have been applied to large-scale virtualized networks

include ant colony and particle swarm optimization, and have performed better in terms of load balancing and fault tolerance. Supervised learning methods, namely decision trees and support vector machines, have also been applied to predictive allocation of resources to train past usage models to optimize container placements strategies [Awaysheh et al., 2021]. Workloads that are GPU-bound are capable of being mitigated with a system like Vortex, which is based on AI-drivable memory orchestration to successfully balance bandwidth and inter-accelerator caching so as to avoid extreme memory capacity constraints [Yuan et al., 2025]. The AI models predict the workload bursts and proactively correct the pod resource on containerized systems such as Kubernetes, such that the SLA violations are reduced and the clusters operate much more efficiently [Chen et al., 2025]. These models do not only automate complex decision making but also reduce manual tuning. In order to utilize them, however, they must also be continuously updated through data collection, feature engineering, and retraining pipelines, which can unfortunately introduce undesirable runtime and development overhead, especially when edge or embedded devices have limited hardware.

Benchmarked Results and Model Comparisons

The empirical standards have demonstrated on multiple occasions that AI-augmented resource allocators outperform traditional heuristics in several important measures, including latency, throughput, and energy efficiency. Serverless frameworks have also discovered AI-based models, through Faasdom fully benchmark suit, to reduce the cold start time by up to 30 percent and improve the method of function level resource packing [Maissen et al., 2020]. Similarly, in high-performance computing (HPC) environments using the AI (star-gen) frameworks, predictive resource modeling is reported to result in large compute utilization and task runtime variance [Chen et al., 2025]. The Vortex-like systems demonstrate nearly 2x the memory footprint in addition to large boosts in bandwidth partition correctness on mixed-precision deep learning workloads [Yuan et al., 2025]. Comparatively, the dynamic schedulers and threshold-based allocators cannot always react to the rapidly varying workloads, thus leading to overprovisioning or underutilization. The quantitative benefits of energy without compromising performance have also been shown by energy-aware AI schedulers over the traditional DVFS-based mechanisms. However, these are relative gains and vary significantly with the kind of system, workload and hardware architecture. Moreover, the benchmarking of different platforms remains not unified, and there is no universal assessment framework that would enable the conduct of comparative studies, which makes it difficult to generalize the models and reproduce them.

Gaps and Open Problems

Despite the grand success, an AI-based resource allocation to modern OS architectures has certain gaps and open problems. The first major limitation is that there are no standard data sets or benchmarking models on the comparison of the effectiveness of various models of various system configurations. Existing benchmarks like Faasdom target a specific area (e.g., serverless), but the frameworks to distribute general-purpose resources are not well developed [Maissen et al., 2020]. This avoids duplication and matching comparison. Furthermore, the cross-platform portability, between the cloud servers and embedded devices, tends to be poor. The GPU-optimized models may not only scale well to the low-memory, low-power edge devices easily [Abid et al., 2020]. Another issue is model interpretability and transparency; the majority of deep learning approaches used in resource scheduling are black boxes, which is an issue in safety-critical applications [Yuan et al., 2025]. In addition, most of the current implementations are reactive rather than proactive and lack real-time visibility of long-term trends of resource utilization. Finally, there is the integration issue between AI models and the kernel rest of the operating system (OS), which leads to the dependency on user-space daemon or external orchestration layers, potentially causing latency and synchronity issues [Awaysheh et al., 2021]. These concerns will need to be resolved through the construction of light-weight, explainable and cross-platform AI models, augmented with universal testing standards and OS-integration APIs.

Predictive Memory Management

Traditional Paging and Cache Policies

Traditional memory management in operating systems has been based on heuristic-based paging and caching policies, including Least Recently Used (LRU), Least Frequently Used (LFU) and FIFO. LRU assumes that the pages that have been recently accessed will be accessed in the nearest future, and LFU is concerned with the pages with the highest access rate to be kept. These techniques are simple to implement and require little computation and hence they can be used in memory-constrained environments. However, they are not optimal in any system where the memory access patterns vary significantly over time or workloads [Heo et al., 2020]. To illustrate, the LRU is typically disadvantaged in throughput-sensitive data applications, such as tiered memory systems with a large page, where the access patterns switch between hot and cold data frequently [Heo et al., 2020]. Even in the case of heterogeneous architecture, such as that which utilizes disaggregated memory and distributed caches, where locality assumptions do not hold due to remote memory access [Vishwarupe, 2024], the techniques are deficient. Moreover, the conventional policies lack the predictive power to anticipate future memory requests and this leads to unnecessary page faults and memory swaps. As computing environments have become more dynamic and data-intensive, static heuristics are becoming more bottlenecks than optimizers of system performance.

Deep Learning and Statistical Models for Memory Prediction

Rule-based memory management limitations are being replaced by data-based and anticipation-based models. Memory forecasters In sequence modeling tasks, Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) models, a type of deep learning, have proven quite effective at solving the memory access problem. These models can be trained to interpret the temporal dependencies in memory access sequences and generate inferences on page requests at later steps that can be utilised to pro-actively cache page or pre-fetch them [Ahmed et al., 2022]. Their more traditional counterparts, multi-sequence LSTM architectures, typically augmented with metaheuristics optimization, have also demonstrated greater accuracy on memory-aware forecasting tasks than their more traditional counterparts when the workload varies [Bouktif et al., 2020]. The trends in the variation of memory usage have been calculated using autoregressive and hybrid time-series forecasting, which can be applied to predict demand spikes or swap requirements [Ahmed et al., 2022]. In disaggregated and cloud environments, deep models can be applied to support intelligent orchestration of memory levels (e.g. DRAM, NVRAM) to realize a tradeoff between latency and throughput [Vishwarupe, 2024]. However, such models are costly to deploy and require good quality training data and a large quantity of computational resources, and they are not conveniently deployed in real-time systems or edge devices with weak memory and power constraints.

