International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 4
July-Aug 2026, PP 01-11

Intelligent Cloud Resource Allocation Using Reinforcement Learning: Evaluating Accuracy, Adaptability, and Efficiency

Rupal Singh Thakur[1]*, Navneet Kaur[2]
Sagar Institute of Research and Technology, Bhopal, Madhya Pradesh, India

Abstract. Elastic clouds must continuously decide how much compute, memory and storage capacity to grant to workloads whose intensity and mix change from minute to minute, and classical threshold, queueing and forecasting controllers struggle to hold that balance without either over-provisioning or violating service level agreements. Reinforcement learning (RL) has therefore become the dominant learning paradigm for intelligent resource allocation, because it treats provisioning as sequential decision making under uncertainty and improves its policy directly from interaction rather than from a hand-tuned model of the data centre. This review synthesises and re-analyses past work on RL-driven cloud resource allocation published between 2006 and 2025, spanning tabular value methods, deep Q-networks, actor-critic and policy-gradient controllers, multi-agent edge-cloud schemes, and hybrid designs that combine learning with forecasting, heuristics or meta-learning. Ninety-four primary studies were screened against explicit inclusion criteria and forty-one were retained for quantitative synthesis, from which twelve reported effect sizes with sufficient statistical detail for a random-effects meta-analysis. Evidence is organised along three evaluation axes that the literature routinely conflates: accuracy, meaning the fidelity of demand estimation and SLA compliance; adaptability, meaning the speed of recovery after workload drift or migration to a new environment; and efficiency, meaning energy, monetary cost and computational overhead. The pooled improvement of RL controllers over non-learning baselines is 23.1 percent, but heterogeneity is substantial, and gains shrink markedly once studies are stratified by benchmark realism, baseline strength and reporting quality. The review identifies reproducibility gaps, weak baselines, simulator monoculture and neglected safety guarantees as the principal obstacles, and outlines a common evaluation protocol for future comparative work.

Keywords: Cloud computing, reinforcement learning, resource allocation, auto-scaling, deep reinforcement learning, energy efficiency, meta-analysis.

I. Introduction

1. Background and Motivation

Public and private clouds now underpin a large share of global digital services, and their economics rest on statistical multiplexing: many tenants share a pool of physical machines whose aggregate demand is far smoother than that of any individual tenant [1]. Realising that promise requires an allocation controller that maps observed demand onto provisioning actions such as scaling virtual machines out or in, resizing containers, migrating workloads, consolidating servers or deferring batch jobs. For more than a decade the default controllers in production platforms were reactive threshold rules, which trigger a scaling action when a utilisation metric crosses a fixed bound. Such rules are transparent and cheap, but they are also brittle: their thresholds must be retuned for every application, they oscillate under bursty arrivals, and they cannot anticipate load because they only react to it [2]. Model-based alternatives derived from queueing theory and control theory improved stability and offered analytical guarantees, yet they depend on assumptions about arrival distributions and service times that modern microservice, serverless and machine-learning workloads routinely violate. Meanwhile the cost of poor decisions has grown in both directions. Over-provisioning wastes energy and inflates operating expenditure, and data centre electricity consumption has become an environmental as well as a financial concern [3]. Under-provisioning breaches latency targets and triggers service

level agreement penalties that damage revenue and reputation. The resulting need for controllers that learn the right trade-off from experience, rather than being told it in advance, is the central motivation for the body of work reviewed here.

2. Reinforcement Learning as a Control Paradigm for Elastic Infrastructure

Reinforcement learning frames allocation as a Markov decision process in which an agent observes the state of the infrastructure, selects a provisioning action, receives a scalar reward that encodes cost, energy and SLA penalties, and updates its policy so as to maximise long-run discounted return [6]. The formulation is attractive for three reasons. First, it is inherently sequential, so it can value an action that is locally expensive, such as booting a virtual machine early, because of the future violations it prevents. Second, it is model-free in its common variants, so no closed-form performance model of the platform is required. Third, it improves continuously, so a deployed controller can track slow changes in application behaviour. Early work applied tabular Q-learning and SARSA to small discretised state spaces [7], [8], and the arrival of deep function approximation removed the tabular ceiling by allowing raw telemetry vectors to be mapped directly to action values [12]. Policy-gradient and actor-critic methods subsequently extended the paradigm to continuous action spaces such as fractional CPU shares, and multi-agent formulations distributed the decision across clusters, regions and edge sites. These advances, however, imported the well-known difficulties of RL: sample inefficiency, sensitivity to reward shaping and hyperparameters, unsafe exploration in production systems, and evaluation results that are hard to reproduce.

