ISSN: 3067-1140

AMERICAN ONLINE JOURNAL OF SCIENCE AND ENGINEERING

VOLUME: 04 ISSUE: 01

RECEIVED: JANUARY 26 REVISED: FEBRUARY 02 ACCEPTED: FEBRUARY 27 PUBLISHED: MARCH 08

AI-Driven Cloud Resource Optimization for Enterprise Applications

Katarzyna Nowak
Asst Professor

Abstract

Cloud resource consumption is vital to the operational costs of enterprise applications, and it is often a main explanatory variable driving the performance experience. Hyper-scale public cloud providers such as Amazon, Microsoft, and Google have been investing in making resource consumption simple and cost-effective across different resource layers: virtual machines, containers, serverless, and database services. However, the rich functionality, flexibility, and cloud-native advantages of enterprise applications come with complexity that transforms any cost benefits into a burden and can negatively affect resource consumption efficiency, especially at very high scale. Empirical studies have shown that a single large-scale business application deployed and operating in the cloud costs many millions of dollars every month. Addressing these cost issues and improving efficiency at such scale require AI, ML, and Data products at their core. Optimization of resource consumption under cost and performance objectives becomes essential. Solutions using AI for data-driven resource optimization, prediction, and forecasting can be utilized on their own whenever a given cost-resource dimension is analysed or combined into a single cost-performance optimization through a case study on a real-world enterprise cloud application.

AI and ML solutions for cloud resource optimization have already been proposed and evaluated in situations such as demand prediction and capacity planning, autoscaling policies, performance-aware workload scheduling, workload-specific service selection, anomaly detection, predictive maintenance, and cost optimization. Promising results have been obtained in terms of cost, performance, and application experience. However, a single solution at a time is not sufficient for producing world-class cost-performance efficiency. A comprehensive AI-powered resource optimization framework leveraging internal or external data sources for the given task is needed to be analysed and deployed with each deployment at every stage of the data-life-cycle process while also considering specific planning needs.

Keywords : Artificial Intelligence in Cloud Computing,Cloud Resource Optimization,Machine Learning for Resource Allocation,Large-Scale Enterprise Applications,Cloud Infrastructure Management,Intelligent Workload Scheduling,Auto-Scaling and Resource Provisioning,Cloud Performance Optimization,Predictive Analytics in Cloud Systems,Distributed Cloud Computing Systems.

1. Introduction

Deploying and operating enterprise applications on public clouds incurs significant costs. Recent reports from Amazon Web Services and Microsoft Azure indicated that cloud service expenses accounted for 22% and 22.8% of total corporate expenditure in 2021, respectively. As cloud service usage continues to escalate at a rapid pace, optimizing the cost-performance trade-off—maximizing performance while maintaining low cloud service expenses—has emerged as a top priority for many organizations.

Despite these considerable investments in public cloud infrastructure, many organizations remain dissatisfied. Indeed, capacity provisioning remains a widely acknowledged concern for domain experts and industry practitioners alike. Predictive demand modeling, especially incorporating time-series forecasting, offers the potential for establishing autoscaling policies that minimize

ISSN: 3067-1140
AMERICAN ONLINE JOURNAL OF SCIENCE AND ENGINEERING
VOLUME: 04 ISSUE: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 02 ACCEPTED: FEBRUARY 27 PUBLISHED: MARCH 08

cost as demand fluctuates. Yet predictive demand models achieve limited accuracy, and hence such cost-minimization approaches are rarely useful in practice. Rather, Informatica's 3D Cloud, a generative cloud resource allocation service based on pure reinforcement learning, epitomizes the state of the art in auto-scaling. Highly scaled and responsive portals such as global e-commerce platforms would benefit from applying similar predictive demand forecasting techniques for cloud autoscaling within these cloud environments. Current demand and supply modeling techniques are applicable only during planned capacity expansions rather than unplanned demand spikes or internal metric-regulated automatic scaling without predictive modeling capabilities. AI-powered cloud resource optimization is therefore alluded to as an important challenge that has yet to be convincingly solved.

1.1. Background and Significance

Massive, global cloud infrastructures provide resizable resources for multiple applications and services, enabling organizations to control operating costs, align on-demand resource allocation with business concomitance and avoid service outages. However, such cloud services are not yet delivered free of charge: a price is paid to leverage the operational and engineering capabilities of hyperscalers. As an outcome, cloud infrastructure operating bills can reach very high values, ranging from several thousand to few million dollars per month. An employed computing resource can cost 10 to 100 times more than on-premise infrastructure, mainly due to the trade-off between the cloud offer flexibility and performance. Thus, organizations demand high Quality of Service (QoS) for their applications.

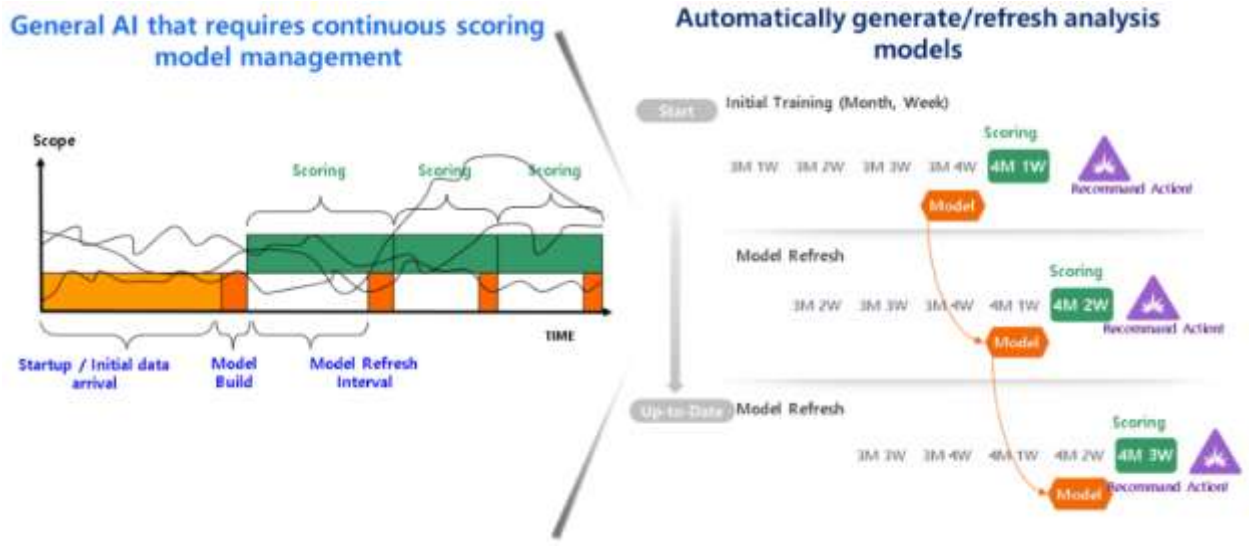


Fig 1: AI-Enhanced Cloud Resource Optimization

1.2. Research design

Research design comprises elements of activity that capture (i) the scope of the study; (ii) data required to conduct it; (iii) the specific research questions and hypotheses guiding the activity; (iv) the experimental setup employed; and (v) the evaluation framework adopted. Research questions for the considered domain can be stated as follows: What reduction in cost could AI-

powered optimization techniques produce during a particular period? Would it be possible to accomplish such reduction without degrading quality? When this question is answered positively, there is room for a second question: What was the estimated saving in monetary terms? To validate these questions these knowledge areas need to be covered with sufficient detail. The activity is then executed with the case study of a global e-commerce platform.

