Patches Are All You Need?

E4040.2025Fall.WAWJ.report
Haipei Xu hx2385, Weixi Huang wh2610
Columbia University

Abstract

This project reproduces the key experimental findings of the ConvMixer architecture under realistic academic computing constraints. Motivated by the question of whether patch-based processing alone can account for the strong performance of modern vision models, we re-implemented ConvMixer-768/32 and ConvMixer-1536/20 and adapted the training pipeline to a single NVIDIA T4 GPU. Due to the scale of ImageNet-1K and limited computational resources, training was conducted on a reduced dataset with modified training settings.Despite these constraints, the reproduced models achieved Top-1 accuracies of 75.96% and 75.62%, closely matching the qualitative behavior and core conclusions reported in the original work. Overall, this project demonstrates the robustness of the ConvMixer design and highlights the impact of data scale, preprocessing choices, and computational limitations on the reproducibility of large-scale vision models.

1. Introduction

Recent advances in vision architectures have increasingly emphasized patch-based representations for image processing. Models such as Vision Transformers [1] and MLP-Mixers [2] have demonstrated strong performance on large-scale image classification benchmarks. However, these approaches often rely on substantial computational resources and complex training pipelines, raising concerns regarding reproducibility and implementation consistency in practical academic settings. Motivated by these challenges, this project investigates whether the performance of the ConvMixer architecture can be reliably reproduced under standard academic computing constraints.

The primary objective of this project is to reproduce the key results reported in the original ConvMixer paper [3], analyze the model's sensitivity to implementation details, and assess how performance varies across different hardware and software configurations. This includes adapting the official training and evaluation pipeline, validating the provided pretrained weights on ImageNet-1K, and comparing the reproduced results with those reported in the original work.

A major challenge in this project arises from differences between our computational environment and that used in the original study. ConvMixer's performance on ImageNet is sensitive to factors such as data preprocessing, library versions, and GPU kernel behavior, making exact replication nontrivial. To mitigate these challenges, we follow the configurations provided in the official repository as closely as possible and systematically examine the effects of hardware, software, and preprocessing variations. Through this analysis, our project aims to provide a clear and reproducible evaluation of ConvMixer and to identify practical factors that influence its reported performance.

2. Summary of the Original Paper

2.1 Methodology of the Original Paper

The paper investigates whether the strong performance of Vision Transformers (ViTs) stems primarily from attention-based mechanisms or from the use of patch-based image representations. To disentangle these factors, the authors propose ConvMixer, a fully convolutional architecture that adopts the same patch representation and isotropic design employed by ViTs and MLP-Mixers. By relying solely on standard convolutional operations, ConvMixer enables an examination of how far patch-based processing alone can go without the use of attention mechanisms or large multilayer perceptrons.

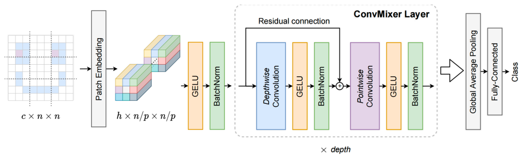

Figure 1: Overview of the ConvMixer architecture

The methodology begins with a patch embedding stage, in which the input image is partitioned into non-overlapping $p \times p$ patches using a single convolutional layer whose kernel size and stride are both equal to the patch size. This operation projects each patch into an h-dimensional embedding while preserving the spatial grid structure. As a result, the model operates on a token grid representation similar to that used in Vision Transformers. The initial projection is followed by Batch Normalization and a GELU activation.

Following the patch embedding stage, the model applies a sequence of identical ConvMixer layers, each explicitly separating spatial and channel mixing. Spatial feature mixing is performed using a depthwise separable convolution with a large kernel size k, followed by a GELU activation and Batch Normalization, and incorporated within a residual connection. This operation expands the receptive field in a manner analogous to the global context aggregation achieved by self-attention. Channel mixing is implemented via a pointwise 1×1 convolution, again followed by GELU and Batch Normalization, serving a role similar to the channel-mixing MLP used in MLP-Mixer and ViT blocks.

