A Comparative Study of the Effect of Various Data Augmentation Techniques on the Performance of CNN and Transformer-based Image Classification

Student Name: Liyuan Zhang
Supervisor Name: Stamos Katsigiannis
Submitted as part of the degree of BSc Computer Science to the
Board of Examiners in the Department of Computer Sciences, Durham University

Abstract—Data augmentation has long been a critical technique for improving the generalization of deep learning models in image classification. This study provides a comprehensive comparison of various data-augmentation methods on the performance of convolutional neural networks (CNNs) versus Transformer-based vision models. Two CNNs (ResNet-18 and ResNet-50) and two Transformer models (Vision Transformer (ViT) and Swin Transformer) were evaluated across four benchmark image datasets (CIFAR-10, CIFAR-100, MNIST, and Fashion-MNIST) under a wide range of augmentation strategies, including traditional geometric and photometric transforms, region erasing (Cutout, Random Erasing), mixup-based methods, automated policy augmentations (AutoAugment, RandAugment, TrivialAugment, AugMix), and five manually crafted multi-step “recipes” evaluated on CIFAR-100. Experiments report test accuracy, precision, recall, and F1-score for each model/augmentation combination (with no augmentation as a baseline), employing early stopping to prevent overfitting. The results indicate that advanced augmentation techniques such as CutMix and TrivialAugment yield the greatest performance gains on more complex datasets (CIFAR-10/100), especially for the CNN models, while simpler augmentations suffice for simpler datasets (e.g. MNIST) where baseline performance is already high. The multi-step pipelines provide an additional ≈ 1 pp boost for the CNNs on CIFAR-100 (with *flip + crop + CutMix* performing best) but deliver only marginal gains for the Transformer models. Transformer-based models achieve higher absolute accuracy than CNNs on these tasks (often exceeding 98–99% on CIFAR-10 and MNIST), leaving less room for improvement from augmentation; nonetheless, certain augmentations still provided modest benefits. The key observation is that augmentation can significantly prolong effective training (delaying overfitting as evidenced by a later point of early stopping), leading to improved generalization. However, not all augmentations are beneficial: for example, horizontal flips degrade performance on digit classification, and random cropping can hurt fashion-item recognition. Overall, this comparative analysis offers practical insights into which augmentation techniques work best for different model architectures and dataset types, and underscores the importance of choosing augmentations appropriate to the data characteristics to maximize performance.

Index Terms—Computer vision, Convolutional neural networks, Deep learning, Transformers

1 Introduction

Deep learning models for image classification often require large and diverse training data to generalize well. Data augmentation is a widely used strategy to synthetically expand the training dataset. It deliberately produces new training samples by applying label-preserving transformations—such as flips, rotations, color jitter, or patch-level mixing—so that the model encounters a richer set of valid input variations, thereby reducing overfitting and improving robustness. Since the breakthrough AlexNet model on ImageNet in 2012, which used random cropping and horizontal flipping to boost performance [1], augmentation techniques have grown increasingly sophisticated. Traditional augmentations (e.g. flips, rotations, scaling) have become standard practice in training convolutional neural networks (CNNs). Recent surveys underscore the variety and importance of augmentation in deep learning pipelines [2].

On the modeling side, CNN architectures such as ResNet [3] have been the dominant approach for image recognition tasks for several years. CNNs incorporate certain inductive biases (e.g. translational invariance and local receptive fields) that help them generalize, sometimes reducing the reliance on external regularization. In the past few years, however, Transformer-based vision models have emerged as a powerful alternative. The Vision Transformer (ViT) introduced by Dosovitskiy *et al.* [4] demonstrated that transformer architectures, when trained on sufficiently large datasets, can achieve excellent image classification accuracy without using convolution. Subsequent models like the Swin Transformer [5] introduced hierarchical attention and shifted windowing to better capture locality, closing the gap between transformers and CNNs on vision tasks.

One key difference between CNNs and Vision Transformers is their reliance on data scale and augmentation. Vision Transformers have far fewer built-in inductive biases about images and thus can underperform when data is limited; strong data augmentation and regularization schemes are often necessary to train them effectively on smaller datasets [6]. Indeed, it has been observed that advanced augmentation techniques (such as mixup [7] and CutMix [8]) are now standard in state-of-the-art Vision Transformer training pipelines [6]. This raises important questions about how different augmentation methods affect CNNs versus Transformers: Do transformers benefit more from augmentation than CNNs, or vice versa? Are certain augmentations particularly well-suited to one architecture type over the other?

In this paper, a comparative study was presented examining a broad spectrum of data augmentation techniques applied to two CNN models (ResNet-18 [3] and ResNet-50 [3]) and two transformer-based models (ViT [4] and Swin Transformer [5]) across four popular image classification datasets of varying complexity (CIFAR-10 [9], CIFAR-100 [9], MNIST [10], and Fashion-MNIST [11]). By evaluating test accuracy, precision, recall, and F1-score for each model under each augmentation, the aim is to identify which augmentations yield the most significant improvements, and how these effects depend on the model architecture and dataset characteristics. There was also a baseline with no augmentation ("None") included to quantify the net gains. Early stopping was employed during training to ensure fair comparison and to illustrate the regularization effect of augmentations.

The contributions of this work are threefold: (1) an extensive empirical evaluation of sixteen augmentation techniques on both CNN and Transformer models, providing insight into their effectiveness on simple vs. complex datasets; (2) summary tables and analysis highlighting trends, such as which augmentations consistently improve performance and which may be detrimental if misapplied; and (3) practical observations on training dynamics (e.g. the role of early stopping) and guidance on selecting augmentations for different scenarios. The findings can inform practitioners in computer vision on how to best leverage data augmentation for different model architectures and data regimes.

2 Related Work

2.1 Data Augmentation Techniques

Augmentation as a regularization technique in deep vision models has a rich history. Early works employed basic transformations: Krizhevsky *et al.* used translations and horizontal reflections to expand the ImageNet training set [1], a practice now common in virtually all CNN training. Beyond these basics, a plethora of augmentation methods have been proposed [2]. Some methods apply simple geometric or photometric transforms. For example, random rotations, affine transforms, or perspective warps can improve invariance to viewpoint changes; adjusting color characteristics (brightness, contrast, etc.) via color jitter or applying blur can mimic lighting and focus variations. These hand-crafted augmentations are easy to implement and have proven effective in many cases.