Numerous public-domain datasets containing telemetry data of cloud resources are now available; some data generation tools offer the advantage of being portable to an environment where the resources are deployed. Having observed the success of such approaches, new models have blended them with time-series prediction to generate realistic synthetic workloads for service-level agreements and performance-testing environments. The predictive models therefore became the often-successful support for experiments. For the effort undertaken, the activity produced a predictive scaling model of demand in a special part of the application dealing with retail data and an intelligence resource-orchestrating policy that applied when the incoming load exceeded the predicted threshold. The model and its associated policies were assessed not just for saving cost but also for achieving these savings without degrading performance.

Equation 1: Multi-Cloud Placement Derivation

must sum to one:

$$\sum_k x_k = 1, \quad x_k \geq 0$$

Step 2: Expected unit execution cost is

$$C = \sum_k c_k \cdot x_k$$

Step 3: Expected latency is

$$L = \sum_k l_k \cdot x_k$$

Step 4: Add transfer or compliance cost m_k if needed:

$$C_{total} = \sum_k (c_k + m_k) \cdot x_k$$

Step 5: A weighted placement objective becomes

$$\min_x \; C_{total} + \theta \cdot \max(0, L - L^*)$$

2. Fundamentals of AI-Driven Resource Optimization

A well-defined problem space and AI-driven solution strategy pave the way for practical optimization in cloud environments. Enterprise applications are resource-intensive and costly to run in the cloud. They typically need to support hundreds of thousands of concurrent users, serve millions of requests per second, and provide low latency. Hundreds of virtual machines process user requests, background tasks, and data analytics in scale. Multiple instances of disordered jobs with different priority levels are scheduled into a bank of containers. A combination of on-demand, reserved, and spot instances is used in production to optimize costs. Satisfying performance objectives with a complex system comes at a high cost. Infrastructure costs can exceed millions of dollars a month, requiring careful optimization. Attempts to cut costs without losing SLA adherence have produced good, but not production-ready results. Automating performance engineering using AI is a natural fit.

The AI techniques used in resource optimization are diverse because the problem space covers different aspects of cloud environments. Reinforcement learning can optimize resource scaling, supervised learning can model output quality to compute cost, and time-series forecasting can predict demand to trigger autoscaling events. AnSLAs. Service-level objectives define target

ISSN: 3067-1140
AMERICAN ONLINE JOURNAL OF SCIENCE AND ENGINEERING
VOLUME: 04 ISSUE: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 02 ACCEPTED: FEBRUARY 27 PUBLISHED: MARCH 08

values for latency, errors rate, and availability, to name just a few resources are costly enough that even slight savings are beneficial. Cloud provider costs differ, so hybrid architectures can reduce expenses when implemented correctly.

Equation 2: Drift Detection and Retraining Derivation

Step 1: Over a reference window of m observations, compute baseline mean error and baseline standard deviation:

$$\bar{e}_{ref} = \frac{1}{m} \sum e_t$$

$$s_{ref} = \sqrt{\frac{1}{m-1} \sum (e_t - \bar{e}_{ref})^2}$$

Step 2: Over a recent window, compute recent mean error $\bar{e}_{new}$.

Step 3: A standardized drift score is

$$z = \frac{\bar{e}_{new} - \bar{e}_{ref}}{s_{ref} / \sqrt{m}}$$

Step 4: If $|z|$ exceeds a control threshold, or if MAPE exceeds an operational threshold, trigger retraining.

MAPE itself is derived as

$$MAPE = \frac{100}{N} \sum \left| \frac{D_t - F_t}{D_t} \right|$$

2.1. Resource Abstractions in the Cloud

Cloud service providers offer infrastructure-level resources corresponding to abstractions in computing, memory, storage, networking, and services. Resource offering and cost structures are tightly related to how resources are abstracted, market dynamics, and growth trends. Key abstractions are physical or virtual machines, containers, and serverless functions. Resource abstractions can be viewed as a hierarchy of three primary offerings: instances, containers, and serverless. Each abstraction can be classified in the context of primary compute vs. primary memory network interactions, and their role in communicating with the network and storage subsystems.

Cloud providers present several cost-performance trade-offs when making reservations for compute resources. When using an instance, a full server is hired during the reservation period, independent of whether the server runs at full capacity or not. More complex demand patterns can be partially addressed by autoscaling policies, but large discrepancies would still incur idle cost and demand spikes lead to SLA breaches. Containers can mitigate these disadvantages by allowing multiple users to support a single server, but their usage can still lead to idle costs. Serverless functions represent a further evolution of the service model by charging only for the time that code is really being executed.

Cloud providers offer different cost–performance trade-offs when reserving compute resources. In traditional instance-based models, users rent an entire server for a fixed reservation period, regardless of whether the server is fully utilized. This often results in wasted resources and unnecessary costs when workloads are low. Although autoscaling policies can handle certain variations in demand, significant fluctuations still create challenges: low demand leads to idle resources and higher costs, while sudden spikes may cause SLA violations due to insufficient capacity. Containers improve resource utilization by enabling multiple applications or users to share a single server, reducing some idle overhead. However, containers may still incur costs when resources remain allocated but underused. Serverless computing further advances this model by eliminating the need to reserve servers entirely; instead, users are billed only for the actual execution time of their code, leading to more efficient cost management and better alignment between usage and payment.

ISSN: 3067-1140
AMERICAN ONLINE JOURNAL OF SCIENCE AND ENGINEERING
VOLUME: 04 ISSUE: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 02 ACCEPTED: FEBRUARY 27 PUBLISHED: MARCH 08

