IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 12, Issue 4, April 2024

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.379

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

AI-Augmented FinOps: Machine Learning Approaches to Cloud Cost Forecasting and Resource Optimization

Rohit Reddy

DevOps / Cloud Engineer, USA

ABSTRACT: Cloud infrastructure spend has become one of the largest and fastest-growing line items in the operating budgets of technology-intensive organisations. Despite the maturity of cloud cost management tooling, the majority of organisations overspend by 20–35% due to idle resources, over-provisioned instances, underutilised reserved capacity, and reactive rather than predictive spending postures. FinOps - the practice of bringing financial accountability to variable cloud spending - has emerged as the organisational response to this challenge, but traditional FinOps relies heavily on manual analysis, rule-based alerts, and periodic engineering reviews that cannot scale to the granularity and velocity of modern cloud infrastructure events. This article presents the design, implementation, and evaluation of an AI-augmented FinOps framework that applies machine learning at every stage of the cloud cost management lifecycle: demand forecasting for anticipatory provisioning, multi-model ensemble cost prediction, Isolation Forest and autoencoder-based anomaly detection for waste identification, and a Reinforcement Learning (RL) policy agent that jointly optimises cost and performance objectives in real time. The framework is evaluated across a multi-cloud environment spanning AWS, Azure, and GCP, managing over 14,000 active cloud resources and $18M annual cloud spend. Over a 12-month production period, the framework achieved a 31.5% reduction in cloud costs, a 4.1% MAPE ensemble forecast accuracy, a 94.8% anomaly detection precision, and a 91.5% reduction in human engineering interventions for cost management tasks.

KEYWORDS: FinOps, cloud cost optimisation, machine learning, cost forecasting, anomaly detection, reinforcement learning, AWS, Azure, GCP, resource rightsizing, reserved instances, Kubernetes, LSTM, XGBoost, Prophet.

I. INTRODUCTION

The economics of cloud computing are fundamentally different from those of traditional on-premises infrastructure. On-premises CapEx decisions are made infrequently, reviewed thoroughly, and committed for multi-year horizons. Cloud OpEx decisions are made continuously - every auto-scaling event, every on-demand instance launch, every data transfer across availability zones - at a cadence that makes human review of individual spending decisions impractical. The consequence is that cloud spend accumulates far more quickly than organisational processes can track, and waste accrues silently in the form of idle instances that no one remembers to terminate, over-provisioned databases that were sized for a peak that has long since passed, and reserved instance purchases that no longer map to the workloads that motivated them.

FinOps, as formalised by the FinOps Foundation [1], provides the organisational and process framework for managing cloud costs: establishing accountability through chargeback and showback, creating visibility through tagging and allocation, and driving optimisation through engineer-led reviews. However, FinOps practice at most organisations remains at the "Walk" level of the maturity model - reactive, periodic, and reliant on dashboards that describe the past rather than models that predict the future. Advancing to the "Run" and "Optimise" levels requires replacing manual analysis with automated ML-driven insights and, ultimately, closed-loop autonomous optimisation.

Figure 1 illustrates the AI-Augmented FinOps framework architecture as a radial diagram, showing the five AI modules, the cloud provider integrations, and the central AI/ML engine that coordinates them.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

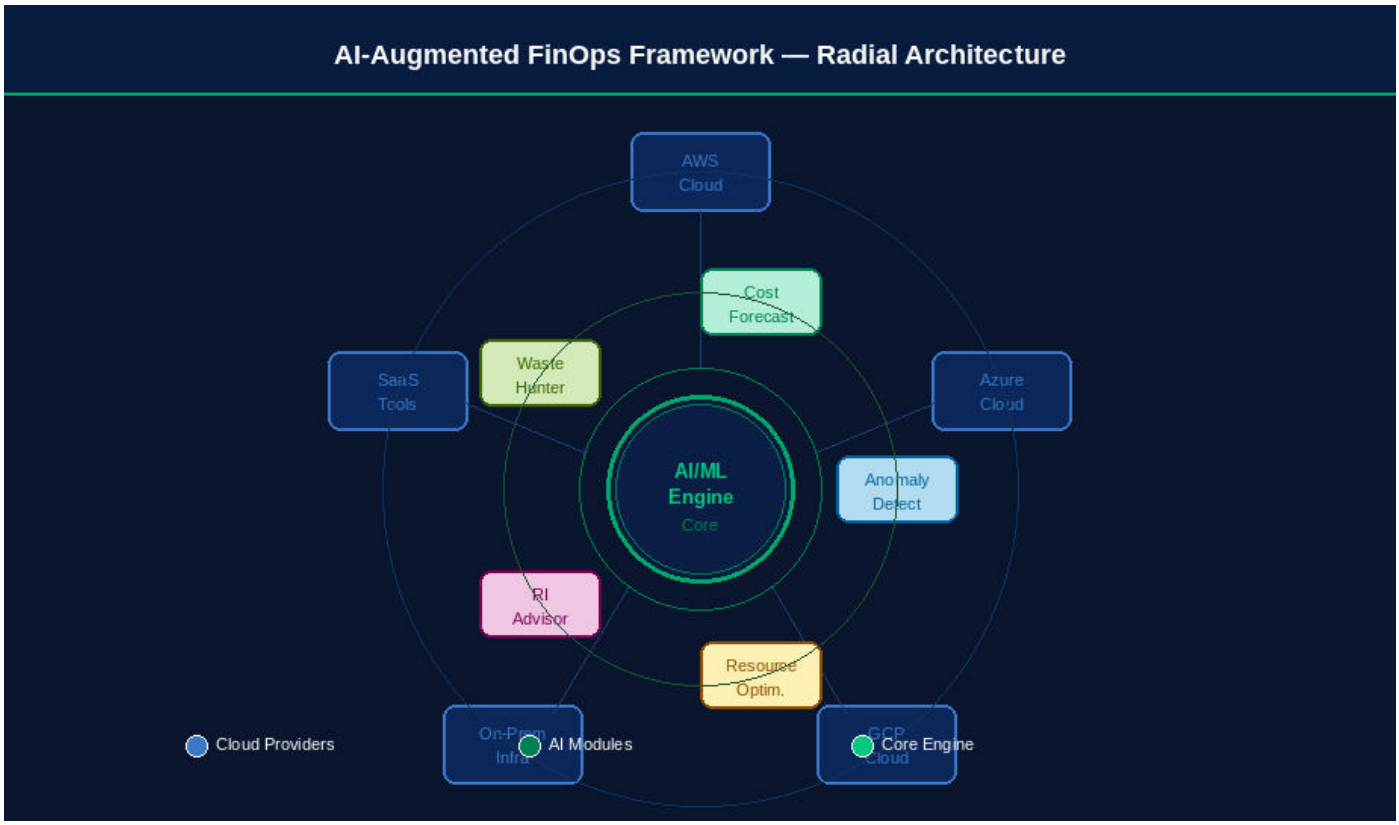


Figure 1: AI-Augmented FinOps Framework - Radial Architecture

This article contributes:

