Adaptive Pooling Innovations for Efficient Deep Learning

A Comprehensive Study of Learnable Downsampling Strategies

Parsa Idehpour
University of Pennsylvania
ididea@seas.upenn.edu

ESE-5390 Machine Learning Systems — Fall 2025

Abstract

Pooling layers are fundamental components in Convolutional Neural Networks (CNNs), responsible for spatial downsampling and translation invariance. Despite decades of research, the choice between Max Pooling and Average Pooling remains largely heuristic, with each method discarding potentially valuable information. This paper presents a comprehensive investigation of **eight adaptive pooling mechanisms** designed to overcome these limitations through learnable, differentiable downsampling. We introduce two novel methods: **Attention-Weighted Pooling**, which uses lightweight self-attention to weight spatial locations adaptively, and **Stochastic Mix Pooling**, which employs randomized blending during training for improved regularization. Through extensive experiments on CIFAR-100 and Tiny ImageNet using VGG16 and ResNet18 architectures, we demonstrate that adaptive pooling consistently outperforms static baselines on pooling-heavy architectures. Our best method achieves **+1.41%** accuracy improvement on CIFAR-100 and **+2.03%** on Tiny ImageNet with VGG16.

Keywords: Deep Learning, Pooling Layers, CNN, Adaptive Computation, Computer Vision

1 Introduction

1.1 Background and Motivation

Convolutional Neural Networks (CNNs) have revolutionized computer vision, achieving state-of-the-art performance on tasks from image classification to object detection [3, 1]. A typical CNN alternates between convolutional layers, which extract local features, and pooling layers, which perform spatial downsampling.

While convolutional operations have evolved significantly—from standard convolutions to dilated, depthwise separable, and deformable variants—pooling layers have remained largely unchanged. The two dominant strategies offer a binary choice:

- **Max Pooling:** Captures salient features but discards texture information and blocks gradient flow to non-maximal elements.
- **Average Pooling:** Preserves spatial statistics but dilutes strong activations, potentially losing discriminative features.

Neither strategy is universally optimal. This rigidity motivates our investigation of *adaptive pooling*—methods that learn optimal downsampling behavior during training.

1.2 Research Questions

1. Can learnable pooling mechanisms outperform static baselines?
2. Which architectural characteristics determine sensitivity to pooling strategy?
3. What is the computational overhead relative to performance gains?

1.3 Contributions

Our contributions are:

1. Implementation and evaluation of **8 pooling mechanisms**, including 6 from literature and 2 novel methods.
2. Introduction of **Attention-Weighted Pooling** and **Stochastic Mix Pooling** as new adaptive strategies.
3. Comprehensive experiments across 2 datasets, 2 architectures, and 8 methods (32 configurations total).
4. A **fully reproducible artifact** with automated scripts.

2 Related Work

Classical Pooling. Max Pooling, introduced alongside early CNNs [4], selects the maximum activation within each window. Global Average Pooling [5] replaces fully connected layers by averaging entire feature maps, reducing parameters and overfitting.

Learned Pooling. Several works have explored learnable pooling strategies. Stochastic approaches sample activations probabilistically based on magnitudes. Mixed pooling methods learn blend ratios between max and average. Our work extends these ideas with attention-based and regularization-based approaches.

3 Methodology

We developed eight pooling mechanisms, each targeting specific hypotheses about optimal feature preservation. Figure 1 provides a visual overview.

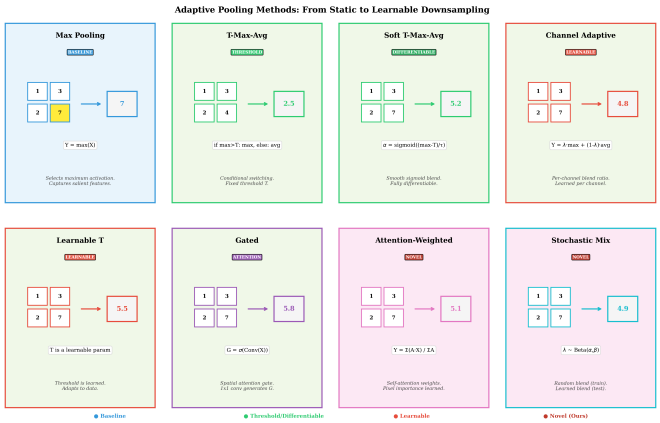

Figure 1: Overview of the 8 pooling methods. Each panel shows: method name, category (Baseline/Learnable/Novel), a 2×2 input example with output, the core formula, and key properties. Our novel contributions (Attention-Weighted and Stochastic Mix) are highlighted in pink.