More advanced techniques involve selectively erasing or mixing image regions. *Cutout* [12] masks out a random square patch of the image during training, forcing the model to focus on less obvious features. A related approach, *Random Erasing*, generalizes this idea by randomly selecting an area and replacing it with random pixels [13]; it improves model robustness especially for fine-grained recognition. Another class of methods are *mixup [7]* techniques, which generate new training examples by mixing multiple images. *Mixup* [7] linearly blends two images and their labels, effectively augmenting the training distribution and encouraging linear interpolation behaviour in the model. *CutMix* [8] patches a rectangular region from one image onto another, with the label similarly mixed, aiming to combine the benefits of Cutout [12] and mixup [7] . These region-level augmentations and mixup [7] strategies have been highly successful in improving generalization for CNNs and are now also used in transformer training routines.

Recently, automated augmentation policy methods have gained prominence. *AutoAugment* [14] employs a search algorithm to find an optimal sequence of augmentation operations for a given dataset . While effective, it requires a separate search phase. *RandAugment* [15] simplifies this by defining a fixed set of operations and randomizing their application with a couple of hyperparameters, eliminating the search phase while still achieving strong results. *TrivialAugment* [16] pushes simplification further by applying a single random augmentation from a predefined set to each image without any tuning, yet reaching state-of-the-art performance comparable to more complex policies. *AugMix* [17] takes a different approach by composing multiple augmentations stochastically and linearly mixing the results, along with an explicit consistency loss, to improve model robustness to distributional shifts. These automated or semi-automated methods often combine multiple simple transformations in diverse ways, yielding augmented data that significantly improves resilience and accuracy.

There have been a few comparative studies and benchmarks of augmentation methods. For instance, Hendrycks et al. compare AugMix [17] with mixup [7] on robustness benchmarks, and others [2] provide taxonomies of augmentation strategies. However, systematic comparisons across a wide range of augmentations on multiple model architectures are less common. Our work aims to fill this gap by evaluating side-by-side many of the aforementioned methods under identical conditions for both CNN and transformer models.

2.2 CNNs vs. Vision Transformers

CNNs have dominated image recognition for years, with architectures like ResNet [3] setting the standard in accuracy and efficiency. Their built-in locality and translation invariance make them naturally suited for vision tasks; simple augmentations often suffice to further improve CNN performance. In contrast, Vision Transformers treat images as sequences of patches and rely on self-attention mechanisms to learn global relationships [4]. The pioneering ViT [4] model achieved excellent results by training on very large datasets (e.g., JFT-300M [18]) with minimal inductive bias. Subsequent work on data-efficient transformers, such as the DeiT model [19], showed that with techniques like heavy data augmentation and knowledge distillation from a CNN teacher, transformers can be trained successfully on smaller datasets (ImageNet-1k) from scratch [19]. These studies indicate that transformers greatly benefit from rich augmentation to make up for their lack of built-in invariances.

Indeed, recent research has noted that strong augmentation schemes (mixup [7], CutMix [8], Random Erasing [13], etc.) are critical for achieving top performance with ViTs [6]. Transformers and CNNs may also respond differently to certain transformations; for example, a CNN might learn invariance to small translations through pooling, whereas a ViT [4] might need to see explicit translated versions of images to learn the same, unless positional embeddings are handled carefully. A study by Kim and Kim even identified that some augmentations can introduce subtle biases in ViT [4]s (e.g., mixup [7] causing positional embedding shifts) and proposed configuration adjustments to mitigate such issues [6].

In light of these differences, it is worth comparing how CNNs versus Transformers handle a common set of augmentations. Prior work provides some expectations: CNNs, with their strong inductive biases, might gain relatively less from augmentation on simple tasks, whereas Transformers might be more augmentation-hungry. The experiments explicitly examine this across multiple datasets. The related work was extended by not only looking at final accuracy but also precision, recall, and F1-score to see if augmentations impact class-wise performance differently for the two architecture types. To the knowledge, this is one of the first studies to comprehensively benchmark a wide range of augmentation techniques on both CNN and Transformer models in a unified setting.

3 Methodology

3.1 Datasets

Models were evaluated on four image-classification datasets that vary in image complexity and number of classes. **CIFAR-10** [9] consists of 60,000 color images of size 32×32 in 10 classes (animals and objects), with 50,000 training and 10,000 test images. **CIFAR-100** [9] is similar but has 100 classes (fine-grained labels), making it a more challenging task with the same image size and data volume. **MNIST** [10] is a classic dataset of 70,000 handwritten-digit images (0–9) in grayscale 28×28 resolution, with 60,000 for training and 10,000 for testing; it is relatively simple and nearly saturated by modern models (human-level performance $\sim 99.5\%$). **Fashion-MNIST** [11] contains 28×28 grayscale images of clothing items (10 classes, such as shirts, shoes, bags), also with 60,000 training and 10,000 test examples; it is intended as a drop-in, more challenging replacement for MNIST [10], featuring greater variability and less distinctive shapes than digits.

These four datasets allow the study to test augmentations on both low-resolution natural images (CIFAR [9]) and monochrome images that range from simplistic (MNIST [10]) to moderately complex (Fashion-MNIST [11]). For each dataset, a portion of the training set was split out for validation during training (typically 20% of the training data) to monitor performance for early stopping. All results reported are on the designated test set of each dataset.

3.2 Models

Two CNN architectures and two Transformer-based architectures were selected for comparison. For CNNs, **ResNet-18 [3]** and **ResNet-50** [3], represent a shallower and a deeper model from the ResNet family. ResNets employ residual skip connections to ease the training of deep networks and have been extremely influential; ResNet-50, with 50 layers, has substantially higher capacity than the 18-layer variant.

For Transformer models, the base **Vision Transformer** (*ViT-B/16*) as introduced by Dosovitskiy et al. [4], and the **Swin Transformer** (base) from Liu et al. [5] were used. ViT-B/16 [4] divides each image into 16×16 patches and feeds them to a standard Transformer encoder for classification, whereas Swin Transformer applies self-attention within local windows and shifts these windows across layers to form a hierarchical representation, achieving lower computational cost on higher-resolution images.

Both Transformer models are of comparable size (approximately 85–88 M parameters—larger than the ResNets [3], which have about 11 M for ResNet-18 and 25 M for ResNet-50 [3]) and were initialized with ImageNet-pretrained weights. Likewise, the ResNets were initialized from ImageNet-1k checkpoints. Using pretrained weights simulates a fine-tuning scenario that mirrors common practice and prevents Transformer performance from being dominated by convergence issues when training ViT [4] from scratch on small datasets [15]. Using pretrained models allows a fairer comparison of augmentation effects without the results being dominated by lack of training convergence on transformers.

3.3 Augmentation Techniques

Sixteen data-augmentation settings were evaluated: a baseline with no augmentation (“None”) and fifteen augmentation methods.