Stacking d such layers results in an isotropic network whose spatial resolution remains constant throughout the architecture. This design mirrors the structural properties of Mixer- and ViT-style models while avoiding their architectural complexity.

After the sequence of ConvMixer layers, the model applies global average pooling, which collapses the spatial dimensions into a single h-dimensional representation. A final linear layer then maps this representation to the output class logits.

The overall architecture is parameterized by four key hyperparameters: the hidden dimension h, network depth d, patch size p, and kernel size k. The authors emphasize that large-kernel depthwise convolutions provide substantial modeling capacity and enable effective long-range spatial interactions even without attention mechanisms. At the same time, the patch-based, fixed-resolution design offers a clean and controlled comparison with Vision Transformers and MLP-Mixer architectures.

2.2 Key Results of the Original Paper

The original paper provides empirical evidence that ConvMixer achieves a favorable accuracy–parameter trade-off on ImageNet-1K. The reported results demonstrate that ConvMixer attains competitive or superior Top-1 accuracy compared to architectures such as ResNet, ResMLP, and DeiT, even at substantially smaller parameter scales.

As shown in Figure 2, ConvMixer-1536/20 achieves approximately 81.4% Top-1 accuracy with 52M parameters, while ConvMixer-768/32 reaches about 80.2% accuracy using only 21M parameters. These results position ConvMixer on a more efficient accuracy–parameter frontier than ResMLP and DeiT, which typically require larger model sizes or stronger regularization to achieve comparable performance. Notably, the authors emphasize that these results are obtained without extensive hyperparameter tuning, highlighting that the combination of patch-based representations and convolutional mixing provides a strong inductive bias and stable learning behavior.

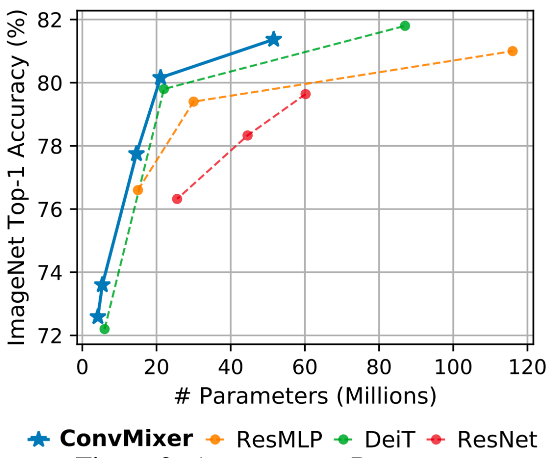

Figure 2: Accuracy vs. Parameters

The paper further evaluates ConvMixer under the standard 224×224 ImageNet-1K training setup and compares it directly with several baseline architectures. As summarized in Table 1, ConvMixer-1536/20 achieves 81.37% Top-1 accuracy with fewer parameters than ResNet-152 (79.64%) and closely matches the performance of DeiT-B (81.8%), despite DeiT relying on stronger regularization and substantially longer training schedules. ConvMixer-768/32 attains 80.16% Top-1 accuracy using only 21M parameters, making it competitive with ResMLP-B24/8 and more parameter-efficient than comparable ResNet variants. These results reinforce the authors' central claim that a simple, fully convolutional, patch-based architecture can rival more complex Transformer-style models, and that the effectiveness of modern isotropic architectures may stem largely from the patch representation itself.

Table 1: Key ImageNet-1k Results: ConvMixer vs. Other Simple Models

Network	Patch Size	Kernel Size	# Params ($\times 10^6$)	# Epochs	ImNet Top-1 (%)
ConvMixer-1536/20	7	9	51.6	150	81.37
ConvMixer-768/32	7	7	21.1	300	80.16
ResNet-152	–	3	60.2	150	79.64
DeiT-B	16	–	86	300	81.80
ResMLP-B24/8	8	–	129	400	81.00

In addition to the ImageNet-1K experiments, the original paper also conducts small-scale studies on CIFAR-10 to examine whether ConvMixer exhibits consistent behavior under limited data regimes. The results show that even small ConvMixer models achieve strong classification accuracy, indicating high data efficiency. Ablation studies over kernel size, patch size, and weight decay reveal several stable trends: increasing the depthwise convolution kernel size consistently improves accuracy, while larger patch sizes substantially reduce spatial resolution and lead to performance degradation. Weight decay is observed to have a smaller but generally positive effect when set to moderate values.