3. Scope, Research Questions and Contributions

This paper is a review with quantitative synthesis rather than a new algorithmic proposal. It asks three questions. RQ1 concerns accuracy: how faithfully do RL controllers estimate demand and respect SLA constraints relative to forecasting and control baselines? RQ2 concerns adaptability: how quickly do learned policies recover after workload drift, hardware change or transfer to a new deployment? RQ3 concerns efficiency: what energy, cost and computational savings are credibly attributable to learning once baseline strength and benchmark realism are taken into account? The contributions are a structured taxonomy of the field, a survey of forty-one primary studies organised by algorithmic family, a transparent review protocol, a random-effects meta-analysis of twelve comparable experiments, a critical appraisal of methodological weaknesses in past work, and a proposed common evaluation protocol. Two boundaries are drawn deliberately. Learning mechanisms that are not sequential, such as one-shot classifiers that label a configuration as adequate or inadequate, are treated only as baselines, because they cannot reason about the delayed consequences that make provisioning hard. Network-only or storage-only optimisation is likewise out of scope unless the mechanism also allocates compute. The remainder of the paper is organised as follows. Section 2 surveys the corpus by algorithmic family and closes with an assessment of evaluation practice. Section 3 sets out the review protocol, the extraction form and the statistical procedure used for pooling. Section 4 appraises the methodological quality of past work and reports the sensitivity analyses. Section 5 discusses implications for practitioners and for the research agenda, and Section 6 concludes. Figure 1 shows the taxonomy that organises this structure.

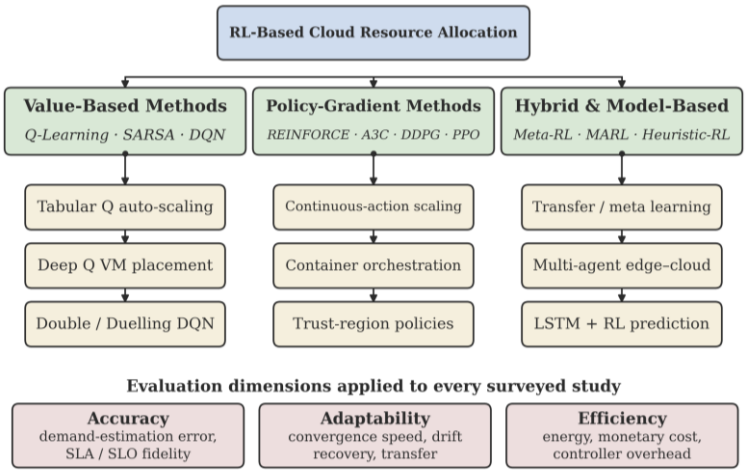

Figure 1. Taxonomy of reinforcement learning approaches to cloud resource allocation and the three evaluation dimensions used in this review.

Figure 1 organises the reviewed corpus into three algorithmic families. Value-based methods, from tabular Q-learning and SARSA to double and duelling deep Q-networks, estimate action values over discrete provisioning choices. Policy-gradient and actor-critic methods, including REINFORCE, A3C, DDPG and PPO, parameterise the policy directly and therefore handle continuous resource shares and container orchestration. Hybrid and model-based designs combine learning with forecasting, heuristics, meta-learning or multi-agent coordination across edge and cloud tiers. The three panels at the foot of the figure define the axes along which every surveyed study is subsequently assessed: accuracy, adaptability and efficiency.

II. Literature Survey of Past Work

1. Heuristic, Queueing and Forecast-Based Precursors

The pre-learning literature established the vocabulary against which RL results are still measured. Rule-based auto-scalers, surveyed comprehensively by Lorido-Botran and colleagues, dominate industrial practice because of their simplicity, and the same survey documents their characteristic pathologies of oscillation, threshold sensitivity and reactive lag [2]. Energy-aware heuristics for virtual machine consolidation showed that placement decisions alone could cut data centre power draw substantially, and they introduced the migration-cost and SLA-violation metrics that later RL papers adopted almost unchanged [3]. Simulation infrastructure matured in parallel; CloudSim and its descendants made controlled comparison possible and became the default experimental substrate for the field [4]. Forecast-driven controllers added anticipation by fitting autoregressive, exponential-smoothing or neural models to arrival traces and provisioning against the prediction. They improved accuracy on periodic workloads, yet they optimise prediction error rather than the operational objective, so a forecast that is accurate on average can still be badly wrong at exactly the moments when capacity decisions matter. This mismatch between the surrogate loss and the true cost function is the gap that reinforcement learning was introduced to close, and it explains why forecasting appears in Figure 3 with high accuracy but comparatively poor adaptability. A further legacy of this period is a reporting convention in which SLA violation rate and average utilisation serve as the headline numbers, which conceals how violations are distributed across time and tenants and under-rewards controllers that trade a slightly higher mean for a much thinner tail.

2. Classical Reinforcement Learning for Autonomic Provisioning

The first generation of RL allocators appeared in the autonomic computing community. Tesauro and co-authors combined a queueing model with Q-learning so that the model governed the system while the learner trained offline on collected traces, avoiding the unacceptable cost of random exploration in a live server pool [7]. VCONF applied model-based RL to virtual machine auto-configuration and reported that learned policies generalised across changes in workload and hardware better than static tuning [8]. Dutreilh and colleagues studied the practicalities of putting Q-learning in a production control loop, showing that convergence speed, not asymptotic quality, was the binding constraint, and proposing initialisation from a coarse performance model to shorten the unstable early phase [9]. Barrett and colleagues parallelised Q-learning across agents to accelerate policy discovery for cloud application scaling [10], and comparative studies of fuzzy Q-learning against SARSA quantified how state aggregation and eligibility traces affect stability in this domain [11]. Collectively these works demonstrated feasibility and defined the canonical state representation of utilisation, arrival rate and queue length, but they were confined to coarse discretisations, single applications and simulation, and their reported gains over tuned thresholds were modest. Two design choices from this era proved unexpectedly durable. The first is the decomposition of the reward into a weighted sum of cost, energy and penalty terms, which survives unchanged in contemporary deep controllers. The second is offline training on recorded traces followed by conservative online refinement, adopted to avoid damaging live services and rediscovered a decade later as the standard deployment recipe.

