A Comparative Study of Knowledge Distillation Methods on Neural Networks

Advanced Computational Learning (52025)

Yaron Otmazgin & Yael Levy

Abstract

Knowledge distillation enables the transfer of knowledge from large, complex teacher models to smaller, more efficient student models. This work provides a systematic comparison of three prominent distillation methods. We evaluate these methods on two datasets of different complexity-MNIST and CIFAR-10-comparing their effectiveness across three key dimensions: classification accuracy, model compression, and inference speed (forward pass time per image). Additionally, we investigate how architectural modifications (residual connections and Batch Normalization) affect distillation performance on the MLP-based MNIST experiments. Our experiments demonstrate that while all distillation methods successfully transfer knowledge from teacher to student, they present different trade-offs between model size, accuracy, and computational efficiency.

1 Introduction

Deep neural networks have achieved remarkable success across various domains, but their deployment in resource-constrained environments remains challenging due to computational and memory requirements. Knowledge distillation (KD), introduced by Hinton et al. [1], addresses this challenge by enabling the training of compact student models that learn to mimic the behavior of larger, more accurate teacher models.

At its core, knowledge distillation is a model compression technique built on a simple intuition: a teacher model is a standard trained neural network, typically large and highly accurate, while a student model is a smaller, more efficient network designed for deployment. Rather than training the student solely on ground-truth labels, knowledge distillation trains the student to imitate the teacher's outputs. This imitation process allows the student to inherit the teacher's learned representations-capturing implicit knowledge about class relationships and data structure that hard labels alone cannot convey.

The key insight of knowledge distillation is that teacher models encode rich information not only in their final predictions but also in their confidence distributions across all classes. For example, a teacher predicting digit "7" with 99% confidence and allocating 1% to digit "1" conveys more information than the hard label alone-it captures the visual similarity between these digits. These probability distributions, called soft targets (or soft labels), are produced by applying temperature scaling (dividing the logits by a temperature parameter T before the softmax function, which produces smoother probability distributions). By learning from these soft targets, student models can achieve performance approaching that of their teachers while using significantly fewer parameters.

Since Hinton's seminal work, various distillation methods have emerged, each proposing different mechanisms for knowledge transfer. FitNets [2] extend distillation to intermediate layers, using hints (intermediate feature maps from the teacher) to guide student training. Relational Knowledge Distillation (RKD) [3] takes a different approach by focusing on preserving pairwise relationships (the relative distances and angles between samples in feature space) rather than matching individual outputs. However, systematic comparisons of these methods under controlled conditions remain limited.

This work provides an empirical comparison of three distillation approaches on image classification tasks using two datasets of different complexity: MNIST (with MLP architectures) and CIFAR-10 (with CNN architectures). We examine how different distillation strategies affect the accuracy-efficiency trade-off and provide insights into when each method might be preferred.

2 Distillation Methods

Hinton's Knowledge Distillation (Vanilla KD). Hinton et al. [1] proposed training a student network using softened output probabilities from a teacher, obtained via temperature scaling, allowing the student to learn from the full class distribution rather than only the predicted label. We implement this approach with temperature $T = 20$ (replicating the setup in Section 3 of [1]) and $T = 4$ for CIFAR-10, and set the distillation weight to $\alpha = 0.1$. The combined loss function is

$$L = \alpha\, L_{\mathrm{CE}}(\mathrm{logits}, y_{\mathrm{true}}) + (1 - \alpha)\, L_{\mathrm{KD}}, \tag{1}$$

where the distillation loss is defined as

$$L_{\mathrm{KD}} = T^2\, \mathrm{KL}(\sigma(\mathrm{logits}/T), \sigma(\mathrm{teacher\_logits}/T))\,. \tag{2}$$

Here, L_{CE} denotes the cross-entropy loss, KL the Kullback–Leibler divergence, σ the softmax function, and the T^2 factor compensates for the reduction in gradient magnitudes introduced by temperature scaling.