Performance Metrics: Hit Rate, Latency, Swapping Efficiency

The hit rate, the memory access latency, and the efficiency of swapping are the most significant measures that can enable us to assess the effectiveness of the memory management strategies employed, both the traditional and the AI-based ones. The percentage of the memory accesses provided by cache is called hit rate and is directly proportional to reduced latency and higher throughput on consumer and high-performance computing systems [Aslanpour et al., 2020]. The AI-based models are more likely to be effective than the static algorithms in predicting page usage, which reduces the cache miss rates and costly disk I/O operations [Ye et al., 2024]. Latency is especially relevant in GPU datacenters, and real-time systems, where microsecond delays can negatively affect the performance of the workloads. In such regions, the prefetching with deep learning-based models has already shown substantial results in the context of task completion time and hardware utilization [Ye et al., 2024]. Memory locality and probability of reuse can be considered in predictive models that are useful in swapping efficiency i.e. minimizing unnecessary page swaps. However, the performance measurements are likely to vary based on the type of workload: AI-based techniques are more effective in HPC and cloud data centers with predictable access patterns, but the difference is not as high in mobile and embedded systems due to the

resource constraints [Aslanpour et al., 2020]. There are also still no standardized benchmarking procedures to allow cross-platform comparisons of predictive memory management techniques to be made fairly.

Challenges and Research Gaps

Nevertheless, there are several essential issues with AI-based memory prediction systems. Training and deployment of deep models, deep specifically LSTM or Transformer-based models, is computationally-expensive, a fact that is a genuine barrier in environments with limited memory and computing resources. The memory prediction must be done in real-time with microsecond latency, but the overhead of most deep models is in the millisecond range, making them incompatible with latency-sensitive applications [Ahmed et al., 2022]. Second, explainability may be viewed as a serious issue. Most AI-based memory models are black boxes and it is difficult to understand or explain why specific pages are prefetched or evicted, which is an unacceptable trade-off in safety-critical or regulated computing domains [Bouktif et al., 2020]. Third, scalability is limited by data dependency and generalization: models trained with one application or a dataset may not perform well when deployed in a different environment or when operating under unexpected memory constraint [Vishwarupe, 2024]. Fourth, lack of integration frameworks to existing OS kernels requires the use of AI models in user space or implementation with middleware to generate coordination overheads and reduce overall efficiency. Finally, predictive memory models lack standardized benchmarks or datasets to evaluate the models and render them challenging to replicate in addition to comparing them with other systems [Aslanpour et al., 2020]. Such gaps will require filling with the development of lightweight, explainable, and OS-native predictive memory frameworks and standard evaluation methodologies.

AI-Enhanced OS Security Monitoring

Conventional OS Security Mechanisms

The conventional OS security mechanisms form the lower level of precaution against ill-minded activities. These include signature IDS intrusion detection systems (IDS), mandatory and discretionary access control (MAC/DAC) and sand boxing. Snort or OSSEC are signature-based IDS which operate using a matching of the system activity with pre-defined malware signatures. Such systems are not able to identify new, zero-day attacks or polymorphic malware, even though they are effective against known threats [Iyer, 2021]. Despite the need to have access control models, they are reactive and have inflexible policies that are not adaptive. Similarly, sandboxing is applied to isolate potentially malicious processes and these are normally avoided through advanced evasion techniques especially in kernel-level attacks. The existence of the above-mentioned techniques is limited by the fact that to the extent to which they operate according to the pre-designed sets of rules and threat databases created by humans, they delay the response to the new threat indeed [Zipperle et al., 2022]. Moreover, the performance overheads are likely to come with such mechanisms due to their constant observation of system calls and context changes. Traditional security measures, though simple in nature, are not keeping up with the pace, volume and complexity of the current threat environment and a shift towards intelligent, behavior-based monitoring is increasingly becoming a necessity.

Anomaly Detection Using AI

AI-based anomaly detection has emerged as a possible solution to signature-based security. Real-time detection of anomalous behavior is being done using autoencoders, unsupervised clustering algorithms and hybrid AI-rule systems. The second problem to address is how to learn a compact representation of normal behavior of the system and detect deviations that indicate unknown attacks Autoencoders, in particular adaptive structures like the WavePCA-Autoencoder (AWPA), can learn the representation of normal behaviors and assist in detecting deviations that indicates possible unknown attacks [Mohamed et al., 2025]. Unsupervised learning models applied in zero-day exploits have been discovered to be detectable without labeled data, k-means, DBSCAN, and Isolation Forest, in particular, significantly enhance the visibility of threats in new spaces [Zoppi et al., 2021]. Hybrid models Hybrid methods that combine statistical baselines with deep learning are more effective in detection and interpretable. Moreover, AI can be used together with provenance-based

monitoring to track the origin and route of data in the OS, which enables high-fidelity intrusion detection [Zipperle et al., 2022]. The flexibility of AI is one of the primary advantages in this case: by continuously learning how the system works, models can adjust to new threats that would otherwise remain undetected by a fixed detection system. Such approaches are however sensitive to the quality of training data and require frequent tuning to prevent model drift especially in dynamic multi-user configuration.