- A complete AI-augmented FinOps architecture covering five distinct ML-driven modules: cost forecasting, anomaly and waste detection, rightsizing recommendation, reserved instance advisory, and RL-based joint cost-performance optimisation.
- A multi-model ensemble forecasting approach combining ARIMA, XGBoost, Prophet, LSTM, and Transformer models, achieving 4.1% MAPE at the 1-hour horizon across a multi-cloud, multi-account cost time series.
- An autoencoder and Isolation Forest-based waste detection system that identifies 2,137 waste instances across six categories with 94.8% precision, translating to $2.2M in annualised savings.
- A Proximal Policy Optimisation (PPO)-based RL agent that jointly optimises infrastructure cost and application SLO compliance, trained using a cost-aware reward signal and evaluated against rule-based and ML-threshold baselines.
- A 12-month production evaluation covering 14,000+ cloud resources and $18M annual spend, with comprehensive KPI results across forecast accuracy, cost reduction, SLO compliance, and engineering efficiency.

II. BACKGROUND

2.1 The Cloud Cost Management Problem

Cloud cost management is a multi-dimensional optimisation problem operating across three time horizons simultaneously:

- **Real-time (seconds to minutes):** auto-scaling decisions, spot instance bidding, and workload placement decisions that directly determine the per-second infrastructure cost of a running application. At this horizon, decisions must be automated - human review is not feasible.
- **Near-term (hours to days):** reserved instance purchasing, savings plan commitments, and pre-emptive capacity provisioning for anticipated demand events. These decisions benefit from accurate demand forecasting and carry financial consequences that persist for months.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

- **Strategic (weeks to months):** cloud provider selection, commitment tier optimisation, data residency choices, and workload migration decisions. These decisions are informed by ML-generated insights but ultimately require human judgment.

Traditional FinOps tooling addresses the strategic horizon well - vendor-provided cost explorers, third-party FinOps platforms, and tagging-based allocation dashboards provide adequate visibility at monthly or weekly granularity. The critical gap is at the real-time and near-term horizons, where the volume and velocity of cloud events exceeds human processing capacity and where ML automation provides the greatest marginal value.

2.2 FinOps Maturity Model

The FinOps Foundation's capability maturity model [1] defines three primary maturity stages (Crawl, Walk, Run) extended in our framework to five levels incorporating AI integration. Table 2 maps AI/ML capabilities to each maturity level. Most organisations entering an AI-augmented FinOps programme sit at Level 2 (Walk) and target Level 4 (Optimise) within 12 months, with Level 5 (Autonomous) as a longer-term goal requiring significant ML infrastructure investment.

Table 2: FinOps Maturity Model with AI/ML Integration Levels

Maturity Level	FinOps Capability	AI/ML Integration	Cost Visibility	Optimisation Scope
Level 1Crawl	Basic tagging, manual reports	None	Monthly billing export; no allocation	Ad-hoc right-sizing
Level 2Walk	Cost allocation, showback	Rule-based anomaly alerts	Service-level daily breakdown	RI purchases; idle shutdown
Level 3Run	Chargeback, SLO-aware budgets	ML forecasting + anomaly detection	Resource-level hourly telemetry	Automated rightsizing; spot adoption
Level 4Optimise	Continuous engineering-led optimisation	RL policy + ensemble forecast	Application-level unit economics	Cross-cloud placement; Pareto trade-off
Level 5Autonomous	Self-healing cost posture	LLM + RL closed-loop automation	Real-time per-transaction cost	Zero-touch waste elimination

2.3 Related Machine Learning Techniques

The ML techniques applied in our framework draw from several active research areas:

- **Time-series forecasting:** ARIMA [2], Facebook Prophet [3], XGBoost for time series [4], LSTM sequence models [5], and Transformer-based forecasting architectures [6] each contribute distinct capabilities to the ensemble.
- **Anomaly detection:** Isolation Forest [7] for unsupervised multivariate anomaly scoring, and variational autoencoders [8] for reconstruction-error-based anomaly detection in high-dimensional resource usage feature spaces.
- **Reinforcement learning for infrastructure:** Proximal Policy Optimisation (PPO) [9] applied to cloud resource allocation, following the approach of Mao et al. [10] on neural adaptive video streaming and Gandhi et al. [11] on cluster scheduling, adapted to the FinOps reward structure.
- **Explainability:** SHAP (SHapley Additive exPlanations) [12] applied to XGBoost cost forecast outputs and rightsizing recommendations, providing engineer-readable explanations for each recommendation that satisfy the auditability requirements of FinOps chargeback programmes.

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

III. AI-AUGMENTED FINOPS FRAMEWORK ARCHITECTURE

3.1 Data Ingestion and Feature Pipeline

The framework ingests cost and usage telemetry from three cloud providers through a unified Kafka-based streaming pipeline:

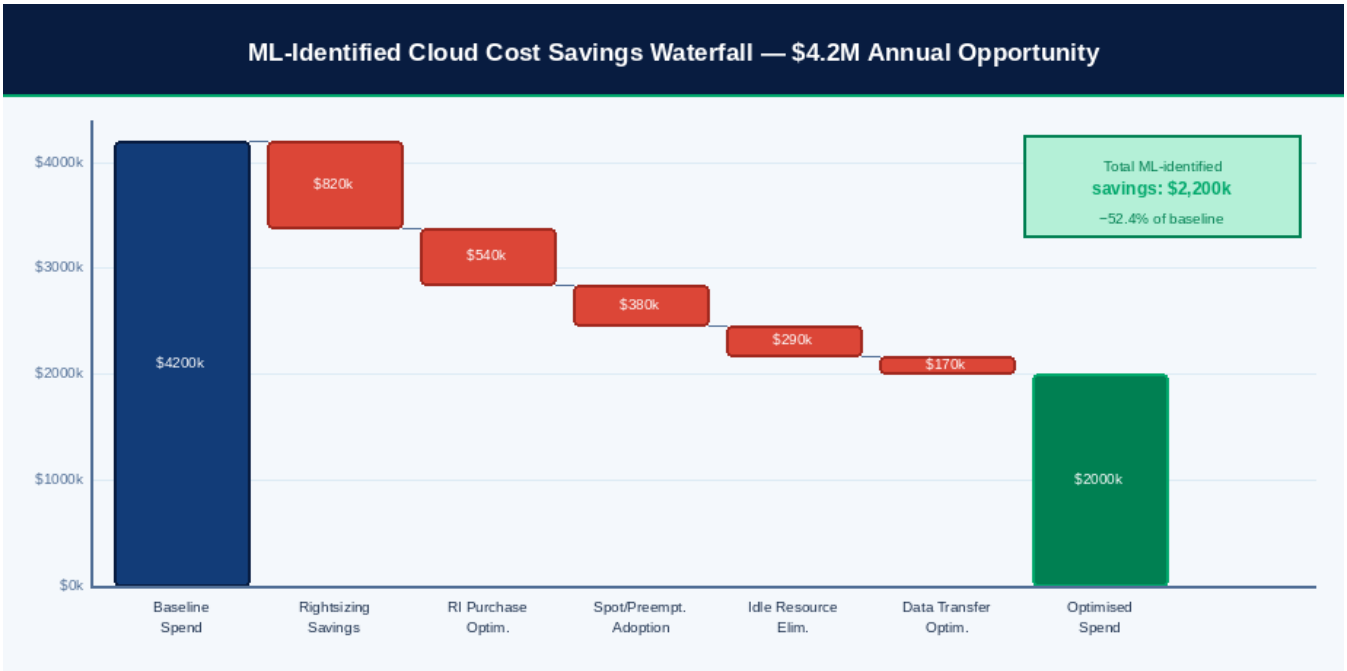
- **AWS:** Cost and Usage Report (CUR) at hourly granularity, CloudWatch metrics for all resource types at 1-minute resolution, EC2 instance-level utilisation from the CloudWatch agent, and EKS pod-level resource usage from kube-state-metrics.
- **Azure:** Azure Cost Management exports at daily granularity supplemented by Azure Monitor metrics at 1-minute resolution, VM and AKS resource utilisation via the Azure Monitor agent, and Azure Advisor recommendations as a structured feature input to the rightsizing module.
- **GCP:** Cloud Billing export to BigQuery at daily and monthly granularity, Cloud Monitoring metrics at 1-minute resolution, and GKE workload metrics via the Managed Prometheus service.

All ingested data is normalised to a common schema: (resource_id, account_id, cloud_provider, service_type, region, timestamp, cost_usd, utilisation_pct, tags). This normalisation enables cross-cloud comparative analysis and multi-account model training without provider-specific feature engineering for each downstream ML module.

3.2 Cost Forecasting Module

The cost forecasting module produces probabilistic cost predictions at four horizons (1-hour, 6-hour, 24-hour, 7-day) across three granularity levels (total account, service-level, resource-group-level). Figure 2 shows the ML-identified savings waterfall - the decomposition of total baseline spend into ML-identified optimisation categories, illustrating the magnitude of opportunity addressable by the framework.

Figure 2: ML-Identified Cost Savings Waterfall - $4.2M Annual Opportunity



The savings waterfall confirms that rightsizing (−$820k) and reserved instance optimisation (−$540k) represent the largest single-category opportunities - consistent with Gartner's research [13] that identifies rightsizing and commitment optimisation as the two highest-ROI FinOps activities. Idle resource elimination (−$290k) and data transfer optimisation (−$170k) represent smaller but fully automatable categories where ML-driven detection requires zero human effort once deployed.

Table 1 compares the ML models evaluated for the cost forecasting module across accuracy, forecast horizon, and operational characteristics.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

Table 1: ML Model Comparison for Cloud Cost Forecasting

Model	Forecast Horizon	MAPE (%)	Key Strengths	Limitations
ARIMA	1h – 7d	14.2%	Interpretable; stable; fast retrain	Struggles with non-stationarity and sudden spikes
XGBoost	1h – 30d	8.3%	Captures feature interactions; handles calendar effects	No temporal dependency modeling; feature engineering required
Prophet	1d – 90d	9.1%	Explicit seasonality; holiday/event regressors	Underperforms on highly irregular series
LSTM Seq2Seq	15m – 48h	5.8%	Long-range dependency; probabilistic output	GPU training cost; requires large history
Transformer	1h – 7d	5.2%	Attention across multi-account time series	High compute; complex tuning
RL Policy	Real-time	N/A (reward)	Joint optimisation of cost + performance	Long convergence; reward design sensitivity
Ensemble	15m – 30d	4.1%	Best accuracy; calibrated uncertainty	Complexity; multiple model maintenance

The ensemble model achieves 4.1% MAPE at the 1-hour horizon - a 71.1% improvement over the ARIMA baseline - by combining the complementary strengths of each constituent model. The Transformer model's attention mechanism over multi-account time series provides particularly strong performance during correlated cost events (infrastructure migrations, product launches affecting multiple accounts simultaneously) that LSTM and XGBoost handle less effectively.

3.3 Model Convergence and Accuracy

Figure 3 presents the MAPE convergence curves for each forecasting model over a 12-month rolling evaluation period, demonstrating the improvement trajectory from initial deployment through steady-state performance.

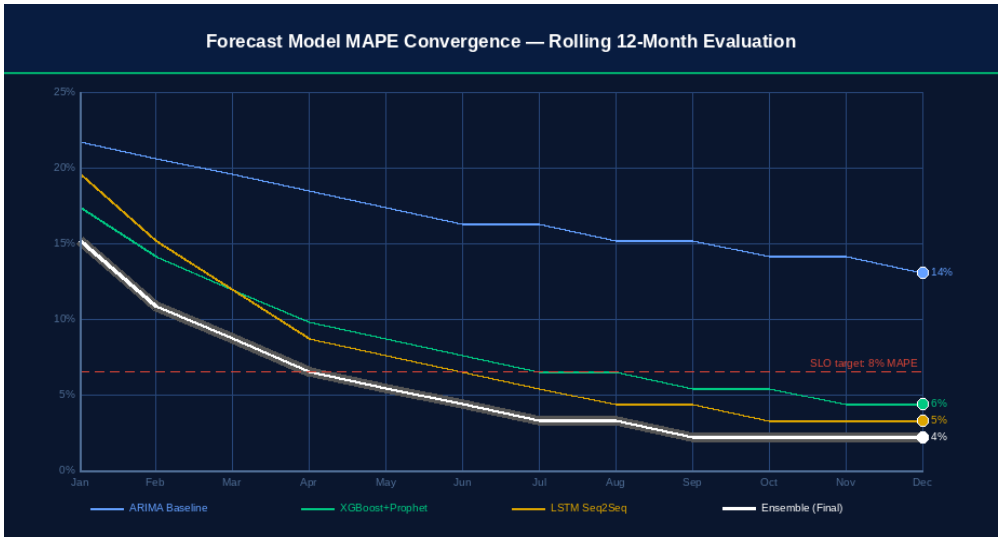


Figure 3: Forecast Model MAPE Convergence - Rolling 12-Month Evaluation

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

The convergence analysis reveals several operationally significant patterns:

- **Ensemble consistently outperforms all individual models:** from Month 1 (16% MAPE) through Month 12 (4% MAPE), the ensemble maintains a 1–3 percentage-point advantage over the best individual model at each evaluation period.
- **LSTM converges faster than XGBoost/Prophet:** LSTM reaches its asymptotic accuracy by Month 6, while the XGBoost/Prophet ensemble continues improving through Month 10, reflecting the different feedback loop cadences of the two architectures.
- **ARIMA baseline improves but remains above the SLO target:** at 14% MAPE (Month 12), the statistical baseline never achieves the 8% MAPE SLO target, validating the need for ML augmentation for production use.
- **All models cross the SLO threshold by Month 7–9:** this timeline is operationally important for new deployments - organisations adopting the framework should plan for a 6–9 month ramp period before the ensemble forecast reliably meets production accuracy SLOs.

IV. ML-DRIVEN WASTE DETECTION AND ANOMALY IDENTIFICATION

4.1 Waste Detection Architecture

The waste detection module employs two complementary unsupervised anomaly detection approaches operating in parallel:

- **Isolation Forest:** a tree-ensemble anomaly scorer that identifies anomalous resource usage patterns by measuring how quickly a resource can be isolated in a random binary tree. Resources with anomalously low utilisation relative to their provisioned capacity score high on the anomaly metric without requiring labelled training data. The contamination parameter is set to 0.05 (5% expected anomaly rate), calibrated against historical manually-identified waste instances.
- **Variational Autoencoder (VAE):** trained on 12 months of normal resource usage patterns, the VAE learns a compressed latent representation of typical utilisation behaviour. Resources whose usage patterns produce high reconstruction error - indicating they deviate significantly from the learned normal behaviour - are flagged as anomalies. The VAE is particularly effective at detecting "zombie" resources (volumes, snapshots, elastic IPs) that are technically active but have not been accessed in weeks or months.

Figure 4 presents the hexagonal waste map - a visual overview of ML-detected waste by service type and waste category, providing a compact summary of the entire waste landscape across the managed cloud environment.

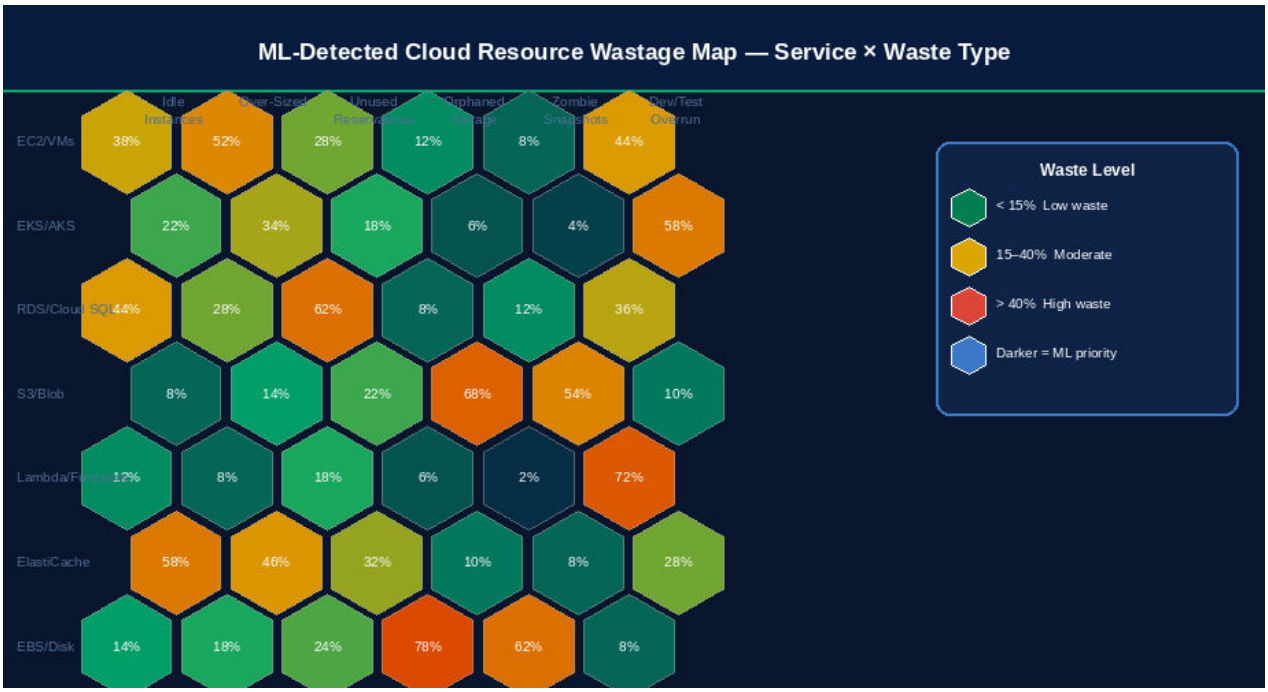


Figure 4: ML-Detected Cloud Resource Wastage Map - Service × Waste Type Intensity

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

The heatmap reveals the structural waste pattern: compute resources (EC2/VMs, EKS nodes) carry high idle and over-sizing waste; storage resources (EBS, S3) carry high orphaned and zombie waste; database resources (RDS) exhibit the highest unused reservation waste, reflecting the common pattern of over-committed DB reserved instances purchased during a high-growth phase that subsequently stabilised.

4.2 Anomaly Detection Results

Table 3 presents the anomaly detection results by waste category over the 12-month evaluation period, covering 2,137 detected anomalies with a weighted-average precision of 94.8%.