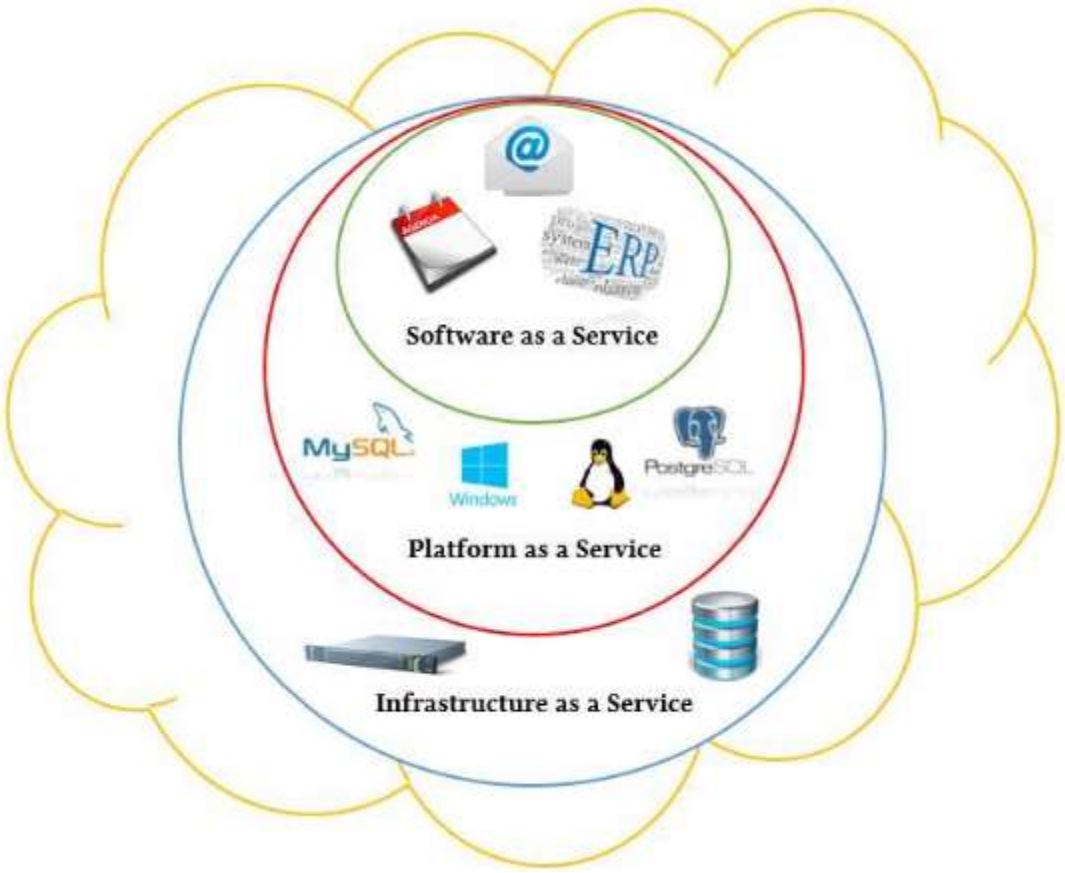


Fig 2: Cloud Computing Abstraction Layers

2.2. Core AI Techniques for Optimization

Reinforcement learning is a generalization of dynamic programming for large or infinite state spaces, allowing agents to learn optimal policies through trial and error from the environment in the form of feedback signals. An agent can directly interact with an environment, keep an internal state, learn good policies using an off-policy strategy, learn when to explore instead of exploiting, and learn such policies that maximize rewards over time. It scales well to action spaces like those seen in workload scheduling and autoscaling and has recently been shown to yield near-optimal results for resource allocation under SLA constraints.

Supervised learning is a classical model-learning framework consisting of a repetitive input-output mapping, where the aim is to uncover the mapping from a set of empirical data pairs. Fully-trained supervised models are able to capture complex input-output relationships, scaling directly with the curse of dimensionality. Such models can be used to support core components of the application lifecycle (e.g., workload characterization), emulate expensive prediction functions (e.g., demand forecasting), reduce

uncertainty in allocated resources (e.g., usage prediction), learn policies (e.g., the predictive scaling model), empirically characterize the environment (e.g., throughput-delay functions), and couple these mathematical and Machine Learning models.

3. Architectural Considerations for Enterprise Deployments

Data governance covers all measures that enable organizations to become competent in the identification and management of data. The primary objectives of data governance are to control data accuracy, quality, privacy, and access. Governance frameworks define the policies, responsibilities, and standards related to data. In enterprise cloud resource optimization, data governance is particularly vital for building and updating cloud resource optimization models. Such models depend on a rich set of telemetry from cloud infrastructures and services. Managing telemetry quality and reliability is, therefore, essential when different definitions and formats are used across providers. It is also crucial for respecting any privacy policies imposed on processed data. Ad-hoc data access controls are often required for governance capabilities to comply with other regulations, such as the GDPR.

Data observability is the capability of monitoring telemetry from applications and their underlying infrastructure. Most organizations have a clear observability strategy for their applications and cloud infrastructure. This strategy typically includes methods to collect and manage metrics, logs, traces, and application performance monitoring data. Adequate observability techniques are paramount for incident detection, diagnosis, root cause identification, and remediation. In cloud resource optimization, observability is vital for ensuring that models provide reliable outputs and all necessary input telemetry is available. Telemetry gaps can severely impact model prediction quality and any dependent optimization action, leading to SLA violations or other critical failures. A well-designed observability strategy incorporates dedicated telemetry health-checking dashboards and alerting mechanisms that can inform the organization when a data quality incident occurs.

ISSN: 3067-1140

AMERICAN ONLINE JOURNAL OF SCIENCE AND ENGINEERING

VOLUME: 04 ISSUE: 01

RECEIVED: JANUARY 26 REVISED: FEBRUARY 02 ACCEPTED: FEBRUARY 27 PUBLISHED: MARCH 08

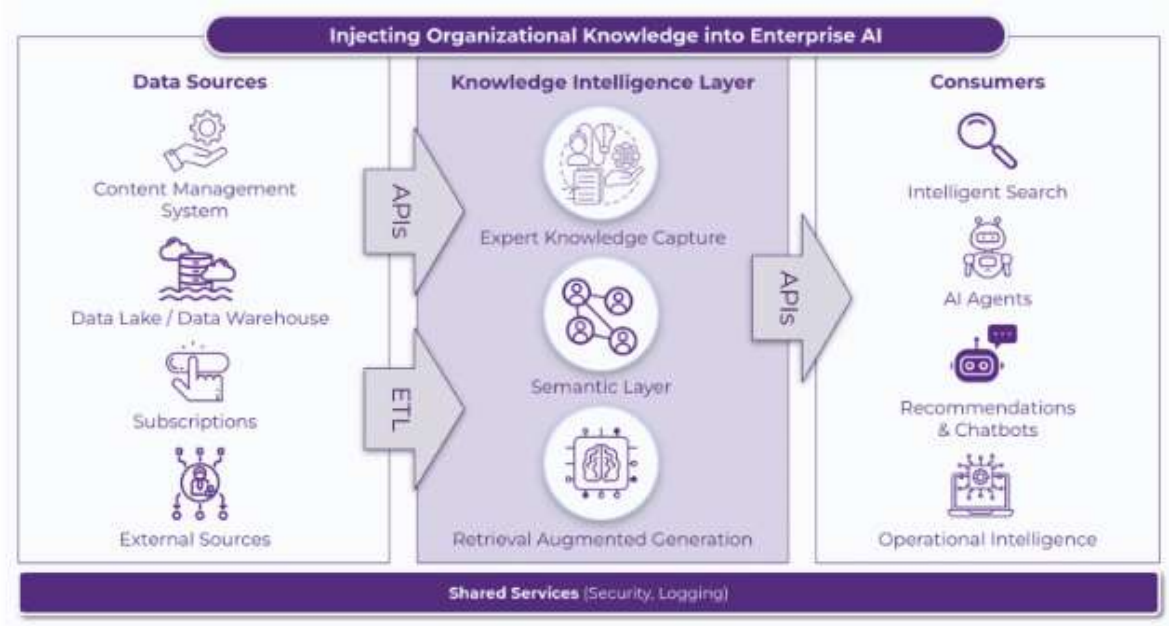


Fig 3: Enterprise AI Architecture Series

3.1. Data Governance and Observability

Data governance for an enterprise requires defining data lineage, quality, protection, and access controls. Data lineage captures where data originates, the steps and transformations applied, where and how it is stored, and where it goes next. Data Quality is about maintaining accuracy, completeness, and consistency for analysis and trained models. Data privacy concerns how to avoid leakage of business-sensitive information and how to avoid developing models based on attributes that are protected. After data lineage, controls determine who has access to data collections and for which purpose; can users connect, query and project data, can data be exported or downloaded, and if so under which conditions.