3.3.1 None

No data augmentation is applied to training images (aside from normalization and any resizing needed for the model input). This serves as the baseline to gauge improvement due to augmentation.

3.3.2 Flip

Random horizontal flip with 50 % probability.This is a standard augmentation for natural images (except where horizontal mirroring would alter class identity, as in non-symmetric digits or text). It helps models learn left-right invariance [1].

3.3.3 Rotate

Random rotation of the image within a specified range (up to $\pm 15^\circ$). Rotation augmentation can help with viewpoint invariance for objects not always upright.

3.3.4 Affine

Random affine transformations, including combinations of scaling, translation, rotation, and shear. This generalizes simple flips/rotations to moderate geometric distortions, emulating different viewing conditions.

3.3.5 Random Perspective

Applies a random perspective warp, which moves the image’s corner points by a random offset. This simulates 3D viewpoint changes and can improve robustness to projective distortions.

3.3.6 Random Resized Crop

Takes a random crop of the image at some scaled size and then resizes back to the original dimensions. This augmentation (widely used in ImageNet training [1]) effectively zooms in on random parts of the image, forcing the model to handle partial views. We ensured the entire image is considered by allowing random scales typically between 0.8 and 1.0 of the original size.

3.3.7 Color Jitter

Randomly changes brightness, contrast, saturation, and hue of the image. This augments the photometric properties and helps models be invariant to lighting/color changes. (For grayscale images like MNIST [10]/Fashion [11], this mainly affects brightness since they have no color channels.)

3.3.8 Gaussian Blur

Applies a Gaussian blur filter with a random kernel size to soften the image. Blurring can mimic depth of field or motion blur effects and challenges the model to recognize blurry inputs.

3.3.9 Cutout

Masks out a random square patch (20–50 % of the image area) of the image by setting its pixels to zero (or mean)[7].

3.3.10 Random Erasing

Similar to Cutout [12], but the masked region is filled with random pixel values (noise) instead of zeros. We applied random erasing [13] with a random rectangle (size and aspect ratio drawn from specified ranges) on each image with a certain probability. This introduces random occlusions and slight noise robustness.

3.3.11 Mixup

Generates a mixed sample by taking two training images and forming a weighted average of their pixel values, with the same weight applied to their one-hot labels [10]. Mixup [7] with a random mixing coefficient was used per batch. Mixup [7] effectively expands the training distribution and encourages linear behavior between classes, improving generalization and label-noise robustness.

3.3.12 CutMix

Creates a new sample by cutting a patch from one image and pasting it onto another image, and labels are mixed in proportion to pixel area. CutMix [8] was used with a random patch size and location per mixed pair. CutMix [8] has been shown to improve performance on tasks requiring localization as well as classification, by exposing models to partial objects.

3.3.13 AutoAugment

Uses a fixed policy learned on a proxy task to apply a sequence of augmentation operations to each image [14].This method was tuned to improve validation accuracy on the target dataset.

3.3.14 RandAugment

Randomly selects and applies $N{=}2$ operations from a fixed list, each with a random magnitude governed by a single strength parameter [15].

3.3.15 TrivialAugment

Chooses one operation at random from a broad set for every image, with a random magnitude and no hyper-parameter tuning [16].

3.3.16 AugMix

Creates three stochastically augmented variants of an image (depth 1–3 simple operations), linearly blends them, and feeds the mixture to the network [17].

3.3.17 Combined Pipelines (CIFAR-100 only)

Five balanced multi-step "recipes" were additionally evaluated on CIFAR-100 [1] to test whether stacking classic transforms with Mixup [7]/CutMix [8] confers extra benefit. The pipelines were frtcj = flip + rotate + color jitter, frcgb = flip + random resized crop + Gaussian blur, arpco = affine + random perspective + Cutout [12], frtcjmx = frtcj + Mixup [7], frccm = flip + random resized crop + CutMix [8]. All other training hyper-parameters were identical to the single-augmentation runs.

For all augmentation methods, they were applied only to the training set images on-the-fly during training. The validation and test images were not augmented (aside from necessary normalization/resizing to fit the network input). This way, improvements in metrics can be attributed to training augmentation improving generalization. The intensity of each augmentation was chosen to be in a typical range as per the original papers or common practice, to avoid either too mild or overly severe transformations.

Each model was trained independently on every $\langle$dataset, augmentation$\rangle$ pair. An automated loop iterated through all combinations, spawning one run per setting. The main hyper-parameters are summarised below; values were kept fixed across datasets and architectures unless stated.

3.4 Training Procedure

Each model was trained on each dataset with each augmentation setting independently. The Adam optimizer was used with an initial learning rate of 1e-5, and a batch size of 32. Training epochs were set at 2000, while early stopping was employed with a patience of 10 epochs based on validation loss. This means training would stop if the validation loss did not improve for 10 consecutive epochs, restoring the best model state. The use of early stopping ensures it does not overfit excessively, and it provides a glimpse into the regularization effect of augmentations: with heavy augmentation, the validation loss often keeps improving for longer, resulting in more training epochs before stopping, as it will be discussed later.

All models were trained on a single GPU. The model with the lowest validation loss was saved for evaluation. After training, each saved model would be evaluated on the respective test set to obtain test accuracy, precision, recall, and F1-score. All metrics are reported on the test set. Precision, recall, and F1 were computed as macro-averages across classes (treating each class equally), appropriate since some datasets (like CIFAR-100 [9]) have many classes where it would ensure balanced performance.

Evaluation and Reporting

Evaluation and Reporting: The results are compiled in tables for each dataset. Each table lists all augmentation methods (including "None") and the performance metrics (Acc, Prec, Rec, F1) achieved by each of the four models under that augmentation. This allows side-by-side comparison of how a given augmentation affects different models, and which model/augmentation pairing yields the best results for that dataset.The test set performance was emphasized to compare generalization outcomes under each augmentation strategy. Validation results were only used for early stopping and are not reported, in line with focusing on generalization rather than the training process.

By analyzing these results, the trends such as which augmentation gives the highest boost in accuracy, how much each model benefits in absolute terms, and whether augmentations that work well for CNNs also do so for Transformers were identified. In the following section, the detailed result tables were presented and the findings for each dataset were discussed, followed by a cross-dataset comparison.

4 Results

The performance of each model (ResNet-18 [3], ResNet-50 [3], ViT [4], Swin [5]) on each dataset under all augmentation settings was presented below. Tables 1–4 summarize the test Accuracy (**Acc**), Precision (**Prec**), Recall (**Rec**), and F1-score (**F1**) for every combination, with all metrics reported as percentages. A "None" augmentation denotes the baseline with no data augmentation.After each table, analysis specific to the corresponding dataset was provided, followed by a discussion of the overall patterns observed.

