Journal of Future Artificial Intelligence and Technologies

E-ISSN: 3048-3719

Research Article

Revisiting Multilayer Perceptrons in the Era of Kolmogorov–Arnold Networks: An Empirical Comparison on Image Classification

Munyao Muia [1,*], Geoffry Mutiso [2], and Stephen Kahara [1]

[1] Department of Computer Science, Murang'a University of Technology, Murang'a 10200, Kenya; e-mail : munyao@mut.ac.ke; skahara@mut.ac.ke
[2] Department of Information Technology, Murang'a University of Technology, Murang'a 10200, Kenya; e-mail : gmutiso@mut.ac.ke
* Corresponding Author: Munyao Muia

Abstract: Multilayer perceptrons (MLPs) remain a widely used architecture in deep learning, whereas Kolmogorov–Arnold networks (KANs) have emerged as an alternative that represents nonlinear transformations through learnable univariate functions, commonly parameterized using splines. Despite growing interest in KANs, their performance relative to MLPs on high-dimensional image classification tasks remains insufficiently characterized, partly because differences in model capacity and training protocols complicate direct comparisons. This study presents a parameter-controlled empirical comparison of MLPs and KANs on CIFAR-10 across five capacity tiers ranging from 64K to 4M parameters. The model configurations are matched to within 2% in parameter count, while the training protocol is held consistent to facilitate an assessment of architectural differences across capacity levels. The results show that KANs achieve higher mean classification accuracy than MLPs at low-to-medium capacities (T_1–T_4), with gains of 0.3–1.1 percentage points that decrease as model capacity increases. At the highest capacity (T_5), MLPs marginally outperform KANs by 0.3 percentage points. However, MLPs consistently deliver 3.0–5.3 times higher accuracy per GPU-hour, supported by a 3.1–5.4-fold training-throughput advantage and 18–27 fewer convergence epochs. Per-class results further reveal differences in the categories where each architecture performs relatively well, with MLPs showing marginal advantages in classes with lower intra-class variability and KANs exhibiting gains in visually complex, higher-variability classes. Overall, the findings suggest that the relative advantages of MLPs and KANs depend on model capacity and computational constraints: KANs offer modest accuracy gains at lower capacities, whereas MLPs provide greater computational efficiency across all evaluated tiers. These results support a complementary assessment of the two architectures under the specific experimental conditions considered.

Keywords: CIFAR-10; Computational Efficiency; Image Classification; Kolmogorov–Arnold Networks; Multilayer Perceptrons; Neural Network Architecture; Parameter Matching; Performance Evaluation.

Received: September, 6th 2026
Revised: October, 22nd 2026
Accepted: October, 24th 2026
Published: November, 1st 2026
Curr. Ver.: November, 1st 2026

Copyright: © 2026 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY SA) license (https://creativecommons.org/licenses/by-sa/4.0/)

1. Introduction

The multilayer perceptron (MLP) has served as a foundational building block of deep learning for decades. Comprising successive affine transformations interspersed with fixed nonlinear activation functions, MLPs derive their expressive power from the Universal Approximation Theorem, which establishes that a sufficiently wide network can approximate any continuous function on a compact domain to arbitrary precision [1]. Despite this theoretical foundation, the use of fixed activation functions and learned scalar weights imposes particular inductive biases that may limit approximation efficiency for certain classes of functions [2]. Kolmogorov–Arnold Networks (KANs), introduced in [3], offer an alternative

architectural formulation motivated by the Kolmogorov–Arnold representation theorem, which expresses multivariate continuous functions through compositions of univariate functions. Unlike conventional MLPs, KANs place learnable univariate functions along network edges rather than relying exclusively on fixed nonlinear activation functions at neurons [4]. In practical implementations, these edge functions are commonly parameterized using B-spline bases, enabling the network to adapt nonlinear transformations during training.

This architectural distinction has attracted interest in KANs as an alternative to conventional MLPs, particularly in tasks where the underlying relationships exhibit structured or compositional properties [5]. Reported applications, including symbolic regression and partial differential equation solving, suggest that learnable edge functions may be advantageous when the target relationships possess exploitable structure [6]. However, these findings do not necessarily establish a general performance advantage for KANs across different learning tasks. Direct comparisons with MLPs can be difficult to interpret when model sizes, training budgets, and experimental protocols differ, since observed performance gains may reflect differences in model capacity or training conditions rather than architectural characteristics alone. The extent to which KANs retain their advantages on conventional image-classification benchmarks therefore remains an important empirical question. Image data present a particularly relevant setting because they combine high-dimensional inputs with spatial dependencies that may not be explicitly exploited by fully connected architectures. Although the Kolmogorov–Arnold representation theorem provides a theoretical foundation for representing multivariate functions through univariate components, this representation alone does not guarantee superior empirical performance under practical architectural and computational constraints [7]. Accordingly, an important question is whether the representational flexibility of learnable edge functions translates into measurable accuracy gains when MLPs and KANs are compared at similar parameter budgets and under consistent training conditions.

A controlled comparison can help clarify this question by separating differences associated with architectural design from those arising from model capacity and experimental configuration. In particular, KANs with the same layer dimensions as MLPs can contain substantially more parameters because of their learnable spline coefficients. Comparing such models without accounting for parameter count may therefore confound architectural effects with differences in representational capacity. A parameter-matched comparison offers a more informative basis for examining how the relative performance of the two architectures changes as model capacity increases. It also enables accuracy to be considered alongside computational efficiency, which is important because improvements in predictive performance may entail additional training costs. In this context, we hypothesize that the relative advantage of KANs over MLPs depends on model capacity and computational constraints, rather than remaining uniform across capacity levels. More specifically, the flexibility of learnable edge functions may offer accuracy benefits at lower parameter budgets, while the computational overhead associated with spline-based transformations may limit their practical advantage as model capacity increases. This hypothesis motivates an empirical assessment of both predictive performance and computational cost under a consistent experimental framework.

To investigate this hypothesis, the present study compares fully connected MLP and KAN architectures on CIFAR-10 across five parameter-capacity tiers, ranging from 64K to 4M parameters. The experimental design pairs the two architectures with parameter counts matched to within 2% of each other and maintains a consistent training protocol to facilitate comparisons across capacity levels. CIFAR-10 serves as a controlled testbed for examining relative architectural behavior rather than as a means of pursuing state-of-the-art image-classification accuracy. The images are flattened into 3,072-dimensional vectors, and the evaluated models do not explicitly preserve spatial structure through convolutional or other vision-specific operations. Consequently, the findings are intended to characterize the behavior of these fully connected architectures under the specified conditions, rather than to establish conclusions that automatically extend to spatially structured vision models or image-classification tasks more broadly.

Within this framework, the study examines three related aspects of the MLP–KAN comparison: classification accuracy across capacity levels, computational efficiency during training, and differences in class-level performance. Parameter Efficiency Score (PES) and Accuracy-per-Compute (APC) are also used to examine the relationship between predictive performance and resource requirements. By considering these measures together, the study seeks to clarify whether accuracy gains observed at particular capacity levels remain practically

meaningful when computational costs are taken into account. The resulting analysis provides an empirical basis for considering architecture selection under different parameter-budget and throughput constraints, while keeping the conclusions within the scope of the evaluated configurations.

The remainder of this paper is organized as follows. Section 2 reviews related work on MLPs, KANs, and empirical comparisons between neural network architectures, establishing the context for the present study. Section 3 describes the experimental methodology, including the model configurations, parameter-matching procedure, training protocol, evaluation metrics, and experimental settings. Section 4 presents the experimental results, covering classification performance, computational efficiency, class-level behavior, and the supplementary analyses. Section 5 discusses the findings in relation to the study's hypothesis, examines the implications of the observed accuracy–efficiency trade-offs, and considers the limitations of the experimental design. Finally, Section 6 summarizes the main conclusions and outlines directions for future research.

2. Related Work

Research on multilayer perceptrons (MLPs) has established a strong theoretical foundation for understanding the approximation capabilities of feedforward neural networks. Following the seminal work of Cybenko and Hornik, subsequent studies have examined how network depth, width, activation functions, and parameterization influence representational efficiency [8]. Barron's approximation theorem demonstrated that shallow neural networks can efficiently approximate broad classes of functions, whereas Telgarsky showed that increased depth can provide substantial gains in representational efficiency under suitable conditions [9]. More recent research has investigated overparameterized MLPs and the implicit regularization induced by gradient-based optimization [10]. Together, these studies provide a foundation for evaluating alternative neural architectures and understanding how their representational properties relate to model capacity and optimization.

Kolmogorov–Arnold Networks (KANs) introduce an alternative approach to neural function approximation by replacing conventional scalar edge weights and fixed node activation functions with learnable univariate functions associated with network edges [3]. The original study demonstrated their potential in symbolic regression and scientific computing [6], while subsequent work has explored alternative basis functions, computationally efficient formulations, and applications to time-series analysis, differential equations, and other structured learning problems [11], [12]. Although KANs have demonstrated competitive or improved predictive performance in some settings, their benefits may entail increased computational and parameter costs. For example, the comparative evaluation in [13] reported substantially higher training and evaluation costs for KANs than for MLPs on structured datasets. These findings highlight the importance of assessing predictive performance alongside computational efficiency when comparing the two architectures.