Model Accuracy, False Positives, and Real-Time Constraints

AI-based intrusion detection systems (IDS) are associated with high detection rates, but they are prone to false positives, real-time inference latency, and scalability. The Conan is a real-time APT detection framework to enable a detection performance of over 95 percent, with sub-second latency, through the combination of model size and data preprocessing pipeline optimization methods [Xiong et al., 2020]. The majority of deep models, though, especially autoencoders and recurrent networks are prone to overfitting, which may result in the detection of benign anomalies (e.g. software updates or maintenance procedures) as an intrusion. This leads to alert fatigue among the system administrators as well as poor confidence in automated decision making [Mohamed et al., 2025]. Besides, in real-time security there is the need of an ultra-fast inference, typically followed by overheads that are likely to result in slower response of the system in resource-constrained devices that use deep models. A problem with this field is the tradeoff between accuracy of detection and the runtime. In other frameworks, layered architectures are employed where the lightweight models are employed as filters before the calls to the more accurate, but more overhead-intensive models, only calling the more accurate models when necessary, at the cost of precision [Xiong et al., 2020]. Despite the improved situation, low false-positive rates with high sensitivity in real-time conditions remain one of the key challenges of AI-based security in OS environments.

Limitations and Research Gaps

Despite the promising features, the following are some of the limitations of AI-based OS security systems, which are very crucial. First, it is extremely hard to integrate with low-latency OS components. Security monitoring is often done in user space, which is not linked with the kernel-level processes where real-time decisions are critical. The latency of the inference, size of the models, and stability are also an issue to the direct integration of AI models into the OS kernel [Xiong et al., 2020]. Second, adversarial AI threats, where an adversary can feed models with input data to mislead them, are becoming a threat. The attackers have succeeded in creating adversarial inputs that can even bypass good anomaly detectors, and thus a robust, explainable AI is urgently required in critical infrastructure [Zoppi et al., 2021]. Third, models have not been studied in terms of adversarial robustness. Most of the existing systems are evaluated in controlled laboratory environments with synthetic data, which limits their applicability to real-life attacks. Fourth, lack of coherent datasets and benchmarking systems does not enable reproducibility and comparable assessment across systems [Zipperle et al., 2022]. And finally, the question regarding trust and interpretability remains. Deep neural models are also effective, but they are black boxes, and one can hardly validate or audit any decision made by the system, especially in regulated areas, like in healthcare or in financial markets [Iyer, 2021]. Future research directions must be on lightweight and interpretable models of AI that are attack-resistant, and tightly integrated with native OS units.

Toward a Unified AI-OS Architecture

Need for Cross-Domain Integration

The current AI-OS ecosystem is disjointed where task scheduling, resource allocation, memory prediction, and security monitoring are isolated and in most instances, they are implemented as independent AI modules or third-party extensions. This is effective in local optimization of the domain, but it does not allow system-wide intelligence and coordination, which makes it inefficient in a shared resource environment such as distributed cloud platform or the edge computing framework [Shankar, 2025]. Using the same example, a given AI scheduler might optimize its task latency without considerations to memory constraints or present security exploits thus providing an unfavorable system behavior. The lack of

interoperability between AI modules that are used on different OS levels does not allow the emergence of a truly adaptive and consistent operating system. The response is the shift to cross-domain integration, in which AI models do not only collaborate across subsystems, but telemetry, feedback, and learned policies are also shared to make joint decisions. Barros (2025) highlights the necessity of such horizontally integrated systems in such areas as telecom, where AI microservices are isolated, which leads to brittle and expensive orchestration. A shared AI-OS would enable synchronization of learning across CPU, I/O and memory planes, increasing efficiency and reliability of dynamic workloads. This is also consistent with the industry trends of the so-called AI-native infrastructure that is able to self-optimize and learn on the fly in multiple areas of operation [Mishra et al., 2024]. The shift between isolated models and integrated intelligence must begin with basic changes in the design of OS, particularly the kernel, and the rejection of the modular AI plug-in model in favor of systemic, co-optimized AI layers.

Frameworks and Architectures Proposed in Literature

Various frameworks have been proposed in the recent literature to attain unified AI-enabled OS architectures. AI-kernels are a novel form of operating system where AI models are embedded into the kernel space and can be applied to make real-time decisions regarding dynamic thread allocation, cache tuning, and anomaly response [Shankar, 2025]. The difference between these systems and the traditional user-space daemons is that they reduce the latency and increase the responsiveness of the critical events in the system. Taking a different turn, Barros (2025) proposes a Horizontal Federated AI Operating System, which is customized to telecommunications, where AI modules installed at base stations learn continuously and synchronize without central control, offering resilience, scalability, and reduced bandwidth demands to telemetry data. Continuing with the same theme, systems like USAS (Universal Sustainable AI System) enable continuous-learning OS environments that are optimized to edge servers, and model drift and energy efficiency are managed via federated updates and real-time adaptation [Mishra et al., 2024]. Further testing of integrated AI strategies can be performed in SIMPLE and other platform simulators to test under simulated workloads that can serve as a testbed to co-design task, resource and memory schedulers [Meyer et al., 2023]. And lastly, open-source work is beginning to breach through as the management of modular, containerised sets of AI-OS toolkits, and industry-academia collaboration [Chen & Zhou, 2025]. However, the majority of these frameworks remain experimental, and not many of them have been applied in production environments due to their extreme complexity and the absence of standard APIs to incorporate AI at the kernel level.

