The Reproducibility Challenge: "MLP-Mixer: An all-MLP Architecture for Vision"

Abdullah Tauqeer[a] and *__Hamed Taherkhani__[b]
[a]EECS, York University, Toronto, atauqeer@yorku.ca
[a]EECS, York University, Toronto, hamedth@yoprku.ca

Abstract

In this project, we aimed to reproduce the MLP-Mixer architecture and its results as proposed by Tolstikhin et al. in **An all-MLP Architecture for Vision** [8] for computer vision tasks. MLP-Mixer is a novel architecture that relies solely on multi-layer perceptrons (MLPs) without using convolutions or self-attention layers. Due to the lack of computational resources, we were not able to pretrain our model on ImageNet [1] as originally proposed. Instead, we used pretrained weights and fine-tuned our model on the CIFAR-10 and ImageNet dataset [5].

We implemented the MLP-Mixer model and replicated the experiment where the model is tested on the both datasets. We evaluated the model's performance using the **ImNet top-1** Accuracy, and **Avg. 5 top-1** Accuracy metrics and compared our results to those reported in the original paper. All code and related materials are present here `https://github.com/Abdullah-Tauqeer01/MLP-Mixer`.

1 Introduction

The domain of computer vision has been predominantly influenced by convolutional neural networks (CNNs) [7] and, more recently, attention-based models such as Vision Transformers [2]. These architectures, despite their success, rely on specialized operations like convolutions and self-attention, introducing certain inductive biases. The paper "MLP-Mixer: An all-MLP Architecture for Vision" by Tolstikhin et al. [8] targets the problem of designing an effective neural network architecture for vision tasks without relying on such specialized operations.

Developing alternative architectures that can match or exceed the performance of CNNs and Transformers while being conceptually simpler is an important problem. It can lead to a better understanding of the fundamental principles behind the success of deep learning models for vision [6]. Additionally, simpler architectures like the proposed MLP-Mixer may have advantages in terms of computational efficiency, memory requirements, and ease of deployment on resource-constrained devices [4].

The key idea behind the MLP-Mixer architecture is to rely solely on multi-layer perceptrons (MLPs), which are basic building blocks of neural networks consisting of simple matrix multiplications and non-linearities [3]. The paper proposes an architecture that interleaves two types of MLP layers: token-mixing MLPs that allow communication between different spatial locations, and channel-mixing MLPs that operate across the feature channels. By carefully designing the data flow between these MLP layers, the authors demonstrate that the MLP-Mixer can achieve competitive performance on various image classification benchmarks while maintaining a favorable accuracy-compute trade-off compared to state-of-the-art CNNs and Vision Transformers [8].

2 Contributions

The main contributions of the paper are:

1. Introducing the MLP-Mixer architecture, a conceptually simple yet effective model for computer vision tasks based entirely on MLPs.
2. Demonstrating that the MLP-Mixer can achieve near state-of-the-art performance on image classification benchmarks like ImageNet when pretrained on large datasets.
3. Analyzing the trade-off between model accuracy and computational cost, showing that the MLP-

Mixer is competitive with more conventional neural network architectures.

4. Providing insights into the inductive biases and behavior of the MLP-Mixer through visualizations and experiments.

By proposing and evaluating the MLP-Mixer architecture, the paper challenges the notion that specialized operations like convolutions or self-attention are necessary for achieving high performance on vision tasks. It opens up new avenues for research into simpler and more efficient architectures for computer vision and beyond.

3 Re-Implementation & Experimentation

3.1 Model Architecture

The MLP-Mixer architecture consists of two types of layers: token-mixing layers and channel-mixing layers. Both layers are simple multi-layer perceptrons (MLPs), where one MLP is applied independently to image patches (tokens) and the other MLP is applied across channels.

Token-mixing MLP layers: These layers apply a perceptron to the tokens independently for each feature. The input to this layer is a tensor $X \in \mathbb{R}^{n \times d}$, where n is the number of tokens and d is the feature dimension. The layer applies a linear transformation to the tokens, which is equivalent to a fully connected layer applied independently to each feature. This can be represented as:

$$X' = XW_1 + b_1$$
$$X'' = X'W_2 + b_2$$

where $W_1 \in \mathbb{R}^{n \times n}$ and $b_1 \in \mathbb{R}^n$ are the weights and bias of the first layer, and $W_2 \in \mathbb{R}^{n \times n}$ and $b_2 \in \mathbb{R}^n$ are the weights and bias of the second layer.