Observability is key for trustworthy deployments. Telemetry should cover metrics, traces, logs, and distributed context. Higher-level dashboards summarize the state of a solution and support operational monitoring. When everything works as expected, assigned teams and owners react and follow a well-defined normalization process. When things do not go as expected, observability is the key to rapid identification of the underlying root cause and remediation of the incident.

3.2. Multi-Cloud and Hybrid Environments

Cloud strategies are increasingly multi-cloud and/or hybrid, a natural evolution of the underlying technologies and market trends: many users operate with multiple cloud vendors, be it deliberately for interoperability reasons or more out of data gravity; cloud technologies are being introduced on-premises, either by businesses that want to preserve investment on existing facilities or by firms for whom the cloud was never really an option, often in regulated industries or countries.

Open ecosystems supporting portability across providers and synchronization with on-premises installations can also support hybrid combinations, whereby a workload runs mainly in the public cloud but can dynamically spill over to an on-premises facility should its capacity be outgrown; this requires no cloud vendor to be the ultimate choice or master of the deployment. A multi-cloud or hybrid strategy, however, complicates the data management. As different vendors provide different services at different quality levels, often with very different pricing shelves, choosing the optimal place to execute a specific task or to store a specific data set becomes a much harder problem. It is no longer sufficient for the data owners to implement data governance policies inside a single-cloud ecosystem. They require a more generic multi-cloud data governance solution handling seamlessly data across providers.

Equation 3: Reinforcement Learning Derivation

Step 1: Define state s_t = [forecast, uncertainty, utilization, latency, instance count, time-of-day, cost state].
Step 2: Define action a_t in {scale in, hold, scale out} or a larger discrete action set.
Step 3: Let immediate reward be negative cost and penalty:
$r_t = - [p_c*n_t + X_t + beta*max(0, L_t - L*)]$
Step 4: The discounted return from time t is
$G_t = sum_{k=0}^infty gamma^k r_{t+k}$
Step 5: The optimal action-value function is
$Q*(s,a) = max_pi E_pi[G_t | s_t=s, a_t=a]$
Step 6: Bellman's optimality equation gives
$Q*(s,a) = E[r_t + gamma * max_{a'} Q*(s_{t+1}, a') | s_t=s, a_t=a]$
Step 7: Q-learning updates estimates by
$Q_new(s_t,a_t) = Q_old(s_t,a_t) + eta * [r_t + gamma*max_a Q(s_{t+1},a) - Q_old(s_t,a_t)]$

4. Modeling and Optimization Strategies

Predictive scaling and autoscaling policies shape the optimization problem surrounding budgeting and provisioning. Demand can be modeled using time-series approaches, driven by location, season, day of the week, hour of the day, or a combination of these factors. Precise workload predictions should minimize transfer costs while remaining accurate enough to prevent performance drop. Temporal guardrails prevent scaling compute capacity before demand is forecasted to exceed baseline service-level objectives (SLOs), and to avoid triggering scaling actions when workload is expected to remain below a minimum threshold.

Scaling behavior is governed either by classic autoscaling policies—based on metrics such as CPU utilization—and potential incapacity alerts contacted by the cloud service provider, or by custom rule sets that specify capacity as a function of occupancy level in a queuing model. Service impacts stem from predicting usage at risk of breaching SLOs, since optimal scaling policy connects demand predictions with capacity provisions to minimize performance drop. Priority dispatching governs workload scheduling, since quality-sensitive capacity must be provisioned for high-priority service before arrival of the affected requests. Support for affinity or anti-affinity preference among workloads also influences scheduling, and occupancy throughput trade-offs must be assessed for queuing optimizations.

ISSN: 3067-1140
AMERICAN ONLINE JOURNAL OF SCIENCE AND ENGINEERING
VOLUME: 04 ISSUE: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 02 ACCEPTED: FEBRUARY 27 PUBLISHED: MARCH 08

Further methods focus on workload characterization and scheduling. Workloads disambiguated by identity, volume, and priority support accurate predictive models. The combined effect of unbalanced, non-static workload distribution remains poorly understood; whether all components of a system should be a single-point failure, and which components should be colocated or separated during scheduling, are also unresolved questions. An open list of characteristic switching points addresses the relationship between the duration of separate runs in distinct configurations—capacity-optimized versus cost-optimized sub-components—and the gains perceived, including occupancy-throughput links and queuing theory together with probability distributions that model wait times as a function of total volume.

Further methods emphasize **workload characterization and scheduling strategies** to improve system efficiency and reliability. Workloads can be differentiated by **identity, volume, and priority**, allowing predictive models to estimate demand patterns and allocate resources more accurately. However, the effects of **unbalanced and dynamically changing workload distributions** remain insufficiently understood, particularly in large-scale or distributed systems. Key architectural questions also arise, such as whether system components should rely on **single-point control or failure domains**, and which components should be **co-located or separated** to achieve optimal scheduling and resilience. Another important research direction involves identifying **characteristic switching points**—thresholds at which systems shift between different configurations, such as **capacity-optimized** and **cost-optimized** operational modes. Understanding the duration and timing of these configuration runs can help quantify performance gains. In this context, relationships between **system occupancy and throughput** become critical, and analytical tools from **queuing theory and probability distributions** are often used to model expected wait times based on overall workload volume and resource availability.

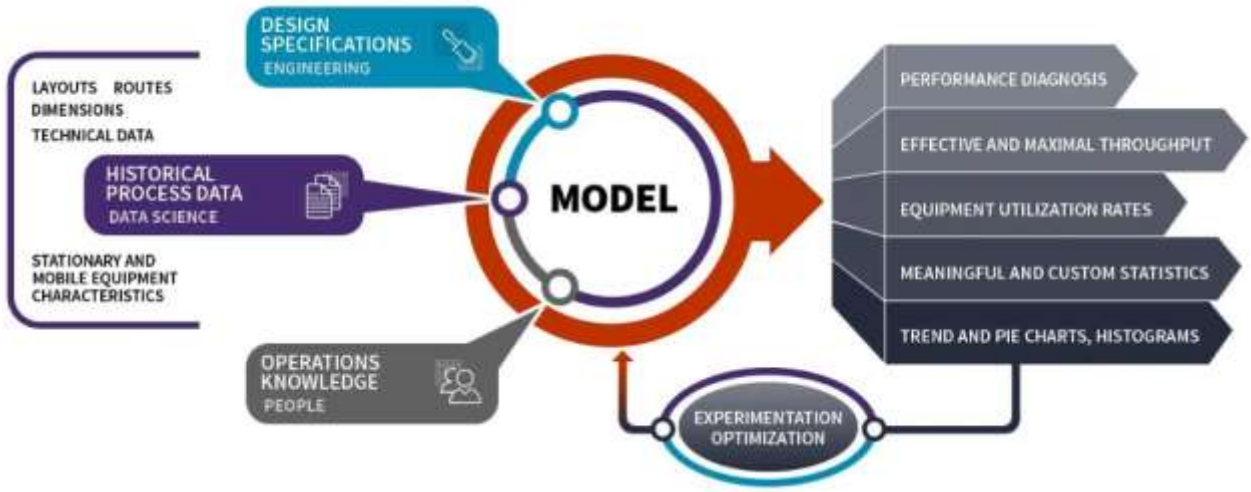


Fig 4: Modelling for Better Operations Scheduling

4.1. Predictive Scaling and Autoscaling Policies

Demand for cloud resources is difficult to predict accurately, especially in highly variable situations such as serverless functions or online retail. Depending on the workload type, overprovisioning may be acceptable or even necessary, while

ISSN: 3067-1140
AMERICAN ONLINE JOURNAL OF SCIENCE AND ENGINEERING
VOLUME: 04 ISSUE: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 02 ACCEPTED: FEBRUARY 27 PUBLISHED: MARCH 08

underprovisioning should be avoided. Costly performance degradation incurs penalties through service-level agreements (SLAs) that could involve loss of business. Hence, service owners often employ autoscaling policies, which receive feedback on demand and automatically adjust resource capacity or provisioning. Such policies augment standard sizing decisions, which determine the resources allocated to a service instance.