Bottlenecks in Implementation

Despite the growing curiosity, using AI models at the low levels of the operating systems has significant bottlenecks in implementing the models. One of the most important problems is hardware dependency: the majority of AI algorithms require accelerators like GPUs or NPUs, which are not evenly distributed in edge, embedded, and legacy systems. As a result, the heterogeneous hardware designs will have to take into account unified AI-OS designs, and optimization and generalization will be difficult [Mishra et al., 2024]. The other challenge is the absence of standardization of APIs and toolchains that allow a seamless interaction of AI modules with the processes at the kernel level. Part of the existing solutions were based on custom kernel patches or user-space wrappers, which offer latency, lack of portability, and challenging update [Shankar, 2025]. In addition, the model drift is a serious risk to continuous-learning systems, as the workload can shift and the learned behavior can become invalid, leading to poor or unsafe decisions. It is particularly unfortunate in real-time systems, where retraining windows are narrow and inference failures may be catastrophic. In addition, the application of AI to low-level system software is linked to security and stability concerns: even a minor error in prediction of task priorities or memory allocation can lead to performance loss or system failure. Besides, most frameworks lack explainability processes of their AI decisions that limit their Foucault basis on safety-based actions like car or industry control [Barros, 2025]. Lastly, the development cost, the complexity of testing, and lack of industry-wide standards of benchmarking of AI-OS platforms also play a role in the inability to scale and

commercially deploy [Chen & Zhou, 2025]. The only way out of these bottlenecks is the collaborative effort of the hardware vendors, OS developers, and AI researchers to co-develop stable, transparent, and hardware-agnostic platforms.

Synthesis of Research Gaps and Future Directions

This research has aimed at critically examining the application of artificial intelligence (AI) models in optimization of functions of core operation system (OS) such as task scheduling, resource allocation, memory management and security monitoring and identify deficits that hinder realization of a unified AI-based system architecture. This literature review has shown a high level of fragmentation in approaches, architectures, and assessment criteria in these areas. Despite the great potential of AI application, most of the work remains local and domain-specific, with a very narrow scope of a single subsystem and no regard to cross-domain interdependencies or system-wide coordination [Shankar, 2025]. It is not a scalable and redundant model of innovation and the AI modules do not result in shared learning or consistent intelligence in this segmented world of innovation. To illustrate, despite the fact that reinforcement learning has been applied to enhance cloud-based task scheduling, it is rarely applied in combination with AI-based memory forecasting or anomaly detection, which creates operational silos in contemporary OSes [Rjoub et al., 2021].

Among the most urgent gaps identified by this review, one can note the lack of AI models that can meet real-time demands with a minimal computational overhead. The examples of deep learning that may be more effective in predictive memory management and zero-day exploit detection, but that may require special accelerators or high power to operate reliably include LSTMs, policy gradient networks, and autoencoders [Ye et al., 2024]. This renders them inapplicable to edge computing, embedded systems or latency-sensitive systems such as industrial automation and vehicle communication systems [Mishra et al., 2024]. Thus, the future of the optimization of OS on the basis of AI must be related to the application of low-weight, low-computing energy optimization that may be capable of operating under the latencies orders of magnitude per microsecond without compromising precision.

The other important theme that emerges is the importance of interpretability and robustness. The majority of existing AI systems are black-box systems, and there is little understanding of how they make decisions, which is a particular issue in security-sensitive or regulated computing applications. To illustrate, deep anomaly detectors and memory prefetching models are likely to produce inexplicable outputs, which cannot be verified or trusted by developers and auditors [Mohamed et al., 2025]. Similarly, few models are trained on adversarial inputs, thus they are vulnerable to sophisticated attacks that can result in false negatives or even system instability [Zoppi et al., 2021]. Future of AI-augmented OS frameworks must therefore be aimed at robust, explainable and verifiable models, perhaps through hybrid AI-rule systems or interpretable learning systems like decision trees with deep learning capabilities.

Is also urgent the issue of cross-platform transferability. The application of AI models that are trained in the cloud data centers does not always compute well on the edge servers or mobile devices due to differences in the load look handling, memory architectures, and the availability of resources [Abid et al., 2020]. This is also partly because there is no platform-independent training information and multi-scale retraining pipelines. A good adaptive OS will be able to generalise across a large range of different hardware types, such as very low-end (ARM-based IoT) to very high-throughput (GPU clusters), without necessarily reentraining the overall model. Such solutions as federated learning frameworks that allow decentralized and privacy-preserving model updates will be possible to propose.

Moreover, the absence of benchmark data and standard evaluation procedures is also an obstacle to further development. Most of the analyzed reports rely on proprietary or synthetic data and, thus, one cannot replicate their findings and implement a fair comparison of the performance between models and systems. Even when dealing with public benchmarks, e.g., Faasdom in the case of serverless or SPEC in the case of general computing, there is still no consensus on key evaluation metrics, e.g., latency, energy, security effectiveness, and model drift resistance [Maissen et al., 2020]. The paper demonstrates that there is a need to access biased datasets, workload

traces and testbeds that reflect real-life OS properties in various and varying conditions. It is also necessary to possess consistent structures and APIs in the integration of AI-kernels, where the current methods are highly dependent on user-space representations or third-party schedulers that increase the coordination overheads and reduce the reliability of faultless its autonomous operation [Shankar, 2025].