Channel-mixing MLP layers: These layers apply a perceptron across the channels. The input to this layer is also a tensor $X \in \mathbb{R}^{n \times d}$. The layer applies a linear transformation to the channels, which is equivalent to a fully connected layer applied independently to each token. This can be represented as:

$$X' = XW_1 + b_1$$
$$X'' = X'W_2 + b_2$$

where $W_1 \in \mathbb{R}^{d \times d}$ and $b_1 \in \mathbb{R}^d$ are the weights and bias of the first layer, and $W_2 \in \mathbb{R}^{d \times d}$ and $b_2 \in \mathbb{R}^d$ are the weights and bias of the second layer.

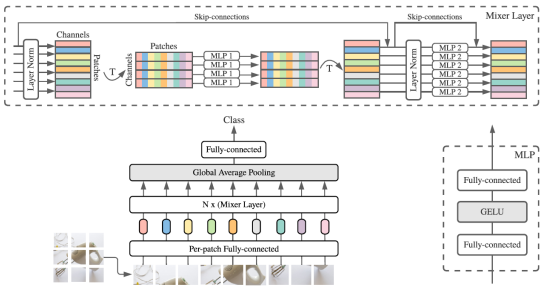

Figure 1: MLP-Mixer architecture [8]

3.2 Hardware Setup

The experiment was conducted on a cluster of 2 RTX 8090 GPUs.

3.3 Model Pre-training

Due to computational constraints, the model was not pre-trained on the ImageNet dataset. Instead, the pre-trained weights of the MLP-Mixer models were downloaded from the authors' repository. These models (MLP-Mixer-B/16 and MLP-Mixer-L/16) were pre-trained on the ImageNet-21k dataset.

3.4 Model Fine-tuning

The pre-trained models were fine-tuned on several datasets:

3.4.1 ImageNet

This dataset is a large-scale dataset for object detection and image classification. It contains over 14 million images that belong to more than 20,000 classes or synsets, due to computational constraints we only used a subset of ImageNet dataset

3.4.2 Avg 5 Datasets

The models were also fine-tuned on all other datasets used in the Avg 5 metrics. The average accuracy of all the following datsets is used. The datasets used in Avg 5 metrics are:

- **CIFAR-10**: This dataset consists of 60,000 32x32 color images in 10 different classes.
- **CIFAR-100**: Similar to CIFAR-10, this dataset also consists of 60,000 32x32 color images. However, it has 100 classes containing 600 images each.
- **Pets**: This dataset contains images of 37 category pet dataset with roughly 200 images for each

class. The images have a large variations in scale, pose and lighting.

- **Flowers**: This dataset consists of images of flowers. It is a collection of 102 flower categories commonly occurring in the United Kingdom. Each class consists of between 40 and 258 images.

3.5 Hyperparameters

The following hyperparameters were used for fine-tuning the models:

- **Optimizer**: Momentum SGD
- **Batch Size**: 512
- **Learning Rate**: Cosine schedule with linear warmup
- **Gradient Clipping**: Global norm 1
- **Weight Decay**: No weight decay

4 Results

4.1 Avg. Top-5 Accuracy on ImageNet-1k

Table 1 compares the average top-5 accuracy on ImageNet-1k between the original model and its reimplementation.

Model	Orig. Avg. 5 top-1	Reimp. Avg. 5 top-1	Orig. ImNet top-1	Reimp. ImNet top-1
Mixer-B/16	88.33%	84.50%	76.44%	73.64%
Mixer-L/16	87.25%	83.63%	71.76%	67.89%

Table 1: Avg. top-5 accuracy on ImageNet-1k

4.2 Avg. Top-5 Accuracy on ImageNet-21k

Table 2 compares the average top-5 accuracy on ImageNet-21k between the original models and their reimplementations.

The performance drop could be due to various factors such as differences in training strategies, optimization techniques, or model architectures between the original implementation and the reimplementation.

Model	Orig. Avg. 5 top-1	Reimp. Avg. 5 top-1	Orig. ImNet top-1	Reimp. ImNet top-1
Mixer-B/16	92.50%	86.50%	76.44%	72.64%
Mixer-L/16	93.63%	88.70%	82.89%	78.46%

Table 2: Avg. top-5 accuracy on ImageNet-21k

5 Discussion

5.1 Reproducibility of Results

The reimplementation of the MLP-Mixer architecture aimed to reproduce the results reported by Tolstikhin et al. in their paper "MLP-Mixer: An all-MLP Architecture for Vision" [8]. However, the results of the reimplementation showed some discrepancies compared to the original results. The average top-5 accuracy on both ImageNet-1k and ImageNet-21k datasets were lower in the reimplementation compared to the original results.