FitNets. Romero et al. [2] extend knowledge distillation by introducing intermediate-layer supervision through a two-stage training process. In **Stage 1**, a regressor is trained to map activations from a designated

student *guided layer* to match those of a teacher *hint layer* by minimizing

$$L_{\text{hint}} = \|\text{Regressor}(h_{\text{student}}) - h_{\text{teacher}}\|^2 \tag{3}$$

In **Stage 2**, the entire student network is trained using the standard knowledge distillation objective, combining supervision from the ground-truth labels and the softened teacher outputs, with the hint-trained layers providing a favorable initialization. This stage-wise training enables deeper and thinner student networks to be trained effectively, achieving higher compression.

Relational Knowledge Distillation (RKD). Park et al. [3] proposed preserving structural relationships between samples rather than matching individual outputs. We extract feature embeddings from the penultimate layers and compute pairwise Euclidean distance matrices D_{Teacher} and D_{Student}, normalized by their mean non-zero values for scale invariance:

$$\hat{D} = D/\mu(D_{\text{nonzero}}) \tag{4}$$

The relational loss uses Huber loss (a loss function less sensitive to outliers than squared error):

$$L_{\text{RKD}} = \text{SmoothL1}(\hat{D}_{\text{student}}, \hat{D}_{\text{teacher}}) \tag{5}$$

The total objective combines task loss with this structural constraint:

$$L_{\text{Total}} = L_{\text{CE}}(\text{logits}, y_{\text{true}}) + \beta \cdot L_{\text{RKD}} \tag{6}$$

where we set $\beta = 100$ to balance gradient magnitudes between the cross-entropy and distance terms.

3 Experiments

3.1 Experimental Setup

We conduct experiments on two datasets representing different complexity levels:[1]

MNIST. The MNIST dataset contains 60,000 training and 10,000 test images of handwritten digits (0–9), each 28×28 grayscale pixels. We use Multi-Layer Perceptron (MLP) architectures for all MNIST experiments. Data augmentation applies random translation of up to 2 pixels during training. All models are trained for 20 epochs using Adam optimizer with learning rate 0.001, weight decay 10^{-4}, and batch size 256.

[1]All experiments are conducted on NVIDIA Tesla T4 GPU with PyTorch.

CIFAR-10. The CIFAR-10 dataset consists of 60,000 32×32 color images in 10 classes, with 50,000 training and 10,000 test images. We use CNN architectures with random horizontal flips for data augmentation. All models are trained for 20 epochs using Adam optimizer with learning rate 0.001 and batch size 128.

3.2 Network Architectures

Given the different complexity levels of our datasets, we employ MLPs for MNIST and CNNs for CIFAR-10 (Figures 1 and 2):

3.2.1 MNIST Architectures (MLP-based)

Teacher Network. Two hidden layers of 1,200 ReLU units each, with dropout ($p = 0.2$). Input dimension 784, output 10 classes. Total parameters: 2,395,210.

Student Network (Vanilla KD & RKD). Two hidden layers of 800 ReLU units each. Total parameters: 1,276,810.

FitNet Student. Four hidden layers of 300 units each (509,410 parameters). The second hidden layer serves as the "guided layer" matching the teacher's hint layer.

3.2.2 CIFAR-10 Architectures (CNN-based)

Teacher Network. Three convolutional blocks with 64, 128, and 256 filters, each followed by ReLU and 2×2 max pooling. Fully-connected layer with 1,024 units and dropout ($p = 0.3$) before the 10-class output. Total parameters: 4,576,394.

Student Network (Vanilla KD & RKD). Two convolutional blocks with 32 and 64 filters, each followed by ReLU and max pooling. Single fully-connected layer with 256 units. Total parameters: 1,070,794.

FitNet Student. Three convolutional layers with 32, 48, and 64 filters. The second layer (48 filters) serves as the "guided layer" matching the teacher's first convolutional layer (64 filters). A 1×1 convolutional regressor maps student to teacher channels during Stage 1 hint training. Total parameters: 83,450.

4 Results on MNIST

Table 1 summarizes the performance of all models on the MNIST test set.