3. Methodology

3.1. Objectives and Technical Challenges

The objective of this project is to reproduce the key experimental results reported in the ConvMixer paper under realistic computational constraints. While the original study trains ConvMixer-1536/20 and ConvMixer-768/32 on the full ImageNet-1K dataset using ten GPUs, our experiments are limited to a single NVIDIA T4 GPU with constrained memory. This disparity in computational resources introduces several technical challenges, including restricted batch-size scalability, reduced data throughput, and significantly longer training times.

A second challenge stems from the scale of the ImageNet-1K dataset, which comprises 1,000 classes and over one million training images. Fully reproducing the original training setup is therefore infeasible within our resource limits. Training on a reduced dataset alters the underlying data distribution and may affect optimization dynamics and generalization behavior.

Finally, the training configuration reported in the original paper is optimized for multi-GPU execution with large effective batch sizes and high hardware parallelism. Adapting these settings to a single-GPU environment is inherently challenging, introducing additional uncertainty when directly comparing our reproduced results with those reported in the original work.

3.2. Problem Formulation and Design Description

We formulate our reproduction task as a supervised image classification problem. Given an input image

$$x \in \mathbb{R}^{3\times224\times224},$$

the ConvMixer network produces a logits vector

$$f_\theta(x) \in \mathbb{R}^{100},$$

and the model is trained to minimize the cross-entropy loss over the ImageNet-1k training set. Compared with the original ConvMixer formulation, our task differs only in the number of output classes.

Our system design follows the same high-level structure as the original paper and consists of four main components: (1) the data preprocessing pipeline, (2) the ConvMixer model construction, (3) the training procedure, and (4) the validation and evaluation workflow. This modular design ensures methodological consistency with the original work while allowing adjustments required for single-GPU execution.

We implement ConvMixer strictly according to the original specification. Patch embedding is performed using a convolution with stride p:

$$x_0 = \mathrm{BN}(\mathrm{GELU}(\mathrm{Conv}_{p\times p}(x))).$$

Each ConvMixer block consists of a depthwise convolution for spatial mixing and a pointwise convolution for channel mixing:

$$z = x + \mathrm{BN}(\mathrm{GELU}(\mathrm{DWConv}_k(x))),$$

$$x_{t+1} = \mathrm{BN}(\mathrm{GELU}(\mathrm{PWConv}(z))).$$

Since our experiments use the first 100 classes in ImageNet-1k dataset, the classification head is changed from 1000 to 100 output units. All other structural details are left unchanged to preserve the fidelity of reproduction.

4. Implementation

This section briefly describes the experimental setup used in our ConvMixer implementation.

All experiments were conducted on Google Cloud Platform using Compute Engine with a single NVIDIA Tesla T4 GPU. The models were implemented and trained using the PyTorch framework with the `timm` library.

Due to computational constraints, training was performed on a reduced subset of the ImageNet-1k dataset and only single-GPU training was used. On this setup, training each of the two ConvMixer models (768/32 and 1536/20) required approximately 140 hours of wall-clock time.

4.1 Data

In this project, we primarily use the ImageNet-1K dataset and additionally conduct small-scale experiments on the CIFAR-10 dataset.

ImageNet-1K is a large-scale benchmark widely used for image classification research. It contains approximately 1.28 million training images and 50,000 validation images spanning 1,000 object categories, including animals, vehicles, tools, household objects, and natural scenes. Images are provided at varying resolutions and are annotated with a single semantic class.

Due to computational constraints, we construct a reduced version of ImageNet-1K by selecting the first 100 classes from both the training and validation sets. This reduces the dataset size by approximately 90%, enabling feasible training on a single NVIDIA T4 GPU.

In addition, the CIFAR-10 dataset is used for preliminary experiments, including initial evaluation and small-scale hyperparameter exploration. CIFAR-10 consists of 60,000 color images of size 32×32, evenly distributed across 10 object categories such as airplanes, automobiles, birds, and cats. The dataset is split into 50,000 training images and 10,000 testing images.