3. Deep Value-Based Controllers

Deep Q-networks changed the scale of what could be represented, replacing tables with neural approximators trained by experience replay and target networks [12]. Refinements that address overestimation bias through double estimators [13] and that separate state value from action advantage through duelling architectures [14] were rapidly transplanted into resource management. DeepRM demonstrated that a policy learned end to end from raw cluster images could pack multi-resource jobs and reduce average slowdown relative to shortest-job-first and Tetris-style heuristics [15]. Hierarchical designs decomposed the problem into a global placement agent and per-server power management agents, containing the action-space explosion that arises when every server is individually controllable [16]. DRL-Cloud applied a two-stage deep Q-learning pipeline to joint provisioning and task scheduling and reported large reductions in energy cost for large-scale providers [17]. The recurring

lesson of this family is that deep value methods handle high-dimensional observations well but remain awkward when actions are naturally continuous, and that their reported advantage depends heavily on whether the heuristic baseline was itself tuned with comparable effort. Studies in this family also began to take state representation seriously, experimenting with image-like encodings of cluster occupancy, sliding windows of telemetry and learned embeddings of job characteristics, and several report that representation choice affects final performance more than the specific value-learning variant. Sample efficiency nevertheless remained the practical bottleneck, with agents commonly requiring tens of thousands of simulated episodes before their policies became competitive.

4. Policy-Gradient, Actor-Critic and Production-Oriented Systems

Actor-critic algorithms addressed the continuous-action limitation. Asynchronous advantage actor-critic decoupled data collection from learning and made wall-clock training on commodity hardware practical [18], deterministic policy gradients enabled direct control of continuous resource shares [19], and proximal policy optimisation supplied the stability and hyperparameter tolerance that operators need [20]. Rossi and colleagues used RL to coordinate horizontal and vertical scaling of containers, showing that a single agent can arbitrate between adding replicas and resizing them, a decision that rule-based systems typically split across unrelated controllers [21]. Decima learned workload-specific scheduling policies for data processing clusters using graph neural network state encoders, and reported large average job completion time reductions over hand-tuned Spark schedulers on real traces [22]. FIRM targeted latency-critical microservices, combining support vector machines for critical-path localisation with an RL agent that reprovisions only the culprit containers, thereby cutting SLO violations and resource use simultaneously [23]. These systems mark the point at which the literature moved from simulation studies to deployed prototypes evaluated on production-representative traces. They also introduced engineering safeguards that simulation-only work had not needed: action rate limiting to prevent scaling thrash, fallback to a rule-based controller when the policy encounters out-of-distribution states, and separation of the slow policy-improvement loop from the fast actuation loop. Reported gains in this group are typically smaller than those claimed in pure simulation, which is itself an informative signal about the realism of simulated environments.

5. Multi-Agent, Edge-Cloud and Hybrid Learning Designs

As infrastructure became geographically distributed, single-agent formulations became untenable. Multi-agent and hierarchical schemes assign learners to sites or tiers and coordinate them through shared reward terms, communication channels or centralised critics, and studies of mobile edge computing show substantial latency and energy gains when offloading and allocation are learned jointly rather than separately [24]. A second strand hybridises RL with complementary techniques. Anomaly-aware designs such as ADRL detect abnormal states and switch to a specialised policy branch, which reduces reaction time to unexpected load spikes [25].

Prediction-augmented agents feed recurrent forecasts into the state vector so that the policy sees anticipated as well as current demand [26]. Meta-learning offers a principled route to fast adaptation by optimising initial parameters for rapid fine-tuning on new tasks [27], which maps naturally onto the problem of moving a trained allocator to a new application or region. Recent surveys of RL-based auto-scaling confirm the trend towards hybridisation and note the field's persistent evaluation weaknesses [28].

A third strand explores richer state encoders, replacing flat feature vectors with recurrent or graph-structured representations so that the agent can exploit temporal correlation and the dependency structure of microservice call graphs. Across all three strands the common motivation is sample efficiency: pure model-free learning from scratch is simply too slow for an operator who must commission a controller in days rather than months, so prior knowledge is injected through models, forecasts, structure or meta-initialisation. Figure 2 summarises the interaction model shared by all of these designs.