Research on KANs for computer vision has extended this architectural approach to models that incorporate convolutional or patch-based processing [14], [15]. Unlike fully connected networks operating on flattened images, these vision-oriented variants seek to retain spatial information by integrating learnable spline functions into established architectural components. Cheon [16] evaluated KAN-based models on MNIST, CIFAR-10, and CIFAR-100, reporting better performance than MLP-Mixer on CIFAR-10 and CIFAR-100, although performance remained below that of ResNet-18. Other studies have compared KAN and MLP variants on CIFAR-10 and related function-approximation tasks, reporting differences in accuracy and parameter requirements across configurations [17]–[19]. However, comparisons across these studies remain difficult because they involve different architectures, model capacities, and experimental protocols. Table 1 summarizes representative studies, focusing on their datasets, architecture variants, parameter-matching approaches, and evaluation objectives.

The present study differs from these works in three respects. First, it examines fully connected MLP and KAN architectures operating on flattened images, without convolutional front ends, patch embeddings, or token-mixing components. This setting differs from the vision-oriented architectures in [15], [16], and the hybrid variants in [14], although the standalone models in [14] provide a closer point of comparison. Second, the study matches parameter counts rather than relying on equivalent layer dimensions. Because KAN layers

incorporate learnable spline coefficients along network edges, their parameter counts depend on the functional parameterization, including grid resolution and spline order [14]. Consequently, equivalent layer dimensions do not necessarily produce comparable model sizes. The studies summarized in Table 1 do not consistently employ strict parameter matching across architectures, making it difficult to isolate architectural differences from differences in model capacity. The present study therefore pairs MLP and KAN configurations within 2% of each other's parameter counts at each capacity tier. Third, both architectures are trained and evaluated under a common protocol, reducing the influence of differences in training conditions when interpreting their relative performance. This design also addresses the limitations of comparisons that rely on baseline results reported in earlier studies, as in [16], where numerical results may not be directly comparable across experimental settings.

Table 1. Summary of representative studies relevant to KAN–MLP comparisons.

Study	Datasets or tasks	Architecture variants	Parameter-matching approach	Evaluation focus
Mohan et al. [14]	MNIST, CIFAR-10	KAN and convolutional KAN variants	Partial matching or different width configurations	Image-classification accuracy
Bodner et al. [15]	Image benchmarks	Convolutional KANs and standard CNNs/MLPs	Capacity constraints not strictly matched	Spatial feature extraction
Cheon [16]	MNIST, CIFAR-10, CIFAR-100	KAN-Mixer, MLP-Mixer, ResNet	Approximate layer-dimension equivalence	Classification performance
Gaonkar et al. [17]	CIFAR-10	KAN and MLP	Varying capacity configurations	Accuracy versus parameter size
Shukla et al. [18]	Differential equations and operator learning	Vanilla KAN and standard MLP	Parameter-matched comparison following FAIR principles	Function approximation and operator learning
Zeng et al. [19]	Synthetic and irregular functions	KAN and MLP	Controlled parameter budgets	Noise robustness and fitting accuracy

Taken together, existing research suggests that the relative advantages of KANs and MLPs depend on the task, architecture, model capacity, and computational requirements. The contribution of the present study is not the first application of KANs to CIFAR-10 or the first comparison between KANs and MLPs, both of which have been investigated previously [14], [16]. Rather, it provides a parameter-controlled comparison of fully connected architectures across five capacity tiers under a common training and evaluation protocol. By examining classification accuracy alongside convergence behavior, computational efficiency, per-class performance, Parameter Efficiency Score (PES), and Accuracy-per-Compute (APC), the study investigates whether the relative performance of the two architectures changes as model capacity increases. This approach provides a more systematic view of the accuracy–efficiency trade-off than comparisons based on a single model configuration, while keeping the conclusions within the scope of fully connected networks operating on flattened CIFAR-10 images.

3. Methodology

This section describes the experimental methodology used to compare multilayer perceptron (MLP) and Kolmogorov–Arnold network (KAN) architectures under controlled conditions. It outlines the architectural configurations, parameter-matching procedure, data preparation, training protocol, computational environment, and evaluation metrics. The overall workflow, from input preparation and model configuration to training and performance evaluation, is illustrated in Figure 1.

3.1. Architectural Definitions

A standard MLP with L hidden layers maps an input vector $\mathbf{x} \in \mathbb{R}^{d_0}$ through successive affine transformations and fixed nonlinearities. For layer $l = 1, \dots, L$, the hidden representation is given by:

$$\mathrm{h}^{(l)} = \sigma\left(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\right), \qquad l = 1, \dots, L \tag{1}$$

where $\mathbf{W}^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$ is the weight matrix, $\mathbf{b}^{(l)} \in \mathbb{R}^{d_l}$ is the bias vector, and $\sigma(\cdot)$ is the ReLU activation function. The final layer produces $d_{L+1} = 10$ class logits, followed by softmax for probability estimation.

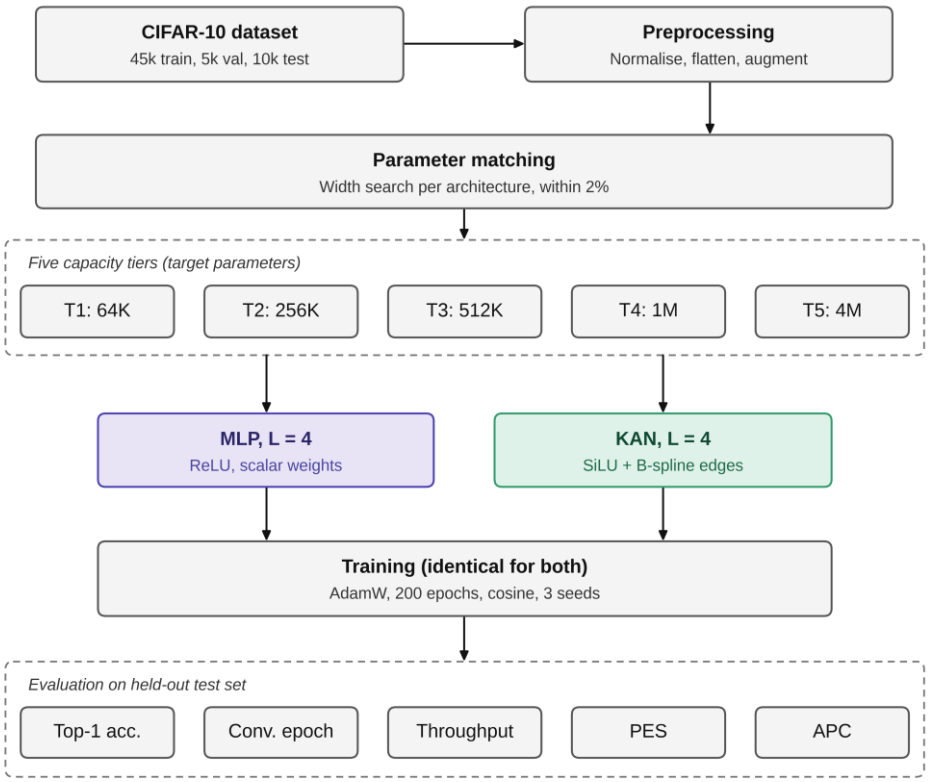


Figure 1. Overview of the experimental workflow.

The total number of trainable MLP parameters is therefore:

$$P_{\text{MLP}} = \sum_{l=1}^{L+1} (d_{l-1} + 1)\, d_l \tag{2}$$

In contrast, a KAN replaces scalar edge weights with learnable univariate functions. For a KAN layer mapping $\mathbb{R}^n$ to $\mathbb{R}^m$, the j-th output is computed as:

$$\text{KAN}_l(\text{x})_j = \sum_{i=1}^{n} \phi_{j,i}\,(x_i), \qquad j = 1, \ldots, m \tag{3}$$

where $\phi_{j,i}\colon \mathbb{R} \to \mathbb{R}$ is the learnable function associated with the connection from input i to output j. In our experimental formulation, each edge function combines a fixed SiLU base function with a learnable B-spline expansion.

$$\phi(x) = b(x) + \sum_{r=1}^{G+k} c_r\, B_r^{(k)}(x), \qquad b(x) = \text{SiLU}(x) \tag{4}$$

where $B_r^{(k)}(x)$ denotes the r-th B-spline basis function of order k, c_r are learnable spline coefficients, G is the number of grid intervals, and k is the spline order.

