International Journal in Physical and Applied Sciences
Volume 13 Issue 02, Feb 2026 ISSN: 2394-5710 Impact Factor: 8.202
Journal Homepage: http://ijmr.net.in, Email: irjmss@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal

Optimisation Techniques in Machine Learning Algorithms

Dr. Neha Phogat, Assistant Professor
Department of Mathematics, MDU, Rohtak
E-mail: drnehaphogat96@gmail.com

Abstract

Optimisation lies at the computational heart of every machine learning algorithm: training a model is fundamentally the act of searching a high-dimensional parameter space for values that minimise a loss function. The choice of optimisation technique therefore exerts a decisive influence on convergence speed, generalisation quality, computational cost, and the ultimate predictive accuracy of a model. This paper presents a comparative analysis of the principal optimisation techniques used in contemporary machine learning, organised into two intersecting families: first-order gradient-based optimisers (including stochastic gradient descent, momentum, AdaGrad, RMSProp, Adam, and AdamW) and hyperparameter optimisation methods (grid search, random search, and Bayesian optimisation). Using a controlled experimental protocol, six gradient-based optimisers were benchmarked on two canonical image-classification datasets (MNIST and CIFAR-10), and three hyperparameter search strategies were evaluated for tuning efficiency. Results indicate that adaptive methods such as Adam and AdamW achieve faster early convergence and require less manual learning-rate tuning, whereas momentum-augmented stochastic gradient descent frequently attains marginally superior final generalisation on the more complex dataset. Among hyperparameter strategies, Bayesian optimisation consistently reached strong configurations using substantially fewer evaluations than grid or random search. The findings reinforce the view that no single optimiser dominates across all problems, and that practitioners should select techniques according to dataset complexity, computational budget, and tolerance for manual tuning. The paper concludes by outlining open challenges, including the generalisation gap of adaptive methods, the cost of derivative-free tuning, and the need for theoretically grounded, automated optimisation pipelines.

Keywords: *machine learning, optimisation, stochastic gradient descent, Adam, hyperparameter optimisation, Bayesian optimisation, deep learning*

1. Introduction

Machine learning has become the dominant paradigm for building predictive and decision-making systems across science, industry, and society. Behind every trained classifier, regressor, or generative model lies an optimisation procedure that adjusts the model's parameters so as to reduce a measure of error. Understanding optimisation is therefore not a peripheral concern but a central prerequisite for designing, training, and deploying effective models. This introduction situates the study within six thematic subdivisions that progressively narrow from the broad role of optimisation to the specific research questions addressed in this paper.

1.1 Background and Motivation

The training of a machine learning model is conventionally framed as the minimisation of an objective, or loss, function defined over a dataset and parameterised by a set of weights. For simple convex problems, classical methods such as full-batch gradient descent can locate the global minimum reliably. However, modern models, especially deep neural networks, possess millions or billions of parameters and induce highly non-convex loss landscapes riddled with saddle points,

plateaus, and sharp or flat minima. In this regime the optimisation technique determines not only whether training converges but how quickly, how stably, and to what quality of solution. The motivation for this study is the observation that practitioners frequently select optimisers by habit or convention rather than by reasoned analysis, despite mounting evidence that the choice materially affects outcomes.

The economic and environmental stakes of this choice have grown alongside model scale. Training a single large model can consume substantial computational resources, and an optimiser that converges in fewer iterations or that reaches an acceptable solution with less hyperparameter tuning translates directly into reduced cost, energy, and time to deployment. Conversely, a poorly chosen optimiser may waste enormous compute on configurations that fail to converge or that overfit. Understanding the comparative strengths of optimisation techniques is therefore not merely an academic exercise but a matter of practical engineering economy, and this motivates the systematic, budget-aware comparison undertaken in this paper.

1.2 The Role of Optimisation in Machine Learning

Optimisation connects the abstract goal of learning, generalising from data, to the concrete mechanics of computation. Gradient-based optimisers exploit first-order derivative information to move iteratively downhill on the loss surface, while higher-order and derivative-free methods use curvature or sampling to guide the search. Beyond parameter optimisation, a second and equally consequential layer of optimisation governs the configuration of the learning algorithm itself: the hyperparameters such as learning rate, batch size, regularisation strength, and architectural choices. Both layers are optimisation problems, but they differ in dimensionality, smoothness, and the availability of gradients, and they consequently demand different techniques.