In cloud environments, predictive demand scaling seeks to optimize time-varying demand over multiple time scales, addressing demand variations longer than minutes to hours. Accurate forecasts enable better decision making, but errors signal uncertainty and must be translated into scaling guardrails. Telemetry data can inform policies that scale resources up and down in response to demand, prioritizing prediction accuracy for scaling up to avoid SLA violation while making scaling down less aggressive to minimize reaction drift.

Equation 4: Workload Scheduling Derivation

Step 1: Define a scheduling score where larger is more urgent:

$$S_i = w1 * p_i - w2 * r_i + w3 * affinity_i - w4 * interference_i$$

Step 2: If server j has free resource vector h_j, job i may be placed only if

$$g_i <= h_j \quad \text{(component-wise)}$$

Step 3: Centralized dispatch chooses

$$j* = argmax_j S_{ij}$$

over feasible servers j. In queue-based scheduling, high-priority jobs can also receive lower waiting penalties, giving a weighted objective:

$$J_sched = sum_i c_i * T_i$$

4.2. Workload Characterization and Scheduling

Demand for cloud resources from services and applications can be highly variable, with both diurnal and aseasonal effects. To mitigate the costs associated with over-provisioning, cloud providers often employ predictive scaling mechanisms that attempt to anticipate upcoming demand and adjust resources accordingly. Although such methods can be effective, they do not guarantee that an SLA will be satisfied. In many enterprise deployments, failure to respect the SLA can have severe business implications. Consequently, it is common practice to bolster predictive scaling with auto-scaling policies based on defined threshold triggers, where the underlying assumption is that sensors can reliably report the state of the system. Typically, these triggers utilize support vectors for horizontal scaling of virtual machine instances or add and remove resources in containers and serverless deployments. Although these mechanisms can be effective in providing low-latency responsiveness, they must be treated with caution to avoid overshooting or undershooting goals. Predictive models can make useful contributions to defining thresholds for these reactive policies, by enabling rules that declare increasing danger ahead well before a threshold breach is expected to occur.

Various workloads place pressure on the services hosted within the cloud environment. Workflows that demand latency or throughput should be prioritized, while workloads that contribute shear and should therefore be kept apart can be delayed until the appropriate resources are available. Affinity and anti-affinity relationships between workloads can be captured and fed to a centralized scheduler that reallocates workloads to the appropriate queue. Depending on load levels, the workload queues at the input and output of an environment can either be operated for occupancy or throughput, thus acting as an explicit trade-off

between latency and resource utilization. Such a holistic approach to workload characterization helps maintain a healthy system, improves high-level metrics for the environment, and maximizes resource utilization.

5. Experiments, Validation, and Case Studies

Empirical validation of AI-driven solutions is crucial for establishing their accuracy, robustness, and practicality. An efficient strategy involves answering how the proposed AI technique connects to predictive scaling or autoscaling policies. First, demand prediction accuracy is assessed prior to testing whether scaling thresholds are appropriately tuned. Connections to scaling policies are then established, critical for guaranteed SLA compliance. This exploratory study is conducted for an unknown class of workloads, with the goal of cost minimization for a predefined level of service quality. A case study featuring a leading global e-commerce platform uses real demand data from multiple countries and underlying cloud services, enabling reliable conclusions.

An empirical analysis was performed to assess the practical relevance of the proposed AI technique. The goal of the method was to minimize costs while ensuring that performance adhered to the service-level objectives. Real demand data for various countries and the related cloud services of a leading global e-commerce platform were employed. The AI solution provided the predicted demand over a forward horizon and was combined with threshold-based predictive scaling policies. The trade-off between cost and performance was analyzed, and the results demonstrated its applicability in real-world scenarios. In particular, the precision of the demand predictions and the tuning of scaling thresholds were identified as relevant aspects that affected the cost-performance relationship.

Equation 5: Queueing-Theory Derivation for Latency

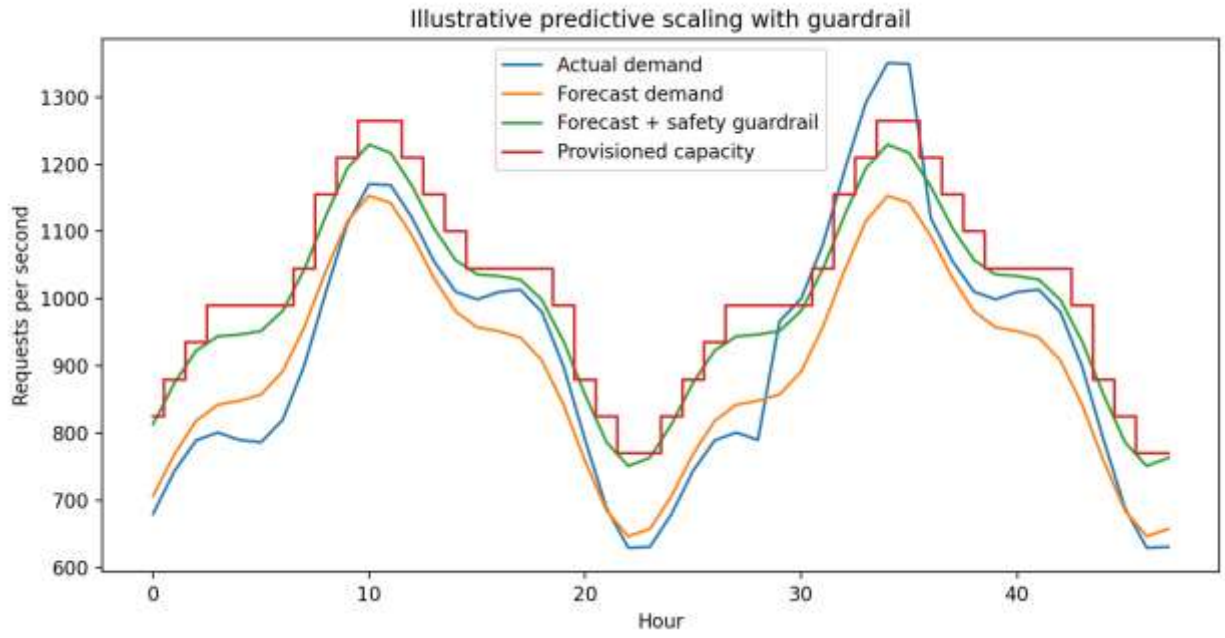
Step 1: Total offered load is
$a = \lambda / \mu$
Step 2: Utilization is
$\rho = \lambda / (c * \mu)$
For stability, $\rho < 1$ must hold.
Step 3: Probability of zero jobs in system is
$P_0 = [\sum_{n=0}^{c-1} (a^n / n!) + (a^c / (c! * (1-\rho)))]^{-1}$
Step 4: Erlang-C waiting probability is
$P_{wait} = (a^c / (c! * (1-\rho))) * P_0$
Step 5: Mean queueing delay is
$W_q = P_{wait} / (c\mu - \lambda)$
Step 6: Mean response time is service plus waiting:
$W = W_q + 1/\mu$
Step 7: An SLA constraint can therefore be written as
$W <= L^*$

AMERICAN ONLINE JOURNAL OF SCIENCE AND ENGINEERING
ISSN: 3067-1140
VOLUME: 04 ISSUE: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 02 ACCEPTED: FEBRUARY 27 PUBLISHED: MARCH 08

5.1. Methodology for Empirical Validation

An experimental design that evaluates real setups, large-scale workloads, and baselines representative of current practices enables realistic assessment of AI-powered cloud optimization solutions. Although private data precludes public sharing of benchmarks, the methodology can be applied to other cases.