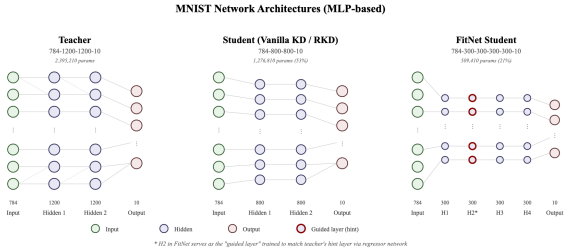

Figure 1: *MLP Architectures*

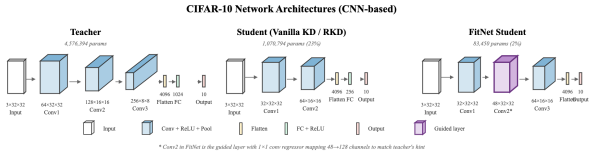

Figure 2: *CNN Architectures*

Table 1: MNIST Test Results

Model	Err.	Acc.	Params	Latency (ms)
Teacher	105	98.95	2.4M	0.216
Student Normal	123	98.77	1.3M	0.167
Vanilla KD	115	98.85	1.3M	0.174
FitNet	117	98.83	509K	0.263
RKD	118	98.82	1.3M	0.154

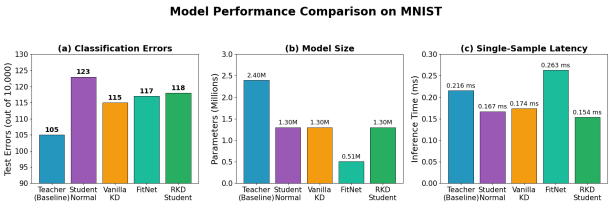

Figure 3: *MNIST Results*

The teacher network achieves 98.95% accuracy (105 errors), while the baseline student without distillation reaches 98.77% (123 errors). [2]

All three distillation methods improve over baseline. Vanilla KD performs best at 98.85% accuracy (115 errors), recovering 44% of the teacher-student gap. RKD and FitNet achieve 98.82% and 98.83% respectively, suggesting soft output matching captures more task-relevant knowledge than embedding distance preservation for MNIST.

Compression-accuracy tradeoffs vary considerably. Vanilla KD and RKD achieve 1.8× compression in params while staying within 0.1% of teacher accuracy. FitNet achieves 4.7× compression with only 0.12% accuracy loss, demonstrating that hint-based training enables thinner networks by trading width for depth.

Inference speed (average forward pass time per image on Tesla T4) shows Vanilla and RKD achieve 1.24–1.40× speedup. FitNet is actually slower than the teacher despite 4.7× fewer parameters - its four sequential layers (versus two in other architectures) increase latency through additional kernel launches, making memory footprint rather than speed its deployment advantage.

[2] At near-saturated accuracy (98–99%), small differences in error counts may reflect training stochasticity rather than systematic effects. We address this with McNemar's test in Section 6.1.

5 Results on CIFAR-10

Table 2 summarizes the performance of all models on the CIFAR-10 test set.

Table 2: CIFAR-10 Test Results

Model	Err.	Acc.	Params	Lat. (ms)
Teacher	1746	82.54	4.6M	0.445
Student Normal	2191	78.09	1.1M	0.344
Vanilla KD	2153	78.47	1.1M	0.317
FitNet	2111	78.89	83K	0.347
RKD	2101	78.99	1.1M	0.311

The teacher network establishes a baseline accuracy of 82.54%, while the normally-trained student achieves 78.09% (2191 errors). All distillation methods provide improvements over this baseline, though a larger gap remains compared to the teacher than observed on MNIST, reflecting the increased task difficulty.

Notably, RKD achieves the best student accuracy at 78.99% (2101 errors), outperforming vanilla KD's 78.47% (2153 errors). This contrasts with MNIST results where vanilla KD was superior, suggesting that preserving relational structure may be more beneficial for complex visual recognition tasks where semantic relationships between object classes are more nuanced. FitNet achieves 78.89% accuracy (2111 errors), performing remarkably well given its dramatically reduced parameter count.

