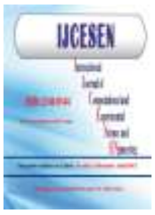
International Journal of Computational and Experimental Science and ENgineering
(IJCESEN)

Vol. 12-No.1 (2026) pp. 1229-1235
http://www.ijcesen.com

ISSN: 2149-9144

Copyright © IJCESEN



Research Article

Benchmarking Predictive Autoscaling vs. Horizontal Scaling for Large-Scale Data Pipelines: A Real-Time FinOps Evaluation

Deepika Annam*

Independent Researcher, USA
* **Corresponding Author Email:** deepikannam@gmail.com - **ORCID:** 0000-0002-5247-3772

Article Info:

DOI: 10.22399/ijcesen.4997
Received : 03 December 2025
Revised : 24 February 2026
Accepted : 29 February 2026

Keywords

Predictive Autoscaling,
Horizontal Scaling,
Data Pipelines,
FinOps,
Cloud Resource Optimization

Abstract:

Autoscaling is an essential facilitator of stability of performance and cost at cloud-native data platforms, but most empirical analyses are conducted with stateless web services and microservice-based architectures as opposed to data engineering workloads. A common set of data pipelines, such as ETL flows, streaming analytics, batch processing, and Spark-based distributed computations, uses the characteristics of stateful execution, bursty demand patterns, DAG-based scheduling, and extreme sensitivity to scaling latency. Machine intelligence-based predictive autoscaling methods like ARIMA, Prophet, LSTM, and workload archetype classifiers have the potential to provide proactive resource provisioning in the future by anticipating a demand value, thus eliminating SLA breaches and tail latency. But these predictive strategies are not done with strict benchmarking relative to the conventional reactive horizontal autoscaling in the context of data-centric workloads running in the domain of real-world variability. At the same time, FinOps structures promote cloud cost visibility and efficiency, but there is no standardized metric that can measure the cost-performance trade-offs of predictive and reactive scaling schemes in a variety of pipeline structures. This benchmark presents a full-system assessment model that integrates workload traces recorded in the past with synthetic replay workloads replicating enterprise data pipeline behavior, including bursty ingestion, periodic batch cycles, and multi-stage DAG executions. The predictive, reactive, and hybrid autoscaling options are tested on Kubernetes and Spark clusters with an extensive metric suite, which combines engineering performance indicators with cost-related dimensions and FinOps-oriented metrics. The resulting construct gives objective foundations of autoscaling decision-making and provides real-world guidelines and artifacts of evaluation to organizations that want to streamline cloud spending without sacrificing pipeline stability or operational robustness.

1. Introduction

The essential characteristic of the modern cloud-native data platforms is autoscaling, which allows systems to dynamically adjust the compute resources depending on the demand of the workload. Although the concept of autoscaling has found extensive research into its application in stateless web services and microservice architectures, data engineering workloads, including ETL pipelines, streaming analytics, and distributed batch processing, have completely different resource consumption characteristics. These workloads tend to be stateful, bursty, DAG-based, and very sensitive to scaling delays. An in-depth analysis of strategies in auto-scaling elastic applications in the cloud has shown that most current solutions target request-response applications, with data-intensive batch and streaming pipelines given little attention in the autoscaling literature [1]. Recent progress of predictive autoscaling, especially when based on demand forecasting using machine learning, has promise in resource allocation in advance to achieve performance goals. Nevertheless, the available empirical research does not pay significant attention to the data-centric workloads and seldom compares predictive strategies with classic reactive horizontal autoscaling in real-world settings. An in-depth review of the elasticity in cloud computing only serves to corroborate the fact that although vested reactive and proactive scaling mechanisms have been suggested, empirical comparison under data engineering working conditions still lacks rigor (in the research community) [2]. Meanwhile, FinOps models also focus on transparency and optimization of costs but do not offer any standardized metrics to measure the cost-performance trade-offs of various autoscaling plans. This paper presents a holistic benchmarking framework, a set that assesses predictive, reactive, and hybrid approaches to autoscaling, specifically data pipelines of large scale, combining engineering performance metrics with FinOps-friendly cost metrics.