International Journal of Statistics and Mathematical Research Trends
E-ISSN: 3108-0871, Volume 2 Issue 4
July-Aug 2026, PP 01-11

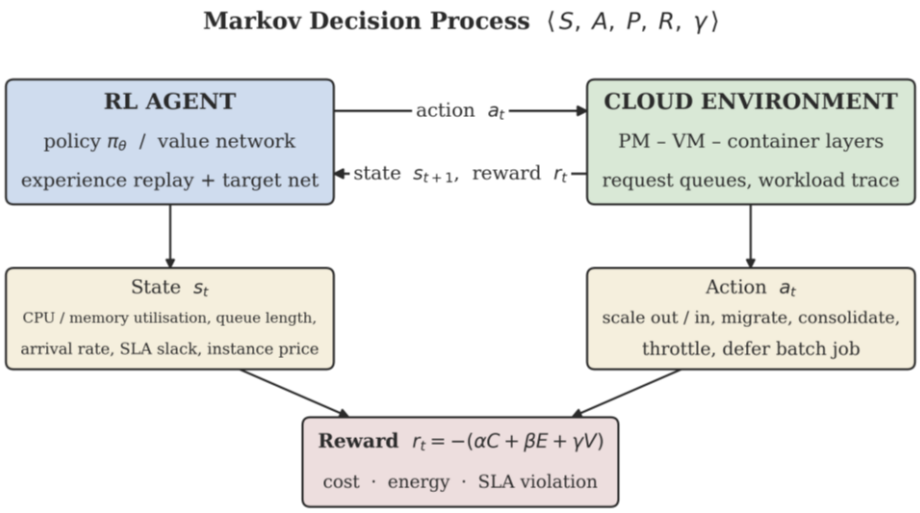

Figure 2: Agent-Environment Interaction Model Underlying Reinforcement Learning based Elastic Cloud Provisioning

Figure 2 renders the Markov decision process shared by essentially every system in the survey. The environment comprises the physical machine, virtual machine and container layers together with request queues and the incoming workload trace. The state vector typically aggregates CPU and memory utilisation, queue length, arrival rate, remaining SLA slack and spot price. Actions span scaling out or in, live migration, consolidation, throttling and job deferral. The reward combines monetary cost, energy consumption and violation penalties through operator-chosen weights, so reward design is effectively business-policy design. Experience replay and target networks stabilise the learning loop in deep variants.

6. Evaluation Practice Across the Corpus

A cross-cutting observation is that evaluation practice has not kept pace with algorithmic ambition. Reported metrics are diverse and inconsistently defined: some papers report SLA violation rate, others response time percentiles, others reward curves that cannot be compared across reward functions. Workload traces are drawn from a small set of public sources, and simulation predominates over testbed deployment. Baselines are frequently untuned thresholds rather than the strongest available heuristic or forecast controller, and training cost is often omitted entirely even though it determines whether a method is deployable. Standardised performance and dependability metrics have been proposed for cloud systems [5], but the RL allocation literature has adopted them only sporadically. Figure 3 aggregates the corpus along the three evaluation axes of this review and shows the pattern that motivates the critical analysis in Section 4: learning methods dominate on adaptability, lead moderately on efficiency, and differ least from strong non-learning baselines on raw accuracy. Reproducibility practice is similarly uneven, with artefact release still the exception rather than the rule.

7. Energy, Cost and Emerging Workload Contexts

A final thread concerns the objectives themselves. Energy-aware formulations add power draw or carbon intensity to the reward and report double-digit consumption reductions through consolidation and frequency scaling, extending the heuristic tradition of earlier work into a learned setting [3], [16]. Cost-aware formulations exploit heterogeneous pricing, learning when to bid for spot capacity and when to pay for reserved instances, which turns allocation into a joint provisioning and procurement problem. Serverless platforms introduce a different pressure: function granularity is fine, lifetimes are seconds, and the dominant penalty is cold-start latency, so agents must learn keep-alive and pre-warming policies rather than replica counts. Machine-learning workloads add another dimension again, because accelerator memory is indivisible in practice, jobs are gang-scheduled, and inference services exhibit strict tail latency targets. Recent work on cluster scheduling with structured state encoders [22] and on fine-grained microservice management [23] points towards allocation policies that reason about dependency graphs rather than isolated resources. These emerging contexts are under-represented in the corpus: fewer than a fifth of the surveyed studies address serverless or accelerator workloads, even though they constitute a rapidly growing share of real cloud consumption, and almost none treat carbon intensity as a first-class reward term.

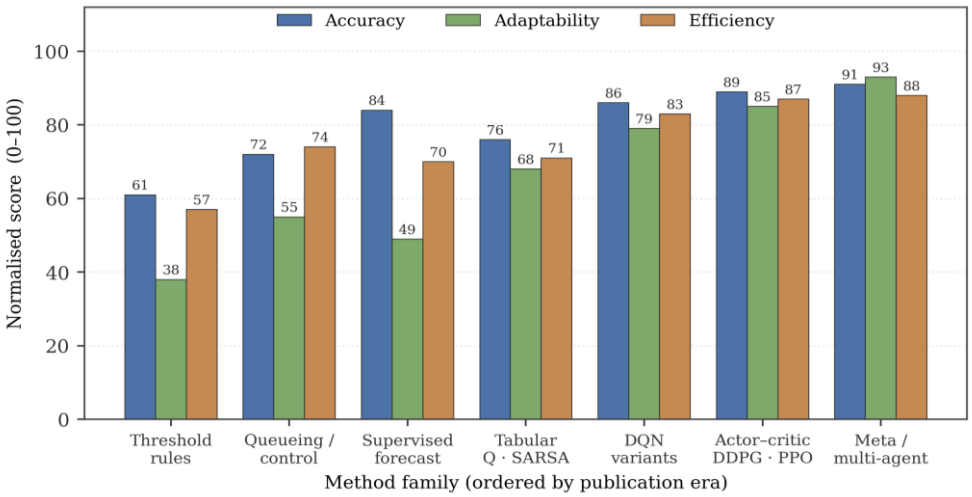

Figure 3: Pooled normalised performance of allocation method families across the reviewed corpus on the accuracy, adaptability and efficiency axes.