The compression-accuracy tradeoffs reveal striking differences across methods. FitNet demonstrates the most aggressive compression, using only ∼1.8% of the teacher's parameters (83,450 vs 4.5M) while achieving accuracy remarkably close to vanilla KD despite having 13× fewer parameters than other students. The standard student architectures (vanilla KD and RKD) provide more moderate 4.3× compression ratios. This confirms that for complex visual data, depth-based compression can be more parameter-efficient than width-based reduction, though the CNN architecture contributes significantly to this efficiency.

Inference speed measurements show students achieving speedups ranging from 1.28× to 1.43× over the teacher baseline. RKD achieves the best latency at 0.311ms, suggesting its training objective doesn't impose architectural overhead at inference time. Despite FitNet's dramatic parameter reduction, its latency (0.347ms) is slightly higher than vanilla KD and RKD students (∼0.31ms), paralleling the MNIST finding that increased depth creates more sequential operations that offset speed gains from reduced parameters.

Training on CIFAR-10 required more epochs to reach stability compared to MNIST. The RKD student showed the most consistent convergence, suggesting that preserving the relational structure of the CIFAR-10 embedding space provides a more robust learning signal for complex object patterns than point-wise soft target matching.

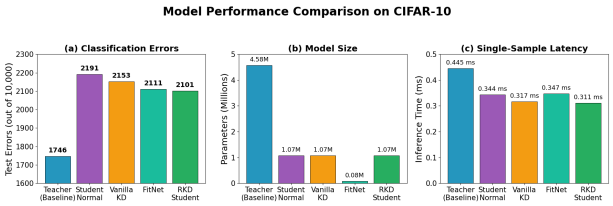

Figure 4: *CIFAR-10 Results*

6 Algorithmic Change

Although the primary focus of this paper is comparing knowledge distillation methods, we also investigate how architectural modifications affect distillation performance.

Knowledge distillation produces lower-magnitude gradients compared to hard labels, making them more susceptible to vanishing during backpropagation. To investigate whether architectural modifications can enhance distillation, we incorporate residual connections and Batch Normalization into all student networks, applied uniformly across Normal Student, vanilla KD, FitNets, and RKD.

We apply this modification exclusively to the MNIST MLP experiments, where uniform hidden layer dimensions allow clean residual additions without the implementation complexities required for CNNs.

For a layer with input x, we replace the standard transformation $h(x) = \sigma(Wx + b)$ with:

$$h(x) = \sigma(\text{BN}(Wx + b)) + x \tag{7}$$

where BN denotes Batch Normalization and the additive x term is the residual connection. The residual path ensures gradient flow of at least $\partial L/\partial x = \partial L/\partial h$ regardless of the learned weights, preventing attenuation of weak distillation signals [4]. Batch Normalization stabilizes activation distributions, improving alignment between student and teacher feature statistics-particularly relevant for FitNets (intermediate activation matching) and RKD (pairwise distance preservation) [5]. To isolate the effect of the architectural change, we used the same pre-trained teacher weights from our baseline experiments. This ensures that any performance difference stems solely from the student's BatchNorm + residual modifications, not from teacher variability.

6.1 MNIST-Results Post Algorithmic Change

Table 3: MNIST Test Results After Algo Change

Model	Err.	Acc.	Params	Lat. (ms)
Teacher	105	98.95	2.4M	0.216
Student Normal	127	98.73	1.3M	0.316
Vanilla KD	108	98.92	1.3M	0.307
FitNet	120	98.80	512K	0.514
RKD	92	99.08	1.3M	0.297

Comparing to the baseline architecture (Table 1) to after algorithmic change (Table 3), RKD improves from 118 to 92 errors (26-error reduction), Vanilla KD from 115 to 108 errors (7-error reduction), while FitNet (117 to 120) and the baseline student (123 to 127) show slight degradations. RKD notably surpasses teacher accuracy (92 vs. 105 errors), which is not contradictory-distillation can act as a regularizer, and Batch Normalization stabilizes activation scales, directly benefiting RKD's distance-based loss by enabling the student to more faithfully preserve the teacher's relational structure.