4.2 Deep Learning Network

In this project, we implement the ConvMixer architecture for image classification.

The network begins with a patch embedding layer that projects non-overlapping image patches into a higher-dimensional feature space. This is followed by a sequence of ConvMixer blocks, each composed of a depthwise convolution for spatial feature mixing and a pointwise convolution for channel-wise mixing, combined with residual connections.

The model is trained in a supervised manner using cross-entropy loss and stochastic gradient-based optimization. Training is performed for multiple epochs, and performance is monitored on a validation set.

4.3 Software Design

The software implementation follows a modular design to support efficient experimentation.

The overall training pipeline consists of model initialization, dataset loading, training, validation, and checkpoint saving. Configuration parameters are used to control model variants and training hyperparameters without modifying core code logic.During training, the model processes mini-batches of data and updates parameters iteratively based on the computed loss.

After each epoch, the model is evaluated on a validation set, and the best-performing model is saved according to validation accuracy, which ensures clarity, reproducibility, and ease of experimentation.

5. Results

5.1 Project Results

Our results include three parts, ConvMixer-768/32 on ImgaeNet-1K(10%), ConvMixer-1536/20 on ImgaeNet-1K(10%), and ConvMixer-256/8 on CIFAR-10.

5.1.1 Hyperparameter tuning

To study the effect of architectural hyperparameters in ConvMixer-256/8, we vary the *kernel size* $3, 5, 7$ and *patch size* $1, 2, 4$ on CIFAR-10, producing nine configurations. Figure 3 reports the training accuracy, test accuracy, per-epoch time, and learning-rate schedule.

Patch size has the strongest impact: larger patches train faster but consistently yield lower accuracy and more unstable test-accuracy curves. Kernel size mainly affects computational cost. Larger kernels increase the per-location convolution cost, while smaller patches increase feature-map resolution. As a result, training becomes slower when kernel size grows and patch size shrinks.

On CIFAR-10, accuracy differences across settings are relatively small, but training-time differences are substantial. These observations guide our hyperparameter choices when scaling to ImageNet-1K, where efficiency is more critical.

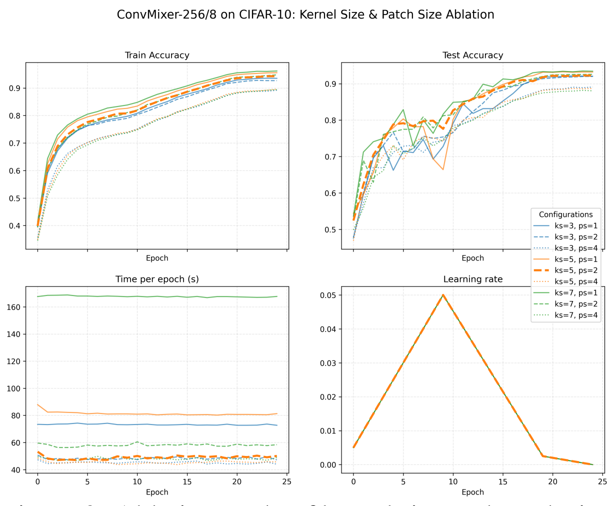

Figure 3: Ablation study of kernel size and patch size on CIFAR-10 using ConvMixer-256/8

5.1.2 Results on ImageNet-1K

After we have a basic understanding of the model and the data, we tried to reproduce the main conclusions of the original paper, which is to testing the performance of ConvMixer-768/32 and -1536/20 on ImgaeNet-1K.

Due to the computational power and time constraints, we cannot reproduce all of the content according to their settings. In our setting, A ConvMixer-1536/20 with 52M parameters can achieve 75.62% top-1 accuracy on ImageNet, and a ConvMixer-768/32 with 21M parameters 75.96%.