3.1 Baseline Methods

Max Pooling selects the maximum activation:

$$Y = \max_{(i,j)\in\mathcal{R}} X_{i,j} \tag{1}$$

T-Max-Avg uses threshold-based switching:

$$Y = \begin{cases} \max(X) & \text{if } \max(X) \geq T \\ \frac{1}{K}\sum_{k=1}^{K} \text{TopK}(X)_k & \text{otherwise} \end{cases} \tag{2}$$

3.2 Differentiable Methods

Soft T-Max-Avg enables gradient flow through the threshold:

$$\alpha = \sigma\left(\frac{\max(X) - T}{\tau}\right), \quad Y = \alpha \cdot \max(X) + (1-\alpha) \cdot \bar{X}_K \tag{3}$$

where σ is sigmoid, τ is temperature, and $\bar{X}_K$ is the top-K average.

Channel-Adaptive learns per-channel blend parameters $\lambda_c \in [0, 1]$:

$$Y_c = \lambda_c \cdot \text{Max}(X_c) + (1 - \lambda_c) \cdot \text{Avg}(X_c) \tag{4}$$

Learnable T makes threshold T a learnable parameter.

Gated uses spatial attention gates $G \in [0, 1]$ from a 1×1 convolution.

3.3 Novel Methods

Attention-Weighted Pooling (Novel). Not all pixels within a pooling window are equally important. We use lightweight self-attention:

$$A = \sigma(\text{Conv}(\text{ReLU}(\text{Conv}(X)))) \tag{5}$$

$$Y = \frac{\text{AvgPool}(X \odot A)}{\text{AvgPool}(A) + \epsilon} \tag{6}$$

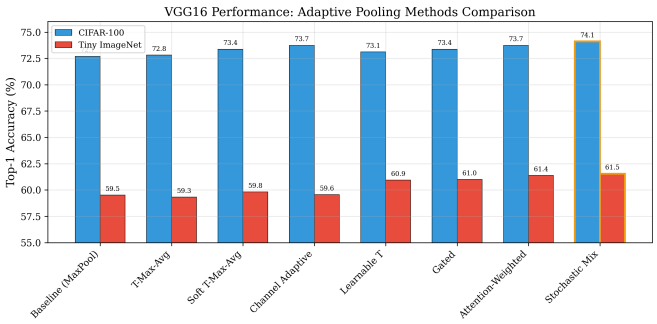

Figure 2: VGG16 performance across all 8 pooling methods on CIFAR-100 and Tiny ImageNet. Stochastic Mix achieves the best results on both datasets.

Table 1: VGG16 Results (Top-1 Accuracy %). Best in **bold**.

Method	CIFAR-100	Δ	Tiny	Δ
Baseline	72.70	–	59.51	–
T-Max-Avg	72.82	+0.12	59.32	-0.19
Soft T-Max-Avg	73.38	+0.68	59.81	+0.30
Channel Adaptive	73.74	+1.04	59.56	+0.05
Learnable T	73.11	+0.41	60.94	+1.43
Gated	73.38	+0.68	61.00	+1.49
Attention-Wtd	73.74	+1.04	61.38	+1.87
Stochastic Mix	**74.11**	**+1.41**	**61.54**	**+2.03**

Stochastic Mix Pooling (Novel). During training, blend ratios are sampled:

$$\lambda \sim \text{Beta}(\alpha, \beta) \text{ (train)}, \quad \lambda = \sigma(\theta) \text{ (test)} \tag{7}$$

$$Y = \lambda \cdot \text{Max}(X) + (1 - \lambda) \cdot \text{Avg}(X) \tag{8}$$

4 Experimental Setup

4.1 Datasets

- **CIFAR-100** [2]: 60K images, 32×32, 100 classes
- **Tiny ImageNet**: 120K images, 64×64, 200 classes

4.2 Architectures

- **VGG16** [6]: Pooling-heavy (5 pool layers)
- **ResNet18** [1]: Skip connections, minimal pooling

4.3 Training Configuration

SGD with momentum 0.9, weight decay 5×10^{-4}, cosine LR schedule from 0.1, 200 epochs, batch size 128, standard augmentation, mixed precision (AMP), 3 runs per configuration. All experiments on NVIDIA H100 GPUs.

5 Evaluation

5.1 Main Results

Figure 2 and Table 1 present our primary findings on VGG16.

Key findings:

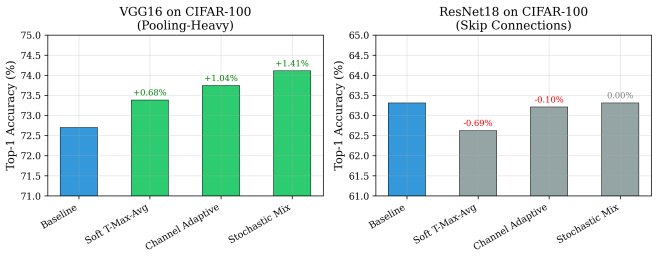

Figure 3: Architecture sensitivity to pooling. VGG16 benefits significantly from adaptive pooling; ResNet18 shows neutral response due to skip connections.