4.1 CIFAR-10 Results

Table 1 shows the test results on CIFAR-10 [9]. CIFAR-10 [9] is a 10-class natural image task of moderate difficulty. Baseline performance with no augmentation is already fairly high for all models, especially the transformers which benefit from pretraining. Still, notable improvements were observed with certain augmentations.

Several clear patterns emerge from Table 1. First, all augmentation methods (except one) improved ResNet-18 [3] over the baseline of 93.97% accuracy. The best augmentation for ResNet-18 on CIFAR-10 [9] was CutMix [8], achieving 95.65% accuracy (a +1.68% gain over no augmentation). Close behind were AutoAugment [14], RandAugment [15], TrivialAugment [16], and others around 95.2–95.5%, indicating that ResNet-18 benefits from a variety of augmentations. For ResNet-50 [3], which started from a stronger baseline of 95.76%, the highest accuracy reached was 96.90% with TrivialAugment [16], about +1.1% improvement. Many augmentations gave ResNet-50 [3] a modest boost in the mid-96% range (CutMix [8], AugMix [17], RandAugment [15] all around 96.6–96.7%). Notably, CutMix [8] and TrivialAugment [16] are standouts for CNNs on CIFAR-10 [9], aligning

TABLE 1
Full CIFAR-10 test metrics for every model–augmentation pair (**ESE** = early-stopping epoch). Abbreviations: **gb**=Gaussian Blur, **cj**=Color Jitter, **re**=Random Erasing, **rp**=Random Perspective, **rrc**=Random Resized Crop.

Mod.	Aug.	ESE	Acc	Prec	Rec	F1
resnet50	affine	15.00	0.9652	0.9652	0.9652	0.9652
	augmix	28.00	0.9667	0.9667	0.9667	0.9666
	autoaugment	22.00	0.9639	0.9640	0.9639	0.9639
	cj	14.00	0.9540	0.9543	0.9540	0.9540
	cutmix	59.00	0.9643	0.9648	0.9643	0.9644
	cutout	27.00	0.9632	0.9633	0.9632	0.9631
	flip	15.00	0.9611	0.9612	0.9611	0.9611
	gb	13.00	0.9555	0.9554	0.9555	0.9553
	mixup	24.00	0.9664	0.9664	0.9664	0.9663
	none	13.00	0.9576	0.9576	0.9576	0.9576
	randaugment	18.00	0.9671	0.9671	0.9671	0.9671
	re	21.00	0.9623	0.9625	0.9623	0.9622
	rp	17.00	0.9606	0.9611	0.9606	0.9605
	rrc	29.00	0.9604	0.9604	0.9604	0.9604
	rotate	15.00	0.9628	0.9629	0.9628	0.9628
	trivialaugment	52.00	0.9690	0.9690	0.9690	0.9690
resnet18	affine	22.00	0.9521	0.9524	0.9521	0.9520
	augmix	26.00	0.9493	0.9493	0.9493	0.9492
	autoaugment	49.00	0.9523	0.9522	0.9523	0.9522
	cj	18.00	0.9404	0.9402	0.9404	0.9402
	cutmix	73.00	0.9565	0.9566	0.9565	0.9564
	cutout	25.00	0.9509	0.9510	0.9509	0.9508
	flip	20.00	0.9451	0.9453	0.9451	0.9451
	gb	19.00	0.9437	0.9438	0.9437	0.9437
	mixup	41.00	0.9522	0.9523	0.9522	0.9522
	none	16.00	0.9397	0.9394	0.9397	0.9395
	randaugment	22.00	0.9545	0.9544	0.9545	0.9544
	re	27.00	0.9493	0.9495	0.9493	0.9492
	rp	19.00	0.9487	0.9489	0.9487	0.9488
	rrc	44.00	0.9433	0.9437	0.9433	0.9434
	rotate	19.00	0.9506	0.9508	0.9506	0.9506
	trivialaugment	32.00	0.9540	0.9539	0.9540	0.9539
swin	affine	15.00	0.9859	0.9860	0.9859	0.9859
	augmix	12.00	0.9871	0.9871	0.9871	0.9871
	autoaugment	16.00	0.9867	0.9868	0.9867	0.9867
	cj	13.00	0.9862	0.9862	0.9862	0.9862
	cutmix	15.00	0.9887	0.9888	0.9887	0.9887
	cutout	22.00	0.9881	0.9881	0.9881	0.9881
	flip	14.00	0.9858	0.9858	0.9858	0.9858
	gb	12.00	0.9848	0.9849	0.9848	0.9848
	mixup	21.00	0.9886	0.9887	0.9886	0.9886
	none	12.00	0.9883	0.9883	0.9883	0.9883
	randaugment	13.00	0.9869	0.9870	0.9869	0.9869
	re	13.00	0.9872	0.9873	0.9872	0.9872
	rp	15.00	0.9842	0.9846	0.9842	0.9842
	rrc	15.00	0.9886	0.9887	0.9886	0.9886
	rotate	18.00	0.9872	0.9872	0.9872	0.9872
	trivialaugment	14.00	0.9872	0.9873	0.9872	0.9872
vit	affine	31.00	0.9858	0.9859	0.9858	0.9858
	augmix	13.00	0.9862	0.9862	0.9862	0.9862
	autoaugment	23.00	0.9861	0.9863	0.9861	0.9861
	cj	12.00	0.9845	0.9846	0.9845	0.9845
	cutmix	25.00	0.9877	0.9877	0.9877	0.9877
	cutout	22.00	0.9879	0.9879	0.9879	0.9879
	flip	12.00	0.9867	0.9867	0.9867	0.9867
	gb	12.00	0.9853	0.9853	0.9853	0.9853
	mixup	49.00	0.9860	0.9861	0.9860	0.9860
	none	18.00	0.9842	0.9843	0.9842	0.9842
	randaugment	17.00	0.9892	0.9892	0.9892	0.9892
	re	23.00	0.9865	0.9866	0.9865	0.9865
	rp	15.00	0.9866	0.9866	0.9866	0.9866
	rrc	22.00	0.9895	0.9895	0.9895	0.9895
	rotate	29.00	0.9846	0.9846	0.9846	0.9846
	trivialaugment	18.00	0.9882	0.9883	0.9882	0.9882

with prior findings that mixing strategies and automated policies can yield superior performance on natural image benchmarks [10], [13].