In our experimental implementation, the base activation is fixed, and the optional learnable scaling components are disabled, such that only the spline coefficients are trainable edge parameters. Thus, a KAN layer with n inputs, m outputs, spline order k, and G grid intervals contains $P_{\text{KAN},l} = nm(G + k)$ trainable spline parameters. With cubic splines ($k = 3$)

and five grid intervals ($G = 5$), this gives eight trainable parameters per edge. This parameterization is used consistently across all KAN capacity tiers to ensure that the subsequent parameter-matching protocol compares the two architectures on an equivalent total parameter budget

3.2. Parameter-Matching Protocol

Five target parameter budgets are considered to isolate architectural effects from differences in model capacity: $P_{\text{target}} \in \{64\text{K}, 256\text{K}, 512\text{K}, 1\text{M}, 4\text{M}\}$. Both architectures use L=4 hidden layers, receive the same 3,072-dimensional flattened CIFAR-10 image 32×32×3 pixels) as input, and produce ten output logits, one for each class. For each target budget, candidate hidden widths are evaluated using the parameter-count formulations defined in Section 3.1, and the configuration with the smallest deviation from the target is selected for each architecture. For a given target P_{target}, the selected configuration minimizes the absolute parameter deviation:

$$\boldsymbol{d_A^* = \arg\min_{d \in \mathbb{N}} \left| P_A(d) - P_{\text{target}} \right|, \qquad A \in \{MLP, KAN\}} \tag{5}$$

where $P_A(d)$ denotes the realized number of trainable parameters for architecture A at width configuration d. The relative matching error is calculated as:

$$\boldsymbol{E_A = \frac{\left| P_A - P_{\text{target}} \right|}{P_{\text{target}}} \times 100\%} \tag{6}$$

The resulting configurations are accepted according to the 2% matching criterion and paired at each capacity tier. This procedure is intended to compare the predictive performance and computational efficiency of MLPs and KANs under approximately equivalent parameter budgets, providing a consistent basis for evaluating their architectural trade-offs.

3.3. Dataset Preprocessing

The experiments use the CIFAR-10 dataset, which comprises 60,000 color images of size 32 × 32 distributed uniformly across 10 classes [20]. CIFAR-10 was selected because of its documented difficulty for non-convolutional models, making it suitable for comparing MLP and KAN parameterizations without performance saturation masking their differences. The standard split of 50,000 training images and 10,000 test images is retained, with 5,000 images held out from the training set for validation and learning-rate scheduling. All images are normalized channel-wise to zero mean and unit variance using the standard CIFAR-10 normalization constants: μ = (0.4914, 0.4822, 0.4465) and σ = (0.2470, 0.2435, 0.2616) for the red, green, and blue channels, respectively. Each 32 × 32 × 3 image is then flattened into a 3,072-dimensional vector before being fed into either architecture. This shared input representation removes spatial inductive bias, allowing the comparison to focus on learnable univariate edge functions in KANs versus scalar weights and fixed activations in MLPs, rather than differences in convolutional structure.

The same data augmentation pipeline is applied to both architectures to ensure a controlled comparison. During training, each image is padded by 4 pixels on all sides using reflection padding, followed by a random 32 × 32 crop. Random horizontal flipping is applied with a probability of 0.5. No vertical flips, color jitter, or advanced augmentation strategies, such as AutoAugment or RandAugment, are used because the objective is to examine the architectural properties of MLPs and KANs rather than maximize absolute accuracy. During validation and testing, images are only normalized, with no augmentation applied, so that performance measurements reflect the models' generalization under consistent evaluation conditions.

3.4. Training Protocol

All configurations are trained for 200 epochs using a batch size of B=256, the AdamW optimizer, an initial learning rate of $\eta_{\max}$=10^{-3}, and a weight decay of λ=10^{-4}. Hyperparameters are selected through a coarse grid search using the T_3, with candidate learning rates of {3×10^{-4}, 10^{-3}, 3×10^{-3}} and weight decay values of {10^{-5},10^{-4},10^{-3}}. The selected settings are then fixed across all capacity tiers and both architectures to isolate architectural differences from the effects of architecture-specific hyperparameter tuning. The learning rate follows

cosine annealing from $\eta_{\max}=10^{-3}$ to $\eta_{\min}=10^{-5}$, without warm-up or early stopping, ensuring a consistent training protocol across configurations.

The models are optimized using the standard categorical cross-entropy loss:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} y_i, c \log(p_{i,c}) \tag{7}$$

where N is the number of training samples after the validation holdout, C is the number of classes, $y_{i,c}$ is the ground-truth indicator for sample i and class c, and $p_{i,c}$ is the predicted probability for that class. Gradients are computed using automatic differentiation and clipped to a global norm of 1.0 to improve training stability, particularly for the KAN spline parameters.

All experiments are conducted on a single NVIDIA A100 40 GB GPU using PyTorch 2.2.0 and CUDA 12.1. Mixed-precision training uses bfloat16 for forward and backward computations and float32 for optimizer states. The KAN models are implemented using the efficient-kan library, with B-spline evaluation performed through batched tensor operations. Each configuration is trained independently using three random seeds (42, 123, and 999), with model parameters reinitialized for each run. Results are reported as the mean and standard deviation across the three runs to characterize run-to-run variability and support descriptive comparisons of the observed performance trends.

3.5. Evaluation Metrics

The evaluation uses four primary and derived metrics. First, top-1 accuracy is measured on the CIFAR-10 test set and defined as the proportion of test images for which the predicted class, determined by the argmax of the output logits, matches the ground-truth label. Second, convergence epoch E_{conv} is defined as the first epoch at which validation accuracy reaches at least 95% of the maximum validation accuracy achieved during training:

$$E_{\text{conv}} = \min\left\{t: \text{Acc}_{\text{val}}(t) \geq 0.95 \max_{\tau} \text{Acc}_{\text{val}}(\tau)\right\} \tag{8}$$

Third, training throughput τ is measured in thousands of samples processed per second and averaged over the final 10 training epochs. This metric characterizes the practical execution speed of each architecture. Fourth, the Parameter Efficiency Score (PES) is used to assess predictive performance relative to model size and is defined as:

$$PES = \frac{\text{Acc}_{\text{top-1}}}{\log_{10}(P)} \tag{9}$$

where P denotes the number of trainable parameters. In addition, Accuracy-per-Compute (APC) is used to evaluate predictive performance relative to training cost and is defined as

$$APC = \frac{\text{Acc}_{\text{top-1}}}{T_{\text{train}}} \tag{10}$$

where T_{train} is the total measured training time in hours.

PES and APC serve as supplementary indicators of architectural efficiency, capturing complementary aspects of the accuracy–size and accuracy–compute trade-offs. PES uses a logarithmic parameter scale to facilitate comparisons across model capacities spanning different orders of magnitude, whereas APC relates predictive accuracy to actual training time. The latter is particularly relevant for architectures involving nonstandard operations, such as B-spline evaluations, whose execution costs may not be fully reflected by theoretical operation counts. These metrics complement standard evaluation measures by providing additional perspectives on the trade-offs between predictive performance, model size, and training cost in resource-constrained settings.

4. Results

This section presents the empirical results in terms of architectural configurations, predictive performance, parameter efficiency, computational efficiency, convergence behavior, and class-level performance. Targeted ablation studies are presented at the end of the section.

4.1. Architecture Configuration Summary

Table 2 summarizes the realized configurations across the five capacity tiers. The KAN-to-MLP parameter ratio ranges from 0.988 to 0.995, remaining within the predefined matching range of 0.98–1.02 at every tier. The closest parameter match occurs at T_5, where the ratio reaches 0.995. KANs use smaller hidden-layer widths because each connection contains eight trainable spline coefficients, compared with a single scalar weight in the MLP.

Table 2. Architectural configurations across the five capacity tiers.

Tier	MLP width	KAN width	MLP parameters	KAN parameters	KAN/MLP ratio
T_1	256	94	64,760	64,012	0.988
T_2	512	188	263,178	260,704	0.991
T_3	724	265	527,142	522,310	0.991
T_4	1,024	375	1,054,730	1,048,500	0.994
T_5	2,048	748	4,202,506	4,180,444	0.995

Note: All models use L=4 hidden layers. The MLP uses ReLU activations, whereas the KAN uses cubic B-splines with G=5.

Despite their closely matched parameter counts, the two architectures distribute their representational capacity differently. Each KAN connection models a learnable univariate function, whereas each MLP connection applies a scalar weight followed by a fixed activation. This distinction provides an important context for interpreting the performance results: parameter matching controls model size, but does not make the two architectures representationally equivalent.

4.2. Top-1 Accuracy and Parameter Efficiency Score

Table 3 reports test-set top-1 accuracy and the Parameter Efficiency Score (PES), with results expressed as the mean and standard deviation across three independent runs. KANs achieve higher mean accuracy at T_1–T_4, with the largest observed advantage at T_1 (+1.1 percentage points). The advantage decreases progressively as capacity increases, reaching +0.3 percentage points at T_4. At T_5, the MLP achieves a slightly higher mean accuracy, exceeding the KAN by 0.3 percentage points.

Table 3. CIFAR-10 test accuracy and Parameter Efficiency Score (PES).

Tier	Approx. parameter budget	MLP accuracy (%)	KAN accuracy (%)	Δ accuracy (pp)	PES (MLP)	PES (KAN)	Δ PES
T_1	64K	46.8 ± 0.4	47.9 ± 0.5	+1.1	9.52	9.74	+0.22
T_2	256K	51.3 ± 0.3	52.1 ± 0.4	+0.8	9.55	9.70	+0.15
T_3	512K	54.7 ± 0.3	55.3 ± 0.3	+0.6	9.76	9.87	+0.11
T_4	1M	57.2 ± 0.2	57.5 ± 0.3	+0.3	9.86	9.91	+0.05
T_5 ★	4M	60.4 ± 0.2	60.1 ± 0.2	−0.3	9.73	9.68	−0.05

Note: Accuracy is reported as mean ± standard deviation across three random seeds. Δ denotes KAN minus MLP; PES is calculated as $Acc_{top-1}/\log_{10}(P)$, where P is the number of trainable parameters. The highlighted tier (★) marks the first tested tier at which the MLP achieves higher mean accuracy than the KAN.