Validation of cloud resource optimization strategies requires experimental evaluation over real deployments with large workloads. Public cloud platforms enable the creation of such conditions by allowing users to consume resources at varying scales and create topologies that resemble global service providers. However, optimizations that target purely academic setups, possibly with unrealistic or unknown cloud costs, often enjoy little interest from industry practitioners. Thus, it is critical to design experiments over setups and workloads that reflect private organizations rather than academia, as is the case with an exploration of open-source models for workload characterization.



5.2. Case Study: Global E-Commerce Platform

A case study on the AI-powered optimization of a leading global e-commerce platform illustrates the utility of the presented framework. The industrial partner sought an approach for optimizing cloud resource usage and cost as part of its multi-cloud strategy, aiming to balance cost with performance indicators such as latency, availability, missed revenue, and service-level agreements. The originating, transient, and service-oriented nature of the e-commerce platform workload provided a relevant context for experimentation and validation.

Research conducted prior to this study identified predictive demand scaling as a promising area for cost savings. An exploration of the accuracy of forecasts and the implications for cost and performance highlighted a practically relevant open question. Multiple supervised models, including regression, boosting, tree-based, and recurrent, were trained to predict incoming requests at a granular level.

6. Challenges, Risks, and Mitigation

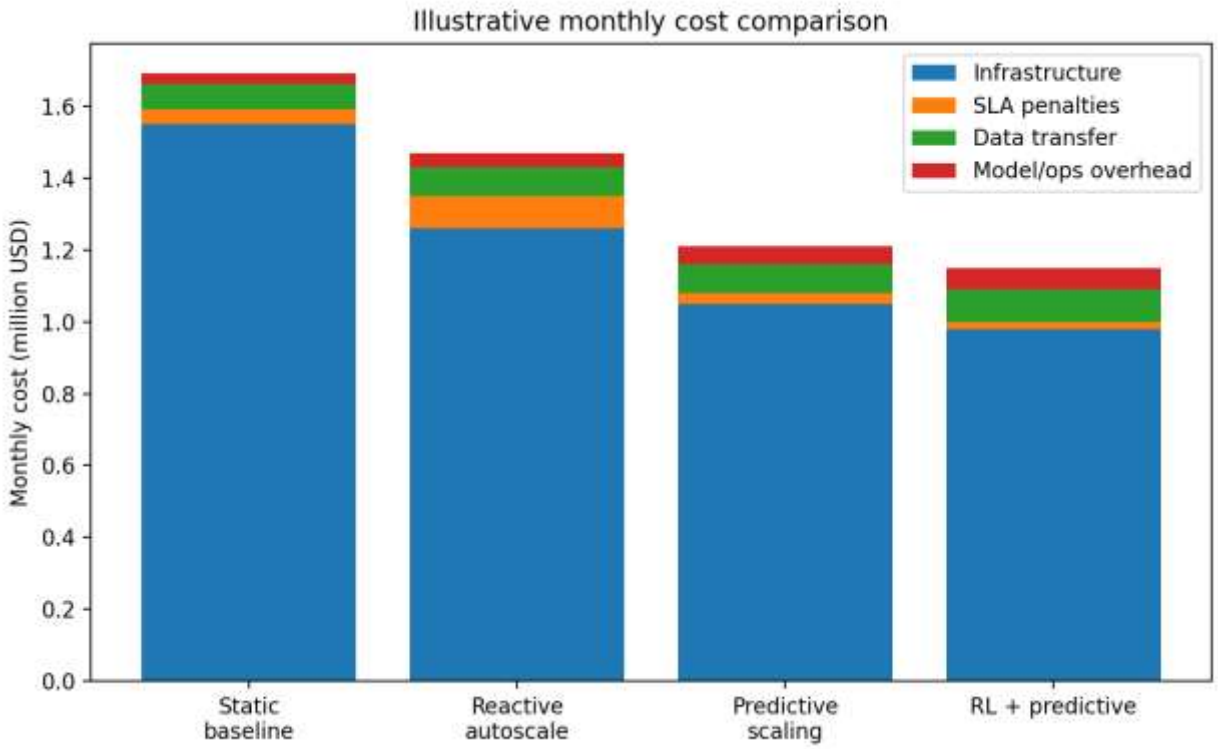
A wide range of factors can negatively affect the quality and stability of predictive models, including but not limited to: data ingestion faults (e.g., misconfigurations, external data sources failing), changes in data lineage affecting telemetry quality, broken telemetry instruments causing data dropouts, and actual changes in the underlying system (e.g., workload or usage profile changes). Warning signals of such issues may be detected in production through a variety of monitoring approaches that assess model performance against SLA objectives (for example, through routine backtesting using a recent period of held-out historical telemetry data), as well as through ad-hoc rule-based detection of telemetry anomalies or drifts. Such monitoring can enable corrective action, including failure investigation, data auditing and cleaning, retraining schedules, and manual evaluation of model quality/validity in production.

Another known risk comes from model drift, which may lead to loss of quality over time relative to SLA objectives. To address this concern, retraining schedules should be established based on model criticality, uncertainty over model quality, detectability of drift, and cost of retraining and deployment – with the simplest approach being periodic retraining at some fixed frequency. When retraining is triggered, instead of possibly incurring costly A/B or shadow testing, the new automatically produced model can instead be evaluated against older retained models before deployment. A conventional evaluation process can be followed for important high-traffic paths or large-scale machine learning (ML) models, using the operations team's historical set of valid data and validating outputs to confirm the model meets the desired SLO.

Interoperability and Vendor Lock-In

Second, system architectures should enable large-scale data transfers with costs kept at a minimum, allowing data to flow in and out without artificial constraints. Third, third-party services that help manage capacity and cost switching by automatically polling providers should be investigated, and, finally, periodic assessment and planning of system utilization should ensure a viable path for data accumulation in a particular provider, either for one service or a group of services.

ISSN: 3067-1140
AMERICAN ONLINE JOURNAL OF SCIENCE AND ENGINEERING
VOLUME: 04 ISSUE: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 02 ACCEPTED: FEBRUARY 27 PUBLISHED: MARCH 08



6.1. Data Quality and Model Drift

Not all deployments are equally affected by data quality. The major requirements for low-quality data are being a supervised task, learning on a drift-prone distribution, and also having a demand for low prediction errors combined with a high utilization of the predictions. In predictive scaling, model drift can be kept under control. Major demand drifts can be detected through quantile-based analysis, and infrequent retraining can keep forecast accuracy high, thereby minimizing scaling mispricings. Similarly, approach to predictive load-based scheduling relies on the underlying queuing model.