The Vision Transformer (ViT [4]) achieved an extremely high baseline of 98.42% on CIFAR-10 [9], likely due to the combination of model capacity and pretraining. Augmentation had relatively little headroom to improve ViT [4], yet yet a small gain is still observed: the best was 98.95% accuracy with Random Resized Crop (the standard cropping augmentation), followed closely by RandAugment [15] (98.92%) and TrivialAugment [16] (98.83%). These improvements ($\sim$+0.5%) are modest but consistent. It is interesting that Random Resized Crop yielded the top result for ViT [4] and also for the Swin [5] Transformer (98.86%, barely edging out its baseline of 98.83%). Random cropping is a staple augmentation for ImageNet training[1], and for these pretrained transformers (which expect 224×224 inputs), fine-tuning was performed with random resized crops likely centered around the object. The result suggests that even for small CIFAR images, letting the model see random zoomed-in portions helped the transformers latch onto features similarly to how they were pre-trained, thus giving a slight boost.The Swin Transformer's performance was essentially saturated; its baseline 98.83% rose to 98.86% at best (Random Crop), an insignificant difference. Swin [5] already nearly achieves the dataset's ceiling, so most augmentations made no appreciable difference—almost every augmentation kept Swin [5] in the 98.5–98.8% range. This underscores that for very high-performing models on simple tasks, augmentation has diminishing returns.

One negative outcome in Table 1 is that Color Jitter slightly decreased ResNet-50 [3]'s accuracy (95.40%) compared to baseline. This small drop (-0.36%) suggests that altering color intensities might have introduced noise without benefit for CIFAR-10 [9], possibly because the model already sees enough color variation or because the jitter sometimes created unrealistic colors. Similarly, Random Perspective marginally reduced Swin [5]'s accuracy (98.42% vs. 98.83% baseline). However, these drops are minor. Almost all other augmentations were neutral or positive.

In terms of precision, recall, and F1, the values track very closely with accuracy for CIFAR-10 [9], as expected in a balanced multi-class setting. The highest F1 for ResNet-18 [3] (95.64 with CutMix [8]) corresponds to its highest accuracy, and likewise for other models. This implies augmentations that improve overall accuracy are not trading off precision vs. recall in any significant way; rather, they generally improve the model's performance across all classes uniformly.

Interestingly, Cutout [12] and Random Erasing [13] provided decent gains for ResNet-18 [3] (95.09% and 94.93% accuracy) and ResNet-50 ($\sim$96.3% and 96.2%). These gains, while not the highest, show that even simpler region-erasing can help CNNs. ViT and Swin [5] also did not degrade with these: ViT [4] actually hit 98.79% with Cutout [12], one of its better results. This suggests that forcing the models to handle missing parts of objects is a beneficial regularisation for both architectures.

Overall, for CIFAR-10 [9] it was found that ResNet [3] models benefited the most from advanced augmentations (CutMix [8], Rand/TrivialAugment [16], AugMix [17]), with up to $\sim$1.5–1.7% accuracy increases. Transformer models were already near peak performance; augmentations gave at most $\sim$0.5% gain to ViT [4] and negligible to Swin [5]. This implies that when a model has high capacity and pretraining (like our transformers), CIFAR-10 [9] is easy enough that augmentations make little difference except perhaps ensuring the model does not overfit early. Indeed, it was observed that, in training, ResNet-50 [3] without augmentation triggered early stopping after only 13 epochs (validation loss stopped improving), whereas with a strong augmentation like CutMix [8] it continued training until 59 epochs before early stopping. This indicates augmentation delayed overfitting significantly, leading to a better model. In contrast, Swin [5]with or without augmentation often converged around a similar epoch because it overfits less readily, given its strong prior.

4.2 CIFAR-100 Results

Table 2 presents the results on the CIFAR-100 [9] dataset, which has 100 classes and is more challenging than CIFAR-10 [9]. CIFAR-100 [9]'s increased difficulty makes augmentation potentially more valuable. Baseline accuracies are generally lower than CIFAR-10 [9], and correspondingly larger relative improvements from augmentation for CNNs.

Augmentations have a significantly larger impact on CIFAR-100 [9], especially for CNN models. ResNet-18 [3]'s baseline accuracy of 77.62% jumped to 81.16% with TrivialAugment [16] – an absolute improvement of +3.54%, which is substantial for this task. This makes intuitive sense: with 100 classes, each class has only 500 training images, so augmentation effectively increases the variety of each class's training examples, yielding a big payoff. TrivialAugment [16]'s simple but diverse one-transform policy turned out best for ResNet-18, slightly ahead of CutMix [8] (80.24%) and others like Random Erasing [13] (80.08%) and Affine (80.10%). Interestingly, even simple flips and rotations gave $\sim$79.5–79.8%, a couple points above baseline, indicating almost any reasonable augmentation helps ResNet-18 on CIFAR-100.

ResNet-50 [3] also gained significantly: from 80.61% to 83.80% with CutMix [8], a +3.19% boost. In fact, CutMix [8] provided the highest accuracy among all methods for ResNet-50, consistent with Yun et al. [10] who showed CutMix [8] excels on challenging classification tasks by effectively augmenting training images with mixed class content. TrivialAugment [16] was a close second for ResNet-50 [3] (83.56%), followed by Mixup [7] (83.03%) and Random Erasing [13]/Cutout [12] ($\approx$82.95–82.97%). AutoAugment [14] performed slightly worse (81.89%) on ResNet-50 [3], possibly because its fixed policy might not perfectly match CIFAR-100 [9] distribution, whereas the on-the-fly random nature of RandAugment [15] (83.17%) and others gave a bit more.

For the ViT [4] on CIFAR-100 [9], the baseline was already very high at 90.58%, reflecting the transformer's superior capacity on this task. Augmentation still improved ViT [4] to 92.30% with TrivialAugment [16] (a +1.72% gain). Notably, ViT [4]'s second-best Random Erasing [13] $\sim$92.17%, then was CutMix [8] at 92.01%, Mixup [7] $\sim$91.88%. It appears ViT [4] benefited most from the same augmentations as ResNet-18 [3] did: trivial and mixup [7]/CutMix [8]. This

TABLE 2
Full CIFAR-100 test metrics for every model–augmentation pair (**ESE** = epoch of early-stopping). Abbreviations: **gb**=Gaussian Blur, **cj**=Color Jitter, **re**=Random Erasing, **rp**=Random Perspective, **rrc**=Random Resized Crop.