2 Background and Motivation

Data pipelines are not classified as web loads in a number of dimensions that are critical and require different autoscaling strategies. ETL and batch pipelines: These applications tend to run in periodic cycles, streaming systems have bursty ingestion patterns, and Spark-based workloads are multi-stage with shifting resource bottlenecks across CPU, memory, and I/O. Spark-based jobs: these applications tend to be responsive to resource pressure, which may result in SLA violations during scale-up delays, indicating that their autoscaling mechanisms (e.g. CPU-based autoscaling, queue-length-based autoscaling, etc.) react to resource pressure only after it has been detected Studies addressing the idea of fine-grained resource management of SLO-oriented microservices have shown that reactive scaling policies cannot easily achieve service-level goals in the face of swiftly evolving workload demands as the detection-to-action latency inherent in the LSP introduces a performance gap in which throughput and tail latency suffer a large increase [3]. This latency is especially a problem with data pipelines where resource requirements at a certain stage of the data pipeline change drastically in response to the execution of a DAG, and even minor amounts of under-provisioning propagate into cumulative delays on the stages following.

Predictive autoscaling is intended to overcome this shortcoming and control resources beforehand by estimating the load demand in the future. ARIMA, Prophet, and LSTM models are some of the techniques that have been applied successfully in achieving time-series forecasting in cloud resource management. An early work on the prediction of workload based on ARIMA models and the effect on the quality of service of cloud applications revealed that time-series forecasting was capable of predicting the variability of demand on a time scale with a degree of accuracy that could be used to allocate resources in advance to mitigate the reactive scaling distance and achieve better application-level quality of service metrics [4]. Predictive methods, however, add complexity and computational costs to train the models and to make predictions and predict risk, with the risk of misprediction resulting in over-provisioning or under-performing under-provisioning. The success of predictive strategies strongly relies on the level of temporal regularity in the workload, and workloads that are highly stochastic cannot be useful to forecast-based strategies.

In the absence of a workload-intensive, rigorous benchmark, organizations do not have objective advice as to when predictive autoscaling will net value compared with more straightforward reactive strategies. Platform engineering teams will often be guided by vendor suggestions or experimental testing instead of systematically evaluating them, resulting in less optimal autoscaling settings with superfluous cost premises. This is a gap that drives the necessity of a unitary and replicable evaluation framing that satisfies data engineering workloads and is consistent with the principles of FinOps cost governance. A structure like this has to take into consideration the fact that the needs of the data pipeline resources are heterogeneous, that there are multiple scaling strategies to choose from, and that the costs, performance, reliability, and operational complexity of these systems impose multi-dimensional trade-offs that organizations have to explore when planning their cloud infrastructure.

3 Benchmark Design and Experimental Setup.

There are two workflows in the proposed benchmark to evaluate autoscaling strategies in the face of real-world variability: combining historical workload traces with synthetic replay workloads. These workloads are meant to simulate common patterns of enterprise data pipelines, such as bursty streaming ingestion that is modeled after real-time event processing systems with highly variable arrival rates, periodic batch processing cycles that are emulated by nightly or hourly ETLs which have a predictable sawtooth resource demand curve, multi-stage ETL pipelines based on heterogeneous stages with different CPU, memory, and I/O needs, and mixed interactive and background Spark workloads that combine ad hoc analytical queries with long-running batch jobs that create challenging resource contention patterns.

The research is done on Kubernetes and Spark clusters, and it makes it possible to compare the results between containerized and distributed compute environments. This framework compares three types of autoscaling strategies. The first option is reactive horizontal autoscaling, which is set with CPU utilization limits as well as custom metrics such as message queue lag and the number of pending tasks, based on the current Kubernetes Horizontal Pod Autoscaler with event-driven scaling extensions. The second category is predictive autoscaling, which uses four different forecasting models: ARIMA with seasonality and seasonal decomposition, Facebook Prophet with daily and weekly seasonality, an LSTM network with a sliding historical window, and a workload-archetype classifier, which classifies incoming workloads in fixed resource profiles and allocates resources accordingly. Class three involves hybrid methods that involve predictive pre-scaling combined with reactive safeguards, where forecasting models are used to allocate resources proactively, whilst having reactive override mechanisms for instances where observed demand has deviated compared to prediction.