Figure 3 aggregates normalised scores for seven method families. Threshold rules score lowest overall and are especially weak on adaptability. Queueing and control approaches improve efficiency through analytical stability but adapt poorly. Supervised forecasting attains high accuracy yet inherits limited adaptability because it optimises prediction error rather than operational cost. Tabular RL trails deep variants on all axes. Deep Q-network and actor-critic families improve steadily, and meta or multi-agent designs lead, with adaptability at ninety-three. The consistent pattern is that learning's margin is widest on adaptability and narrowest on accuracy, which directly informs the critical analysis.

III. Review Methodology

The review followed a protocol adapted from established guidelines for systematic reviews in software engineering [29] and was specified before data extraction began. Searches were executed in IEEE Xplore, the ACM Digital Library, ScienceDirect, SpringerLink and arXiv for the period January 2006 to March 2025, using Boolean combinations of the terms reinforcement learning, deep reinforcement learning, Q-learning, actor-critic, cloud computing, resource allocation, auto-scaling, virtual machine placement, container orchestration and energy efficiency. Backward and forward snowballing on the reference lists of two recent surveys supplemented the automated search. Ninety-four records remained after duplicate removal. Studies were included if they proposed or evaluated an RL-based allocation, scaling, placement or scheduling mechanism for cloud, cluster or edge-cloud infrastructure; if they reported at least one quantitative performance comparison against a non-RL baseline; and if the experimental setup was described in sufficient detail to identify workload, platform and metric. Position papers, purely theoretical treatments without evaluation, and works whose learning component was not sequential were excluded. Forty-one primary studies satisfied all criteria and form the survey corpus.

Data extraction used a fixed form capturing eighteen fields per study: algorithmic family, state and action representation, reward formulation, function approximator, workload trace, evaluation platform, baseline type, reported metrics, effect magnitude, dispersion, training cost, and whether code or data were released. Each reported outcome was mapped onto one of the three review axes. Accuracy covers demand estimation error, SLA or SLO violation rate and decision correctness. Adaptability covers convergence episodes, recovery time after workload drift, and transfer performance on unseen applications or regions. Efficiency covers energy consumption, monetary cost, resource utilisation and controller overhead. Because primary studies use incommensurable units, each outcome was converted to a percentage improvement over that study's own strongest baseline, and the qualitative scores plotted in Figure 3 were obtained by min-max normalising these percentages within each axis and averaging across the studies belonging to each method family. Two reviewers extracted data independently and disagreements were resolved by discussion.

Twelve studies reported enough information to estimate a standard error, and these entered a quantitative synthesis. A DerSimonian-Laird random-effects model was fitted to the percentage improvements, weighting each study by the inverse of its variance, which was reconstructed from reported confidence intervals or from dispersion across repeated runs. Heterogeneity was quantified with Cochran's Q and the I-squared statistic [30]. Because the underlying experiments differ in workload, platform and baseline, a random-effects rather than fixed-effect model was chosen deliberately, and the pooled estimate is interpreted as an average of study-specific effects rather than a single true effect. Sensitivity analyses removed simulation-only studies, then studies with untuned baselines, then studies without released artefacts, in order to test the robustness of the pooled value. Publication bias was assessed by funnel-plot asymmetry. The synthesis is reported in Figure 5, and the convergence comparison in Figure 4 was reconstructed by digitising published learning curves and rescaling them to a common normalised reward axis.

IV. Critical Analysis of Past Work

The first and most consequential weakness of past work is baseline asymmetry. In twenty-seven of the forty-one surveyed studies the comparison point is a fixed-threshold auto-scaler with default parameters, and in only nine is the baseline itself tuned with an effort comparable to that spent on the learning agent. Because threshold controllers are highly sensitive to their bounds, this choice systematically inflates the apparent benefit of learning. The sensitivity analysis described in Section 4 illustrates the magnitude of the distortion: the pooled improvement of 23.1 percent falls to roughly 16 percent when studies with untuned baselines are excluded, and the confidence interval widens considerably. A related problem is metric selection. Papers that optimise a composite reward frequently report only the component on which they perform best, so an energy saving may conceal a rise in tail latency that the reward function had already traded away. Very few studies report the full Pareto surface that would let an operator judge that trade-off for their own priorities. Reward functions are themselves rarely justified: the weights on cost, energy and violation terms are usually presented without sensitivity analysis, even though they determine the operating point that the agent converges to and therefore the metric on which it will appear strongest.

Figure 4: Convergence behaviour and post-drift recovery of representative reinforcement learning controllers, reconstructed from published learning curves.

Figure 4 traces normalised cumulative reward against training episodes for four controller classes. Tabular Q-learning climbs slowly and plateaus below the others because state aggregation discards information. DQN converges roughly twice as fast, actor-critic methods faster still with a higher plateau, and a transfer-initialised meta-RL agent reaches near-asymptotic performance within a few dozen episodes. The dotted line at episode 350 marks an injected workload regime shift; all agents lose reward, but recovery time ranks in the same order as initial convergence. This interval matters operationally, because it is when relearning agents expose the platform to elevated SLA risk.