Despite the transformative potential of AI in the design of operating systems, methodological fragmentation, inability to be used in real-time, low model interpretability, poor platform generalizability, and the absence of benchmarking standards are challenges on the path to truly intelligent, secure, and responsive operating systems. Additional guidance should include the construction of modular yet integrated architectures enabling cross-domain learning, low-latency AI inference in kernel space, and transparent decision logic, all of which should be supported by community-built datasets and APIs to enable the capacity to critically assess models in a fair and robust way.

Table 1: Summary of Key Contributions, Gaps, and Future Directions

Author(s)	Theme	Key Findings	Gap Identified	Future Direction
Shankar (2025)	AI in Linux kernel optimization	AI models improve scheduler and resource decisions when integrated with kernel modules	Lack of API support and kernel-level integration	Development of AI-native kernel APIs and modular OS learning frameworks
Rjoub et al. (2021)	Deep RL in task scheduling	Deep Q-networks significantly outperform FCFS and RR in cloud-based systems	No multi-domain coordination with memory/security models	Holistic AI scheduling models with resource-awareness and threat-awareness
Mishra et al. (2024)	Continuous learning at the edge	Federated learning minimizes model drift and enables localized model updates	Hardware dependency and high model complexity	Federated low-power AI models with adaptive compression and energy tuning
Mohamed et al. (2025)	AI for zero-day threat detection	Autoencoders and WavePCA detect novel attacks with high accuracy	Limited explainability and high false positives	Design of hybrid interpretable models and adversarially robust frameworks
Maissen et al. (2020)	Benchmarking in serverless systems	Faasdom provides baseline comparisons for serverless function scheduling	No standard for memory or security model evaluation	Unified benchmarking toolsets across task, memory, and security domains
Zoppi et al. (2021)	Unsupervised anomaly detection	Clustering and Isolation Forests can identify unknown intrusions	No validation under adversarial scenarios	Adversarially-trained, online-updating anomaly detectors
Ye et al. (2024)	Deep learning for GPU workload management	Predictive models optimize memory and throughput in data centers	Not scalable to mobile/embedded due to inference cost	Lightweight DL models or neural pruning for low-power OS deployment
Barros (2025)	Horizontal federated AI-OS in telecom	Decentralized AI coordination reduces bandwidth and improves responsiveness	Not transferable to general-purpose OS environments	Scalable, industry-agnostic horizontal AI-OS architectures

Chen & Zhou (2025)	Open-source AI-OS collaboration	Open-source ecosystems accelerate co-development of AI-OS prototypes	Lack of modular design and test environments	Containerized, open-source AI OS development environments

Conclusion

This research has examined the integration of Artificial Intelligence (AI) in fundamental operating system (OS) functions, that is, the task schedules, resource scheduling, memory management, and security surveillance in a systematic way by conducting a comprehensive literature review. The findings show that despite the fact that AI techniques have recorded impressive improvements over the conventional heuristic-based techniques in individual areas, their application remains fragmented. An example is that reinforcement learning has been applied to significantly enhance the efficiency of scheduling in cloud systems, and deep learning architectures like LSTM and autoencoders have been applied to conduct predictive memory management and anomaly detection. Nevertheless, these developments are not usually interoperable. The subsystems are optimized individually, typically with no coordination or even awareness of other systems resulting in architectural-level silos that cancel out the whole system intelligence and flexibility [Shankar, 2025]. This cross-domain integration has turned out to be one of the most persistent and detrimental limitations in the current AI-OS literature.

The second valuable lesson is that the existing AI models cannot be used in real-time systems. The majority of the high-performance AI techniques are associated with computational overhead and latency issues, which is why they cannot be used in resource-constrained or latency-sensitive systems such as embedded systems, mobile systems, or industrial IoT. This incompatibility between the capabilities of AI models and constraints of the operating system proves the urgent need of highly-efficient, flexible, and lightweight models capable of inference and learning on-device [Mishra et al., 2024]. In addition, AI models are typically not transparent, even though they can be used to automate and optimize decision-making, raising concerns of trust, explainability, and debugging in security-critical systems [Mohamed et al., 2025]. As a result, the future of the AI-OS integration is supposed to incorporate intelligent machine learning techniques, or mixed models, which join symbolic logic and statistical learning.

The literatures also show that there is a serious lack of benchmarking infrastructures and assessment protocols. Datasets, testbeds, and cross-platform benchmarks are not standardized, and therefore it is difficult to compare the AI-enhanced OSes models or replicate the results on different hardware or other applications. This reproducibility is not only impeding academic advancement but it is also not appealing to adoption in the industry where an issue of operational reliability and performance assurances is paramount [Maissen et al., 2020]. Another persistent problem this is because over time, an AI model trained on a particular dataset will become inaccurate as a system develops. Although the proposed frameworks like federated and continuous learning may assist in solving this issue, they are not yet developed and implemented in OS-level applications [Barros, 2025].

Repeating the research question, *How have AI models been utilized to optimize operating system performance in the domains of task scheduling, resource allocation, memory prediction, and security monitoring, and what gaps remain unaddressed in current literature?*, the research has arrived at the conclusion that despite the positive developments, a unified, modular, and explainable AI-OS architecture has not been reached. To accomplish it, the AI models must be more tightly integrated with the kernels of any operating system, capable of coordinating across boundaries, and demonstrated by repeatable, comparable scientific experiments. Moreover, the OS is to be turned into an active, intelligent entity, capable of self-optimization, autonomous reaction to threats, and learning.

