Improving MLP-Mixer Performance on Small Image Datasets Using Locality-Aware Modifications

George Kanazi Georgek@post.bgu.ac.il
Ammar Mnaa mnaa@post.bgu.ac.il
Adam Eisa eisa@post.bgu.ac.il

Abstract

The MLP-Mixer architecture challenges the dominance of Convolutional Neural Networks (CNNs) and Transformers in computer vision by relying entirely on Multi-Layer Perceptrons (MLPs). However, while MLP-Mixer achieves state-of-the-art results on massive datasets (e.g., JFT-300M), it struggles to generalize on small datasets due to a lack of inductive bias, specifically the inability to capture local spatial relationships efficiently. In this project, we propose a "Hybrid Mixer" architecture that injects a lightweight spatial inductive bias using a Depthwise Separable Convolutional stem immediately after patch embedding. Our experiments on CIFAR-10 and CIFAR-100 demonstrate that this minimal architectural change significantly improves generalization, achieving a $+5.56\%$ and $+6.83\%$ accuracy increase respectively, while adding negligible parameters compared to standard convolutions.

Code repository: `https://github.com/GeorgeKanazi/HybridMixer/`

1 Introduction

Deep learning for computer vision has long been dominated by Convolutional Neural Networks (CNNs), which possess a strong inductive bias for processing grid-structured data: they assume that pixels usually share relationships with their immediate neighbors. Recently, Vision Transformers (ViT) and the MLP-Mixer [1] have shown that it is possible to achieve competitive results without standard convolutions, provided the model is trained on massive datasets.

However, a significant gap remains when applying these modern architectures to "small" data regimes. The standard MLP-Mixer treats image patches as independent tokens and must learn spatial relationships from scratch. Without millions of training examples, the model tends to overfit, failing to capture the local structures (edges, textures) that CNNs extract naturally.

In this project, we address this limitation by proposing a novel extension to the MLP-Mixer: a **Hybrid Convolutional Stem**. We introduce a locality-aware processing block consisting of a 3×3 Depthwise Convolution followed by a 1×1 Pointwise Convolution. This modification is designed to "smooth" the patch representations using local neighbor information before the global mixing layers begin processing. Our approach bridges the gap between the efficiency of MLPs and the spatial awareness of CNNs.

2 Background

2.1 The MLP-Mixer Architecture

The MLP-Mixer architecture is based on two distinct mixing layers applied to input patches:

- **Token-mixing MLPs:** These act on the columns of the input matrix (mixing spatial locations). They allow communication between different patches across the image.
- **Channel-mixing MLPs:** These act on the rows (mixing features). They allow communication between different channels within a single patch.

Mathematically, if the input X has shape (S, C) where S is the number of patches (sequence length) and C is the hidden dimension, the mixer block performs:

$$U = X + W_2\sigma(W_1 \text{LayerNorm}(X)^T)^T \tag{1}$$

$$Y = U + W_4\sigma(W_3 \text{LayerNorm}(U)) \tag{2}$$

where σ is a nonlinearity (GELU). Crucially, the Token-mixing MLP treats all patches equally and globally, lacking the concept of "neighbors."

2.2 The Inductive Bias Problem

Inductive bias refers to the set of assumptions a model makes about the data to predict outputs for unseen inputs. CNNs have a *translation equivariance* and *locality* bias. The MLP-Mixer lacks this. On small datasets like CIFAR-10 (50k images), the Mixer struggles to learn that pixel (i, j) is correlated with pixel $(i + 1, j)$. This results in slower convergence and lower validation accuracy compared to CNN baselines.

3 Contributions and Innovations

Our primary contribution is the design and implementation of a **Hybrid Stem** that injects locality with minimal computational overhead.

3.1 Architectural Modification

In the standard MLP-Mixer, the input image is linearly projected into patches and immediately flattened into a sequence. We modify this pipeline by inserting a convolutional block *before* the flattening operation, as illustrated in Figure 1.

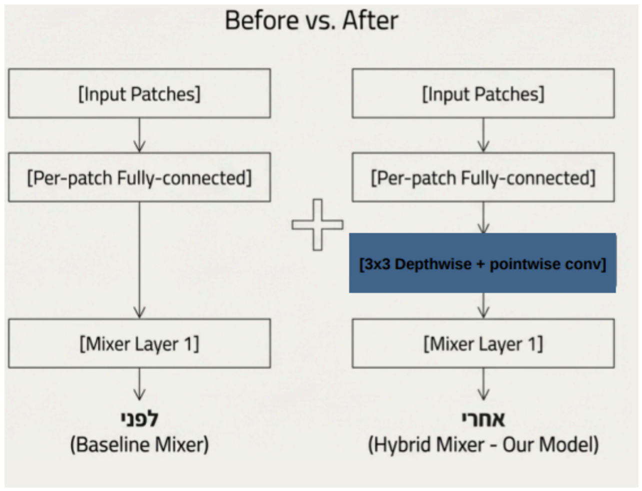
Before vs. After
[Input Patches]
[Per-patch Fully-connected]
[Mixer Layer 1]
לפני
(Baseline Mixer)
[Input Patches]
[Per-patch Fully-connected]
[3x3 Depthwise + pointwise conv]
[Mixer Layer 1]
אחרי
(Hybrid Mixer - Our Model)

Figure 1: Architectural Comparison: The standard MLP-Mixer pipeline (Left) vs. our Hybrid Mixer (Right). We insert a Depthwise Separable Convolutional block immediately after the patch embedding to inject spatial awareness.