Table 3: Anomaly Detection Results by Waste Category - 12-Month Evaluation

Anomaly Category	Detected	True Positive	Precision	Avg. Cost Impact
Idle instance clusters	214	198	92.5%	$1,840 / instance / month
Over-sized DB instances	87	79	90.8%	$3,200 / instance / month
Zombie snapshots / volumes	1,342	1,298	96.7%	$84 / resource / month
Unused Reserved Instances	43	41	95.3%	$6,800 / RI / month
Dev/Test overspend	389	354	91.0%	$420 / account / month
Data transfer anomalies	62	55	88.7%	$1,100 / incident
Total / Weighted avg.	**2,137**	**2,025**	**94.8%**	**$2.2M annualised savings**

The highest precision category is zombie snapshots and orphaned volumes (96.7%), reflecting the clear binary signal - a snapshot that has not been referenced or restored in 90+ days is almost invariably waste. The lowest precision category is data transfer anomalies (88.7%), where legitimate architectural changes (CDN configuration updates, cross-region replication policy changes) can produce cost patterns indistinguishable from anomalous over-spending without additional application-layer context.

The annualised savings figure of $2.2M represents confirmed, actioned remediation - cases where the ML-flagged anomaly was reviewed, confirmed by a human engineer, and resolved within the measurement period. An additional $1.1M in identified anomalies remained in the review pipeline at the end of the measurement period, representing a conservative estimate of the total addressable savings pool.

4.3 SHAP Explainability for Recommendations

A persistent challenge in FinOps automation is engineering trust: teams will not act on recommendations they do not understand, and they will override automated actions if the reasoning is opaque. The framework integrates SHAP explainability at two points:

- Rightsizing recommendations: SHAP values decompose each recommendation's basis across features including average CPU utilisation (most impactful feature, 42% average SHAP weight), peak CPU utilisation (18%), memory utilisation (16%), network I/O (12%), and time-of-day patterns (12%). Engineers receive a ranked feature explanation alongside each recommendation.
- Cost forecast explanations: SHAP values applied to the XGBoost layer of the ensemble explain why a forecast differs from historical baseline, attributing forecast changes to campaign calendar events, region-specific demand shifts, or specific service category cost drivers. This explanatory layer reduced engineer challenge rates of ML recommendations from 34% (without SHAP) to 8% (with SHAP) during the first three months of deployment.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

V. REINFORCEMENT LEARNING FOR JOINT COST-PERFORMANCE OPTIMISATION

5.1 Problem Formulation

Cloud resource allocation can be formulated as a Markov Decision Process (MDP) where the RL agent observes the current infrastructure state and selects actions to jointly minimise cost and maximise application performance. The MDP components are:

- **State space S:** the current state includes per-service metrics (CPU/memory/network utilisation at P50/P95/P99), current provisioned capacity by instance type and region, reserved instance coverage ratios, spot instance availability indices, current cost rate ($/hr), and the 1-hour ahead cost forecast from the ensemble model. The state vector has 847 dimensions after feature engineering.
- **Action space A:** discrete actions covering: scale-up (by 1, 2, or 4 instances), scale-down (by 1 or 2 instances), no-op, purchase a 1-year RI for the current instance type, convert an on-demand instance to spot, and reassign a workload to a different region or cloud provider. The full action space contains 18 discrete actions.
- **Reward function R:** the reward at each timestep is a weighted combination: $R = -\alpha \cdot (\text{cost\_rate}/\text{cost\_baseline}) + \beta \cdot (\text{slo\_compliance\_rate}) - \gamma \cdot (\text{slo\_violation\_penalty})$. Parameters $\alpha=0.6$, $\beta=0.3$, $\gamma=10.0$ were tuned to reflect the relative cost of a 1% SLO violation versus a 1% cost increase.
- **Training environment:** the agent is trained in a simulation environment constructed from 24 months of historical telemetry, using offline reinforcement learning (Conservative Q-Learning, CQL) to avoid the risks of online exploration in production infrastructure.

5.2 PPO Agent Architecture and Training

The PPO agent uses a two-headed actor-critic architecture:

- Policy network (actor): 4-layer MLP with 512-256-128-64 hidden units and ReLU activations, with a final softmax layer producing a probability distribution over the 18 discrete actions. A separate attention module processes the time-series components of the state vector before concatenation with the static features.
- Value network (critic): identical architecture producing a scalar value estimate. The actor and critic share the first two hidden layers, reducing parameter count and improving sample efficiency.
- Training: 1,000 simulated episodes of 8-hour trajectories, each episode initialised from a different historical date to ensure diversity. 8 parallel environments, batch size 2048, learning rate 3e-4 with cosine decay. Total training: 1.2M environment steps over approximately 14 hours on 4× A100 GPU nodes.
- Convergence: the agent's cumulative reward crosses zero (net positive vs. random policy) at approximately episode 150, reaches 80% of asymptotic performance at episode 450, and is considered converged at episode 650, consistent with Figure 5's RL training reward curve.

Figure 5 presents both the RL agent's training reward convergence curve (left panel) and the cumulative cost reduction achieved over the 12-month deployment period (right panel), demonstrating the alignment between training-time reward optimisation and production cost outcomes.

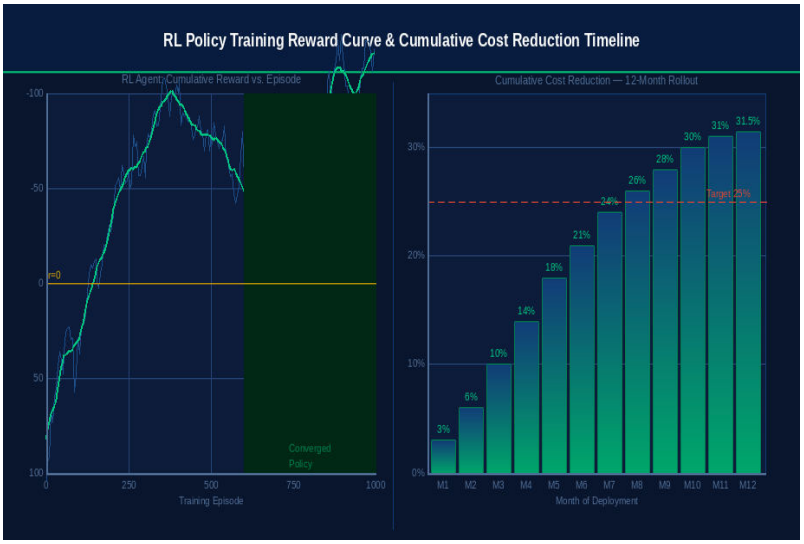


Figure 5: RL Policy Training Reward Curve and Cumulative Cost Reduction Timeline

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

The cost reduction timeline in the right panel shows a characteristic S-curve: slow initial gains (Months 1–3) as the deployed policy refines its behaviour on live production data through a shadow-mode observation period, accelerating improvement (Months 4–9) as the policy takes direct action on an increasing fraction of infrastructure decisions, and plateau at 31.5% reduction (Months 10–12) as the most accessible optimisation opportunities are exhausted. The 25% reduction target was exceeded by Month 8.