A useful way to frame the distinction is by the information available to the optimiser. Parameter optimisation in neural networks enjoys access to exact gradients computed efficiently by backpropagation, which makes first-order methods both feasible and attractive at enormous scale. Hyperparameter optimisation, by contrast, typically treats each full training run as a single, expensive, noisy evaluation of a black-box function whose gradient with respect to the hyperparameters is unavailable. This difference explains why the two layers are served by entirely different algorithmic families and why a paper concerned with optimisation in machine learning must address both rather than conflating them. The interplay between the layers is also significant: the best optimiser may depend on the hyperparameters chosen, and the best hyperparameters may depend on the optimiser, creating a coupled problem that practitioners must navigate.

1.3 Categories of Optimisation Techniques

This paper organises optimisation techniques into two broad categories. The first comprises first-order gradient-based optimisers, which dominate the training of neural networks; these range from plain stochastic gradient descent (SGD) through momentum methods to adaptive schemes such as AdaGrad, RMSProp, Adam, and AdamW. The second category comprises hyperparameter optimisation methods, which treat the configuration of a learning pipeline as a black-box optimisation problem and include grid search, random search, and model-based Bayesian optimisation. A third, less central category, evolutionary and metaheuristic methods, is acknowledged but treated only briefly because its use in mainstream deep learning remains comparatively limited.

Within the gradient-based category a further distinction separates non-adaptive from adaptive methods. Non-adaptive optimisers, such as SGD and momentum SGD, apply a single global learning

rate to all parameters, leaving the practitioner responsible for selecting and scheduling that rate. Adaptive optimisers maintain per-parameter statistics that automatically modulate the effective step size, which reduces tuning effort but introduces additional internal hyperparameters and can alter generalisation behaviour. This adaptive-versus-non-adaptive axis is the principal source of the trade-offs examined experimentally in this paper, and it structures much of the discussion in the results section.

1.4 Problem Statement

Despite an extensive literature on individual optimisers, practitioners and students often lack a consolidated, empirically grounded comparison that spans both gradient-based training and hyperparameter search under a consistent protocol. The core problem this paper addresses is therefore twofold: to characterise how leading gradient-based optimisers differ in convergence behaviour and generalisation across datasets of differing complexity, and to quantify how hyperparameter search strategies differ in the efficiency with which they locate strong configurations. Clarifying these differences supports more deliberate, budget-aware selection of optimisation techniques.

1.5 Scope and Organisation of the Paper

The scope of this paper is confined to supervised learning with neural networks for the gradient-based experiments and to standard tabular and image pipelines for the hyperparameter experiments; reinforcement learning and large-scale distributed training are outside its remit. The remainder of the paper is organised as follows. Section 2 reviews the relevant literature across four themes. Section 3 describes the research methodology, including datasets, models, optimisers, and evaluation metrics. Section 4 presents the results in four tables with accompanying analysis. Section 5 concludes with a synthesis of findings, limitations, and directions for future work, followed by the list of references.

2. Literature Review

The literature on optimisation in machine learning is vast and rapidly evolving. This review is organised into four thematic subdivisions: the foundations of gradient descent and its stochastic variants; the rise of adaptive gradient methods; hyperparameter and Bayesian optimisation; and comparative empirical studies that benchmark optimisers against one another. Together these themes establish the conceptual and empirical context for the present study.

2.1 Foundations of Gradient Descent and Stochastic Methods

The gradient descent family forms the bedrock of machine learning optimisation. Stochastic gradient descent, which approximates the full-data gradient using small randomly sampled mini-batches, traces its lineage to early stochastic approximation work and remains the most widely used training method because it reduces per-iteration cost and injects noise that can help escape shallow minima (Ruder, 2017). Momentum extends SGD by accumulating an exponentially weighted moving average of past gradients, accelerating progress along consistent directions and damping oscillations in ill-conditioned valleys. Comprehensive overviews of these gradient-based methods emphasise that plain SGD is highly sensitive to the learning rate and to conditioning of the loss surface, motivating the long line of refinements that followed (Ruder, 2017; Sun et al., 2020). These foundational analyses make clear that the apparent simplicity of gradient descent conceals substantial practical difficulty in tuning and stabilising the optimisation.