Model	Aug.	FSE	Acc	Prec	Rec	F1
resnet50	affine	19.00	0.8287	0.8318	0.8287	0.8288
	augmix	19.00	0.8245	0.8278	0.8245	0.8246
	autoaugment	26.00	0.8189	0.8210	0.8189	0.8185
	cj	17.00	0.8046	0.8086	0.8046	0.8044
	cutmix	38.00	0.8380	0.8404	0.8380	0.8377
	cutout	19.00	0.8295	0.8337	0.8295	0.8300
	flip	18.00	0.8222	0.8265	0.8222	0.8220
	gb	16.00	0.8035	0.8082	0.8035	0.8038
	mixup	35.00	0.8303	0.8334	0.8303	0.8302
	none	16.00	0.8061	0.8114	0.8061	0.8068
	randaugment	20.00	0.8317	0.8351	0.8317	0.8316
	re	19.00	0.8297	0.8335	0.8297	0.8300
	rp	18.00	0.8234	0.8280	0.8234	0.8239
	rrc	36.00	0.8215	0.8256	0.8215	0.8221
	rotate	17.00	0.8211	0.8245	0.8211	0.8214
	trivialaugment	27.00	0.8356	0.8388	0.8356	0.8353
resnet18	affine	30.00	0.8010	0.8040	0.8010	0.8009
	augmix	29.00	0.7906	0.7925	0.7906	0.7899
	autoaugment	36.00	0.7952	0.7965	0.7952	0.7947
	cj	24.00	0.7836	0.7856	0.7836	0.7833
	cutmix	40.00	0.8024	0.8039	0.8024	0.8018
	cutout	34.00	0.7995	0.8017	0.7995	0.7992
	flip	27.00	0.7982	0.8012	0.7982	0.7982
	gb	23.00	0.7838	0.7874	0.7838	0.7841
	mixup	34.00	0.7971	0.7981	0.7971	0.7960
	none	22.00	0.7762	0.7799	0.7762	0.7764
	randaugment	31.00	0.7991	0.8018	0.7991	0.7990
	re	30.00	0.8008	0.8035	0.8008	0.8005
	rp	29.00	0.7941	0.7981	0.7941	0.7942
	rrc	49.00	0.7883	0.7922	0.7883	0.7883
	rotate	28.00	0.7954	0.7992	0.7954	0.7955
	trivialaugment	45.00	0.8116	0.8129	0.8116	0.8110
swin	affine	14.00	0.9171	0.9194	0.9171	0.9172
	augmix	16.00	0.9178	0.9194	0.9178	0.9178
	autoaugment	15.00	0.9209	0.9228	0.9209	0.9209
	cj	14.00	0.9170	0.9183	0.9170	0.9171
	cutmix	17.00	0.9304	0.9320	0.9304	0.9305
	cutout	14.00	0.9182	0.9205	0.9182	0.9183
	flip	13.00	0.9163	0.9187	0.9163	0.9165
	gb	14.00	0.9161	0.9186	0.9161	0.9162
	mixup	17.00	0.9256	0.9268	0.9256	0.9256
	none	13.00	0.9180	0.9202	0.9180	0.9179
	randaugment	14.00	0.9188	0.9206	0.9188	0.9187
	re	14.00	0.9216	0.9231	0.9216	0.9214
	rp	13.00	0.9194	0.9215	0.9194	0.9194
	rrc	19.00	0.9239	0.9252	0.9239	0.9238
	rotate	13.00	0.9169	0.9188	0.9169	0.9168
	trivialaugment	16.00	0.9220	0.9233	0.9220	0.9219
vit	affine	20.00	0.9125	0.9145	0.9125	0.9120
	augmix	19.00	0.9174	0.9193	0.9174	0.9176
	autoaugment	23.00	0.9128	0.9150	0.9128	0.9132
	cj	14.00	0.9127	0.9143	0.9127	0.9127
	cutmix	26.00	0.9201	0.9214	0.9201	0.9202
	cutout	17.00	0.9181	0.9197	0.9181	0.9183
	flip	15.00	0.9139	0.9157	0.9139	0.9142
	gb	14.00	0.9117	0.9135	0.9117	0.9120
	mixup	36.00	0.9188	0.9199	0.9188	0.9188
	none	14.00	0.9058	0.9095	0.9058	0.9062
	randaugment	20.00	0.9169	0.9187	0.9169	0.9171
	re	18.00	0.9217	0.9228	0.9217	0.9217
	rp	16.00	0.9178	0.9196	0.9178	0.9179
	rrc	22.00	0.9221	0.9241	0.9221	0.9223
	rotate	19.00	0.9104	0.9133	0.9104	0.9105
	trivialaugment	22.00	0.9230	0.9246	0.9230	0.9231

suggests that even for a high-performing model, adding diverse or mixed-sample augmentations confers a small yet measurable improvement in classification accuracy on CIFAR-100 [9].

The Swin Transformer [5] had an impressive baseline of 91.80% and reached 93.04% with CutMix [8], a +1.24% increase.Mixup [7] also gave Swin [5] 92.56%.Swin [5]'s best (93.04%) slightly surpasses ViT [4]'s best (92.30%), reflecting that Swin [5]'s inductive bias and architecture help it on CIFAR-100 [9] as well. Both transformers clearly outstrip the CNNs here in absolute accuracy, but the CNNs narrowed the gap with augmentation (ResNet-50 [3] went from ∼10% less accuracy than ViT [4] to only ∼8.5% less after augmentation, for instance).

One observation is that CutMix [8] was extremely effective across the board on CIFAR-100 [9]: it is the top method for ResNet-50 [3] and Swin [5], and among the top three for ResNet-18 [3] and ViT [4]. This aligns with the idea that mixing image regions is very helpful when you have many classes and finer details – by cut-and-pasting between images, the models see more varied compositions and learn features that generalize better [8]. TrivialAugment [16] also shone for ResNet-18 [3] and ViT, showing that even without tuning, a random augmentation per image is powerful for diverse data. Perhaps the sheer number of classes benefits from a wide range of distortions (each class gets augmented in different random ways). Mixup [7] and Random Erasing [13]/Cutout [12] all gave solid improvements to both CNNs and ViT [4] as well. Basic augmentations like flip and rotate provided 1.5–2% gains for ResNet-18/50 [3], which is useful but clearly inferior to the advanced methods. This indicates that while traditional augmentations are beneficial, modern techniques can compound multiple distortions or generate novel samples to achieve greater improvement on complex datasets. AutoAugment [14]'s underperformance relative to simpler RandAugment [15] or TrivialAugment [16] for ResNet-50 [3] may be due to its policy being optimized for CIFAR-10 [9] originally; it might not have been optimal for CIFAR-100 [9]'s fine-grained classes (some recent works note that learned policies do not always transfer perfectly).