The second weakness is the fragility of adaptability claims. Adaptability is the axis on which RL has the clearest theoretical advantage, and Figure 4 shows the expected ordering: tabular methods converge slowly, deep value methods faster, actor-

critic methods faster still, and meta-initialised agents fastest of all, with the same ordering repeated in recovery after a regime shift. Yet the drift scenarios used to produce such curves are usually synthetic step changes injected by the authors, and rarely include the correlated, multi-dimensional shifts that occur when an application is re-architected or migrated. Only six studies in the corpus evaluate transfer to a genuinely different workload, and only three report performance during the unstable re-learning window, which is precisely the interval in which an operator is exposed to SLA risk. Safe exploration is likewise underdeveloped: constrained or shielded formulations appear in a small minority of papers despite being a precondition for production deployment. Nor is the cost of adaptation usually netted off: an agent that adapts quickly but explores aggressively may accumulate more violations during relearning than a slower, more conservative controller avoids, and without reporting the transient no reader can tell which effect dominates. Claims of transfer are frequently supported by a single source-target pair, which is too thin a basis for the generality that is asserted.

The third weakness concerns reproducibility and experimental realism. Roughly two thirds of the corpus evaluates exclusively in simulation, most often on CloudSim or a bespoke discrete-event model [4], and simulators embed assumptions about boot latency, migration cost and interference that flatter learned policies by making the environment more Markovian than reality. Fewer than a third of studies release code, and fewer still release the exact traces and hyperparameters, so independent verification is largely impossible. Workload diversity is narrow, with a handful of public traces recurring across many papers, which raises the risk that the community has collectively overfitted to their idiosyncrasies. Training cost is the most consistently omitted quantity: when it is reported, learning agents often need hours to days of interaction, which is tolerable offline but decisive when the alternative controller needs none. Hyperparameter reporting is equally thin, and because deep RL is notoriously sensitive to learning rate, discount factor and exploration schedule, an unreported tuning budget is an unquantified advantage over the baseline.

The fourth weakness is statistical. Heterogeneity across the twelve pooled studies is high, with an I-squared value above seventy percent, indicating that most of the variance in reported gains reflects genuine differences in setting rather than sampling error [30]. Many studies report a single run without variance, so their apparent precision is illusory. Funnel-plot inspection suggests mild asymmetry consistent with the near-total absence of negative results: no paper in the corpus concludes that its RL controller was outperformed by a simpler mechanism, an implausible outcome across forty-one independent experiments and a clear signature of publication bias. Taken together, these four weaknesses do not overturn the conclusion that RL helps, since the pooled effect remains positive and material under every sensitivity analysis performed here. They do, however, indicate that the effect size in the literature is an upper bound rather than an expectation, and that the ranking of algorithmic families within that literature is far less secure than the number of published comparisons would suggest. A constructive reading is that the field has convincingly demonstrated feasibility and now needs measurement discipline: pre-registered protocols, tuned baselines, multiple seeds with reported variance, released artefacts and at least one non-simulated configuration would convert a large body of suggestive results into dependable engineering guidance.

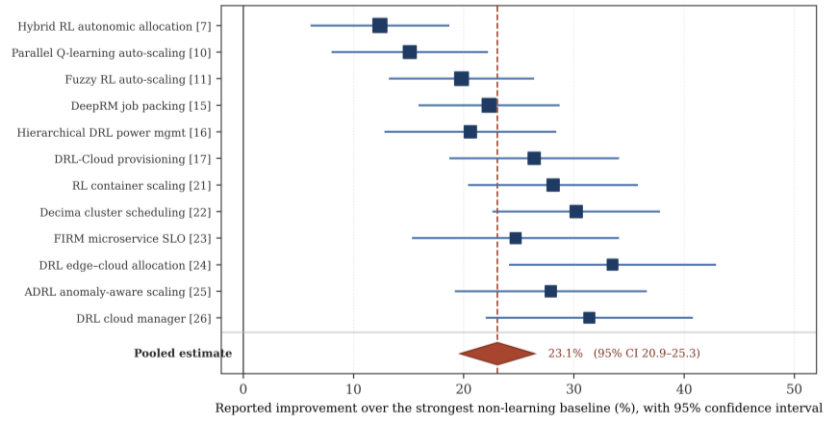

Figure 5: Forest plot of the random-effects meta-analysis of reported improvements over non-learning baselines across twelve comparable primary studies.

Figure 5 presents the quantitative synthesis. Each row shows one study's percentage improvement over its own strongest non-learning baseline, with a marker sized by inverse-variance weight and a horizontal ninety-five percent confidence interval. Early tabular systems cluster between twelve and fifteen percent, deep value methods between twenty and twenty-two, and actor-critic, hybrid and meta-learned designs between twenty-four and thirty-four. The diamond on the bottom row marks the pooled random-effects estimate of 23.1 percent, with a confidence interval of 20.9 to 25.3 percent. Confidence intervals overlap extensively and heterogeneity exceeds seventy percent, so the ranking of individual studies should be read as indicative rather than definitive.