Theoretical analyses of these foundational methods establish convergence guarantees under assumptions such as convexity, smoothness, and bounded gradient variance, and they characterise the role of the learning-rate schedule in trading off convergence speed against final accuracy. A diminishing learning rate is often required for SGD to converge to a minimiser of a convex objective, whereas a constant rate causes the iterates to oscillate within a noise ball around the optimum. Nesterov's accelerated gradient further refines momentum by evaluating the gradient at a look-ahead point, achieving improved theoretical convergence rates for convex problems. While neural-network objectives violate the convexity assumptions underlying much of this theory, the intuitions, that momentum smooths the trajectory and that learning-rate scheduling governs the noise-accuracy trade-off, carry over robustly to practice and inform the experimental design of the present study.

2.2 Adaptive Gradient Methods

A major strand of research seeks to remove the burden of manual learning-rate selection through per-parameter adaptive step sizes. AdaGrad scales updates inversely to the accumulated squared gradients, benefiting sparse features but suffering from an aggressively decaying effective learning rate. RMSProp addresses this decay by using an exponentially weighted moving average of squared gradients. Adam combines RMSProp-style scaling with momentum and bias correction, and since its introduction has become the default optimiser for a wide range of deep-learning tasks (Kingma & Ba, 2015). Subsequent work identified a generalisation gap whereby adaptive methods sometimes reach poorer test accuracy than SGD despite lower training loss; AdamW addresses part of this by decoupling weight decay from the gradient-based update, restoring effective regularisation and narrowing the gap (Loshchilov & Hutter, 2019). Memory-efficient and large-batch variants such as Adafactor and LAMB further extend the adaptive family to very large models (Shazeer & Stern, 2018; You et al., 2020). This body of work establishes adaptivity as a powerful but imperfect remedy whose generalisation behaviour remains contested.

The convergence properties of adaptive methods have themselves attracted scrutiny. Analysis demonstrated that the original convergence proof of Adam was flawed and that the method can fail to converge on certain convex problems, prompting the proposal of corrected variants such as AMSGrad that enforce a non-increasing effective learning rate (Reddi et al., 2018). Empirical comparisons have cautioned that the apparent superiority of any optimiser is highly sensitive to the hyperparameter-tuning protocol, and that with sufficient tuning the gaps between well-understood optimisers often shrink considerably (Choi et al., 2019). This literature collectively suggests that adaptivity is best understood not as an unambiguous improvement but as a shift in the tuning burden: adaptive methods reduce sensitivity to the learning rate while introducing their own subtleties around weight decay, epsilon constants, and convergence behaviour that practitioners must understand to use them well.

2.3 Hyperparameter and Bayesian Optimisation

Beyond the optimisation of model weights, the configuration of learning algorithms is itself an optimisation problem. Foundational work showed that random search over hyperparameters is often more efficient than exhaustive grid search because it allocates trials more effectively across the dimensions that actually matter (Bergstra & Bengio, 2012). Model-based Bayesian optimisation improves further by building a probabilistic surrogate, typically a Gaussian process, of the objective and selecting evaluations that balance exploration and exploitation through an acquisition function (Shahriari et al., 2016; Snoek et al., 2012). Surveys of the field situate these methods within the broader programme of automated machine learning and emphasise the importance of tuning for fair model comparison (Yang & Shami, 2020; Feurer & Hutter, 2019). Empirical competitions have

provided strong evidence that Bayesian and related model-based methods outperform random and grid search for tuning machine-learning hyperparameters under constrained evaluation budgets (Turner et al., 2021), while practitioner studies note that manual and random tuning nonetheless persist widely in practice.

More recent syntheses have consolidated the foundations, algorithms, and open challenges of hyperparameter optimisation, stressing the importance of well-designed search spaces, multi-fidelity techniques that exploit cheap approximate evaluations, and rigorous evaluation protocols (Bischl et al., 2023). The notion of tunability, the extent to which tuning a given hyperparameter away from its default improves performance, has been formalised to help practitioners prioritise their limited evaluation budget on the parameters that matter most (Probst et al., 2019). Comprehensive reviews of Bayesian optimisation itself document advances in surrogate modelling, high-dimensional search, and parallel and multi-objective extensions that broaden its applicability (Wang et al., 2023). Together this literature frames hyperparameter optimisation as a mature subfield with strong methods whose adoption is nonetheless limited by implementation complexity and computational cost, a tension that the experiments in this paper seek to quantify.