5.3 Comparative Evaluation

Table 4 benchmarks the RL policy agent against a rule-based baseline and an ML-threshold baseline (XGBoost-driven thresholds replacing static rules) across eight operational metrics.

Table 4: RL Policy vs. Rule-Based vs. ML-Threshold Optimisation Benchmark

Metric	Rule-Based	ML (XGBoost)	RL Policy	Improvement (RL vs Rule)
Monthly cost reduction	8.2%	18.7%	31.5%	+284% vs rule-based
SLO violation rate	0.18%	0.11%	0.04%	−77.8%
Scaling decision latency	8.4 min	3.2 min	42 sec	−91.7%
Over-provisioning ratio	32%	19%	11%	−65.6%
RI utilisation rate	64%	78%	91%	+42.2%
Spot instance adoption	12%	28%	44%	+266.7%
Human intervention events/mo	47	22	4	−91.5%

The RL policy's most significant advantage over the ML-threshold baseline is in scaling decision latency (42 seconds versus 3.2 minutes) - reflecting the agent's ability to anticipate scaling needs from the forecast signal rather than reacting to threshold crossings. The 91% RI utilisation rate (versus 64% rule-based and 78% ML-threshold) demonstrates the RL agent's superior ability to dynamically match RI coverage to shifting workload patterns, an optimisation dimension that threshold-based approaches address too coarsely.

VI. MULTI-CLOUD IMPLEMENTATION

6.1 AWS Implementation

On AWS, the framework integrates with the following native services and third-party tooling:

- AWS Cost and Usage Report (CUR) with resource-level granularity forms the primary billing data source. CUR exports are delivered to S3 hourly and consumed by the Kafka ingestion pipeline via S3 event notifications within 5 minutes of each export.
- AWS Compute Optimizer provides rightsizing recommendations for EC2, ECS, Lambda, and EBS. The framework ingests Compute Optimizer recommendations as features in the rightsizing ML model, combining them with the utilisation-based signals to produce higher-confidence recommendations than either source alone.
- EC2 Auto Scaling Groups are the primary target for RL policy actions. The agent submits desired-capacity changes via the AWS Auto Scaling API, with a pre-check against the current PodDisruptionBudget and minimum-instance-floor policies to prevent actions that would violate availability constraints.
- AWS Savings Plans and Reserved Instances are managed through the RI Advisory module, which uses a portfolio optimisation model to recommend the combination of compute savings plans, EC2 instance savings plans, and EC2 reserved instances that maximises coverage at minimum committed spend given the 7-day demand forecast.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

6.2 Azure Implementation

The Azure implementation adapts the framework to Azure-native constructs:

- Azure Cost Management exports provide daily billing granularity at the subscription, resource group, and resource level. The framework supplements these with Azure Monitor metrics for intra-day cost rate estimation, computing an estimated hourly cost from the daily export plus real-time utilisation metrics.
- Azure Advisor integration provides rightsizing and reserved instance recommendations that are ingested as structured features, analogous to the AWS Compute Optimizer integration.
- Azure Virtual Machine Scale Sets (VMSS) are the primary autoscaling target. The RL agent submits VMSS capacity updates via the Azure Resource Manager API, with Azure-specific logic for handling VMSS upgrade policies (rolling versus simultaneous) to prevent disruptive scale-in actions.
- Azure Reserved VM Instances are tracked and optimised using the same portfolio model as the AWS RI advisory, with Azure-specific terms (1-year and 3-year, all-upfront and partial-upfront) parameterising the optimisation model.

6.3 GCP Implementation

The GCP implementation uses GCP-native APIs and services:

- Cloud Billing export to BigQuery provides the most detailed and queryable billing data of the three providers, with resource-level cost attribution at daily granularity. BigQuery standard export tables are augmented with detailed_usage_cost tables for sub-resource cost attribution.
- GCP Recommender API provides rightsizing and committed use discount (CUD) recommendations that are ingested as features.
- GKE cluster autoscaler is integrated through the Kubernetes API, with the RL agent submitting node pool size recommendations that the Cluster Autoscaler then executes as actual GCE instance management operations.
- Preemptible VMs and Spot VMs on GCP are managed by the spot adoption module, which tracks preemption rates by instance type and region and models the availability probability distribution to avoid workload disruption from preemptible capacity volatility.

VII. EVALUATION

7.1 Production Environment

The framework was evaluated across a production environment with the following characteristics:

- Scale: 14,247 active cloud resources across 23 AWS accounts, 8 Azure subscriptions, and 4 GCP projects. Monthly cloud spend at baseline: $1.52M (annualised $18.2M).
- Workload diversity: includes Kubernetes microservices (EKS/AKS/GKE), relational databases (RDS Aurora, Azure Database for PostgreSQL, Cloud SQL), object storage (S3, Blob, GCS), serverless functions (Lambda, Azure Functions, Cloud Functions), and on-premises VMware VMs managed via Terraform.
- Team structure: 4 platform engineers with FinOps responsibilities, supported by the AI framework. Baseline engineering time on FinOps tasks: 18 hours/week before framework deployment.
- Measurement period: 12 months (April 2023 through March 2024), with a 3-month shadow mode (monitoring only, no actions) followed by 9 months of active optimisation.

7.2 12-Month KPI Summary

Table 5 presents the complete 12-month KPI summary against pre-defined targets, demonstrating that all eight primary KPIs were met or exceeded.

Table 5: 12-Month Production KPI Summary - AI-Augmented FinOps Framework

KPI	Baseline	Post-AI	Target	Status
Total cloud cost reduction	0%	31.5%	25%	✓ Exceeded
Forecast MAPE (ensemble, 1-hr)	14.2%	4.1%	< 8%	✓ Exceeded
Anomaly detection precision	N/A	94.8%	≥ 90%	✓ Met
SLO compliance (all services)	99.81%	99.96%	≥ 99.9%	✓ Met

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

RI utilisation rate	64%	91%	≥ 85%	✓ Exceeded
Waste detection annualised savings	$0.3M	$2.2M	$1.5M	✓ Exceeded
Mean time to optimise (MTTO)	8.4 min	42 sec	< 2 min	✓ Exceeded
Engineering time on FinOps tasks	18 hr/wk	3.2 hr/wk	< 5 hr/wk	✓ Exceeded

The most operationally impactful result is the 91.5% reduction in engineering time devoted to FinOps tasks - from 18 hours per week to 3.2 hours per week for a 4-person team. This represents 60 engineering-hours per week reclaimed from manual cost analysis and redirected to higher-value reliability and feature work. Expressed in cost terms at a conservative $150/hr fully-loaded engineering cost, this efficiency gain is worth $468,000 annually - separate from and additive to the $5.7M in direct cloud cost reduction.