Lastly, the application of AI in the creation of the next generation of intelligent operating systems is not just in enhancing the performance but also in the re-engineering of the role of the OS as a decision-making engine. Decentralized, autonomous, and heterogeneous computing environments will further impose new requirements on the management of such

complexity and offer resilience and support seamless user experiences only with the help of AI-integrated operating system platforms. However, the change cannot occur without technological innovation, but also transdisciplinary collaboration between AI research, systems engineering, and hardware design. This literature review preconditions the stage of such integration, describes the achievements and failures of the field, and asks to make a united effort in the direction of the creation of intelligent, adaptive, and secure operating systems of the future.

REFERENCES

Abid, A., Manzoor, M. F., Farooq, M. S., Farooq, U., & Hussain, M. (2020). Challenges and issues of resource allocation techniques in cloud computing. *KSII Transactions on Internet and Information Systems (TIIS)*, *14*(7), 2815-2839. https://koreascience.kr/article/JAKO202022762159479.pdf

Adeyeye, O. J., & Akanbi, I. (2024). Artificial intelligence for systems engineering complexity: a review on the use of AI and machine learning algorithms. *Computer Science & IT Research Journal*, *5*(4), 787-808. https://www.researchgate.net/profile/Olade-Adeyeye/publication/385379144_ARTIFICIAL_INTELLIGENCE_FOR_SYSTEMS_ENGINEERING_COMPLEXITY_A_REVIEW_ON_THE_USE_OF_AI_AND_MACHINE_LEARNING_ALGORITHMS/links/67222cba77b63d1220cda98c/ARTIFICIAL-INTELLIGENCE-FOR-SYSTEMS-ENGINEERING-COMPLEXITY-A-REVIEW-ON-THE-USE-OF-AI-AND-MACHINE-LEARNING-ALGORITHMS.pdf

Ahmed, D. M., Hassan, M. M., & Mstafa, R. J. (2022). A review on deep sequential models for forecasting time series data. *Applied computational intelligence and soft computing*, *2022*(1), 6596397. https://onlinelibrary.wiley.com/doi/pdf/10.1155/2022/6596397

Aslanpour, M. S., Gill, S. S., & Toosi, A. N. (2020). Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research. *Internet of Things*, *12*, 100273. https://qmro.qmul.ac.uk/xmlui/bitstream/handle/123456789/66658/Gill%20Performance%20Evaluation%20Metrics%202020%20Accepted.pdf?sequence=4

Awaysheh, F. M., Alazab, M., Garg, S., Niyato, D., & Verikoukis, C. (2021). Big data resource management & networks: Taxonomy, survey, and future directions. *IEEE Communications Surveys & Tutorials*, *23*(4), 2098-2130. https://ieeexplore.ieee.org/abstract/document/9478917/

Barros, S. (2025). The Case for a Horizontal Federated AI Operating System for Telcos. *arXiv preprint arXiv:2506.17259*. https://arxiv.org/pdf/2506.17259

Bikkasani, D. C., & Yerabolu, M. R. (2024). Ai-driven 5g network optimization: A comprehensive review of resource allocation, traffic management, and dynamic network slicing. *American Journal of Artificial Intelligence*, *8*(2), 55-62. https://www.researchgate.net/profile/Dilees-Chandra-Bikkasani/publication/385214600_AI-Driven_5G_Network_Optimization_A_Comprehensive_Review_of_Resource_Allocation_Traffic_Management_and_Dynamic_Network_Slicing/links/674f9dee790d154bf9c27b6f/AI-Driven-5G-Network-Optimization-A-Comprehensive-Review-of-Resource-Allocation-Traffic-Management-and-Dynamic-Network-Slicing.pdf

Bosch, J., Olsson, H. H., & Crnkovic, I. (2021). Engineering ai systems: A research agenda. *Artificial intelligence paradigms for smart cyber-physical systems*, 1-19. https://arxiv.org/pdf/2001.07522

Bouktif, S., Fiaz, A., Ouni, A., & Serhani, M. A. (2020). Multi-sequence LSTM-RNN deep learning and metaheuristics for electric load forecasting. *Energies*, *13*(2), 391. https://www.mdpi.com/1996-1073/13/2/391/pdf

Chakraborty, P. (2023). *Operating Systems: Evolutionary Concepts and Modern Design Principles*. Chapman and Hall/CRC. https://www.taylorfrancis.com/books/mono/10.1201/9781003383055/operating-systems-pranabananda-chakraborty

Chen, P., Mo, Q., Xu, Z., Zhang, X., & Lu, Y. (2025). Star-gen: an HPC-AI framework for constructing large-scale computational materials database. *CCF Transactions on High Performance Computing*, *7*(2), 85-99. https://www.researchgate.net/profile/Pin-Chen-3/publication/390615364_Star-gen_an_HPC-AI_framework_for_constructing_large-scale_computational_materials_database/links/6840eeec6b5a287c304942c3/Star-gen-an-HPC-AI-framework-for-constructing-large-scale-computational-materials-database.pdf