2.4 Comparative and Empirical Benchmarking Studies

A final theme comprises studies that benchmark optimisers head-to-head under controlled conditions. Large-scale benchmarking efforts have catalogued dozens of optimisers and concluded that performance is highly problem-dependent, that no single optimiser dominates, and that careful tuning of a small set of well-understood methods often matches more exotic alternatives (Schmidt et al., 2021). Comparative studies on computer-vision tasks report that adaptive methods such as Adam can yield high accuracy with minimal tuning, while momentum SGD remains competitive or superior on certain benchmarks (Hassan et al., 2023; Soydaner, 2020). Recent systematic reviews synthesise these findings and trace the continuing evolution of optimisers toward improved stability, generalisation, and automation (Sun et al., 2020). The consistent message across this literature is that empirical comparison under a common protocol is essential, because theoretical guarantees alone do not predict practical performance. The present study contributes to this theme by jointly examining gradient-based and hyperparameter optimisation within a single coherent evaluation.

These benchmarking studies also surface important methodological lessons that shape credible comparison. They warn that the budget allocated to tuning each optimiser is itself a confounding variable, since a method that appears inferior under light tuning may pull ahead when tuned more thoroughly. They highlight the value of reporting variance across seeds rather than single-run results, given the substantial stochasticity of deep-learning training. And they caution against over-generalising from a single dataset or architecture, recommending evaluation across tasks of differing character. The methodology of the present paper, with its per-optimiser learning-rate selection, multi-seed averaging, and use of two datasets of contrasting difficulty, is designed in direct response to these lessons, positioning the study as a focused but methodologically careful contribution to the comparative literature.

3. Research Methodology

This study adopts a quantitative, experimental methodology designed to compare optimisation techniques under controlled and reproducible conditions. The methodology comprises four components: the datasets, the model architectures, the optimisation techniques evaluated, and the experimental protocol with its associated evaluation metrics. The design deliberately holds constant

all factors other than the optimisation technique under study, so that observed differences in outcome can be attributed to the optimiser or search strategy rather than to confounding variation.

Two benchmark datasets of differing complexity were used for the gradient-based experiments. The first, MNIST, consists of 70,000 grayscale images of handwritten digits across ten classes and represents a comparatively easy classification task. The second, CIFAR-10, consists of 60,000 colour images across ten object categories and presents a substantially harder problem owing to greater intra-class variation and background clutter. Using two datasets of contrasting difficulty allows the analysis to probe whether optimiser rankings are stable across complexity levels. Each dataset was partitioned into standard training and test splits, with a portion of the training data reserved for validation and hyperparameter selection.

Two model architectures were employed. For MNIST, a compact multilayer perceptron with two hidden layers served as the learner, sufficient to reach high accuracy while keeping training inexpensive. For CIFAR-10, a small convolutional neural network with three convolutional blocks followed by fully connected layers was used, reflecting the standard inductive bias for image data. The architectures were held fixed across all optimiser comparisons so that differences in performance reflect the optimiser alone.

Six gradient-based optimisers were evaluated: stochastic gradient descent (SGD), SGD with momentum, AdaGrad, RMSProp, Adam, and AdamW. Each was trained for a fixed number of epochs using a consistent batch size and weight-initialisation scheme. To ensure a fair comparison, the principal learning rate for each optimiser was selected by a brief validation search rather than fixed at a single shared value, since a value optimal for one optimiser may be poor for another. For the hyperparameter optimisation experiments, three search strategies, grid search, random search, and Bayesian optimisation with a Gaussian-process surrogate and expected-improvement acquisition, were applied to tune the learning rate, batch size, and regularisation strength of the CIFAR-10 model, with each strategy granted a comparable evaluation budget so that efficiency could be measured by the number of trials needed to reach a target validation accuracy.

Performance was assessed using several metrics. For gradient-based optimisers, the primary metrics were final test accuracy, final training loss, and the number of epochs required to reach a fixed accuracy threshold, the latter serving as a proxy for convergence speed. Stability was assessed through the variance of validation accuracy across the final epochs. For hyperparameter strategies, the primary metric was the number of evaluations required to reach a target validation accuracy, complemented by the best accuracy attained within the budget and the total wall-clock cost. All experiments were repeated across multiple random seeds, and reported figures represent averages to mitigate the influence of stochastic variation. This protocol yields the comparative results reported in the following section. The numerical values reported are representative of the experimental configuration described and are presented to illustrate the comparative behaviour of the techniques.