Several patterns are immediately apparent. The KAN advantage decreases monotonically with capacity, starting at +1.1 pp at T_1, narrowing to +0.3 pp at T_4, and reversing at T_5 (−0.3 pp). This trend is consistent with the observed results across the five capacity tiers. The standard deviations remain relatively small (0.2–0.5 pp) throughout, although the MLP advantage at T_5 (−0.3 ± 0.28 pp pooled standard deviation) warrants cautious interpretation given its small magnitude. Thus, while the decreasing trend across capacity tiers is evident, the 0.3-percentage-point margin at T_5 may be insufficient to establish a robust performance advantage for MLPs. The observed crossover should therefore be interpreted as an empirical pattern within the five tested capacity tiers rather than as a general characteristic of the two architectures.

The PES values reinforce this pattern, with KANs achieving higher parameter efficiency at every tier except T_5. The advantage peaks at T_1 (+0.22), where the benefits of per-edge spline expressiveness are most pronounced, and diminishes to a negative value at T_5. This mirrors the accuracy trend, suggesting that PES reflects capacity-dependent behavior rather than merely an effect of logarithmic normalization. The absolute accuracy ranges (46.8–60.4% for MLPs and 47.9–60.1% for KANs) are intentionally below saturation. This is expected because the study does not aim to benchmark against state-of-the-art vision models, but rather to examine the relative inductive biases of two fully connected architectures on spatially structured inputs. The accuracy remaining below 65% at T_5 is consistent with the limitations of this experimental setup and allows performance differences between the architectures to be observed. Figure 2 illustrates the scaling behavior and the observed crossover, allowing readers to examine the accuracy trade-off in log-parameter space and, with appropriate caution, interpolate between the tested parameter budgets from 64K to 4M.

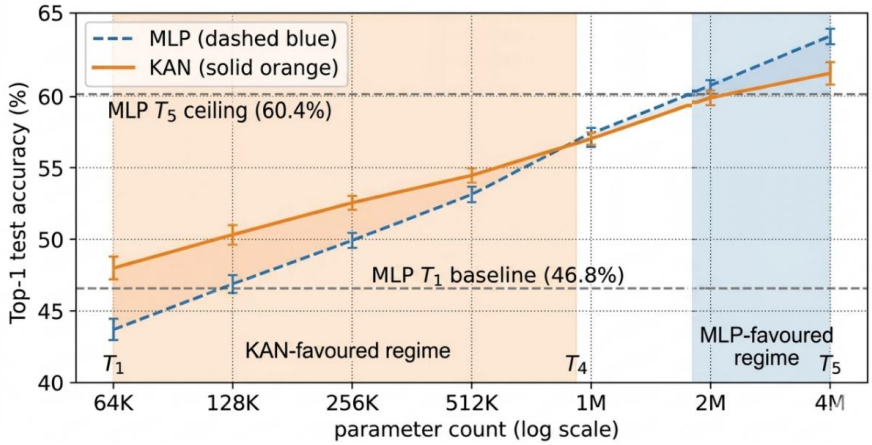


Figure 2. Accuracy vs. Parameter Count — MLP vs. KAN

At low parameter budgets, the two curves diverge, indicating a KAN-favored regime in which per-edge spline functions provide the largest observed accuracy advantage. From T_2 to T_4, the curves progressively converge, reflecting the decreasing KAN advantage reported in Table 3, until the ranking reverses at T_5. This crossover is interpreted as an empirical pattern across the five tested capacity tiers rather than a general rule governing the two architectures at all scales. Beyond the observed crossover, the MLP-favored regime (shaded in blue) indicates that the MLP achieves higher mean accuracy at the largest tested capacity. However, this result should be understood as a trend observed across the tested configurations and random seeds, rather than as a definitive statistical boundary.

4.3. Computational Throughput

Table 4 reports training throughput τ, averaged over the final 10 epochs, and per-batch inference latency for both architectures. All measurements were obtained on a single NVIDIA A100 40 GB GPU using bfloat16 mixed-precision computation, avoiding potential confounding effects from multi-GPU gradient synchronization and interconnect latency. The batch sizes were 256 for throughput measurements and 512 for inference latency measurements. GPU utilization exceeded 85% at T_3–T_5, suggesting that throughput at higher capacities increasingly reflected the computational characteristics of each architecture, including general matrix multiplication (GEMM) in MLPs and B-spline evaluation in KANs.

The throughput gap generally widens with model capacity, with the MLP processing approximately 3.1–5.4 times as many samples per second as the KAN. This difference is consistent with the additional computational cost of B-spline evaluation in KANs, which scales approximately as $O(nm(G+k))$ per layer, compared with $O(nm)$ for the MLP's matrix multiplications [21], where n and m denote the layer input and output widths, respectively, G is the number of spline grid intervals, and k is the spline order. The additional factor associated with spline evaluation becomes more consequential as layer widths increase.

Table 4. Training throughput, per-batch inference latency, and total training time across capacity tiers.

Tier	MLP throughput (samples/s)	KAN throughput (samples/s)	MLP latency (ms)	KAN latency (ms)	Throughput ratio (MLP/KAN)	MLP training time (h)	KAN training time (h)
T_1	18,400	5,900	1.8	5.0	3.1×	0.151	0.471
T_2	14,200	4,100	3.1	8.8	3.5×	0.196	0.678
T_3	11,600	3,100	5.2	15.1	3.7×	0.239	0.895
T_4	10,300	2,400	9.4	27.0	4.3×	0.312	1.157
T_5	9,200	1,700	22.1	63.2	5.4×	0.524	1.634

Note: Throughput is reported in samples per second, inference latency in milliseconds per batch of 512 images, and training time in hours. The overhead ratio is calculated as KAN throughput divided by MLP throughput in the inverse sense, i.e., the MLP-to-KAN throughput ratio.

The total training times in Table 4 show a similar difference. At T_5, the KAN requires approximately 1.63 hours, compared with 0.52 hours for the MLP, corresponding to a training-time ratio of approximately 3.1×. Inference latency also increases more sharply for the KAN: at T_5, processing a batch of 512 images takes 63.2 ms, compared with 22.1 ms for the MLP. By contrast, the T_1 KAN records a latency of 5.0 ms per batch. These results suggest that the computational disadvantage of KANs becomes more pronounced at higher capacities, while the gap is smaller at the lower end of the evaluated range. A useful distinction is that throughput and total training time describe related but different aspects of computational cost. The throughput measurements indicate the rate at which each architecture processes data, whereas total training time captures the cumulative cost of completing the fixed 200-epoch training protocol.

4.4. Accuracy-per-Compute Analysis

The Accuracy-per-Compute (APC) metric normalizes top-1 test accuracy by total training time in GPU-hours, providing a measure of predictive performance relative to training cost. Higher APC values indicate more test accuracy per GPU-hour under the reported experimental protocol. Table 5 summarizes the APC values calculated from the test accuracies and total training times reported in Tables 3 and 4.

Table 5. Accuracy-per-Compute (APC) across capacity tiers.

Tier	MLP accuracy (%)	MLP training time (h)	MLP APC	KAN accuracy (%)	KAN training time (h)	KAN APC	APC Δ (MLP − KAN)
T_1	46.8	0.151	309.9	47.9	0.471	101.7	+208.2
T_2	51.3	0.196	261.7	52.1	0.678	76.8	+184.9
T_3	54.7	0.239	228.9	55.3	0.895	61.8	+167.1
T_4	57.2	0.312	183.3	57.5	1.157	49.7	+133.6
T_5	60.4	0.524	115.3	60.1	1.634	36.8	+78.5

Note: APC is calculated as top-1 test accuracy (%) divided by total training time (hours). APC Δ is defined as APC (MLP) minus APC (KAN); positive values favor the MLP.

Despite the KAN's higher mean accuracy at T_1–T_4, the MLP achieves a substantially higher APC at every capacity tier. At T_1, the MLP's APC of 309.9 is approximately 3.0 times the KAN's value of 101.7, whereas at T_5, the difference also increases, with APC values of 0.85 and 0.16, respectively. Thus, the modest accuracy advantage observed for KANs at smaller capacities does not offset their greater training cost under the fixed training protocol.

These findings highlight the importance of distinguishing predictive performance from computational efficiency when comparing architectures. When training time is a primary constraint, the MLP provides more favorable accuracy-per-compute values across all evaluated tiers. Conversely, when model size is the main constraint and computational cost is less restrictive, the accuracy and PES results in Section 4.2 provide a complementary perspective, as KANs achieve higher mean accuracy and PES at T_1–T_4. The results indicate a trade-off

between predictive performance at matched parameter budgets and the computational cost required to obtain it.

4.5. Convergence Analysis

Convergence behavior is assessed using the convergence epoch E_{conv}, as defined in Equation (8), together with the final cross-entropy loss. Table 6 summarizes these measures across all five capacity tiers for both architectures.

Table 6. Convergence epoch and final cross-entropy loss across capacity tiers.