Chen, X., & Zhou, Y. (2025). Open-Source Collaboration and Technological Innovation in the Industrial Software Industry: A Multi-Case Study. *Systems*, *13*(6), 433. https://www.mdpi.com/2079-8954/13/6/433

Djordjevic, B., Timcenko, V., Kraljevic, N., & Macek, N. (2021). File System Performance Comparison in Full Hardware Virtualization with ESXi, KVM, Hyper-V and Xen Hypervisors. *Advances in Electrical & Computer Engineering*, *21*(1). https://www.academia.edu/download/101783345/AECE.2021.0100220230503-1-8wd8gc.pdf

Hammad, A., & Abu-Zaid, R. (2024). Applications of AI in decentralized computing systems: harnessing artificial intelligence for enhanced scalability, efficiency, and autonomous decision-making in distributed architectures. *Applied Research in Artificial Intelligence and Cloud Computing*, *7*(6), 161-187. https://core.ac.uk/download/pdf/620852567.pdf

Hè, H. (2024). PerOS: Personalized Self-Adapting Operating Systems in the Cloud. *arXiv preprint arXiv:2404.00057*. https://arxiv.org/pdf/2404.00057?

Heo, T., Wang, Y., Cui, W., Huh, J., & Zhang, L. (2020). Adaptive page migration policy with huge pages in tiered memory systems. *IEEE Transactions on Computers*, *71*(1), 53-68. https://ieeexplore.ieee.org/iel7/12/4358213/09252863.pdf

Ilager, S., Muralidhar, R., & Buyya, R. (2020, October). Artificial intelligence (ai)-centric management of resources in modern distributed computing systems. In *2020 IEEE Cloud Summit* (pp. 1-10). IEEE. https://arxiv.org/pdf/2006.05075

Ismael, G. A., Salih, A. A., AL-Zebari, A., Omar, N., Merceedi, K. J., Ahmed, A. J., ... & Ibrahim, I. M. (2021). Scheduling Algorithms Implementation for Real Time Operating Systems: A Review. *Asian Journal of Research in Computer Science*, *11*(4), 35-51. https://www.academia.edu/download/101913461/56810.pdf

Iyer, K. I. (2021). From Signatures to Behavior: Evolving Strategies for Next-Generation Intrusion Detection. *European Journal of Advances in Engineering and Technology*, *8*(6), 165-171. https://www.academia.edu/download/122415883/Research_Article_From_Signatures_to_Behavior_Evolving_Strategies_for_NextGeneration_Intrusion_Detection.pdf

Joloudari, J. H., Alizadehsani, R., Nodehi, I., Mojrian, S., Fazl, F., Shirkharkolaie, S. K., ... & Acharya, U. R. (2022). Resource allocation optimization using artificial intelligence methods in various computing paradigms: A Review. *arXiv preprint arXiv:2203.12315*. https://www.researchgate.net/profile/Javad-Hassannataj-Joloudari/publication/359435036_Resource_allocation_optimization_using_artificial_intelligence_methods_in_various_computing_paradigms_A_Review/links/623bd7973818892e0a6c79e6/Resource-allocation-optimization-using-artificial-intelligence-methods-in-various-computing-paradigms-A-Review.pdf

Kumar, A., & Priyadarshini, S. (2024). Adaptive AI Infrastructure: A Containerized Approach For Scalable Model Deployment. *International Research Journal of Modernization in Engineering Technology and Science*, *6*(11), 5827-5834. https://www.researchgate.net/profile/Apurva-Kumar-10/publication/386382602_ADAPTIVE_AI_INFRASTRUCTURE_A_CONTAINERIZED_APPROACH_FOR_SCALABLE_MODEL_DEPLOYMENT/links/674fe36ca7fbc259f1ab0ad0/Adaptive-AI-Infrastructure-A-Containerized-Approach-for-Scalable-Model-Deployment.pdf

Maged, S. A., & Mikhail, B. H. (2020). Deep reinforcement learning collision avoidance using policy gradient optimisation and Q-learning. *International Journal of Computational Vision and Robotics*, *10*(3), 260-274. https://www.inderscienceonline.com/doi/abs/10.1504/IJCVR.2020.107253

Maissen, P., Felber, P., Kropf, P., & Schiavoni, V. (2020, July). Faasdom: A benchmark suite for serverless computing. In *Proceedings of the 14th ACM international conference on distributed and event-based systems* (pp. 73-84). https://arxiv.org/pdf/2006.03271

McDermott, T. A., Blackburn, M. R., & Beling, P. A. (2021). Artificial intelligence and future of systems engineering. In *Systems engineering and artificial intelligence* (pp. 47-59). Cham: Springer International Publishing. https://link.springer.com/chapter/10.1007/978-3-030-77283-3_3

Meyer, E. L., Mielke, T., Parke, T., Jacko, P., & Koenig, F. (2023). SIMPLE—A modular tool for simulating complex platform trials. *SoftwareX*, *23*, 101515. https://www.sciencedirect.com/science/article/pii/S235271102300211X

Mishra, C. S., Sampson, J., Kandemir, M. T., Narayanan, V., & Das, C. R. (2024, March). Usas: A sustainable continuous-learning framework for edge servers. In *2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA)* (pp. 891-907). IEEE. https://par.nsf.gov/servlets/purl/10552532

Mohamed, A. A., Al-Saleh, A., Sharma, S. K., & Tejani, G. G. (2025). Zero-day exploits detection with adaptive WavePCA-Autoencoder (AWPA) adaptive hybrid exploit detection network (AHEDNet). *Scientific Reports*, *15*(1), 4036. https://www.nature.com/articles/s41598-025-87615-2.pdf