Precision, Recall, and F1 in Table 2 again track accuracy closely. One thing to note: for ResNet-18 [3], the precision (81.29) is slightly higher than recall (81.16) under TrivialAugment [16], suggesting it might be marginally more precise (fewer false positives) than exhaustive in recall. But the differences are tiny. The macro metrics being so close to accuracy means the augmentations improved performance uniformly across classes rather than just a subset: an important consideration for CIFAR-100 [9], where each class has few samples, is whether augmentation disproportionately helps some classes. The results imply most classes benefited roughly equally (otherwise we would see divergence between accuracy and macro-averaged precision/recall).

In summary, on CIFAR-100 [9] augmentation is crucial for CNNs, yielding up to 3–4% absolute improvements in test accuracy. Even for the high-performing Transformer models, augmentations like CutMix [8] and TrivialAugment [16] provided noticeable gains (∼1–2%). The best augmentations overall were CutMix [8] and TrivialAugment [16], with Mixup [7], Random Erasing [13], and RandAugment [15] also being effective. Simpler augmentations helped but to a lesser extent. There were no cases of severe degradation: the worst-case was perhaps AugMix [17] for ResNet-18 [3] (79.06%, baseline 77.62%—actually, AugMix [17] still improved ResNet-18 [3] by +1.44%, so even AugMix [17] helped; it is just on the lower end compared to others). Thus, all augmentations provided some benefit to CNNs on CIFAR-100 [9], though the magnitude varied.

4.3 MNIST Results

Table 3 contains the results on the *MNIST [10]* dataset (handwritten digits). MNIST [10] is a much simpler task, and indeed it could be seen that all models achieve $> 99.5\%$ accuracy even with no augmentation. Augmentation in such a scenario is expected to have minimal impact, and possibly could hurt if it alters digit shapes significantly.

As expected, the differences are extremely small. All models without augmentation already achieve between 99.5–99.7% accuracy on MNIST [10], indicating only a handful of test errors out of 10 000. Augmentations mostly hover around those values, within $\pm 0.2\%$ of baseline.

The only clearly detrimental augmentation is *Horizontal Flip*. Flipping digits horizontally can create images that are not valid digits (e.g., a "2" flipped is not a standard digit). The results show this: every model's performance drops with Flip. ResNet-18 [3] goes from 99.59% to 99.18% (a noticeable -0.41%), ResNet-50 from 99.56% to 99.17% (-0.39%), ViT [4] from 99.63% to 99.09% (-0.54%), and Swin from 99.66% to 99.27% (-0.39%). These drops, while small in absolute terms, are significant on MNIST [10] (since baseline error rates are so low, Flip roughly doubled the error count for some models). This confirms that applying an augmentation violating the data's inherent symmetries can hurt performance; in practice, one would avoid flips for MNIST [10], so this serves as a sanity check.

Apart from Flip, most augmentations did not significantly change performance. The best for ResNet-18 was Mixup [7] at 99.68% (a mere +0.09% over baseline). For ResNet-50, Color Jitter randomly gave 99.72% (+0.16%, likely within variance). For ViT [4], Cutout [12] gave 99.69% (+0.06%). For Swin [5], Random Erasing [13] reached 99.74% (+0.08%). These tiny improvements are likely not meaningful—essentially all models are at the asymptote of MNIST performance, and differences could be due to chance or slight regularisation preventing one or two mistakes.

Some augmentations show trivial differences: *Rotation*, because MNIST [10] digits can appear slightly rotated in the dataset, uses a small range (we applied $\pm 15°$) that neither harms nor meaningfully helps (ResNet-18 [3]: 99.62% vs. 99.59% baseline). Affine and Perspective transformations had similarly negligible effects. This is expected: slight warping of digits is tolerated because humans can still recognise them, and the models likely learn to as well.

Mixup [7] and CutMix [8] on MNIST [10] are somewhat curious. Mixup [7] linearly blends two digit images, producing a superimposed or blurred image that may not resemble either digit clearly (e.g., a mix of "1" and "7"). Given MNIST [10]'s simplicity, one might think Mixup [7] could harm clarity, yet it showed no ill effect and even marginally improved ResNet-18 [3]. Possibly Mixup [7] helped regularise the network on boundaries between classes. CutMix [8],

TABLE 3
Full MNIST test metrics for every model–augmentation pair (**ESE** = early-stopping epoch). Abbreviations: **gb**=Gaussian Blur, **cj**=Color Jitter, **re**=Random Erasing, **rp**=Random Perspective, **rrc**=Random Resized Crop.