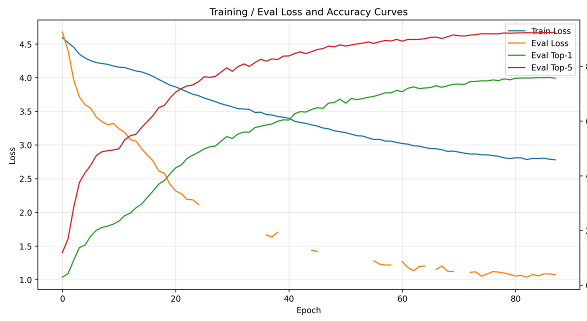

Figure 4: ConvMixer-768/32

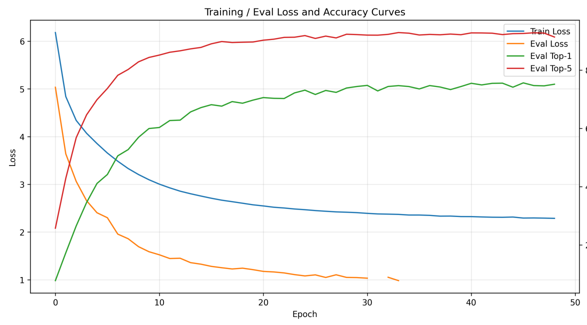

Figure 5: ConvMixer-1536/20

Figure 4 presents the training dynamics of ConvMixer-768/32 on the ImageNet-100 subset, including training loss, evaluation loss, and evaluation Top-1 and Top-5 accuracy over epochs. During the early phase of training, the model improves rapidly. Training loss drops quickly, Top-1 accuracy rises from 2.92% to above 40%. This steep improvement suggests that the model quickly acquires coarse-grained semantic information such as shapes, edges, and color regions. These early patterns align with the inductive bias of convolutional layers.

Between epochs 20 and 40, the improvement becomes more gradual but remains steady. Top-1 accuracy increases from around 43% to nearly 60%, while Top-5 accuracy rises from 70% to about 84%. This stage likely reflects the model learning finer-grained distinctions among categories. It also shows progressive refinement of feature composition and greater stability in the internal representation space. Training loss continues to decrease during this period, and whenever the evaluation loss is valid, it follows the same downward trend. This indicates that the optimization process remains stable.

In the later phase, the model approaches convergence, with accuracy curves gradually flattening. The slow but continuing decline in training loss, combined with stable evaluation accuracy, suggests that the architecture is effectively utilizing the available information without overfitting.

A notable phenomenon is that evaluation loss becomes NaN in many epochs. However, evaluation accuracy continues to improve smoothly, and no signs of instability appear throughout training. This indicates that the NaN values stem from numerical issues rather than optimization failure. Under mixed-precision training, extremely small predicted probabilities can cause FP16 log operations to overflow; similarly, uneven validation batches or rare preprocessing anomalies may lead to invalid loss values. Since accuracy metrics remain fully reliable, these NaN values do not affect the assessment of model performance.

Figure 5 illustrates the evolution of training and evaluation loss, together with evaluation Top-1 and Top-5 accuracy, across training epochs. We observe a rapid decrease in both training and evaluation loss during the early stages of training, indicating effective optimization and stable gradient updates. Concurrently, evaluation accuracy improves sharply, with Top-5 accuracy increasing particularly fast within the first 10 epochs, suggesting that the model quickly learns coarse-grained semantic representations.

After approximately 25–30 epochs, both evaluation Top-1 and Top-5 accuracy begin to plateau, while training loss continues to decrease slowly. This divergence indicates diminishing returns from further training and suggests that the model has largely converged under the current architecture and hyper-parameter configuration. Importantly, no significant overfitting is observed, as the evaluation loss remains stable and does not increase in later epochs.

Based on these observations, training was terminated early to avoid unnecessary computational cost, and the final model was selected according to the best validation accuracy.

Overall, the results demonstrate stable convergence behavior and confirm that ConvMixer is capable of learning effective visual representations on the ImageNet-1k subset used in this experiment.

5.2 Comparison of the Results Between the Original Paper and Students' Project

5.2.1 Results on Evaluation

We evaluated the official ConvMixer-1536/20 checkpoint on the ImageNet-1K validation set and obtained a Top-1 accuracy of 80.65%, which is slightly lower than the 81.37% reported by the authors. This difference is modest and falls within the range of variation commonly observed in large-scale ImageNet evaluations. Several factors may account for this gap.