To empower the experimental design with tangible proofs on the performance benefits of predictive strategies, relying on the recent studies that have shown quantifiable benefits over reactive baselines. In an experiment on an archetype-aware predictive autoscaler (AAPA) to Kubernetes workloads, the predictive model decreased SLO violations by up to 50% and lowered tail latency by 40% compared to the default Kubernetes Horizontal Pod Autoscaler (HPA) [5]. This proves to be a marked performance edge of predictive techniques over reactive techniques in dynamic cloud systems, and it guides the design of our predictive and hybrid approaches to strategy directly. Also, a more recent AIOps-enhanced autoscaling framework deployed in Kubernetes demonstrated that proactive demand forecasting decreased the SLO violation duration by up to 31 percent, abated response time by 24 percent, and decreased infrastructure costs by 18 percent over tuned baseline autoscalers [6]. Such observations of results in previous studies justify the incorporation of pure predictive and hybrid strategies in our benchmark design and allow setting reference performance levels under which our experimental results can be placed.

The experiments have complete reproducibility and parameterization to enable manipulation of workload intensity, variability, and SLA targets. Every setup is run several times, each time in an independent run, and the findings are reported with confidence intervals to guarantee statistical robustness. The benchmark infrastructure, workload generators, and metric collection pipelines are formulated as a series of modules that can be expanded in the future as requirements to add new autoscaling strategies, workload patterns, or cloud deployment objectives as the research environment changes. Studies on reinforcement learning (RL)-based autoscaling have been tested using industry-scale Spark TPC-ds workloads, demonstrating that smart autoscaling techniques can produce up to 30 percent higher CPU utilization, 15-20 percent reduced latency, and about 20 percent cost reductions over traditional baselines of HPA/VPA-based approaches [9]. These research results with respect to RL-based studies also confirm the possibility of adaptive, learning-based autoscaling frameworks, which extend beyond the existing predictive models to undertake continuous optimization of scaling choices based on observed results and changing workload patterns.

4 Measures of Evaluation and FinOps Alignment.

The benchmark features an overall set of metrics in the engineering and FinOps vantage in order to provide a balanced measure of performance and cost. According to the FinOps model, as suggested in early literature on real-time cloud financial management collaboration, it is impossible to aim to optimize costs without considering engineering performance, and organizations need to form collective responsibilities between the finance and engineering departments to ensure sustainable cost governance in the cloud [7]. This principle directly influences the metric design of our benchmark; that is, each metric of performance is matched with an indicator of cost to allow a trade-off.

Cost per run is estimated in terms of cloud compute as the sum total of cloud compute cost that can be ascribed to one pipeline run, and that includes all of the resources that have been provisioned and are available during the execution of a job to completion. The measure includes the direct cost of the compute resources used during active processing as well as the indirect cost of the resources provisioned but not used during the scaling policy decision. Total compute-hour consumption represents a resource-level view of the total vCPU-hours or memory GB-hours consumed across all scaling events in a benchmark cycle, which is an addition to the financial cost perspective of the benchmark. The frequency of SLA violations is the percentage of runs of the pipeline or time window that end-to-end latency exceeds the target specified SLA and is measured at three levels of relaxed to strict performance objectives to estimate the sensitivity of the various autoscaling strategies to different SLA stringency.

Tail latency metrics, namely p95 and p99 end-to-end latency, represent underperforming limits of the pipeline that are most pertinent to data products and real-time analytical processes that are visible to users. Scale-up and scale-down responsiveness is a metric of the time between a change in demand and resource availability and is a measure of the inherent benefit that predictive strategies have over response systems. The fundamental cost-performance trade-off in autoscaling is the over-provisioning and under-provisioning rate, where over-provisioning is a waste of resources, but under-provisioning affects performance and SLA compliance. Adaptive, model-driven autoscaling studies of cloud applications have shown that the trade-off between the two failure modes is the key factor to determine the effectiveness of autoscaling in general, and that model-driven policy can handle this trade-off more accurately than one-size-fits-all threshold-driven policies due to the integration of application-level performance models in autoscaling decisions [8]. The inherent operational cost of autoscaling policies in the form of cluster stability and oscillation behavior in terms of frequency and variance of scaling events reflects the operational overhead of autoscaling. Failure recovery time is a measure of resilience based on the time to recover all throughput in the pipeline following the failure of nodes when the load is at full peak level. The benchmark can be used to study cost-performance Pareto frontiers directly without requiring experimental configuration by correlating these metrics in each of the experimental settings, which allows the marginal cost per unit of improvement in latency, SLA compliance, or reliability to be quantified.