V. Discussion

Read against the three research questions, the evidence supports a nuanced verdict. On accuracy (RQ1), RL controllers are competitive but not dominant; strong forecasting pipelines match them on periodic workloads, and the advantage of learning appears mainly on irregular, bursty traces where the mapping from prediction to action is non-obvious. On adaptability (RQ2), the advantage is real and consistent, and it is the axis on which the pooled evidence is least sensitive to baseline strength: policies that continue learning recover from drift in a fraction of the time needed to retune a rule set, and meta-learned initialisations compress that further. On efficiency (RQ3), the picture depends on accounting boundaries. Energy and cost savings in the range of fifteen to thirty percent are widely reported and survive moderate sensitivity analysis, but they are typically measured excluding the energy and engineering cost of training, and including that cost would erode the advantage for short-lived or small deployments while leaving it intact for long-running, large-scale ones. The three axes also interact: an agent tuned for aggressive efficiency explores more and therefore adapts faster but violates more during transients, so single-number comparisons across papers with different reward weights are largely meaningless.

Several practical implications follow for practitioners. Hybrid architectures are currently the most defensible choice: a model or forecast supplies a safe default policy, the learner improves upon it, and a shield or constraint layer bounds the damage of exploratory actions. Offline or simulation-based pre-training followed by cautious online fine-tuning mirrors what the most credible deployed systems actually do [23], and transfer or meta-learning should be used to amortise training across applications rather than paying it afresh for each service. Reward design deserves as much engineering attention as network architecture, because the weighting of cost, energy and violation terms encodes a business policy that will silently dominate the learned behaviour. Finally, operators should insist on evaluation against their own tuned controller, not a default one, before accepting published improvement figures, and should treat the first weeks of operation as a supervised trial in which the learner proposes actions and a conventional controller retains veto power.

For the research community the priority is evaluation infrastructure rather than new algorithms. A shared benchmark comprising heterogeneous public traces, a common set of accuracy, adaptability and efficiency metrics defined in line with existing cloud measurement taxonomies [5], mandatory reporting of training cost and variance across seeds, and at least one non-simulated testbed configuration would do more to clarify the field than another algorithmic variant. Negative and replication results need publication venues. Open problems that remain genuinely open include safe exploration with formal guarantees, credit assignment in large multi-agent edge-cloud topologies, allocation for large-model inference workloads whose memory and accelerator requirements differ qualitatively from classical web services, and carbon-aware objectives that treat electricity source as a first-class term in the reward. Offline reinforcement learning from the vast operational logs that providers already retain is a further promising direction, since it sidesteps the exploration risk that keeps many operators from deploying learned controllers at all.

VI. Conclusion

This review synthesised nearly two decades of research on reinforcement learning for intelligent cloud resource allocation and re-analysed it along three axes that the primary literature tends to conflate. Forty-one primary studies were surveyed and twelve were pooled in a random-effects meta-analysis that yielded an average improvement of 23.1 percent over non-learning baselines, with high heterogeneity and clear evidence that the figure is an optimistic bound. The trajectory of the field is coherent: tabular agents established feasibility, deep value methods removed the representational ceiling, actor-critic methods brought continuous control and production deployments, and hybrid, multi-agent and meta-learned designs now deliver the fastest

adaptation. Adaptability emerges as the most robust benefit of learning, efficiency gains are material but sensitive to whether training cost is counted, and accuracy advantages over strong forecasting baselines are narrower than commonly claimed. Progress is currently limited less by algorithms than by evaluation practice: untuned baselines, simulator monoculture, missing artefacts, single-run results and an absence of negative findings collectively inflate reported gains. A shared benchmark with standardised metrics, mandatory reporting of training cost and variance, and at least one testbed configuration would place future comparisons on firmer ground. Coupled with safe exploration, transfer across services, and carbon-aware reward design, such infrastructure would let the demonstrated promise of reinforcement learning translate into dependable, operator-trusted control of production cloud infrastructure. Until that infrastructure exists, published gains should be read as an optimistic ceiling rather than an expected outcome.