Our added block is a **Depthwise Separable Convolution**, composed of two layers:

1. **Depthwise Convolution** (3×3)**:** This layer applies a single convolutional filter per input channel. It captures spatial context by aggregating information from the 8 immediate neighbors of every patch.

2. **Pointwise Convolution** (1×1)**:** This layer projects the channels linearly, mixing the features learned in the depthwise step.

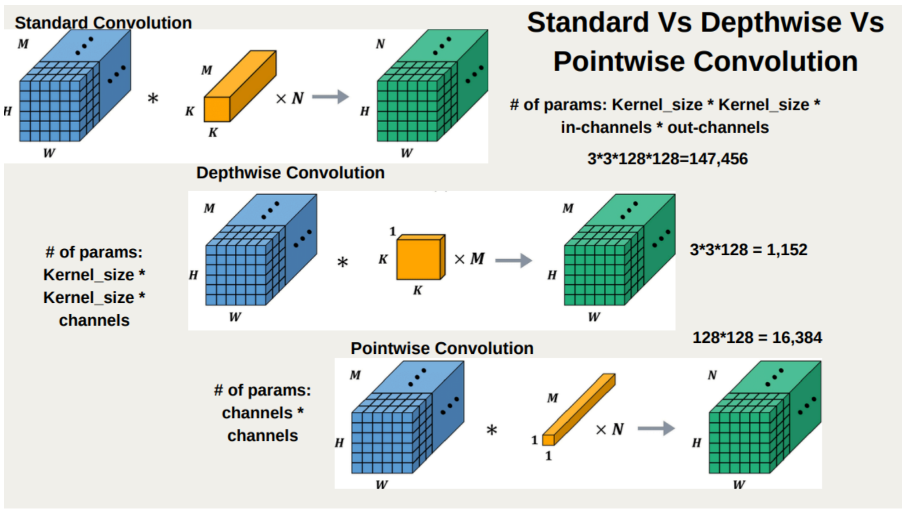
Standard Vs Depthwise Vs Pointwise Convolution
Standard Convolution
# of params: Kernel_size * Kernel_size * in-channels * out-channels
3*3*128*128=147,456
Depthwise Convolution
# of params: Kernel_size * Kernel_size * channels
3*3*128 = 1,152
Pointwise Convolution
128*128 = 16,384
# of params: channels * channels
M
N
H
W
K
1
× N
× M

Figure 2: Efficiency Mechanism: Visualizing the difference between a Standard Convolution and our Depthwise Separable approach. By decoupling spatial mixing (Depthwise) from feature mixing (Pointwise), we significantly reduce the parameter count.

This design, visualized in Figure 2, allows the model to form 'spatially aware' patch representations before they enter the permutation-invariant Mixer layers.

3.2 Parameter Efficiency Analysis

A key constraint of our project was to improve performance without bloating the model size. A standard convolution would be prohibitively expensive in terms of parameters.

For a kernel size K, input channels C_{in} and output channels C_{out}, a standard convolution requires:

$$\text{Params}_{std} = K \times K \times C_{in} \times C_{out} \tag{3}$$

For our configuration ($K = 3, C = 128$), this results in $\approx 147,456$ parameters.

In contrast, our Depthwise Separable approach requires:

$$\text{Params}_{dw} = K \times K \times C_{in} \tag{4}$$
$$\text{Params}_{pw} = 1 \times 1 \times C_{in} \times C_{out} \tag{5}$$
$$\text{Total} = (3 \times 3 \times 128) + (1 \times 1 \times 128 \times 128) \approx 17,536 \tag{6}$$

This represents an $\approx 88\%$ **reduction** in parameters compared to a standard convolutional layer, maintaining the "lightweight" philosophy of the MLP-Mixer.

4 Results

We evaluated our Hybrid Mixer against the baseline MLP-Mixer on the CIFAR-10 and CIFAR-100 datasets. Both models were trained for 30 epochs using the Cross Entropy loss function.

4.1 Accuracy Comparison

As shown in Table 1 and Figure 3, our Hybrid model significantly outperformed the baseline on both datasets. The improvement is particularly notable on CIFAR-100, which is a harder task, suggesting that the added spatial bias helps the model learn more robust features that generalize better to diverse classes.