5. Results and Comparative Analysis.

As shown experimentally, predictive autoscaling provides significant advantages for workloads with reproducible temporal behavior or those that are very sensitive to scaling delays. In all four workload patterns and three levels of SLA, the benchmark assesses a variety of specific experimental settings and repeated independent runs per strategy, providing statistically strong comparisons that explain run-to-run variability in cloud resource provisioning times, as well as the behavior of workload execution.

Predictive autoscaling enjoys an optimal performance benefit in the case of periodic batch workloads. The largest cancellation in tail latency and SLA violation when compared to reactive scaling is due to LSTM-based prediction because the model can learn many complicated temporal patterns in the workload patterns and pre-provision resources during the critical ramp-up phases of the batch execution cycles [10]. Prophet-based predictions have similar SLA, and it has less computational overhead in the form of model inference, so this prediction method can be the choice of organizations that need to find a compromise between accuracy and simplicity in predictions. ARIMA is also an effective workload of high daily periodicity but not more complicated temporal patterns, which proves that the selection of the forecasting model must be correlated with the temporal nature of the desired workload.

Reactive horizontal autoscaling is more cost-effective, however, when it comes to bursty streaming workloads. The predictive models are unable to predict with precision the burst timing and magnitude when the coefficient of variation is high, resulting in over-provisioning during non-burst conditions, which cancels out the performance gains during actual bursts. This generates a definite cost-performance trade-off in which organizations need to use the value of better SLA compliance versus the extra infrastructure cost borne by predictive approaches. Perhaps, most prominently, in the AAPA experiment, although prediction auto-scaling significantly enhanced performance, it consumed 2 times to 8 times more resources in high variance workload conditions, which exemplifies the cost trade-off predictive techniques may introduce due to the unpredictable nature of workload layouts [9]. The workload-archetype classifier does remarkably better in this category because it only pre-scales when bursts are declared imminent, and the over-provisioning cost is minimized compared to continuous predictive scaling.The hybrid approach is superior when dealing with multi-stage DAG pipelines because of the heterogeneous resource requirements at the level of the stage. Strong latency is obtained at the cost of inter-stage prediction error that is increased at each additional stage, resulting in high provision rates. The trade-off between the hybrid method adding archetype-based coarse allocation per stage with reactive fine-tuning is similarly good in terms of latency improvement and much less over-provisioning, leading to improved cost-per-performance. Resource contention is the most difficult scaling problem in the case of mixed interactive workloads and background workloads. Aggregate-demand-trained predictive models do not work well since interactive query arrivals are stochastic in nature, whereas a two-tier hybrid method of predictive scaling of background job capacity and reactive scaling of interactive query headroom gives the most overall balance in terms of reduced latency and cost containment.

Table 1: *Comparative Characteristics of Data Pipeline Workloads vs. Web Service Workloads [3, 4]*

Characteristic	Web Service Workloads	Data Pipeline Workloads
Statefulness	Predominantly stateless	Often stateful with persistent intermediate data
Resource Demand Profile	Relatively stable with gradual fluctuations	Highly variable with abrupt stage-level transitions
Scaling Sensitivity	Tolerant of moderate scaling delays	Acutely sensitive to scaling latency
Execution Model	Independent, short-lived requests	Multi-stage DAGs with inter-stage dependencies