The main scaling approach tries to model all application and system characteristics that determine the adequacy of resources for the demand. The operating principle is simple. A set of thresholds is defined, each defined for a set of application-level service responses related by some property (e.g., similar latency), so that when any of those thresholds is crossed a resource is added or removed. The basic unit is taken to be a single unit addition/removal of a resource per threshold crossover; decisions of whether a throughput or latency or user response time or some combination should be imposed depend on the actual application being considered.

ISSN: 3067-1140
AMERICAN ONLINE JOURNAL OF SCIENCE AND ENGINEERING
VOLUME: 04 ISSUE: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 02 ACCEPTED: FEBRUARY 27 PUBLISHED: MARCH 08

6.2. Interoperability and Vendor Lock-In

Despite the large scale of cloud computing, many organizations rely on a single provider, raising concerns over long-term availability policies and possible monopolistic tendencies. Given the massive resource and data requirements, cloud computing remains the only practical option for many applications, but such a concentration poses a potential risk of not surviving in the long run. To mitigate that risk, and to meet requirements for low latencies and/or compliance, organizations deploy their applications in hybrid setups, combining the use of on-premises and multiple clouds. Private clouds often interoperate with those of one or two vendors but across clouds, little standardization of interfaces exists. The current approach is typically cloud-specific, and cost models do not account for data transfer between clouds, which can influence workload placement decisions. The absence of standard interfaces limits the extent of workload migration between clouds, and so does the use of proprietary formats and protocols. In addition to lowering costs, multi-cloud deployments can also improve resiliency; the failure of one vendor does not necessarily imply application downtime. However, operation across cloud vendors adds a layer of complexity to data governance and privacy, since those different clouds are governed by the compliance rules of different providers.

Adopting open-source software, known open interfaces, and cloud-agnostic formats minimizes these risks. Vendors provide tools for the seamless transfer of data in their environments, but users need to perform a cost-benefit analysis before relying on these tools. Solutions to interconnect their clouds should incorporate the cost of data transfer. Long-term, a clear cloud-vendor-agnostic exit strategy should also be in place; even if the chosen vendor currently meets all requirements, the business model may very well change over the years.

7. Conclusion

AI and ML techniques integrate natively with public cloud services, enabling teams to harness their potential for cloud resource optimization without incurring additional capital or operational expenditures. Consequently, AI- and ML-powered optimization techniques have amassed a robust portfolio of success stories, supported by various industries and published in high-impact media. However, few of these techniques have penetrated the enterprise sphere, where long-standing commitments to on-premises infrastructure investments have precluded the adoption of cloud-native solutions. The ensuing popularity of hybrid and multi-cloud deployments, along with the maturation of open-source and open-systems solutions, have turned this situation around: enterprise workloads can now be deployed on the public cloud or shared services offered by global cloud providers without compromising organizational needs and goals.

Several factors can be expected to accelerate the adoption of AI and ML techniques for public cloud resource optimization within enterprises. Multi-cloud environments governed by established IT service providers—teams who can ensure data quality and trustworthy models—are in a better position to capitalize on these cloud resource cost-saving opportunities. Within a multi-cloud environment, cost-performance trade-offs become much more relevant than for a single-cloud setting. Often, these public cloud-sensitive parameters become a key selection factor. Whenever native integration between AI- and ML-powered techniques and the public cloud resources in the selected deployment scenario is possible, the benefits are amplified, and the implementation does not require significant investments in a scalable production-governed AI-centric pipeline. However, as observed in the preliminary analysis of a real-world global e-commerce platform, the most relevant asset is still the governance, management, alerting, and enablement of the data. Inclusion at this level is indeed the foundation of any production-driven AI- and ML-power optimization initiative.

ISSN: 3067-1140
AMERICAN ONLINE JOURNAL OF SCIENCE AND ENGINEERING
VOLUME: 04 ISSUE: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 02 ACCEPTED: FEBRUARY 27 PUBLISHED: MARCH 08

Locking in a particular cloud service provider (CSP) implies possible dependence on such vendor and even on some particular services, which may lead to concerns about continuity and costs. To counteract such risks, several strategies that focus on cloud agility and interoperability can be followed. First, an openness of interfaces and formats, such as standard programming interfaces (APIs) or open data formats, together with documentations, enables development of wrappers, known connectors, and libraries that implement the same standard and facilitate using other providers' services.

Concept	Equation	Interpretation
Total cost	$J\_cost = \Sigma[p\_c\ n\_t + X\_t + \alpha \max(0, L\_t-L^*)]$	Minimize cloud spend plus SLA exposure
Forecast guardrail	$Guardrail\_t = F\_t + z\_q\ \sigma\_t$	Reserve headroom for uncertainty
Instance count	$n\_t = ceil((F\_t + k\sigma\_t)/\mu)$	Predictive scaling rule
Utilization	$\rho = \lambda/(c\mu)$	Measures proximity to saturation
Queue waiting	$W\_q = P\_wait/(c\mu-\lambda)$	Mean queue delay from Erlang-C
Response time	$W = W\_q + 1/\mu$	Latency used in SLA constraint
Autoscaling rule	$n\_\{t+1\}$ depends on thresholded U_t or $U\hat{}g\_t$	Reactive or hybrid control
RL reward	$r\_t = -[cost + SLA\ penalty]$	Lets policy learn long-run optimum
MAPE	$(100/N)\ \Sigma\ \|(D\_t-F\_t)/D\_t\|$	Forecast quality monitoring
Placement cost	$C\_total = \Sigma(c\_k+m\_k)x\_k$	Multi-cloud routing objective

Table : Derived Equation Summary Table

7.1. Future Directions

Several opportunities exist to advance AI-driven resource optimization for enterprise applications. Reducing the need for historical data by developing accurate ML-based scaling predictions can facilitate application of the technology to deployments that lack monitoring. Expanding optimization strategies beyond scaling, for example employing ML-driven workload models, could yield further cost savings. Addressing resource blended scaling, via optimization of combined storage and compute systems such as AWS Redshift, may enable practical resource optimizations for many organizations. Enabling complex workflows through specialized workload characterization and scheduling could reduce operational lattency in data-intensive systems. Adding support for resource placement across multiple cloud providers could enable enterprises to reduce their cloud service fees by taking advantage of larger, lower-cost service batches, while also overcoming regulatory requirements. Exploring the applicability of AI resource predictions within multi-service applications and under ultra-low-latency traffic requirements would contribute to performance within defined SLO boundaries. Integrating detailed resource health monitoring with predictive control theory could provide greater cloud utilization under defined SLOs.

Investigation of how ML resource predictions could enhance enterprise control of controlled-but-non-deterministic data flows or orders would yield insights for additional business applications. Accelerating and streamlining enterprise acceptance of AI resource predictions by harnessing simple, common-sense business perspectives and articulating the advantages in an accessible manner would increase the efforts probability of advancement and adoption.