First, our evaluation was conducted on a single NVIDIA T4 GPU with a relatively lightweight CPU configuration, differing from the higher-performance hardware used in the original experiments. Variations in GPU architecture, CUDA kernel scheduling, and floating-point nondeterminism can introduce accuracy fluctuations of up to approximately 1%. Second, our PyTorch, CUDA, and torchvision versions do not exactly match those used by the authors. Minor differences in image decoding, interpolation, or center-crop implementations across library versions can lead to measurable accuracy changes on ImageNet-scale benchmarks.

Overall, the small discrepancy between our reproduced accuracy and the reported result indicates that our dataset setup, preprocessing pipeline, model configuration, and software environment are well aligned with the official implementation. This evaluation therefore serves as a successful sanity check, confirming that our experimental environment is correctly configured for subsequent experiments.

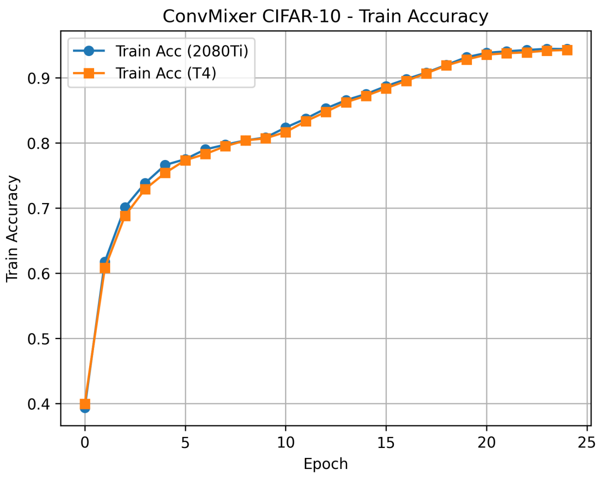

Figure 6: Training accuracy on CIFAR-10

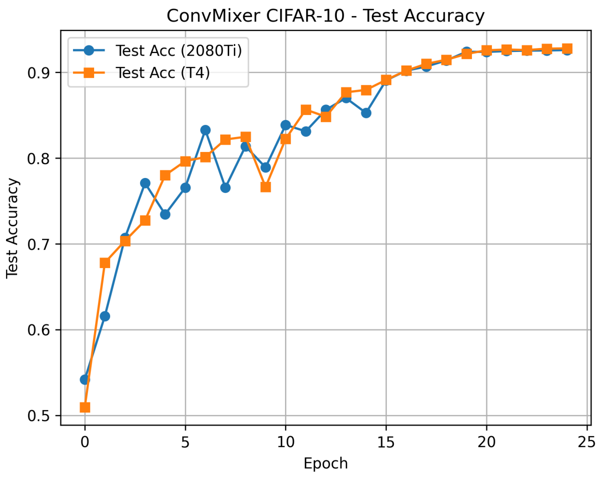

Figure 7: Test accuracy on CIFAR-10

5.2.2 Results on CIFAR-10

Before conducting the main experiments on the large-scale ImageNet dataset, we first performed a small-scale reproduction test on CIFAR-10 to test for the time consumption and accuracy. We trained the ConvMixer-256/8-2-5 model for 25 epochs.

Figure 6 shows that the training accuracy steadily improves and reaches approximately 94%, which closely matches the early-epoch behavior reported in the original ConvMixer paper. The test accuracy shown in Figure 7 reaches about 93–94% by the end of training, which also matches the result in original paper.

We also measured the per-epoch training time across two GPU types (2080Ti and T4), not only to compare hardware behavior but also to estimate the computational cost of running large-scale ImageNet experiments.

Before conducting the full experiments, we performed a FLOPs-based runtime estimation, summarized in Table 2, to approximate the expected training time and monetary cost across different GPU configurations. Based on this estimate, a single epoch of ConvMixer-1536/20 requires approximately 1.9×10^4 GFLOPs, corresponding to an expected runtime of about 40 minutes on a T4 GPU when assuming an effective throughput of 8 TFLOPs.