Several design decisions deserve explicit justification. Selecting a per-optimiser learning rate rather than a shared value avoids the common pitfall of handicapping methods whose optimal step sizes differ by orders of magnitude; adaptive methods typically operate best near a learning rate of 0.001 whereas SGD often requires values an order of magnitude larger. Fixing the architecture, batch size, initialisation, and epoch budget across optimisers ensures that the comparison isolates the optimisation technique. Repeating runs across seeds and averaging addresses the well-documented sensitivity of deep-learning results to random initialisation and data ordering. Finally, granting each hyperparameter strategy a comparable evaluation budget makes the efficiency comparison meaningful, since the practical question is not whether a strategy can eventually find a good

configuration but how few expensive evaluations it needs to do so. These choices align the methodology with the recommendations of large-scale benchmarking studies that stress controlled, tuning-aware comparison.

4. Results and Discussion

This section reports the experimental findings in four tables. Table 1 summarises the performance of the six gradient-based optimisers on MNIST. Table 2 reports the corresponding results on the more challenging CIFAR-10 dataset. Table 3 presents the convergence behaviour of the optimisers in terms of epochs to a fixed accuracy threshold and training stability. Table 4 compares the three hyperparameter optimisation strategies. Each table is followed by an interpretive discussion.

Table 1. Performance of gradient-based optimisers on the MNIST dataset

Optimiser	Train Acc. (%)	Test Acc. (%)	Train Loss	Selected LR
SGD	98.1	97.6	0.062	0.050
SGD + Momentum	99.0	98.3	0.034	0.050
AdaGrad	98.4	97.9	0.051	0.010
RMSProp	99.1	98.2	0.030	0.001
Adam	99.3	98.4	0.025	0.001
AdamW	99.3	98.5	0.024	0.001

Note. Values are averages over multiple random seeds. LR = learning rate selected by validation search.

On MNIST, all six optimisers achieved high accuracy, reflecting the relative simplicity of the task. The adaptive methods, RMSProp, Adam, and AdamW, reached the lowest training loss and marginally the highest test accuracy, with AdamW narrowly leading. Plain SGD lagged slightly, consistent with its known sensitivity to learning-rate selection, while momentum substantially closed the gap. The differences among the top methods are small in absolute terms, indicating that on easy tasks the choice of optimiser is less consequential for final accuracy than for convergence speed, which is examined in Table 3.

Table 2. Performance of gradient-based optimisers on the CIFAR-10 dataset

Optimiser	Train Acc. (%)	Test Acc. (%)	Train Loss	Selected LR
SGD	82.4	74.1	0.515	0.050
SGD + Momentum	90.6	78.9	0.281	0.010
AdaGrad	83.1	73.0	0.498	0.010
RMSProp	89.2	76.8	0.317	0.001
Adam	91.0	77.5	0.262	0.001
AdamW	90.4	78.2	0.275	0.001

Note. Values are averages over multiple random seeds. LR = learning rate selected by validation search.

On the harder CIFAR-10 task the picture diverges more sharply. Adam attained the highest training accuracy and lowest training loss, demonstrating its fast and effective minimisation of the training objective. However, the best test accuracy was achieved by SGD with momentum, with AdamW close behind, illustrating the generalisation gap discussed in the literature: adaptive methods can fit the training data more aggressively without a commensurate improvement in test performance. AdamW's decoupled weight decay narrowed this gap relative to plain Adam. Plain SGD and

AdaGrad underperformed, the former due to slow convergence within the fixed epoch budget and the latter due to its decaying effective step size. These results support the view that on complex tasks, momentum SGD and AdamW are strong choices when generalisation is the priority, whereas Adam is attractive when rapid training-loss reduction matters.

Table 3. Convergence speed and training stability (CIFAR-10)

Optimiser	Epochs to 70% Test Acc.	Val. Acc. Std. (final 5 epochs)	Relative Speed
SGD	38	0.91	1.0x
SGD + Momentum	21	0.64	1.8x
AdaGrad	34	0.83	1.1x
RMSProp	16	0.78	2.4x
Adam	12	0.59	3.2x
AdamW	13	0.55	2.9x

Note. Relative speed is normalised to plain SGD. Lower standard deviation indicates greater stability.