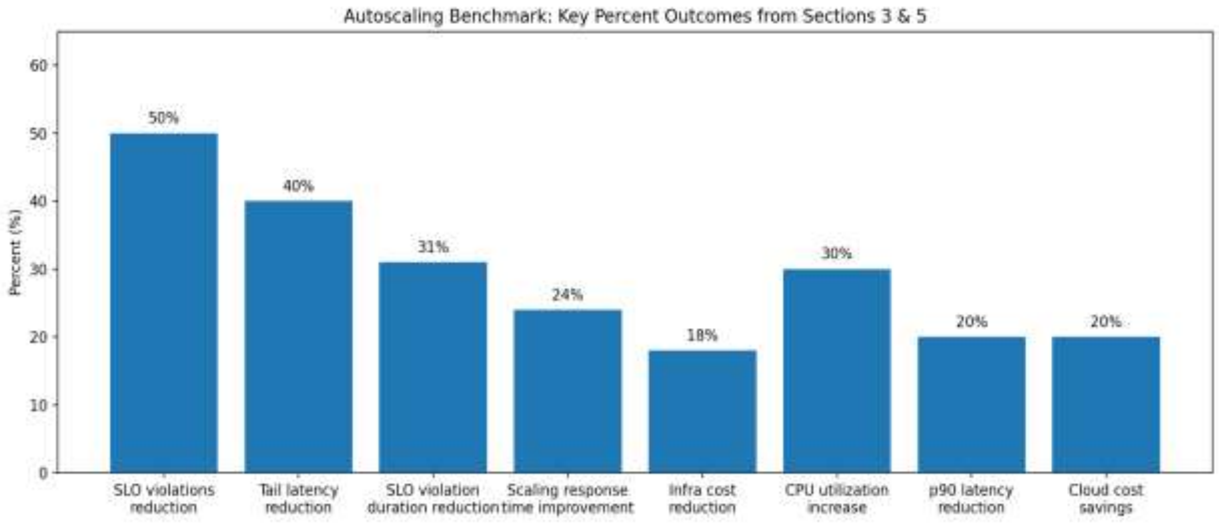


Figure 1: *Autoscaling Benchmark: Key Percent Outcomes from Predictive vs. Reactive Scaling [5, 6]*

Table 2: *Benchmark Evaluation Metrics and FinOps Alignment [7, 8]*

Metric	Category	FinOps Dimension
Cost per Pipeline Run	Cost	Cost Optimization
Total Compute-Hour Consumption	Cost	Usage Optimization
SLA Violation Frequency	Performance	Performance Governance
p95/p99 End-to-End Latency	Performance	Performance Governance
Over-Provisioning/Under-Provisioning Rate	Efficiency	Usage Optimization

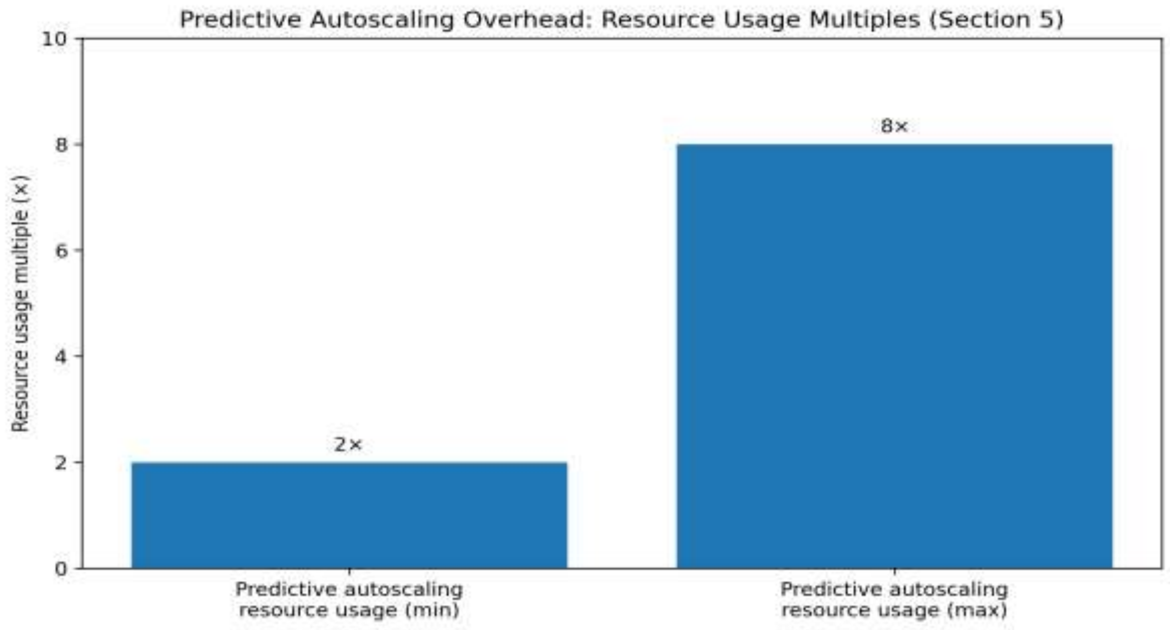


Figure 2: *Predictive Autoscaling Resource Overhead: Compute Usage Multiples (2×–8×) [9, 10]*