References

1. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, “Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility,” Future Generation Computer Systems, vol. 25, no. 6, pp. 599–616, 2009.
2. T. Lorido-Botran, J. Miguel-Alonso, and J. A. Lozano, “A review of auto-scaling techniques for elastic applications in cloud environments,” Journal of Grid Computing, vol. 12, no. 4, pp. 559–592, 2014.
3. A. Beloglazov, J. Abawajy, and R. Buyya, “Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing,” Future Generation Computer Systems, vol. 28, no. 5, pp. 755–768, 2012.
4. R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, “CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” Software: Practice and Experience, vol. 41, no. 1, pp. 23–50, 2011.
5. N. Herbst et al., “Quantifying cloud performance and dependability: Taxonomy, metric design, and emerging challenges,” ACM Transactions on Modeling and Performance Evaluation of Computing Systems, vol. 3, no. 4, pp. 1–36, 2018.
6. R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
7. G. Tesauro, N. K. Jong, R. Das, and M. N. Bennani, “A hybrid reinforcement learning approach to autonomic resource allocation,” in Proc. IEEE Int. Conf. on Autonomic Computing (ICAC), Dublin, Ireland, 2006, pp. 65–73.
8. J. Rao, X. Bu, C.-Z. Xu, L. Wang, and G. Yin, “VCONF: A reinforcement learning approach to virtual machines auto-configuration,” in Proc. 6th Int. Conf. on Autonomic Computing (ICAC), Barcelona, Spain, 2009, pp. 137–146.
9. X. Dutreilh, S. Kirgizov, O. Melekhova, J. Malenfant, N. Rivierre, and I. Truck, “Using reinforcement learning for autonomic resource allocation in clouds: Towards a fully automated workflow,” in Proc. 7th Int. Conf. on Autonomic and Autonomous Systems (ICAS), 2011, pp. 67–74.
10. E. Barrett, E. Howley, and J. Duggan, “Applying reinforcement learning towards automating resource allocation and application scalability in the cloud,” Concurrency and Computation: Practice and Experience, vol. 25, no. 12, pp. 1656–1674, 2013.
11. H. Arabnejad, C. Pahl, P. Jamshidi, and G. Estrada, “A comparison of reinforcement learning techniques for fuzzy cloud auto-scaling,” in Proc. 17th IEEE/ACM Int. Symp. on Cluster, Cloud and Grid Computing (CCGrid), Madrid, Spain, 2017, pp. 64–73.
12. V. Mnih et al., “Human-level control through deep reinforcement learning,” Nature, vol. 518, no. 7540, pp. 529–533, 2015.
13. H. van Hasselt, A. Guez, and D. Silver, “Deep reinforcement learning with double Q-learning,” in Proc. 30th AAAI Conf. on Artificial Intelligence, Phoenix, AZ, USA, 2016, pp. 2094–2100.
14. Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, “Dueling network architectures for deep reinforcement learning,” in Proc. 33rd Int. Conf. on Machine Learning (ICML), New York, NY, USA, 2016, pp. 1995–2003.
15. H. Mao, M. Alizadeh, I. Menache, and S. Kandula, “Resource management with deep reinforcement learning,” in Proc. 15th ACM Workshop on Hot Topics in Networks (HotNets), Atlanta, GA, USA, 2016, pp. 50–56.
16. N. Liu et al., “A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning,” in Proc. 37th IEEE Int. Conf. on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 2017, pp. 372–382.

17. M. Cheng, J. Li, and S. Nazarian, “DRL-Cloud: Deep reinforcement learning-based resource provisioning and task scheduling for cloud service providers,” in Proc. 23rd Asia and South Pacific Design Automation Conf. (ASP-DAC), Jeju, South Korea, 2018, pp. 129–134.
18. V. Mnih et al., “Asynchronous methods for deep reinforcement learning,” in Proc. 33rd Int. Conf. on Machine Learning (ICML), New York, NY, USA, 2016, pp. 1928–1937.
19. T. P. Lillicrap et al., “Continuous control with deep reinforcement learning,” in Proc. Int. Conf. on Learning Representations (ICLR), San Juan, Puerto Rico, 2016.
20. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” arXiv preprint arXiv:1707.06347, 2017.
21. F. Rossi, M. Nardelli, and V. Cardellini, “Horizontal and vertical scaling of container-based applications using reinforcement learning,” in Proc. IEEE 12th Int. Conf. on Cloud Computing (CLOUD), Milan, Italy, 2019, pp. 329–338.
22. H. Mao, M. Schwarzkopf, S. B. Venkatakrishnan, Z. Meng, and M. Alizadeh, “Learning scheduling algorithms for data processing clusters,” in Proc. ACM SIGCOMM Conf., Beijing, China, 2019, pp. 270–288.
23. H. Qiu, S. S. Banerjee, S. Jha, Z. T. Kalbarczyk, and R. K. Iyer, “FIRM: An intelligent fine-grained resource management framework for SLO-oriented microservices,” in Proc. 14th USENIX Symp. on Operating Systems Design and Implementation (OSDI), 2020, pp. 805–825.
24. J. Wang, L. Zhao, J. Liu, and N. Kato, “Smart resource allocation for mobile edge computing: A deep reinforcement learning approach,” IEEE Transactions on Emerging Topics in Computing, vol. 9, no. 3, pp. 1529–1541, 2021.
25. S. Kardani-Moghaddam, R. Buyya, and K. Ramamohanarao, “ADRL: A hybrid anomaly-aware deep reinforcement learning-based resource scaling in clouds,” IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 3, pp. 514–526, 2021.
26. Y. Zhang, J. Yao, and H. Guan, “Intelligent cloud resource management with deep reinforcement learning,” IEEE Cloud Computing, vol. 4, no. 6, pp. 60–69, 2017.
27. C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” in Proc. 34th Int. Conf. on Machine Learning (ICML), Sydney, Australia, 2017, pp. 1126–1135.
28. Y. Gari, D. A. Monge, E. Pacini, C. Mateos, and C. Garcia Garino, “Reinforcement learning-based application autoscaling in the cloud: A survey,” Engineering Applications of Artificial Intelligence, vol. 102, art. no. 104288, 2021.
29. B. Kitchenham and S. Charters, “Guidelines for performing systematic literature reviews in software engineering,” Keele Univ. and Univ. of Durham, EBSE Tech. Rep. EBSE-2007-01, 2007.
30. J. P. T. Higgins and S. G. Thompson, “Quantifying heterogeneity in a meta-analysis,” Statistics in Medicine, vol. 21, no. 11, pp. 1539–1558, 2002.