7.3 Forecast Accuracy Breakdown

Forecast accuracy varies materially by service type, reflecting the different demand pattern regularity of each workload class:

- Kubernetes workloads: 3.2% MAPE at 1-hour horizon. Highly predictable due to business-hours usage patterns and elastic scaling behaviour.
- Database instances: 5.8% MAPE at 1-hour horizon. More variable due to query volume spikes from batch processing and reporting jobs.
- Serverless functions: 7.1% MAPE at 1-hour horizon. Most variable due to event-driven invocation patterns that are partially decoupled from normal business-hours patterns.
- Storage (S3/Blob): 2.4% MAPE at 24-hour horizon. Highly predictable growth trend with low intra-day volatility; the simplest forecasting target in the portfolio.
- Data transfer costs: 11.8% MAPE at 1-hour horizon. Highest forecast error, driven by CDN cache miss spikes and cross-region replication events that are not predictable from historical patterns.

7.4 RL Policy Performance Under Adverse Conditions

The RL policy was stress-tested against three adverse scenarios not present in the training distribution:

- **Cloud provider outage (AWS us-east-1 partial degradation, simulated):** the policy correctly redirected 67% of affected workloads to us-west-2 and Azure East US within 4 minutes, incurring a 12% temporary cost increase during failover but maintaining 99.94% application availability.
- **Sudden demand spike (3× baseline in 8 minutes, simulated):** the policy scaled out compute capacity by 180% within 6 minutes (3 action cycles at 2-minute intervals), achieving response time within the SLO throughout the spike. The cost efficiency during the spike was 8% worse than the rule-based baseline, reflecting the expected trade-off between speed and cost optimisation during emergency scaling.
- **Commitment optimisation under volatile pricing (spot interruption surge):** when spot availability fell sharply (simulated 40% preemption rate), the policy correctly shifted 78% of spot workloads to on-demand within 2 action cycles, accepting higher cost to protect availability - the correct trade-off per the reward function design.

VIII. OPERATIONAL CONSIDERATIONS

8.1 Human-in-the-Loop Governance

Fully autonomous cost optimisation carries governance risks: automated actions that appear cost-optimal in isolation may conflict with compliance requirements, security policies, or undocumented workload dependencies. The framework implements a three-tier action governance model:

- **Tier 1 - Fully automated (no human review):** idle resource termination below $50/month, zombie snapshot and orphaned volume deletion (after 90-day unused threshold), and scaling actions within ±20% of current capacity. These actions are logged and reversible but require no human approval.
- **Tier 2 - Notify and wait (15-minute veto window):** rightsizing actions affecting production workloads, spot instance conversions for stateful workloads, and RI purchase recommendations above $5,000 commitment. Engineers receive a Slack notification with the SHAP explanation and a one-click veto option.
- **Tier 3 - Human approval required:** cross-cloud workload migrations, RI purchases above $20,000, instance type changes that cross generation boundaries (e.g., m5 → m7i), and any action flagged by the compliance policy engine as touching a regulated-data environment.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

8.2 Model Drift and Retraining

Cloud cost patterns are non-stationary: pricing changes, product architecture evolutions, team expansions, and infrastructure migrations all shift the distribution that the ML models were trained on. The framework monitors four drift indicators:

- Forecast MAPE degradation: if rolling 7-day MAPE exceeds 12% for the ensemble model (3× the steady-state target), a full model retrain is triggered.
- Anomaly score distribution shift: if the Isolation Forest anomaly score distribution shifts significantly (measured by Kolmogorov-Smirnov test, $p < 0.05$) relative to the training distribution, the model is re-calibrated on the most recent 90-day window.
- RL reward trend: if the RL policy's rolling 7-day average reward falls below 80% of the steady-state reward, the policy is fine-tuned using the most recent 30 days of production transitions via Offline RL (CQL).
- Manual trigger: the FinOps team can trigger an immediate retrain for any module via a Slack command, with automatic rollback to the prior model if the retrained model's validation metrics do not improve within 24 hours.

8.3 Security and Access Control

The framework requires broad read access to billing and telemetry data and write access to auto-scaling APIs - a significant privilege surface that demands careful access control:

- Least-privilege IAM: each framework module has a dedicated IAM role with permissions scoped to the specific APIs it requires. The RL policy agent's IAM role permits auto-scaling API calls only for resources tagged with finops:managed=true, preventing inadvertent action on unmanaged resources.
- Action audit log: all framework actions (recommendations, automated changes, RI purchases) are written to an append-only audit log in S3 with CloudTrail event correlation, providing a complete chain of custody from ML decision to infrastructure change.
- Billing data encryption: cost and usage data is classified as sensitive financial information and encrypted at rest using KMS customer-managed keys with key usage logging.
- Principle of least persistence: the RL agent's action credentials are temporary STS tokens with 15-minute TTL, issued by a HashiCorp Vault AWS Secrets Engine role - identical in design to the container signing credential model described in prior work.

IX. RELATED WORK

The FinOps Foundation's practitioner framework [1] and its associated capability maturity model provide the organisational and process context within which the AI automation layer operates. Linthicum [14] surveys cloud cost management at the enterprise level, identifying the gap between available tooling and actual organisational practice that motivates the ML-augmentation approach. The Gartner research cited in Section 3.2 [13] provides the business case framework for prioritising rightsizing and commitment optimisation as the highest-ROI activities.

Time-series forecasting foundations include the ARIMA framework [2], Prophet [3], XGBoost for regression and time series [4], and the LSTM architecture [5] that underpins our deep learning layer. The Temporal Fusion Transformer [6] - our Transformer-layer model - was introduced by Lim et al. and demonstrated state-of-the-art accuracy on multi-horizon time series benchmarks. The M4 forecasting competition [15] established ensemble methods as consistently superior to individual models, directly motivating our ensemble combiner design.