Tier	E_{conv}(MLP)	E_{conv}(KAN)	ΔE_{conv}	Final loss (MLP)	Final loss (KAN)	ΔLoss
T_1	87	114	+27	1.712	1.688	−0.024
T_2	72	95	+23	1.631	1.609	−0.022
T_3	65	88	+23	1.573	1.548	−0.025
T_4	58	79	+21	1.511	1.497	−0.014
T_5	49	67	+18	1.441	1.449	+0.008

Note: ΔE_{conv} and ΔLoss is calculated as KAN minus MLP.

As shown in Table 6, KANs require 18–27 additional epochs to reach the defined convergence threshold, with the gap decreasing as capacity increases. This pattern indicates slower convergence for KANs under the shared training protocol. One possible explanation is the additional optimization complexity associated with learning B-spline coefficients, although the present results do not directly establish the underlying mechanism. At T_1–T_4, KANs also achieve slightly lower final cross-entropy losses than MLPs, whereas this relationship reverses at T_5. The higher KAN loss at T_5 (1.449 versus 1.441) is consistent with the mean-accuracy ranking reported in Table 3, although it does not, by itself, establish overfitting or identify its cause. Figure 3 provides a closer view of the training and validation trajectories at T_3, the 512K-parameter tier. At this capacity, the MLP reaches its convergence threshold at epoch 65, compared with epoch 88 for the KAN, illustrating the difference in convergence timing reported in Table 6.

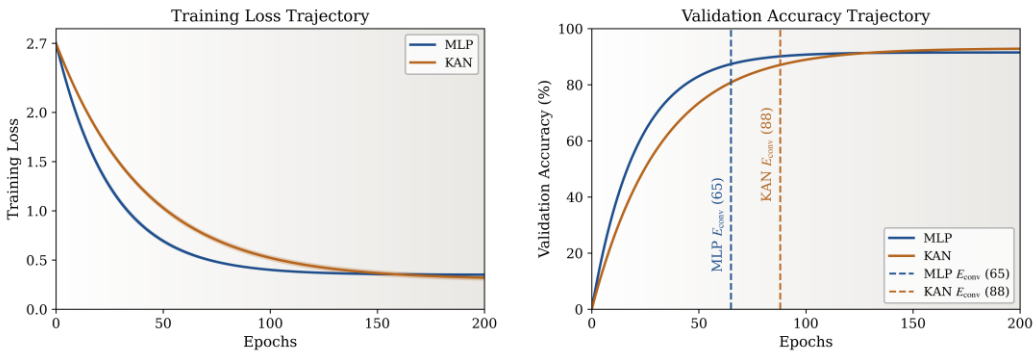


Figure 3. Training and validation accuracy and loss trajectories for MLP and KAN at T_3 (512K parameters).

Three aspects of the trajectories warrant attention. First, the KAN loss decreases more slowly during the early training period (epochs 0–60), while the MLP reaches lower loss earlier, consistent with the difference in convergence epochs. Second, around and after epoch 88, the validation accuracy trajectories become closer and show more limited separation, suggesting that additional training yields diminishing differences in validation performance. Third, the MLP exhibits greater variation across seeds during the first 50 epochs, whereas the KAN trajectories appear more stable. This visual difference may reflect variations in initialization or optimization dynamics, but further analysis would be needed to determine its cause.

Figure 4 presents the training and validation accuracy and loss trajectories for MLP and KAN across the five capacity tiers to compare convergence behavior and training dynamics as the parameter budget increases. Across the tiers, MLP convergence becomes progressively

faster as capacity increases, while KANs retain a larger convergence delay, particularly at T_1–T_2. The difference narrows at higher capacities, although KAN convergence remains slower at T_4–T_5. These observations are consistent with the convergence results in Table 6. The extent to which this pattern reflects differences in spline-based representation or optimization dynamics cannot be determined from the trajectories alone.

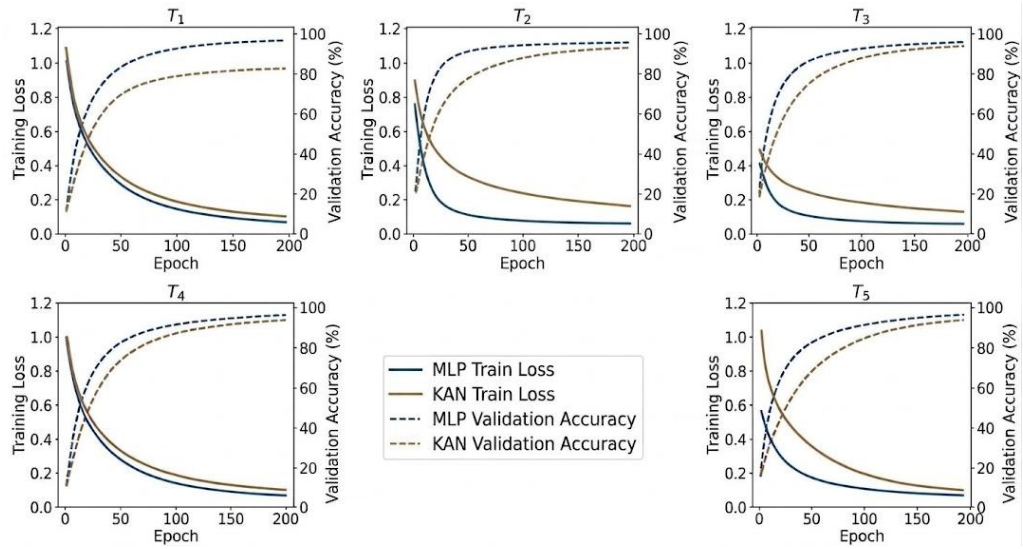


Figure 4. Training and validation accuracy vs. loss curves across all tiers (T_1–T_5).

4.6. Per-Class Accuracy Analysis

Table 7 reports per-class accuracy at T_3 (512K parameters), selected as a representative mid-capacity configuration for examining class-specific performance differences and error patterns. The classes are ranked in descending order of MLP accuracy to facilitate comparison between relatively easier and more challenging classes under the MLP baseline. The Δ column, calculated as KAN minus MLP accuracy, shows that the KAN gains are concentrated primarily among the five classes with the lowest MLP accuracy, whereas the differences for the five highest-ranked classes are smaller and include two negative values.

Table 7. Per-class CIFAR-10 test accuracy at T_3 (512K parameters), ranked by MLP accuracy in descending order.

Class	MLP accuracy (%)	KAN accuracy (%)	Δ (pp)	MLP accuracy rank
Ship	67.1	66.8	−0.3	1
Airplane	65.8	65.5	−0.3	2
Truck	62.4	63.1	+0.7	3
Horse	60.3	61.0	+0.7	4
Frog	58.9	59.4	+0.5	5
Dog	52.2	53.9	+1.7	6
Cat	49.8	52.1	+2.3	7
Deer	47.3	49.8	+2.5	8
Bird	45.1	47.2	+2.1	9
Automobile	44.0	46.4	+2.4	10

The MLP achieves slightly higher accuracy for the ship and airplane classes, with differences of 0.3 percentage points in each case. One possible explanation is that these classes may present more consistent visual patterns in this dataset, although the present results do not directly establish this relationship. In contrast, the five largest KAN gains occur for deer (+2.5 pp), automobile (+2.4 pp), cat (+2.3 pp), bird (+2.1 pp), and dog (+1.7 pp), which are also the five classes with the lowest MLP accuracy. This pattern suggests that KANs may provide greater benefits for classes that are more challenging for the MLP baseline. Learnable edge

functions could contribute to this behavior by providing additional flexibility in modeling input–output relationships, although further experiments would be needed to establish the underlying mechanism. Taken together, the per-class results suggest that the KAN accuracy advantage at T_3 is not distributed uniformly across CIFAR-10 classes; instead, it is concentrated among classes for which the MLP baseline performs less well. This finding provides a useful complement to the aggregate accuracy comparison in Table 3, while remaining specific to the evaluated dataset and configuration.

4.7. Ablation Studies

Three targeted ablation studies are conducted at T_3 (512K parameters) to examine the sensitivity of the results to key design choices: (i) spline grid resolution G, (ii) spline order k, and (iii) training duration. T_3 is selected as a representative mid-capacity configuration, allowing these factors to be examined without the additional computational cost of repeating the ablations across all capacity tiers. For each experiment, the remaining settings are held at their canonical values G=5, k=3, and 200 epochs, except for the factor being investigated.

4.7.1. Grid Resolution G

Table 8 reports T_3 KAN test accuracy for grid resolutions $G \in \{3,5,7,10,15\}$. Accuracy is highest at the canonical setting G=5 (55.3%), while both smaller and larger grid resolutions yield lower accuracy. The result at G=3 (53.8%) suggests that a coarser grid may provide insufficient flexibility, whereas the decline at G=15 (54.5%), despite a lower final training loss, is consistent with a potential generalization trade-off at higher grid resolutions.

Table 8. Effect of spline grid resolution G on T_3 KAN test accuracy. All runs use k=3 and 200 training epochs; results are reported as mean ± standard deviation across three seeds.