However, empirical measurements showed that a single epoch completed in approximately 20 minutes, indicating that the practical throughput of our T4 instance is closer to 16 TFLOPs. After accounting for the difference between the 25 training epochs used in our experiments and the 100-epoch basis assumed in Table 2, the actual runtime and cost are approximately 1.3 times higher than the values reported in the table.

These calibrated measurements directly informed our experimental planning for ImageNet. By more accurately estimating the effective compute capability of the T4 GPU, we refined our expectations for runtime and cost, and adjusted batch sizes, training schedules, and GPU selection accordingly.

Table 2: FLOPs-based runtime and cost estimation for ConvMixer(single card, 100 epochs).

Model	FLOPs	T4 (5T)	V100 (15T)	A100 (40T)
768/32	7.5×10^{18}	420 h 150$	140 h 350$	52 h 190$
1536/20	1.9×10^{19}	1040 h 360$	346 h 860$	130 h 480$
CIFAR-10	/1000	/1000	/1000	/1000

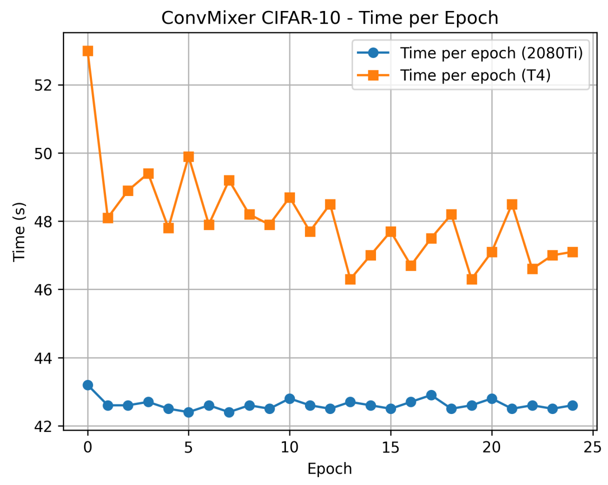

Figure 8: Training time per epoch on CIFAR-10

5.2.3 Results on ImageNet-1K

As shown in Table 3, our reproduced ConvMixer models achieve competitive Top-1 accuracy compared to the original implementation.

While the reproduced results do not exactly match the reported numbers, the overall performance trend is consistent with the original work, indicating that the ConvMixer architecture generalizes well under our experimental setup.

Table 3: Comparison of Experimental Settings and Accuracy

Model	Source	Data	Batch	Worker	Epoch	Top-1 acc
Conv-768	Us	10%	32	4	90	75.96
	Author	Full	64	10	300	80.16
Conv-1536	Us	10%	16	4	50	75.62
	Author	Full	64	10	150	81.37

5.3 Discussion / Insights Gained

The observed performance differences between our reproduction and the original ConvMixer results can be attributed to several practical factors. First, our experiments were conducted using a single GPU, whereas the original work employed a multi-GPU setup with up to 10 GPUs. This constraint limits the effective batch size and training throughput, which can affect optimization stability and final accuracy.

Second, due to computational limitations, training was performed on a reduced subset of the ImageNet dataset rather than the full ImageNet-1K training set used in the original study. The reduced data scale limits sample diversity and exposure, further contributing to the remaining performance gap.

Finally, we observed occasional NaN values in the evaluation loss during training, likely caused by numerical instability under limited batch size and aggressive optimization settings. Importantly, evaluation accuracy remained stable throughout training, and model selection was therefore based on validation accuracy rather than loss. Despite these differences, the reproduced results remain competitive, highlighting the robustness of the ConvMixer architecture under constrained computational resources.

6. Future Work

We believe future research on ConvMixer can proceed along several promising directions.

First, as noted in [3], a more complete comparison with modern architectures such as Vision Transformers and MLP-Mixers requires evaluating ConvMixer under large-scale pre-training. Due to computational constraints, our work could not reproduce the full training regime of the original study. Extending experiments to substantially larger datasets would help assess ConvMixer's scalability, convergence behavior, and transfer performance at foundation-model scale.