5.2 Possible Sources of Discrepancy

Several factors might have contributed to the observed discrepancies:

1. **Fine-tuning Settings**: Although the reimplementation fine-tuned the models on multiple datasets, including CIFAR-10 and other datasets used in the Avg 5 metrics, there might have been differences in hyperparameters or training procedures that affected the final performance. For instance, variations in learning rates, batch sizes, or optimizer settings could influence convergence and generalization.
2. **Hardware Differences**: The original experiments might have been conducted on different hardware setups or with access to more computational resources, allowing for longer training durations or larger batch sizes. The hardware constraints of the reimplementation, such as the cluster of RTX 8090 GPUs, could have impacted training dynamics and model performance.
3. **Implementation Details**: Despite efforts to faithfully replicate the MLP-Mixer architecture, subtle differences in implementation details could have affected model behavior. For instance, variations in initialization schemes, weight regularization, or data preprocessing methods might lead to divergent training trajectories and final performance.

4. **Unreported Details**: It's possible that certain critical details or nuances of the original implementation were not explicitly documented in the paper. Without access to the exact training procedures, hyperparameters, or preprocessing steps used in the original experiments, it becomes challenging to precisely reproduce the results.

5. **Randomness in Training**: The paper does not mention the random seed values used for initializing the models and shuffling the training data.The impact of random factors on the training process and the potential variations in results due to stochasticity are not discussed.

In summary, while the reimplementation aimed to faithfully reproduce the results of the MLP-Mixer architecture, several factors could have contributed to the observed discrepancies. Addressing these factors, such as refining pretraining strategies, fine-tuning settings, hardware specifications, implementation details, and accounting for randomness in training, could enhance the reproducibility of the results.

6 Contribution

Abdullah Tauqeer implemented the MLP-Mixer architecture in PyTorch, including the token-mixing and channel-mixing MLP layers according to the specifications in the original paper. He developed the overall model structure and pipeline for training and evaluation. Abdullah also handled the setup and configuration of the computational resources (RTX 8090 GPUs) used for experimentation. Additionally, he analyzed the results and investigated potential factors contributing to the discrepancies between the reproduced and original results. Abdullah coordinated the collaboration efforts and the division of tasks between team members.

Hamed Taherkhani implemented the data loading and augmentation pipelines required for the experiments. He configured to load the pre-training weights for the MLP-Mixer models using the pre-trained weights from the authors' repository. Hamed handled the fine-tuning of the pre-trained models on the ImageNet dataset and the additional datasets used for the "Avg 5" metric. He also managed the hyperparameter tuning and optimization settings during fine-tuning, evaluated the models' performance on the validation sets, and computed the Top-1 and Top-5 accuracy metrics. Hamed also wrote most of the project report.

7 Conclusion

In conclusion, this report presents a reimplementation and analysis of the MLP-Mixer architecture proposed by Tolstikhin et al. for computer vision tasks. Despite efforts to faithfully reproduce the results reported in the original paper, the reimplementation showed some discrepancies in the performance metrics compared to the original results.

Various factors may have contributed to these differences, including variations in fine-tuning settings, hardware configurations, implementation details, and randomness in the training process. Addressing these factors could improve the reproducibility of the results and provide deeper insights into the behavior of the MLP-Mixer architecture.

Overall, while the reimplementation did not perfectly match the original results, it contributed to the understanding of the MLP-Mixer architecture and its performance on image classification tasks. Further research and experimentation are needed to fully explore the capabilities and limitations of this novel approach to vision tasks.

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: CVPR09, 2009.

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, arXiv preprint arXiv:2010.11929.

[3] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, MIT press, 2016.

[4] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861.

[5] A. Krizhevsky, Learning multiple layers of features from tiny images, Tech. rep., Citeseer (2009).

[6] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.

[7] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document

recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.

[8] I. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, D. Keysers, J. Uszkoreit, M. Lucic, A. Dosovitskiy, Mlp-mixer: An all-mlp architecture for vision, arXiv preprint arXiv:2105.01601.