8. References

1. Ding, D., Fan, X., Zhao, Y., Kang, K., Yin, Q., & Zeng, J. (2020). Q-learning based dynamic task scheduling for energy-efficient cloud computing. Future Generation Computer Systems, 108, 361–371.
2. Lu, H., Gu, C., Luo, F., Ding, W., & Li, X. (2020). Optimization of lightweight task offloading strategy for mobile edge computing based on deep reinforcement learning. Future Generation Computer Systems, 102, 847–861.
3. Miao, Y., Wu, G., Li, M., Zhang, Y., & Li, C. (2020). Intelligent task prediction and computation offloading based on mobile-edge cloud computing. Future Generation Computer Systems, 102, 925–931.
4. Mishra, S. K., & Manjula, R. (2020). A meta-heuristic based multi objective optimization for load distribution in cloud data center under varying workloads. Cluster Computing, 23(4), 3079–3093.
5. Kolla, S. H., & Mattaparthi, R. (2025). Hybrid Gen AI Systems: Integrating Small LMs with Large Language Models for Cost-Efficient Enterprise Automation and Decision Intelligence. International Journal of Research Publications in Engineering, Technology and Management (IJRPETM), 8(6), 13345-13357.
6. Monge, D. A., Pacini, E., Mateos, C., & others. (2020). CMI: An online multi-objective genetic autoscaler for scientific and engineering workflows in cloud infrastructures with unreliable virtual machines. Journal of Network and Computer Applications, 165, 102464.
7. Ni, X., Li, J., Yu, M., & others. (2020). Generalizable resource allocation in stream processing via deep reinforcement learning. Proceedings of the AAAI Conference on Artificial Intelligence, 34(01), 857–864.
8. Pandiyan, S., Lawrence, T. S., Sathiyamoorthi, V., & others. (2020). A performance-aware dynamic scheduling algorithm for cloud-based IoT applications. Computer Communications, 160, 512–520.

9. Ragmani, A., Elomri, A., Abghour, N., Moussaid, K., & others. (2020). FACO: A hybrid fuzzy ant colony optimization algorithm for virtual machine scheduling in high-performance cloud computing. Journal of Ambient Intelligence and Humanized Computing, 11(10), 3975–3987.
10. Ren, H., Wang, Y., Xu, C., & others. (2020). Smig-rl: An evolutionary migration framework for cloud services based on deep reinforcement learning. ACM Transactions on Internet Technology, 20(4), 1–18.
11. Rodrigues, T. K., Suto, K., Nishiyama, H., Liu, J., & Kato, N. (2020). Machine learning meets computation and communication control in evolving edge and cloud: Challenges and future perspective. IEEE Communications Surveys & Tutorials, 22(1), 38–67.
12. Loganathan, R. (2025). AGENTIC AI FRAMEWORKS FOR AUTONOMOUS RISK DETECTION AND COMPLIANCE REMEDIATION IN ENTERPRISE DATA CENTER OPERATIONS. Lex Localis-Journal of Local Self-Government, 23 (S6), 9672–9697.
13. Sardaraz, M., & Tahir, M. (2020). A parallel multi-objective genetic algorithm for scheduling scientific workflows in cloud computing. International Journal of Distributed Sensor Networks, 16(8), 1550147720949142.
14. Tong, Z., Chen, H., Deng, X., & others. (2020). A scheduling scheme in the cloud computing environment using deep Q-learning. Information Sciences, 512, 1170–1191.
15. Wan, B., Dang, J., Li, Z., & others. (2020). Modeling analysis and cost-performance ratio optimization of virtual machine scheduling in cloud computing. IEEE Transactions on Parallel and Distributed Systems, 31(7), 1518–1532.
16. Wang, H., Liu, N., Zhang, Y., & others. (2020). Deep reinforcement learning: A survey. Frontiers of Information Technology & Electronic Engineering, 21(12), 1726–1744.
17. Welsh, T., & Benkhelifa, E. (2020). On resilience in cloud computing: A survey of techniques across the cloud domain. ACM Computing Surveys, 53(3), 1–36.
18. Kim, M.-H., Lee, J.-Y., Shah, S. A. R., Kim, T.-H., & Noh, S.-Y. (2021). Min-max exclusive virtual machine placement in cloud computing for scientific data environment. Journal of Cloud Computing, 10, 2.
19. Mahil, M., & Jayasree. (2021). Combined particle swarm optimization and ant colony system for energy efficient cloud data centers. Concurrency and Computation: Practice and Experience.

20. Peng, Y., Bao, Y., Chen, Y., & others. (2021). DL2: A deep learning-driven scheduler for deep learning clusters. IEEE Transactions on Parallel and Distributed Systems, 32(8), 1947–1960.
21. Reddy, V. A. R. (2023). Orchestrating the Future Autonomous Healthcare Data Pipeline Management through Agentic AI Architectures. International Journal of Advanced Research in Computer Science & Technology (IJARCST), 6(2), 7979-7992.
22. Wang, J., Hu, J., Min, G., & others. (2021). Fast adaptive task offloading in edge computing based on meta reinforcement learning. IEEE Transactions on Parallel and Distributed Systems, 32(1), 242–253.
23. Zhou, G., Tian, W., Buyya, R., Xue, R., & Song, L. (2024). Deep reinforcement learning-based methods for resource scheduling in cloud computing: A review and future directions. Artificial Intelligence Review, 57, 124.
24. Chen, X., Zhu, F., Chen, Z., & others. (2022). Resource allocation for cloud-based software services using prediction-enabled feedback control with reinforcement learning. IEEE Transactions on Cloud Computing, 10(2), 1117–1129.
25. Chen, X., Yang, L., Chen, Z., & others. (2023). Resource allocation with workload-time windows for cloud-based software services: A deep reinforcement learning approach. IEEE Transactions on Cloud Computing, 11(2), 1871–1885.
26. Chen, G., Qi, J., Sun, Y., & others. (2023). A collaborative scheduling method for cloud computing heterogeneous workflows based on deep reinforcement learning. Future Generation Computer Systems, 141, 284–297.
27. Karat, C., & Senthilkumar, R. (2022). Optimal resource allocation with deep reinforcement learning and greedy adaptive firefly algorithm in cloud computing. Concurrency and Computation: Practice and Experience, 34(4), e6657.
28. Uma, J. (2022). Optimized intellectual resource scheduling using deep reinforcement Q-learning in cloud computing. Transactions on Emerging Telecommunications Technologies.
29. Tuli, S., Ilager, S., Ramamohanarao, K., & Buyya, R. (2022). Dynamic scheduling for stochastic edge-cloud computing environments using A3C learning and residual recurrent neural networks. IEEE Transactions on Mobile Computing, 21(3), 940–954.

30. Bilal, M., Canini, M., Fonseca, R., & Rodrigues, R. (2023). With great freedom comes great opportunity: Rethinking resource allocation for serverless functions. Proceedings of the 18th ACM European Conference on Computer Systems, 427–443.
31. Smendowski, M., & Nawrocki, P. (2024). Optimizing multi-time series forecasting for enhanced cloud resource utilization based on machine learning. Knowledge-Based Systems, 304, 112489.
32. Meda, R. (2025). Optimizing Quota Planning and Territory Management through Predictive Analytics: Segmenting Sales Reps and Accounts within National Sales Zones. Advances in Consumer Research, 2(4).
33. Pintye, I., Kovács, J., & Lovas, R. (2024). Enhancing machine learning-based autoscaling for cloud resource orchestration. Journal of Grid Computing.
34. Kayalvili, S., Senthilkumar, R., Yasotha, S., & Kamalakannan, R. S. (2025). An optimized resource allocation in cloud using prediction enabled reinforcement learning. Scientific Reports, 15, 36088.