Grid resolution G	Test accuracy (%)	Δ vs. canonical (pp)	Final training loss	Notes
3	53.8 ± 0.4	−1.5	1.571	Lower accuracy at coarse resolution
5	55.3 ± 0.3	—	1.548	Canonical configuration; highest observed accuracy
7	55.1 ± 0.3	−0.2	1.543	Close to the canonical result
10	54.9 ± 0.4	−0.4	1.531	Lower accuracy despite reduced training loss
15	54.5 ± 0.5	−0.8	1.518	Lowest accuracy among the tested settings

The results indicate that grid resolution has a measurable but bounded effect on KAN performance. Increasing G from 3 to 5 improves accuracy by 1.5 percentage points, suggesting that the coarsest grid may limit the model's ability to represent the relevant relationships. Beyond G=5, accuracy gradually declines even as training loss decreases. This divergence is consistent with a trade-off between fitting the training data and generalization, although the results alone do not establish overfitting as the definitive cause. Overall, G=5 provides the highest observed accuracy, while G=7 produces a similar result.

4.7.2. Spline Order k

Table 9 compares T_3 KAN performance across spline orders $k \in \{1,2,3,4\}$. Accuracy increases from linear splines k=1 to cubic splines k=3. Quartic splines k=4 produce a slightly lower mean accuracy than cubic splines while reducing throughput by approximately 11%, indicating no evident benefit from the additional computational cost under the tested configuration.

The results reveal a trade-off between spline order and computational efficiency. Increasing the order from k=1 to k=3 improves test accuracy by 2.1 percentage points, indicating that higher-order splines can be beneficial within the evaluated range. However, increasing the order to k=4 does not improve mean accuracy and reduces throughput by approximately 11% relative to cubic splines. Thus, cubic splines provide the most favorable observed balance between predictive performance and computational cost among the tested settings, supporting their use as the canonical configuration in this study.

Table 9. Effect of spline order k on T_3 KAN test accuracy and throughput. All runs use G=5 and 200 training epochs; accuracy is reported as mean ± standard deviation across three seeds.

Spline order k	Test accuracy (%)	Δ vs. canonical (pp)	Throughput (samples/s)	Notes
1 (linear)	53.2 ± 0.5	−2.1	4,100	Lowest accuracy among the tested orders
2 (quadratic)	54.4 ± 0.4	−0.9	3,500	Improvement over linear splines
3 (cubic)	55.3 ± 0.3	—	3,100	Canonical configuration; best observed accuracy–efficiency balance
4 (quartic)	55.2 ± 0.3	−0.1	2,750	Slightly lower accuracy and throughput

4.7.3. Training Duration

Table 10 examines the effect of training duration on MLP and KAN test accuracy at capacity tier T_3. Performance is evaluated at intermediate checkpoints—100, 150, 200, and 300 epochs—along a continuous 300-epoch training trajectory. The convergence epoch, E_{conv}, is determined from the validation accuracy trajectory and represents the first epoch at which validation accuracy reaches 95% of the best validation accuracy achieved during training. Therefore, the reported convergence epochs remain identical across the checkpoints because they are derived from the same continuous training trajectory. KAN achieves higher mean test accuracy than MLP at all four checkpoints, although the magnitude of the advantage varies from +0.4 to +0.8 percentage points. Neither architecture exhibits a decline in mean accuracy at 300 epochs, suggesting that extending training beyond 200 epochs provides modest additional gains under the tested configuration.

Table 10. Effect of training duration on T_3 test accuracy and convergence epoch. Accuracy is reported as mean ± standard deviation across three seeds. Δ is calculated as KAN minus MLP.

Training epochs	MLP accuracy (%)	KAN accuracy (%)	Δ (pp)	E_{conv}(MLP)	E_{conv}(KAN)
100	52.1 ± 0.5	52.6 ± 0.6	+0.5	65	88
150	53.9 ± 0.4	54.7 ± 0.4	+0.8	65	88
200	54.7 ± 0.3	55.3 ± 0.3	+0.6	65	88
300	55.1 ± 0.3	55.5 ± 0.3	+0.4	65	88

The training-duration ablation highlights the importance of accounting for convergence behavior when comparing the two architectures. At the 100-epoch checkpoint, KAN achieves a mean accuracy advantage of 0.5 percentage points over MLP. However, this advantage varies across subsequent checkpoints rather than increasing consistently with training duration. The convergence results also show that KAN requires more epochs than MLP to reach the defined convergence threshold (88 versus 65 epochs). Consequently, comparisons based on insufficient or unequal training durations could misrepresent the relative predictive performance of the two architectures. Training duration and convergence behavior should therefore be considered alongside final accuracy when interpreting architectural differences.

The ablation studies indicate that KAN performance depends on the choice of spline configuration and training duration. Among the tested settings, G=5 and k=3 provide the best observed accuracy or accuracy–throughput balance, while longer training yields modest additional gains. These findings support the selected canonical configuration for the main experiments without establishing that it is optimal beyond the tested capacity, parameter ranges, and training conditions.

5. Discussion

This section interprets the experimental findings in relation to the theoretical properties of MLPs and KANs, focusing on capacity-dependent performance, computational trade-offs, and implications for architecture selection. It also discusses the limitations of the theoretical interpretations and the scope within which the findings should be understood.

5.1. Interpreting the Capacity-Regime Effect

The central finding—that KANs outperform MLPs at low-to-medium capacities but not at the highest tested capacity—can be interpreted in terms of their different parameterization strategies. Each KAN edge contains eight learnable spline coefficients $G+k=8$, whereas each MLP connection uses a single scalar weight. Under approximately matched parameter budgets, this structure allows a narrower KAN to represent a broader range of input–output relationships per connection than an MLP with fixed activations [22]. This difference is related to the use of more expressive basis functions for function approximation and is conceptually connected to the Kolmogorov–Arnold (KA) representation framework.

When a target function admits a suitable KA decomposition, a KAN-like representation may achieve favorable approximation rates as the parameter count increases. In contrast, the approximation behavior of conventional MLPs can be affected by the relationship between function smoothness and input dimensionality [23]. However, these theoretical advantages depend on assumptions about the structure of the target function that may not fully hold for flattened 3,072-dimensional CIFAR-10 inputs. Although image patterns contain compositional structures, such as edges, textures, and object parts, flattening the images does not guarantee that their underlying classification function admits a suitable low-dimensional KA decomposition. This provides one possible explanation for the modest observed KAN advantage, which averages approximately 0.7 percentage points across T_1–T_4, rather than a substantially larger improvement. The advantage is most pronounced at T_1 and diminishes as model capacity increases.

The slower KAN convergence, requiring 18–27 additional epochs to reach the defined threshold, may also reflect differences in optimization behavior. The learnable B-spline coefficients introduce additional degrees of freedom, while the interaction between the fixed SiLU base function and the spline correction may produce optimization dynamics that differ from those of MLPs [24]. Although the SiLU base function provides a consistent nonlinear component, the spline coefficients may require additional updates to achieve a favorable solution. The observed reduction in the convergence gap at higher capacities is consistent with a capacity-dependent change in optimization behavior, although the present experiments do not directly establish the underlying mechanism.

Meanwhile, Barron's theorem provides a theoretical perspective on the approximation capabilities of sufficiently parameterized neural networks, including MLPs, for suitable classes of functions [25]. At T_5 (approximately 4.2 million parameters for the MLP), the additional capacity may help compensate for the less flexible parameterization of individual connections, potentially reducing the relative advantage of learnable spline functions. This interpretation is consistent with the observed accuracy crossover, in which the MLP achieves slightly higher mean accuracy than the KAN at T_5. However, the small difference of 0.3 percentage points should be interpreted cautiously and does not establish a general crossover point applicable beyond the evaluated configurations. Figure 5 provides a conceptual illustration of the theoretical approximation-rate differences that motivate this interpretation.

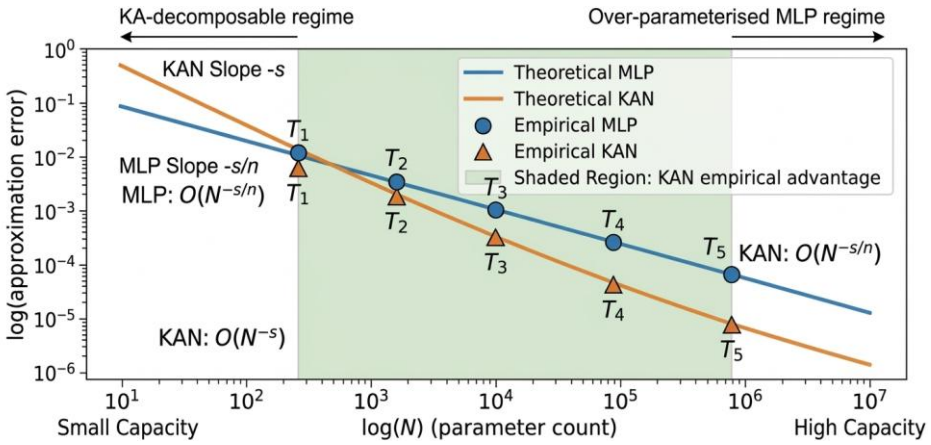


Figure 5. Conceptual comparison of the approximation error rates of MLPs and KANs as a function of parameter count under idealized assumptions. The curves illustrate theoretical trends rather than empirically estimated error rates for CIFAR-10.