Second, concurrent work such as [4] shows that very large depthwise kernels (e.g., 31×31) can yield receptive-field sizes and long-range modeling capabilities comparable to attention mechanisms. This suggests a natural extension: exploring whether increasing ConvMixer's depthwise kernel size, potentially with stabilizing techniques like structural reparameterization, could further enhance its representational power or narrow the gap between convolutional and attention-based models.

Third, alternative isotropic architectures, including graph-based vision backbones proposed in [5], represent an intriguing direction. Since both ConvMixer and graph-based isotropic models operate at the patch level, systematic comparisons or hybrid architectures may provide new insights into token-mixing design choices.

Finally, recent analyses such as [6] show that several later lightweight models, including FastViT, inherit key design elements from ConvMixer, particularly convolution-based token mixing and structural re-parameterization. Understanding this architectural lineage may help guide future improvements in efficiency, scalability, and generalization.

7. Conclusion

This project aimed to reproduce the key findings of the ConvMixer paper under realistic computational constraints. Despite operating with a single NVIDIA T4 GPU and using only 10% of the ImageNet-1K dataset, our experiments successfully reproduced the qualitative behavior and core conclusions of the original work. ConvMixer-768/32 and ConvMixer-1536/20 both exhibited stable convergence and achieved Top-1 accuracies of 75.96% and 75.62%, respectively, demonstrating strong performance under constrained settings.

Overall, the objectives of the project were achieved. We reproduced the main experimental findings of ConvMixer, analyzed its learning dynamics, and identified practical challenges that arise when large-scale vision models are adapted to limited computational environments. This process provided deeper insight into patch-based convolutional architectures, inductive biases, and the sensitivity of large-scale evaluation to numerical precision and preprocessing choices.

Several directions remain for future work. Expanding the dataset beyond 100 classes would allow a more faithful comparison with the original results, while improving numerical stability could further enhance reproducibility. In addition, exploring larger batch sizes, alternative training schedules, or distributed training may offer further insight into how ConvMixer scales with increased computational resources.

8. Acknowledgement

We would like to thank Professor Zoran Kostic for his guidance throughout this course and for giving us the opportunity to explore neural network techniques through this project. We also appreciate the infrastructure and software support provided by Google Cloud and TensorFlow, which enabled efficient GPU-based experimentation. In addition, we acknowledge the authors of the original paper for offering valuable insights that shaped our understanding and motivated our exploration of convolutional neural network architectures. We are also grateful to all the TAs for their timely and helpful responses on Ed, which greatly supported our progress throughout the project. Finally, we extend special thanks to Yaqi Wang from Carnegie Mellon University for helping us get started with several fundamental concepts in model training.

9. References

[1] 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*, 2020.

[2] I. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, D. Keysers, J. Uszkoreit, M. Lucic, *et al.*, “Mlp-mixer: An all-mlp architecture for vision,” *arXiv preprint arXiv:2105.01601*, 2021.

[3] A. Trockman and J. Z. Kolter, “Patches are all you need?,” 2022.

[4] X. Ding, X. Zhang, J. Han, and G. Ding, “Scaling up your kernels to 31x31: Revisiting large kernel design in cnns,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11963–11975, June 2022.

[5] K. Han, Y. Wang, J. Guo, Y. Tang, and E. Wu, “Vision gnn: An image is worth graph of nodes,” 2022.

[6] O. Elassiouti, G. Hamed, D. Khattab, and H. Ebbied, “Comparative analysis of lightweight deep learning models for classification of medical images,” *International Journal of Intelligent Computing and Information Sciences*, vol. 25, no. 3, pp. 93–106, 2025.

10. Appendix

10.1 Individual Student Contributions in Fractions

	hx2385	**wh2610**
Last Name	Xu	Huang
Fraction of (useful) total contribution	1/2	1/2
What I did 1	Trained, evaluated, and tuned ConvMixer-768/32 on CIFAR-10.	Built the ImageNet download and preprocessing pipeline.
What I did 2	Write Chap4-10 in report and mange the overall format	Write Chap1-5 and Chap7 in report
What I did 3	Trained ConvMixer-1536/20 on ImageNet-1K.	Trained ConvMixer-768/32 on ImageNet-1K.

Table 4: Individual Student Contributions in Fractions