Ratnayake, S. (2024). A COMPREHENSIVE REVIEW OF AI-DRIVEN OPTIMIZATION, RESOURCE MANAGEMENT, AND SECURITY IN CLOUD COMPUTING ENVIRONMENTS. https://www.researchgate.net/profile/Sanjeewa-Ratnayake/publication/386250503_A_COMPREHENSIVE_REVIEW_OF_AI-DRIVEN_OPTIMIZATION_RESOURCE_MANAGEMENT_AND_SECURITY_IN_CLOUD_COMPUTING_ENVIRONMENTS/links/674a1d28f309a268c0173faa/A-COMPREHENSIVE-REVIEW-OF-AI-DRIVEN-OPTIMIZATION-RESOURCE-MANAGEMENT-AND-SECURITY-IN-CLOUD-COMPUTING-ENVIRONMENTS.pdf

Rjoub, G., Bentahar, J., Abdel Wahab, O., & Saleh Bataineh, A. (2021). Deep and reinforcement learning for automated task scheduling in large-scale cloud computing systems. *Concurrency and Computation: Practice and Experience*, *33*(23), e5919. https://www.researchgate.net/profile/Gaith-Rjoub/publication/341538799_Deep_and_Reinforcement_Learning_for_Automated_Task_Scheduling_in_Large-Scale_Cloud_Computing_Systems/links/5ec5ef4d299bf1c09acfadcb/Deep-and-Reinforcement-Learning-for-Automated-Task-Scheduling-in-Large-Scale-Cloud-Computing-Systems.pdf

Safarzadeh, V. M., & Loghmani, H. G. (2021). Artificial Intelligence in the Low-Level Realm--A Survey. *arXiv preprint arXiv:2111.00881*. https://arxiv.org/pdf/2111.00881

Shankar, V. (2025). Machine Learning for Linux Kernel Optimization: Current Trends and Future Directions. *International Journal of Computer Sciences and Engineering*, *13*(3), 56-64. https://www.academia.edu/download/122243023/Machine_Learning_for_Linux_Kernel_Optimization_Current_Trends_and_Future_Directions.pdf

Shyalika, C., Silva, T., & Karunananda, A. (2020). Reinforcement learning in dynamic task scheduling: A review. *SN Computer Science*, *1*(6), 306. https://www.researchgate.net/profile/Chathurangi-Shyalika-2/publication/344477959_Reinforcement_Learning_in_Dynamic_Task_Scheduling_A_Review/links/5f7b448e92851c14bcaf0d16/Reinforcement-Learning-in-Dynamic-Task-Scheduling-A-Review.pdf

Vishwarupe, J. (2024). *Distributed Caching with Disaggregated Memory: High-Performance Computing in the Modern World* (Doctoral dissertation, University OF Pennsylvania). https://www.cis.upenn.edu/wp-content/uploads/2024/05/Thesis-Final-Vishwarupe.pdf

Wang, Y., & Xing, S. (2025). AI-Driven CPU Resource Management in Cloud Operating Systems. *Journal of Computer and Communications*, *13*(6), 135-149. https://www.scirp.org/pdf/jcc_1733192.pdf

Xiong, C., Zhu, T., Dong, W., Ruan, L., Yang, R., Cheng, Y., ... & Chen, X. (2020). Conan: A practical real-time apt detection system with high accuracy and efficiency. *IEEE Transactions on Dependable and Secure Computing*, *19*(1), 551-565. https://ieeexplore.ieee.org/abstract/document/8979384/

Ye, Z., Gao, W., Hu, Q., Sun, P., Wang, X., Luo, Y., ... & Wen, Y. (2024). Deep learning workload scheduling in gpu datacenters: A survey. *ACM Computing Surveys*, *56*(6), 1-38. https://dl.acm.org/doi/pdf/10.1145/3638757

Yuan, Y., Iyer, A., Ma, L., & Talati, N. (2025). Vortex: Overcoming Memory Capacity Limitations in GPU-Accelerated Large-Scale Data Analytics. *arXiv preprint arXiv:2502.09541*. https://arxiv.org/pdf/2502.09541

Zhang, Y., Zhao, X., Li, Z., Yin, J., Zhang, L., & Chen, Z. (2024). Integrating Artificial Intelligence into Operating Systems: A Comprehensive Survey on Techniques, Applications, and Future Directions. *arXiv preprint arXiv:2407.14567*. https://arxiv.org/pdf/2407.14567

Zhang, Y., Zhao, X., Yin, J., Zhang, L., & Chen, Z. (2024). Operating system and artificial intelligence: A systematic review. *arXiv e-prints*, arXiv-2407. https://ui.adsabs.harvard.edu/abs/2024arXiv240714567Z/abstract

Zipperle, M., Gottwalt, F., Chang, E., & Dillon, T. (2022). Provenance-based intrusion detection systems: A survey. *ACM Computing Surveys*, *55*(7), 1-36. https://dl.acm.org/doi/pdf/10.1145/3539605

Zoppi, T., Ceccarelli, A., & Bondavalli, A. (2021). Unsupervised algorithms to detect zero-day attacks: Strategy and application. *Ieee Access*, *9*, 90603-90615. https://ieeexplore.ieee.org/iel7/6287639/9312710/09461213.pdf

.