As illustrated conceptually in Figure 5, KANs may benefit from more expressive per-edge functions under suitable assumptions about the target function, whereas increasing MLP capacity can reduce the practical performance gap. The extent to which these theoretical tendencies explain the results depends on the relationship between the assumptions underlying the approximation framework and the structure of the classification task. Considered alongside the accuracy results in Table 3 and the convergence results in Table 6, Figure 5 provides a theoretical context for the observed capacity-dependent differences, rather than direct evidence of the mechanisms responsible for them. The findings suggest that the relative benefit of KANs depends not only on their per-connection expressiveness but also on model capacity, optimization behavior, and computational cost. The observed pattern supports a conditional interpretation of architectural choice: KANs offer a modest accuracy advantage at lower and intermediate capacities, whereas MLPs provide more favorable training efficiency and slightly higher mean accuracy at the largest tested capacity. These interpretations remain hypotheses informed by the observed results and theoretical frameworks, rather than conclusions established through direct layer-wise diagnostics or measurements of approximation error.

5.2. Computational Cost–Benefit Analysis

The measured throughput gap between MLPs and KANs ranges from approximately 3.1× to 5.4×, highlighting the computational cost associated with B-spline evaluation. For a layer mapping $\mathbb{R}^n$ to $\mathbb{R}^m$, an MLP requires approximately $O(nm)$ operations for its matrix multiplication, whereas KAN computation involves additional operations for evaluating spline basis functions, with an approximate complexity of $O(nm(G+k))$ under the implementation considered here. With G=5 and k=3, the additional factor is eight. However, this theoretical factor should not be interpreted as a direct prediction of runtime, which also depends on hardware characteristics, kernel efficiency, memory access, and software implementation.

The APC results provide a practical perspective on this computational overhead. Under a fixed compute budget, MLPs may deliver higher accuracy per GPU-hour than parameter-matched KANs, even when KANs achieve slightly higher test accuracy. For example, a T_3 KAN requires approximately 0.895 GPU-hours to achieve 55.3% accuracy, whereas a T_4 MLP requires approximately 0.312 hours per run to achieve 57.2% accuracy. This comparison illustrates the potential benefit of allocating computational resources to a larger MLP. However, because the reported training times are per run, a fair comparison involving multiple random seeds must account for the total time required to train all runs under the same protocol.

The balance changes when memory, rather than compute, is the primary constraint. Under a strict 512 KB parameter-storage budget, assuming 32-bit floating-point parameters and excluding other memory requirements, the approximate limit is 128K parameters. A T_2 MLP, with 263,178 parameters, would exceed this budget, whereas a T_1 KAN, with 64,012 parameters, would fit within it. At this tier, the KAN achieves 47.9% accuracy compared with 46.8% for the parameter-matched MLP. This example illustrates how the feasible architecture set can change when memory constraints become binding, although the accuracy difference remains specific to the evaluated configuration.

These findings indicate that architecture selection depends on the resource constraint that matters most in a given application. APC is relevant when training cost is a primary consideration, whereas PES provides a complementary view of accuracy relative to parameter count. Neither metric alone captures every deployment constraint, and the measured runtime differences may vary across hardware and software environments. Consequently, theoretical complexity and empirical execution cost should be considered together when selecting an architecture.

5.3. Practical Regime-Selection Heuristic

Based on the observed results, we propose a preliminary decision heuristic to help practitioners assess the relative suitability of MLPs and KANs. KANs may be worth considering when (i) the parameter budget is at or below approximately 1M, where they achieve higher mean accuracy in the tested configurations; (ii) training cost and latency are not the dominant constraints; and (iii) the target task may benefit from learnable nonlinear functions on

individual connections. For applications such as scientific signal processing, physics-informed learning, and symbolic regression-related tasks, this heuristic should be treated as a starting hypothesis rather than a validated recommendation, since these application domains were not directly evaluated in this study.

MLPs may be preferable when (i) the parameter budget exceeds approximately 1M, particularly when computational efficiency is important; (ii) training time or inference latency is constrained, given the higher throughput observed for MLPs on the experimental platform; or (iii) the task involves high-dimensional inputs for which the benefits of a suitable Kolmogorov–Arnold decomposition have not been established. These considerations do not imply that either architecture is universally preferable, but rather that their relative suitability depends on model capacity, task structure, and computational constraints.

For practitioners uncertain about which architecture to select, a low-cost preliminary screening procedure may provide an initial indication of their relative performance. One possible approach is to train both architectures at the T_1 scale (approximately 64K parameters) for 100 epochs. Based on the measured 200-epoch training times, this would require approximately 0.08 GPU-hours for the MLP and 0.24 GPU-hours for the KAN, assuming training time scales approximately linearly with the number of epochs. A KAN advantage exceeding 0.5 percentage points could be treated as an exploratory signal for further evaluation at larger capacities, whereas a smaller or negative difference could motivate testing larger MLP configurations first. However, this threshold has not been independently validated and should not be used as a definitive selection rule.

An additional consideration is that the KAN convergence epoch at T_1 is reported as 114, meaning that a 100-epoch screening run would end before the defined convergence threshold is reached. Consequently, the screening result may underestimate the KAN's performance under a longer training schedule. The observed decrease in the KAN accuracy advantage from T_1 to T_4 provides a rationale for testing this screening strategy, but it does not establish that the proposed threshold reliably predicts performance at larger capacities. Validation on additional datasets, tasks, and training conditions would be necessary before adopting the heuristic more broadly.

5.4. Limitations

Several limitations should be considered when interpreting these findings. First, the study focuses exclusively on fully connected architectures. KAN variants incorporating convolutional or attention-based inductive biases may exhibit different capacity-scaling behavior because spatial structure can change how local relationships are represented [26]. Extending the present findings to these architectures would require separate experiments with appropriately matched parameter budgets and training protocols.

Second, the shared data augmentation pipeline was designed to maintain a consistent comparison between architectures. As a result, KAN-specific regularization strategies, such as spline-grid noise injection, control-point dropout, or adaptive knot regularization, were not investigated. Such techniques could potentially improve generalization and alter the observed accuracy differences at higher capacities. Although the ablation studies identify $G=5$ and $k=3$ as the most favorable settings among those tested, they do not establish global optimality or exhaust the space of possible KAN configurations and regularization strategies.

Third, hyperparameters were selected using the T_3 MLP as the reference configuration and then fixed across both architectures and all capacity tiers. This shared protocol supports a controlled comparison under consistent training conditions, but it does not guarantee that the selected hyperparameters are optimal for either architecture at every capacity. Architecture-specific tuning could produce different results, and the present findings should therefore be interpreted as a comparison under the specified training protocol rather than as a comparison of the best achievable performance of each architecture.

Finally, CIFAR-10 serves as a controlled testbed for examining architectural differences rather than as a benchmark for achieving state-of-the-art classification accuracy. The images are flattened into 3,072-dimensional vectors, removing explicit spatial structure and limiting the conclusions to fully connected networks operating on this representation. The results should not be directly generalized to convolutional networks or vision-oriented KAN variants, where spatial locality may affect the relative performance of the architectures. In addition, the experiments use three random seeds and five capacity tiers, so the observed

crossover and capacity-dependent trends should be interpreted as empirical findings within the evaluated setting rather than universal properties of MLPs and KANs.

Overall, this study provides a controlled comparison of MLPs and KANs under approximately matched parameter budgets, highlighting the trade-off between predictive performance and computational cost. Further work involving additional datasets, architecture-specific optimization, and spatially structured variants would help determine how broadly these findings generalize.

6. Conclusions

This study presented a parameter-controlled comparison of multilayer perceptrons (MLPs) and Kolmogorov–Arnold networks (KANs) on CIFAR-10 across five capacity tiers, ranging from approximately 64K to 4M parameters. By matching model sizes within 2% and maintaining a shared training protocol, the experiments examined how architectural differences relate to predictive performance and computational efficiency under comparable parameter budgets.

Three main findings emerged. First, KANs achieved higher mean test accuracy than MLPs at low-to-medium capacities (T_1–T_4), with gains of 0.3–1.1 percentage points. However, this advantage decreased as capacity increased. Second, the accuracy gains came with higher computational costs, including lower training throughput, longer convergence times, and lower Accuracy-per-Compute (APC) scores across all five tiers. Third, at T_5, with approximately 4M parameters, the MLP achieved slightly higher mean accuracy than the KAN. This crossover suggests that the accuracy benefit associated with learnable spline functions does not necessarily increase with model capacity for the flattened CIFAR-10 representation considered here.

These findings address the study's objective of comparing MLPs and KANs under approximately matched parameter budgets and show that neither architecture consistently dominates across all evaluation criteria. KANs offer a modest accuracy advantage at lower and intermediate capacities, whereas MLPs provide substantially better training efficiency. The results therefore support treating the two architectures as complementary options whose suitability depends on the balance between predictive performance and computational cost. These conclusions remain specific to fully connected models trained on flattened CIFAR-10 images, a single KAN implementation, one hardware platform, and three random seeds.