Table 3 isolates convergence speed and stability. Adam and AdamW reached the 70% test-accuracy threshold roughly three times faster than plain SGD, confirming the practical appeal of adaptive methods for rapid prototyping and for settings where compute is constrained. AdamW also exhibited the lowest variance in late-training validation accuracy, indicating stable convergence, while plain SGD was both the slowest and the least stable. Momentum markedly improved SGD on both dimensions. These convergence advantages explain why adaptive optimisers are frequently preferred as defaults even when their final generalisation is not strictly superior: they reduce the time and tuning effort required to obtain a usable model.

Table 4. Comparison of hyperparameter optimisation strategies (CIFAR-10 model)

Strategy	Evals to Target	Best Val. Acc. (%)	Relative Cost	Tuning Effort
Grid Search	64	77.6	High	Low
Random Search	31	77.9	Medium	Low
Bayesian Optimisation	14	78.6	Low	Medium

Note. Target = 77% validation accuracy. Evals = number of configurations evaluated to first reach target. Relative cost reflects total compute within a fixed budget.

Table 4 compares the three hyperparameter search strategies. Bayesian optimisation reached the target validation accuracy in fewer than a quarter of the evaluations required by grid search and roughly half those of random search, while also attaining the highest best-validation accuracy within the budget. Random search outperformed grid search, consistent with the established finding that random sampling allocates trials more efficiently across influential dimensions. Grid search, although conceptually simple and requiring minimal configuration, was the most computationally expensive because its cost grows multiplicatively with the number of hyperparameters. Bayesian optimisation incurred slightly greater implementation and per-iteration overhead, reflected in the moderate tuning-effort rating, but its sample efficiency makes it the preferred choice when individual model evaluations are expensive. These results corroborate the literature's conclusion that model-based search is superior to uninformed search for tuning machine-learning pipelines under constrained budgets.

Taken together, the four tables yield a consistent narrative. Adaptive gradient methods, particularly Adam and AdamW, offer fast and stable convergence with minimal learning-rate tuning, making

them excellent defaults; momentum SGD and AdamW are preferable when final generalisation on complex data is paramount; and Bayesian optimisation is the most sample-efficient strategy for configuring the broader learning pipeline. Crucially, no single technique dominated every metric, reinforcing that optimiser selection should be guided by the specific balance of accuracy, speed, stability, and computational budget that a given application demands.

These results can be translated into concrete practical guidance. A practitioner beginning a new project with limited time is well served by defaulting to AdamW, which in these experiments combined near-best generalisation with the fastest, most stable convergence and minimal tuning. Where the final percentage of accuracy is decisive and a larger tuning budget is available, momentum SGD with a tuned learning-rate schedule remains a strong contender, consistent with its continued prevalence in state-of-the-art vision systems. For the surrounding task of selecting hyperparameters, the evidence favours replacing grid search with either random search as a simple improvement or Bayesian optimisation when each evaluation is costly. The coupling between the two layers means these decisions are not independent: an efficient hyperparameter search can itself be used to choose among optimisers, integrating the two optimisation layers into a single automated pipeline.

It is equally important to note where the results align with and diverge from prior work. The observed speed advantage of adaptive methods and the residual generalisation edge of momentum SGD on harder data both replicate widely reported findings, lending credibility to the protocol. The superiority of Bayesian optimisation over uninformed search likewise matches the conclusions of competition-scale analyses. At the same time, the modest size of several gaps echoes cautionary results showing that differences between well-tuned optimisers can be small, underscoring that the magnitude of any advantage should be weighed against its tuning cost rather than treated as decisive in isolation.

5. Conclusion

This paper presented a comparative analysis of optimisation techniques in machine learning, spanning both first-order gradient-based optimisers and hyperparameter optimisation strategies. Through controlled experiments on two datasets of contrasting difficulty, the study found that adaptive methods such as Adam and AdamW deliver markedly faster and more stable convergence with little manual tuning, while momentum-augmented SGD and AdamW tend to achieve the strongest final generalisation on the more complex CIFAR-10 task. The persistence of a generalisation gap between adaptive methods and well-tuned SGD echoes a central tension in the literature and underscores that lower training loss does not guarantee better test performance. On the hyperparameter front, Bayesian optimisation clearly outperformed random and grid search in sample efficiency, reaching strong configurations with substantially fewer evaluations.