At near-saturated performance (98–99% accuracy), small differences in error counts may arise from train-

ing stochasticity rather than systematic effects of the architectural modifications. For instance, a 26-error reduction in the RKD could result from correcting 26 previously misclassified samples, or from correcting 100 samples while simultaneously causing 74 new errors-fundamentally different outcomes that yield identical net changes. To distinguish systematic improvements from error exchanges, we apply McNemar's test [6]-comparing each method's predictions before and after the architectural change on the same test set. The test focuses on samples where predictions differ: those that were wrong before but correct after (improved samples, denote as n_{01}), and those that were correct before but wrong after (degraded samples, denote as n_{10}). If the architectural change truly helps, we expect significantly more improvements than degradations. The test statistic is:

$$\chi^2 = \frac{(|n_{01} - n_{10}| - 1)^2}{(n_{01} + n_{10})} \tag{8}$$

A significant p-value ($<$ 0.05) indicates the improvement-to-degradation ratio is too asymmetric to attribute to chance.

Table 4 shows only RKD exhibits statistically significant improvement ($p = 0.0099$): the architectural modifications corrected 60 previously misclassified samples while causing 34 new errors, a nearly 2:1 improvement ratio. In contrast, Vanilla KD ($p = 0.34$: 23 improved, 16 degraded), FitNet ($p = 0.77$: 22 improved, 25 degraded), and baseline student ($p = 0.81$: 73 improved, 77 degraded) show balanced error exchanges, providing no evidence of systematic benefit from the architectural changes.

Table 4: McNemar's test

Method	Improved	Degraded	χ^2	p-val	Sig.
Normal Stud.	73	77	0.060	0.81	No
Vanilla KD	23	16	0.923	0.34	No
FitNet	22	25	0.085	0.77	No
RKD	60	34	6.649	0.0099	Yes

Note: "Improved" = samples wrong before algorithmic change and correct after; "Degraded" = samples correct before, wrong after. Significance threshold: $\alpha = 0.05$.

Model Compression. The architectural modification introduces a small increase in parameter count due to the BatchNorm layers (learnable scale and shift parameters per layer). For the 2×800 students, parameters increased from 1,276,810 to 1,280,010-a negligible 0.25% overhead. FitNet similarly grew from 509,410 to 511,810 parameters. Importantly, the compression ratio relative to the teacher remains largely unchanged: students retain $\sim$53% of teacher parameters, and FitNet remains 4.7$\times$ smaller. Thus, the architectural change does not compromise the fundamental compression benefit of distillation.

Inference Time. The BatchNorm and residual operations introduce notable computational overhead. Student inference time roughly doubled (e.g., RKD: 0.154 $\rightarrow$ 0.297 ms/sample), and FitNet increased from 0.263 to 0.514 ms/sample. This slowdown stems from the additional BatchNorm statistics computation and the residual addition at each layer. Consequently, the modified students are now slower than the teacher (speedup dropped from 1.29$\times$ to 0.52$\times$ for vanilla KD). This presents a deployment trade-off: for RKD, the statistically significant 22% error reduction ($p = 0.0099$) may justify the latency increase in accuracy-critical applications, while for vanilla KD and FitNet-neither showing significant improvements-the doubled latency offers poor cost-benefit ratio. Practitioners should weigh these factors based on whether accuracy or latency is the primary deployment constraint.

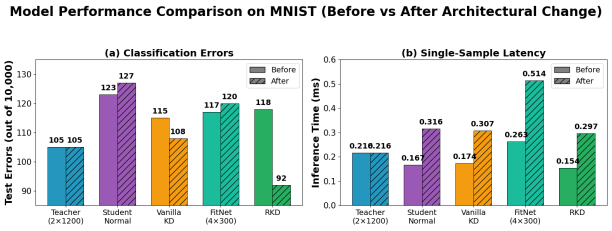

Figure 5: *MNIST Results Before/After Algo Change*

7 Discussion

7.1 Method Comparison Across Datasets

Distillation method effectiveness varies systematically with dataset complexity. On MNIST, vanilla KD achieved highest accuracy (98.85%), slightly outperforming RKD (98.82%) and FitNet (98.83%). On CIFAR-10, this pattern reversed: RKD achieved best performance (78.99%), outperforming vanilla KD (78.47%). This suggests that preserving relational structure becomes more valuable for complex visual tasks where semantic relationships are nuanced, while simpler tasks are well-served by output-level soft targets.