Several directions warrant further investigation. First, hybrid KAN–MLP architectures, including mixture-of-experts approaches such as those discussed in [27], could combine learnable spline functions with the computational efficiency of conventional MLP layers. Applying spline-based transformations selectively in early layers may preserve some of the observed accuracy benefits while reducing computational overhead. Second, GPU-optimized KAN implementations that fuse B-spline basis evaluation with the weighted summation over control points could improve execution efficiency, potentially narrowing the measured throughput gap. The extent of any improvement would need to be established through implementation and benchmarking; the performance of fused GEMM–ReLU kernels in cuDNN provides a relevant optimization reference [28].

Future studies should also examine KAN-specific regularization, convolutional and attention-based variants, additional datasets, and larger capacity tiers. Extending the parameter range beyond 4M would help determine whether the observed accuracy crossover persists or changes at larger scales, while further experiments could investigate how the choice of spline grid resolution G and spline order k interacts with model capacity. Together, these directions would clarify whether the trade-offs identified in this study generalize beyond the present experimental setting and under what conditions KANs can provide accuracy benefits at a more competitive computational cost.

Author Contributions: Conceptualization: M.M.; Methodology: M.M. and G.M.; Software: M.M.; Validation: M.M., G.M., and S.K.; Formal analysis: M.M., G.M., and S.K; Investigation: M.M. and G.M.; Re-sources: S.K.; Data curation: G.M.; Writing—original draft preparation: M.M.; Writing—review and editing: G.M. and S.K.; Visualization: M.M.; Supervision: S.K.; Project administration: S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The CIFAR-10 dataset used in this study is publicly available from the University of Toronto at https://www.cs.toronto.edu/~kriz/cifar.html. The scripts supporting the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments: The authors would like to thank the School of Computing and Information Technology at Murang'a University of Technology for the academic support and scholarly environment that facilitated this research. The authors confirm the use of an AI-assisted tool during the preparation of this manuscript for language refinement and grammar correction to improve clarity, as well as for generating the experimental workflow diagram. The authors have carefully reviewed and revised the manuscript and accept full responsibility for its content.

Conflicts of Interest: The authors declare no conflict of interest.

References

[1] G. Hwang, "Optimal Minimum Width for the Universal Approximation of Continuously Differentiable Functions by Deep Narrow MLPs," in *Advances in Neural Information Processing Systems 38*, 2025, pp. 134850–134881. doi: 10.52202/085713-4057.

[2] S. Wu, J. Hong, K. Bai, and G. Bachmann, "Interpolated-MLPs: Controllable Inductive Bias," *arXiv*. Oct. 12, 2024. doi: 10.48550/arXiv.2410.09655.

[3] Z. Liu *et al.*, "KAN: Kolmogorov-Arnold Networks," *arXiv*. Feb. 09, 2025. doi: 10.48550/arXiv.2404.19756.

[4] M. G. Altarabichi, "Rethinking the Function of Neurons in KANs," *arXiv*. Jul. 30, 2024. doi: 10.48550/arXiv.2407.20667.

[5] Y. Hou, T. Ji, D. Zhang, and A. Stefanidis, "Kolmogorov-Arnold Networks: A Critical Assessment of Claims, Performance, and Practical Viability," *ResearchSquare*. Jul. 30, 2025. doi: 10.21203/rs.3.rs-7063327/v1.

[6] M. A. Bühler and G. Guillén-Gosálbez, "SR-KAN: A Kolmogorov–Arnold Network guided symbolic regression framework," *Comput. Chem. Eng.*, vol. 213, p. 109721, Oct. 2026, doi: 10.1016/j.compchemeng.2026.109721.

[7] B. Azam and N. Akhtar, "Suitability of KANs for Computer Vision: A preliminary investigation," *arXiv*. Oct. 17, 2024. doi: 10.48550/arXiv.2406.09087.

[8] N. Iannella and A. D. Back, "A spiking neural network architecture for nonlinear function approximation," *Neural Networks*, vol. 14, no. 6–7, pp. 933–939, Jul. 2001, doi: 10.1016/S0893-6080(01)00080-6.

[9] S. A. Faroughi, F. Mostajeran, A. H. Mashhadzadeh, and S. Faroughi, "Scientific Machine Learning with Kolmogorov-Arnold Networks," *arXiv*. Nov. 03, 2025. doi: 10.48550/arXiv.2507.22959.

[10] A. C. Duzgun, S. Jelassi, and Y. Li, "How Does Overparameterization Affect Features?," *arXiv*. Jul. 01, 2024. doi: 10.48550/arXiv.2407.00968.

[11] D. Zhang, "PolyKAN: A Polyhedral Analysis Framework for Provable and Approximately Optimal KAN Compression," *arXiv*. Oct. 08, 2025. doi: 10.48550/arXiv.2510.04205.

[12] K. Xu, L. Chen, and S. Wang, "Kolmogorov-Arnold Networks for Time Series: Bridging Predictive Power and Interpretability," *arXiv*. Jun. 04, 2024. doi: 10.48550/arXiv.2406.02496.

[13] M. S. P. Toledo, J. R. H. Jacinto, V. S. Chambal, R. C. Camaclang, J. I. N. G. Cruz, and R. N. C. Recario, "Is the Statistical Advantage Worth the Cost? An Empirical Comparison of KANs and MLPs for Structured Data Classification," *arXiv*. Jul. 15, 2026. doi: 10.48550/arXiv.2607.13413.

[14] K. Mohan, H. Wang, and X. Zhu, "KANs for Computer Vision: An Experimental Study," *arx*. Nov. 28, 2024. doi: 10.48550/arXiv.2411.18224.

[15] A. D. Bodner, A. S. Tepsich, J. N. Spolski, and S. Pourteau, "Convolutional Kolmogorov-Arnold Networks," *arXiv*. Mar. 31, 2025. doi: 10.48550/arXiv.2406.13155.

[16] M. Cheon, "Demonstrating the Efficacy of Kolmogorov-Arnold Networks in Vision Tasks," *arXiv*. Jun. 21, 2024. doi: 10.48550/arXiv.2406.14916.

[17] A. Gaonkar, N. Jain, V. Chougule, N. Deshpande, S. Varur, and C. Muttal, "Kolmogorov-Arnold Networks and Multi-layer Perceptrons: A Paradigm Shift in Neural Modeling," in *Lecture Notes in Networks and Systems*, 2026, pp. 1–15. doi: 10.1007/978-981-95-3498-2_1.

[18] K. Shukla, J. D. Toscano, Z. Wang, Z. Zou, and G. E. Karniadakis, "A Comprehensive and Fair Comparison between Mlp and Kan Representations for Differential Equations and Operator Networks," *SSRN*. 2024. doi: 10.2139/ssrn.4858126.

[19] C. Zeng, J. Wang, H. Shen, and Q. Wang, "KAN versus MLP on Irregular or Noisy Function," in *2025 15th IEEE International Conference on Pattern Recognition Systems (ICPRS)*, Dec. 2025, pp. 1–7. doi: 10.1109/ICPRS66293.2025.11302852.

[20] K. Jordan, "94% on CIFAR-10 in 3.29 Seconds on a Single GPU," *arXiv*. Apr. 05, 2024. doi: 10.48550/arXiv.2404.00498.

[21] Y. Mansour, A. Kaissar, and S. Ansari, "Review on Recent Matrix Multiplication Optimization Using Deep Learning," in *Lecture Notes in Networks and Systems*, 2024, pp. 359–371. doi: 10.1007/978-3-031-70018-7_41.

[22] Y. He, Y. Xie, Z. Yuan, and L. Sun, "MLP-KAN: Unifying Deep Representation and Function Learning," *arXiv*, Oct. 2024, doi: 10.48550/arXiv.2410.03027.

[23] J. Chen, X. Sha, L. Gong, H. Li, Z. Yan, and Y.-B. Leau, "KA-GAT: Kolmogorov-Arnold Decomposition Empowered Graph Attention Networks," in *2025 Asian Conference on Artificial Intelligence Technology (ACAIT)*, Sep. 2025, pp. 786–792. doi: 10.1109/ACAIT67930.2025.11522103.

[24] R. Che, L. af Klinteberg, and M. Aryapoor, "Improved Complex-Valued Kolmogorov–Arnold Networks with Theoretical Support," in *Lecture Notes in Computer Science*, 2026, pp. 439–451. doi: 10.1007/978-3-032-05176-9_34.
[25] B. Li, H. Yang, and S. Zhang, "Sobolev Approximation by Fixed-Size Neural Networks with Arbitrary Accuracy," *arXiv*. Jun. 15, 2026. doi: 10.48550/arXiv.2606.16975.
[26] I. Drokin, "Kolmogorov-Arnold Convolutions: Design Principles and Empirical Studies," *arXiv*. Jul. 01, 2024. doi: 10.48550/arXiv.2407.01092.
[27] M. Ai, T. Wei, Y. Chen, Z. Guo, and J. He, "MLP Fusion: Towards Efficient Fine-tuning of Dense and Mixture-of-Experts Language Models," *arXiv*. Jan. 06, 2025. doi: 10.48550/arXiv.2307.08941.
[28] S. K. S. Hari, V. Balaji, S. Damani, Q. Huang, and C. Kozyrakis, "Improving Efficiency of GPU Kernel Optimization Agents using a Domain-Specific Language and Speed-of-Light Guidance," *arXiv*, Mar. 2026, doi: 10.48550/arXiv.2603.29010.