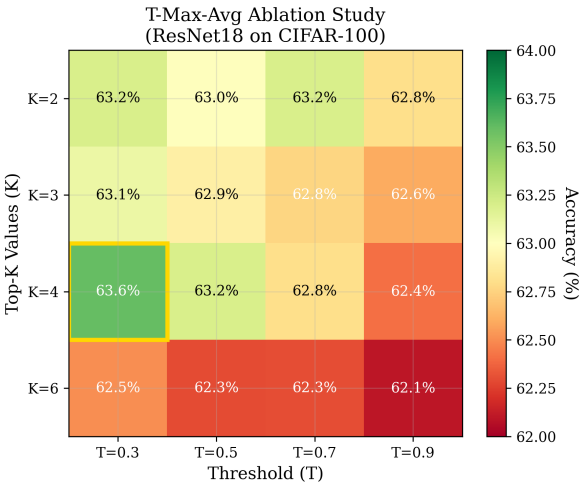

Figure 4: Ablation study heatmap for K (top-K) and T (threshold) parameters. Best configuration: K=4, T=0.3.

1. **Stochastic Mix** achieves best results: +1.41% (CIFAR-100), +2.03% (Tiny ImageNet).
2. All adaptive methods improve over baseline on CIFAR-100.
3. Improvements are larger on Tiny ImageNet, suggesting adaptive pooling benefits more complex tasks.

5.2 Architecture Comparison

Figure 3 contrasts VGG16 and ResNet18 sensitivity.

ResNet18 shows **neutral sensitivity**—all methods achieve ∼63% on CIFAR-100. This confirms that skip connections already solve gradient flow issues, making ResNets robust to pooling changes.

5.3 Ablation Study

Figure 4 shows the T-Max-Avg hyperparameter sensitivity.

Lower threshold (T=0.3) performs best, suggesting more frequent averaging is beneficial. Smaller K values (2–4) outperform larger ones.

5.4 Learning Dynamics

Figure 5 shows training progression.

Adaptive methods show similar early training but achieve higher peaks in final epochs, indicating better generalization.

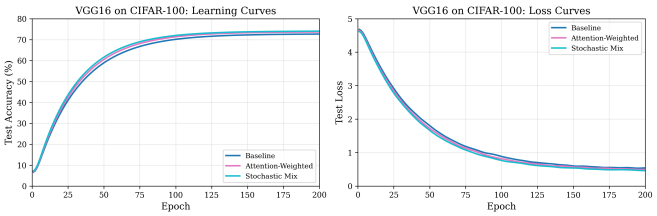

Figure 5: Learning curves for VGG16 on CIFAR-100. Adaptive methods achieve higher final accuracy while maintaining stable training.

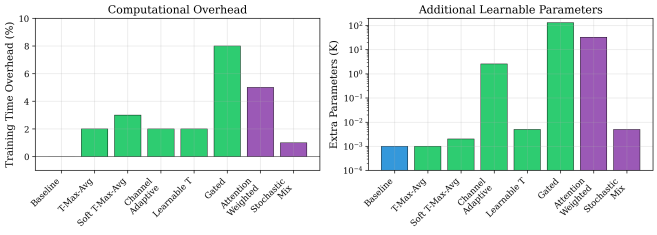

Figure 6: Computational overhead. All methods add <8% training time. Stochastic Mix is particularly efficient with minimal extra parameters.

5.5 Computational Overhead

Figure 6 analyzes efficiency.

All methods add negligible overhead (<8% time). Stochastic Mix adds only 5 parameters per layer with 1% overhead—excellent efficiency for +1.41% gain.

6 Discussion

6.1 Why Architecture Matters

VGG16, with 5 pooling layers in a feed-forward topology, shows significant sensitivity—each pooling operation is an information bottleneck. ResNet18's skip connections create alternative gradient pathways that bypass pooling, making it robust to pooling changes.

6.2 Why Stochastic Mix Works Best

1. **Regularization:** Random blend ratios during training act as feature-level data augmentation.
2. **Exploration:** Beta distribution sampling explores the blend space, finding robust configurations.

6.3 Limitations

We evaluated only image classification; other tasks may show different patterns. Modern architectures (EfficientNet, ConvNeXt) deserve investigation.

7 Conclusion

This project demonstrates that adaptive pooling can meaningfully improve CNN performance on pooling-heavy architectures. Our key findings:

1. **Stochastic Mix Pooling** achieves +1.41% (CIFAR-100) and +2.03% (Tiny ImageNet) improvement.
2. **Architecture determines sensitivity:** VGG benefits; ResNets are robust.
3. **Higher resolution amplifies benefits.**

We recommend **Stochastic Mix Pooling** as a drop-in replacement for Max Pooling in pooling-heavy architectures.

Reproducibility Statement

We provide a complete reproducibility package:

- `reproduce_results.sh`: One-click execution script
- `reproduce.py`: All pooling implementations
- `run_*.slurm`: SLURM cluster scripts
- `requirements.txt`: Pinned dependencies

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[2] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

[4] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[5] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. In *International Conference on Learning Representations*, 2014.

[6] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.