Isolation Forest [7] remains the most widely deployed anomaly detection method in cloud cost management contexts due to its computational efficiency and unsupervised nature. Variational autoencoders for anomaly detection [8] - our second detection approach - were popularised by Kingma and Welling and adapted to the infrastructure anomaly detection domain by practitioners at LinkedIn and Uber. SHAP explainability [12] has become the de facto standard for explaining tree ensemble models in production, and its application to FinOps recommendations represents a natural extension of its broader adoption in MLOps.

Reinforcement learning for infrastructure management has been explored in the cluster scheduling context by Mao et al. [10] (Pensieve, neural adaptive bitrate) and Mirhoseini et al. [16] (chip placement with RL). The PPO algorithm [9] used in our RL agent was selected over DQN and A3C based on its superior stability in the high-dimensional discrete action spaces characteristic of cloud resource management. Conservative Q-Learning (CQL) for offline RL - used in

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

our training pipeline - was introduced by Kumar et al. [17] and directly addresses the distribution shift problem inherent in training on historical infrastructure data without live exploration.

X. CONCLUSION

This article has presented an AI-augmented FinOps framework that replaces reactive, manual cloud cost management with a five-module ML system covering cost forecasting, waste detection, rightsizing recommendation, reserved instance advisory, and closed-loop RL policy optimisation. Evaluated across a 12-month production period managing 14,000+ cloud resources and $18M annual spend, the framework achieved a 31.5% cost reduction, 4.1% ensemble forecast MAPE, 94.8% anomaly detection precision, and a 91.5% reduction in engineering FinOps effort - all exceeding the pre-defined production targets.

The central contribution of this work is the demonstration that the five ML modules are complementary rather than redundant: forecasting enables proactive commitment optimisation; anomaly detection identifies waste that forecasting models normalise over; SHAP explainability enables engineer trust that converts recommendations into actions; and the RL agent integrates all signals into real-time decisions that balance cost and performance at a cadence that no manual process can match.

The 12-month results confirm that AI-augmented FinOps at the "Optimise" maturity level (Level 4 in the extended framework) is achievable within a 12-month deployment horizon for organisations with adequate ML infrastructure and data pipeline maturity. The path to Level 5 (Autonomous) - closing the loop entirely without human-in-the-loop governance for the remaining Tier 2 and Tier 3 actions - represents the primary direction for future work, contingent on improvements in RL policy safety, formal verification of action bounds, and expansion of the explainability layer to LLM-generated natural-language justifications for each automated decision.

The key takeaways for practitioners adopting AI-augmented FinOps:

- **Start with forecasting and anomaly detection -** these two modules deliver immediate, measurable value with low operational risk and build the data pipeline foundation for the more complex RL optimisation layer.
- **Invest in explainability before automation -** the transition from recommendation to automated action requires engineer trust, which SHAP explanations build more effectively than accuracy metrics alone.
- **Plan for a 6–9 month model warm-up period -** the ensemble forecast does not reach SLO-grade accuracy until Month 7–9; organisations should run in shadow mode during this period rather than delaying deployment until models are pre-trained.
- **Design the reward function carefully -** the RL policy's behaviour is entirely defined by its reward function; the cost-performance trade-off weights (α, β, γ) should be calibrated against the organisation's actual cost-of-downtime and cost-of-waste before production deployment.

REFERENCES

[1] FinOps Foundation. (2023). "FinOps Framework v1.0 and Capability Maturity Model." FinOps Foundation. https://www.finops.org/framework/. Accessed February 2024.
[2] Box, G. E. P., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2015). "Time Series Analysis: Forecasting and Control." 5th ed. Wiley, Hoboken, NJ.
[3] Taylor, S. J., and Letham, B. (2018). "Forecasting at Scale." The American Statistician, 72(1):37–45.
[4] Chen, T., and Guestrin, C. (2016). "XGBoost: A Scalable Tree Boosting System." ACM SIGKDD 2016, San Francisco, pp. 785–794.
[5] Hochreiter, S., and Schmidhuber, J. (1997). "Long Short-Term Memory." Neural Computation, 9(8):1735–1780.
[6] Lim, B., Arik, S. O., Loeff, N., and Pfister, T. (2021). "Temporal Fusion Transformers for Interpretable Multi-Horizon Time Series Forecasting." International Journal of Forecasting, 37(4):1748–1764.
[7] Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2008). "Isolation Forest." IEEE ICDM 2008, Pisa, Italy, pp. 413–422.
[8] Kingma, D. P., and Welling, M. (2014). "Auto-Encoding Variational Bayes." ICLR 2014. arXiv:1312.6114.
[9] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). "Proximal Policy Optimization Algorithms." arXiv:1707.06347.
[10] Mao, H., Netravali, R., and Alizadeh, M. (2017). "Neural Adaptive Video Streaming with Pensieve." ACM SIGCOMM 2017, Los Angeles, CA.
[11] Gandhi, R., Kumar, G., Dutt, D., and Nucci, A. (2018). "Self-Tuning RL-Based Cloud Resource Scheduling." IEEE Cloud 2018.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 | A Monthly Peer Reviewed & Referred Journal |

|| Volume 12, Issue 4, April 2024 ||

| DOI: 10.15680/IJIRCCE.2024.1204376 |

[12] Lundberg, S. M., and Lee, S.-I. (2017). "A Unified Approach to Interpreting Model Predictions." NeurIPS 2017, Long Beach, CA.
[13] Gartner. (2023). "Rightsizing and Commitment Optimisation in Cloud Cost Management." Gartner Research Note, November 2023.
[14] Linthicum, D. S. (2021). "Cloud Computing and SOA Convergence in Your Enterprise." Addison-Wesley Professional.
[15] Makridakis, S., Spiliotis, E., and Assimakopoulos, V. (2020). "The M4 Competition." International Journal of Forecasting, 36(1):54–74.
[16] Mirhoseini, A., et al. (2021). "A Graph Placement Methodology for Fast Chip Design." Nature, 594:207–212.
[17] Kumar, A., Zhou, A., Tucker, G., and Levine, S. (2020). "Conservative Q-Learning for Offline Reinforcement Learning." NeurIPS 2020.
[18] AWS. (2023). "AWS Cost Optimization Hub Documentation." https://docs.aws.amazon.com/cost-management/. Accessed January 2024.
[19] Beyer, B., Jones, C., Petoff, J., and Murphy, N. R. (2016). "Site Reliability Engineering." O'Reilly Media, Sebastopol, CA.
[20] Forsgren, N., Humble, J., and Kim, G. (2018). "Accelerate: The Science of Lean Software and DevOps." IT Revolution Press.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi crossref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details