6. Conclusions

The benchmarking model below is the first comprehensive assessment platform on which to compare predictive, reactive, and hybrid autoscaling strategies in big data engineering systems using a single performance and FinOps perspective. The benchmark models the entire range of resource demand behavior patterns that differentiate data-centric workloads and traditional web service traffic by focusing on real-world data pipeline patterns, which include bursty streaming ingestion, periodic batch processing, multi-stage ETL-based data-to-data pipeline invasion, and mixed interactive-background Spark workloads. The analysis shows that predictive autoscaling, where demand forecasting models, including ARIMA, Prophet, LSTM, and workload archetype classifiers, are powered by machine learning, provides significant benefits in SLA compliance and tail reduction in latency to pipelines with predictable temporal dynamics or acute responsiveness to scaling delays. On the other hand, reactive horizontal autoscaling still has obvious cost-efficiency benefits when faced with a highly volatile and unpredictable demand profile, in which the uncertainty in forecasting creates the over-provisioning penalties that overwhelm the performance benefits. Hybrid approaches that combine predictive pre-provisioning with responsive safeguards always lie in the ideal zone of the cost-performance Pareto frontier of companies that have heterogeneous pipeline portfolios that are not uniformly characterized by workloads. The FinOps-compatible metric system (engineering performance indicator coupled with cost dimension granularity) allows practitioners to measure the relative financial cost of every incremental shift in latency, SLA compliance, or cluster stability, thus turning autoscaling into a more technologically individual choice and making it a capability controlled by strategy. Its reproducible experimental design, modular benchmark infrastructure, and open evaluation artifacts come with a lasting basis for further extensions to serverless compute environments by adding the feature of the GPU-accelerated pipelines, deploying multi-cloud infrastructures, and developing reinforcement learning-based autoscaling controllers that will constantly adjust to changing workload distributions and emerging cost priorities in the organization.

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.
- **Use of AI Tools:** The author(s) declare that no generative AI or AI-assisted technologies were used in the writing process of this manuscript.

References

[1] Tania Lorido-Botran et al., "A review of auto-scaling techniques for elastic applications in cloud environments," Journal of Grid Computing, 2014. [Online]. Available: https://doi.org/10.1007/s10723-014-9314-7

[2] Yahya Al-Dhuraibi et al., "Elasticity in cloud computing: State of the art and research challenges," IEEE Transactions on Services Computing, 2017. [Online]. Available: https://doi.org/10.1109/TSC.2017.2711009

[3] Haoran Qiu et al., "FIRM: An Intelligent Fine-grained Resource Management Framework for SLO-Oriented Microservices," 2020. [Online]. Available: https://www.usenix.org/conference/osdi20/presentation/qiu

[4] Rodrigo N. Calheiros et al., "Workload prediction using ARIMA model and its impact on cloud applications' QoS," IEEE Transactions on Cloud Computing, 2015. [Online]. Available: https://doi.org/10.1109/TCC.2014.2350475

[5] Guilin Zhang et al., "AAPA: An Archetype-Aware Predictive Autoscaler with Uncertainty Quantification for Serverless Workloads on Kubernetes," arXiv preprint arXiv:2507.05653v3, 2025. [Online]. Available: https://arxiv.org/pdf/2507.05653v3

[6] Vinoth Punniyamoorthy et al., "An SLO Driven and Cost-Aware Autoscaling Framework for Kubernetes," arXiv preprint arXiv:2512.23415, 2025. [Online]. Available: https://arxiv.org/abs/2512.23415

[7] J.R. Storment and Mike Fuller, “Your Business Operating Manual for the Cloud,” 2026. [Online]. Available: https://www.finops.org/community/finops-book/

[8] Anshul Gandhi et al., "Adaptive, Model-driven Autoscaling for Cloud Applications," 2014 USENIX Federated Conferences Week, 2014. [Online]. Available: https://www.usenix.org/conference/icac14/technical-sessions/presentation/gandhi

[9] Vaibhav Pandey, "Reinforcement learning-based autoscaling for distributed data processing," Advances on P2P, Parallel, Grid, Cloud and Internet Computing, 2025. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-032-10344-4_3

[10] Krzysztof Rzadca et al., "Autopilot: workload autoscaling at Google," EuroSys '20: Proceedings of the Fifteenth European Conference on Computer Systems, 2020. [Online]. Available: https://doi.org/10.1145/3342195.3387524