FitNet demonstrated consistent extreme compression across both datasets (4.7$\times$ on MNIST, 55$\times$ on CIFAR-10) while maintaining competitive accuracy, confirming that

depth-guided training compensates for dramatic width reduction. However, FitNet's deeper architecture consistently ran slower than teachers despite fewer parameters, indicating deployment advantages manifest primarily in memory footprint rather than inference speed.

7.2 Factors Affecting Performance

Dataset Complexity. MNIST's ceiling effect (baseline student at 98.77%) left only 0.18% improvement room, causing all methods to cluster tightly. CIFAR-10's larger gap (4.45%) allowed clearer differentiation, with methods spanning 0.52% accuracy range.

Architecture Matching. Vanilla KD and RKD use identically-structured students differing only in width, enabling straightforward transfer. FitNet's thinner-deeper architecture requires explicit intermediate supervision to bridge structural gaps and prevent optimization failure from vanishing gradients in narrow layers.

Architectural Modifications. RKD benefited significantly from residual connections and Batch Normalization (22% error reduction, $p = 0.0099$, surpassing teacher at 99.08%) because normalization stabilizes activation scales, directly improving pairwise distance preservation. Vanilla KD showed non-significant improvement ($p = 0.34$), while FitNet degraded slightly ($p = 0.77$), suggesting residual connections may disrupt the activation space that hint matching expects during two-stage training.

7.3 Practical Considerations

Choose Vanilla KD when: working with simple-to-moderate tasks, architecturally similar teacher-student pairs, prioritizing implementation simplicity, or targeting moderate compression (2–4×). Provides reliable baseline performance with single-stage training.

Choose FitNets when: memory footprint is the critical deployment constraint, requiring maximum compression (4–55×), can afford increased inference latency and two-stage training complexity. Best for edge devices with strict storage limits.

Choose RKD when: working with complex visual tasks with rich semantic structure, using Batch Normalization (strong synergy), embedding quality matters for downstream tasks, or avoiding temperature hyperparameter tuning. Particularly effective when teacher-student capacity gap is large.

8 Conclusions

We systematically compared three knowledge distillation methods-vanilla KD, FitNets, and RKD-across MNIST (MLP) and CIFAR-10 (CNN), providing controlled evaluation of accuracy-compression-speed trade-offs.

Key findings reveal three factors governing distillation effectiveness: (1) Teacher-student performance gap-larger gaps enable clearer knowledge transfer benefits, explaining why CIFAR-10 showed stronger method differentiation than MNIST; (2) Architectural compatibility-both structural similarity between teacher-student and synergy between student architecture and distillation mechanism (normalization benefits RKD's distance-based loss but disrupts FitNet's hint matching); (3) Task semantic richness-output distributions suffice for simple patterns, but relational structure captures more information in complex visual recognition.

The architectural modification experiment revealed method-specific responses: RKD achieved statistically significant 22% error reduction ($p = 0.0099$), even surpassing the teacher, while vanilla KD and FitNet showed no significant benefits. This demonstrates that architectural enhancements must align with distillation objectives rather than being universally beneficial.

For practitioners, these findings translate to clear selection criteria based on deployment constraints and task characteristics. Future work could extend this comparison to progressive distillation methods, such as online distillation where students and teachers are trained simultaneously, or self-distillation where a model learns from its own predictions, to investigate whether these approaches offer different accuracy-compression trade-offs or benefit differently from architectural modifications like residual connections and Batch Normalization.

Code availability: The code used for all experiments and analyses in this study is publicly available at our repo: https://github.com/Yaronot/KD-comparative-study

References

[1] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," arXiv preprint arXiv:1503.02531, 2015.

[2] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: Hints for thin deep nets," arXiv preprint arXiv:1412.6550, 2014.

[3] W. Park, D. Kim, Y. Lu, and M. Cho, "Relational knowledge distillation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3967–3976.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *CVPR*, 2016.

[5] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *ICML*, 2015.

[6] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Computation*, vol. 10, no. 7, pp. 1895–1923, 1998.