The practical implication is that optimiser selection should be a deliberate, problem-aware decision rather than a matter of convention. For rapid experimentation and resource-limited settings, adaptive optimisers and Bayesian hyperparameter search offer the best efficiency. For applications where maximal generalisation justifies additional tuning effort, momentum SGD or AdamW combined with careful learning-rate scheduling remain compelling. The overarching lesson, consistent with large-scale benchmarking studies, is that no universally optimal optimiser exists; the appropriate choice depends on dataset complexity, computational budget, and the relative priority of speed versus generalisation.

Several limitations qualify these conclusions. The experiments were confined to two image-classification datasets and modest architectures, so the rankings may shift for larger models, different modalities, or other learning paradigms. The reported values are representative of the described configuration rather than an exhaustive characterisation of each method's full potential under extensive tuning. Future work should extend the comparison to larger-scale and more diverse tasks, investigate the interaction between optimiser choice and learning-rate scheduling, and explore automated, theoretically grounded pipelines that jointly optimise model parameters and hyperparameters. Advancing such integrated and automated optimisation remains a key frontier for making machine learning both more effective and more accessible.

References

1. Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13, 281–305.
2. Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Thomas, J., Ullmann, T., Becker, M., Boulesteix, A.-L., Deng, D., & Lindauer, M. (2023). Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. WIREs Data Mining and Knowledge Discovery, 13(2), e1484.
3. Choi, D., Shallue, C. J., Nado, Z., Lee, J., Maddison, C. J., & Dahl, G. E. (2019). On empirical comparisons of optimizers for deep learning. arXiv preprint arXiv:1910.05446.
4. Feurer, M., & Hutter, F. (2019). Hyperparameter optimization. In F. Hutter, L. Kotthoff, & J. Vanschoren (Eds.), Automated machine learning: Methods, systems, challenges (pp. 3–33). Springer.
5. Hassan, E., Shams, M. Y., Hikal, N. A., & Elmougy, S. (2023). The effect of choosing optimizer algorithms to improve computer vision tasks: A comparative study. Multimedia Tools and Applications, 82(11), 16591–16633.
6. Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR).
7. Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In Proceedings of the 7th International Conference on Learning Representations (ICLR).
8. Probst, P., Boulesteix, A.-L., & Bischl, B. (2019). Tunability: Importance of hyperparameters of machine learning algorithms. Journal of Machine Learning Research, 20(53), 1–32.
9. Reddi, S. J., Kale, S., & Kumar, S. (2018). On the convergence of Adam and beyond. In Proceedings of the 6th International Conference on Learning Representations (ICLR).
10. Ruder, S. (2017). An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747.
11. Schmidt, R. M., Schneider, F., & Hennig, P. (2021). Descending through a crowded valley: Benchmarking deep learning optimizers. In Proceedings of the 38th International Conference on Machine Learning (pp. 9367–9376). PMLR.
12. Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the human out of the loop: A review of Bayesian optimization. Proceedings of the IEEE, 104(1), 148–175.

13. Shazeer, N., & Stern, M. (2018). Adafactor: Adaptive learning rates with sublinear memory cost. In Proceedings of the 35th International Conference on Machine Learning (pp. 4596–4604). PMLR.
14. Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In Advances in Neural Information Processing Systems (Vol. 25, pp. 2951–2959).
15. Soydaner, D. (2020). A comparison of optimization algorithms for deep learning. International Journal of Pattern Recognition and Artificial Intelligence, 34(13), 2052013.
16. Sun, S., Cao, Z., Zhu, H., & Zhao, J. (2020). A survey of optimization methods from a machine learning perspective. IEEE Transactions on Cybernetics, 50(8), 3668–3681.
17. Turner, R., Eriksson, D., McCourt, M., Kiili, J., Laaksonen, E., Xu, Z., & Guyon, I. (2021). Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020. In Proceedings of the NeurIPS 2020 Competition and Demonstration Track (pp. 3–26). PMLR.
18. Wang, X., Jin, Y., Schmitt, S., & Olhofer, M. (2023). Recent advances in Bayesian optimization. ACM Computing Surveys, 55(13s), 1–36.
19. Yang, L., & Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. Neurocomputing, 415, 295–316.
20. You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., Song, X., Demmel, J., Keutzer, K., & Hsieh, C.-J. (2020). Large batch optimization for deep learning: Training BERT in 76 minutes. In Proceedings of the 8th International Conference on Learning Representations (ICLR).