Table 1: Validation Accuracy Comparison

Dataset	Baseline Accuracy	Hybrid (Ours) Accuracy	**Improvement** (Δ)
CIFAR-10	76.71%	**82.27%**	+5.56%
CIFAR-100	48.20%	**55.03%**	+6.83%

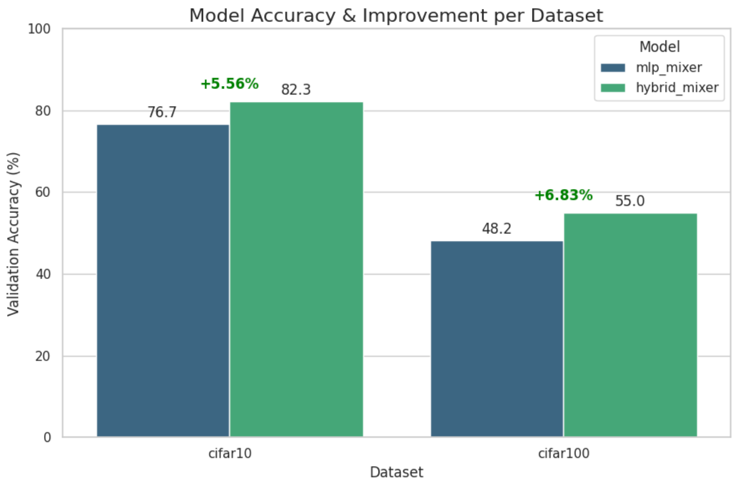

Figure 3: Validation Accuracy Comparison. Our Hybrid model (Green) consistently outperforms the Baseline MLP-Mixer (Blue) on both CIFAR-10 and CIFAR-100 datasets.

4.2 Training Dynamics

We monitored the Training and Validation loss curves to assess the learning stability and convergence speed.

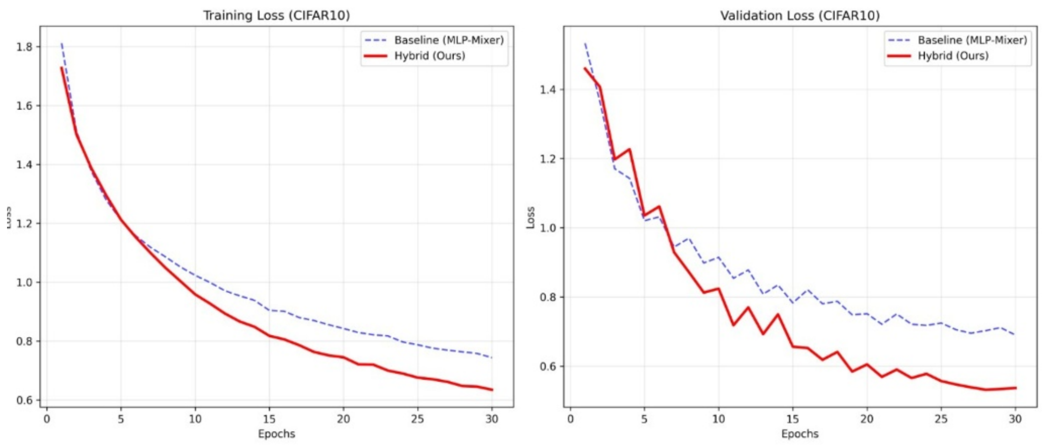

Figure 4: Training and Validation Loss comparison on CIFAR-10. The Hybrid model (Red) consistently achieves lower loss values than the Baseline (Blue dashed) across both metrics.

As illustrated in Figure 4, both models exhibit a stable training process with a similar gap between training and validation loss. However, the Hybrid Mixer (Red solid line) achieves consistently lower loss values than the Baseline (Blue dashed line) throughout the entire training duration.

Crucially, the Hybrid model's validation loss drops more rapidly in the early epochs and settles at a significantly lower final value (≈ 0.55 vs ≈ 0.70). This indicates that the added spatial inductive bias allows the model to optimize more efficiently and learn more robust features, leading to superior generalization performance (lower absolute error) without the need for additional regularization to close the generalization gap.

5 Conclusions and Future Directions

In this work, we demonstrated that the performance of MLP-Mixer on small datasets is hindered by a lack of spatial inductive bias. By introducing a lightweight Depthwise Separable Convolutional stem, we successfully injected locality into the model without significantly increasing the parameter count.

Our results show a clear improvement in generalization on CIFAR-10/100. This suggests that "Hybrid" architectures—combining the efficiency of MLPs/Transformers with the local feature extraction of CNNs—are a promising direction for efficient computer vision.

Future Directions:

- **Inference Speed:** While we optimized for parameter count, future work should benchmark the inference latency, as adding layers (even lightweight ones) adds sequential operations.
- **Interleaved Convolutions:** We experimented with adding the stem at the beginning. Future work could explore interleaving these convolutional blocks inside the Mixer layers (e.g., every 3rd block) to periodically re-introduce spatial awareness deep in the network.
- **Larger Datasets:** Testing if these gains persist on medium-sized datasets like ImageNet-1k, and how well the hybrid model performs on large scale models like JFT-300M.

References

[1] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34:24261–24272, 2021.