Mod.	Aug.	ESE	Acc	Prec	Rec	F1
resnet50	affine	23.00	0.9959	0.9959	0.9959	0.9959
	augmix	21.00	0.9954	0.9954	0.9954	0.9954
	autoaugment	23.00	0.9960	0.9960	0.9960	0.9960
	cj	31.00	0.9972	0.9972	0.9972	0.9972
	cutmix	53.00	0.9959	0.9959	0.9959	0.9959
	cutout	22.00	0.9965	0.9965	0.9964	0.9965
	flip	13.00	0.9917	0.9917	0.9914	0.9915
	gb	16.00	0.9955	0.9955	0.9955	0.9955
	mixup	38.00	0.9963	0.9964	0.9963	0.9963
	none	17.00	0.9956	0.9955	0.9956	0.9956
	randaugment	22.00	0.9966	0.9966	0.9966	0.9966
	re	31.00	0.9964	0.9964	0.9964	0.9964
	rp	18.00	0.9953	0.9954	0.9953	0.9953
	rrc	35.00	0.9952	0.9953	0.9951	0.9952
	rotate	29.00	0.9960	0.9960	0.9959	0.9960
	trivialaugment	20.00	0.9969	0.9969	0.9968	0.9969
resnet18	affine	19.00	0.9954	0.9955	0.9954	0.9954
	augmix	19.00	0.9959	0.9960	0.9958	0.9959
	autoaugment	23.00	0.9962	0.9962	0.9961	0.9962
	cj	24.00	0.9961	0.9962	0.9961	0.9961
	cutmix	79.00	0.9963	0.9963	0.9962	0.9963
	cutout	37.00	0.9954	0.9955	0.9953	0.9954
	flip	25.00	0.9918	0.9918	0.9915	0.9916
	gb	16.00	0.9958	0.9958	0.9958	0.9958
	mixup	76.00	0.9968	0.9968	0.9967	0.9968
	none	19.00	0.9959	0.9959	0.9958	0.9959
	randaugment	17.00	0.9960	0.9961	0.9959	0.9960
	re	22.00	0.9950	0.9950	0.9949	0.9950
	rp	23.00	0.9957	0.9958	0.9956	0.9957
	rrc	41.00	0.9958	0.9959	0.9958	0.9958
	rotate	17.00	0.9962	0.9962	0.9962	0.9962
	trivialaugment	26.00	0.9966	0.9966	0.9965	0.9966
swin	affine	16.00	0.9959	0.9960	0.9958	0.9959
	augmix	22.00	0.9960	0.9960	0.9959	0.9960
	autoaugment	18.00	0.9948	0.9949	0.9946	0.9947
	cj	15.00	0.9927	0.9928	0.9929	0.9928
	cutmix	29.00	0.9969	0.9969	0.9969	0.9969
	cutout	20.00	0.9965	0.9965	0.9965	0.9965
	flip	12.00	0.9927	0.9926	0.9926	0.9926
	gb	13.00	0.9961	0.9961	0.9961	0.9961
	mixup	30.00	0.9968	0.9968	0.9968	0.9968
	none	15.00	0.9966	0.9966	0.9965	0.9966
	randaugment	22.00	0.9961	0.9961	0.9960	0.9961
	re	22.00	0.9974	0.9974	0.9974	0.9974
	rp	15.00	0.9960	0.9960	0.9960	0.9960
	rrc	33.00	0.9943	0.9942	0.9942	0.9942
	rotate	18.00	0.9955	0.9955	0.9955	0.9955
	trivialaugment	18.00	0.9961	0.9961	0.9961	0.9961
vit	affine	32.00	0.9959	0.9959	0.9959	0.9959
	augmix	26.00	0.9953	0.9954	0.9953	0.9953
	autoaugment	26.00	0.9959	0.9960	0.9959	0.9959
	cj	25.00	0.9932	0.9932	0.9931	0.9931
	cutmix	33.00	0.9955	0.9955	0.9954	0.9955
	cutout	27.00	0.9969	0.9969	0.9969	0.9969
	flip	22.00	0.9909	0.9909	0.9908	0.9908
	gb	34.00	0.9963	0.9963	0.9963	0.9963
	mixup	24.00	0.9956	0.9956	0.9955	0.9956
	none	31.00	0.9963	0.9963	0.9962	0.9963
	randaugment	17.00	0.9948	0.9949	0.9947	0.9948
	re	25.00	0.9965	0.9965	0.9965	0.9965
	rp	30.00	0.9959	0.9959	0.9959	0.9959
	rrc	43.00	0.9960	0.9960	0.9960	0.9960
	rotate	29.00	0.9959	0.9959	0.9959	0.9959
	trivialaugment	36.00	0.9955	0.9955	0.9955	0.9955

which pastes part of one digit onto another (producing hybrid digits), did not significantly degrade performance; all models stayed around 99.5–99.6%. Networks appear robust enough to ignore the small pasted patch if it is incoherent, or such hybrids are too rare to matter.

The automated policies (AutoAugment [14], RandAugment [15], TrivialAugment [16], AugMix [17]) all had essentially no effect, landing in the 99.5–99.6% range. They neither helped nor hurt in any meaningful way. Although AutoAugment [14] has a learned policy for MNIST [10], the dataset is already so easy that gains are marginal.

In summary, MNIST [10] shows that data augmentation is largely unnecessary when the model already achieves near-perfect performance. The sole cautionary tale is that augmentations violating task semantics (flipping non-symmetric digits) will degrade accuracy and should be avoided. Otherwise, models are robust enough that augmentations cause minimal change. Early stopping triggered very quickly on MNIST [10] runs—many models converged in under 10 epochs even without augmentation. Augmentations sometimes extended training by an epoch or two at most, as there was little overfitting to combat. Thus, MNIST [10] primarily highlights that augmentation has diminishing returns on extremely simple and well-sampled tasks.

4.4 Fashion-MNIST Results

Table 4 shows results on *Fashion-MNIST [11]*, a more challenging 10-class problem than MNIST [10] (clothing images are more complex than digits). Baseline accuracies are around 93–95%, leaving some room for improvement. Indeed, we see augmentations providing roughly 1% gains—similar to CIFAR-10 [9], though not as large as for CIFAR-100 [9] .

Augmentation yields modest improvements on Fashion-MNIST [11], roughly on par with CIFAR-10 [9] in magnitude (since baseline accuracy $\sim 94\%$). Key observations:

ResNet-18 [3] baseline 93.77% improved to 94.68% with CutMix [8] (+0.91%). Mixup [7], Erasing [13], Cutout [12], and RandAugment [15] all gave around 94.4–94.5%, so ~ 0.6–0.7% gains. These are small but consistent; every augmentation even Gaussian Blur (94.18%) improved ResNet-18 at least slightly. The best single augmentation for ResNet-18 [3] was CutMix [8], as on CIFAR-10 [9]. This suggests that mixing clothing images yields new training samples that help the CNN pick up additional features (e.g., a patch of a shoe on a shirt forces attention to detail).

ResNet-50 [3] baseline 93.78% rose to 94.88% with RandAugment [15] (+1.10%), the highest among augmentations. Interestingly, for ResNet-50 [3], RandAugment [15] edged out CutMix [8] (94.57%) and Mixup [7] (94.59%). This might indicate that a balanced combination of simple transforms (as RandAugment [15] does) was especially useful for the larger CNN, possibly because it can already fit the training data well and benefits from varied regular distortions more than from synthetic image mixing. However, the differences are small; CutMix [8] and TrivialAugment [16] (94.58%) are practically the same as RandAugment [15]'s result within 0.3%. So one could say many augmentations help ResNet-50 [3] about $\sim 1\%$ on Fashion-MNIST [11]. The only ones that seem not to help are those that possibly remove too

TABLE 4
Full Fashion-MNIST test metrics for every model–augmentation pair (**ESE** = early-stopping epoch). Abbreviations: **gb**=Gaussian Blur, **cj**=Color Jitter, **re**=Random Erasing, **rp**=Random Perspective, **rrc**=Random Resized Crop.

Mod.	Aug.	ESE	Acc	Prec	Rec	F1
resnet50	affine	16.00	0.9426	0.9424	0.9426	0.9424
	augmix	15.00	0.9430	0.9428	0.9430	0.9427
	autoaugment	19.00	0.9452	0.9453	0.9452	0.9449
	cj	13.00	0.9367	0.9387	0.9367	0.9373
	cutmix	32.00	0.9457	0.9456	0.9457	0.9455
	cutout	15.00	0.9434	0.9432	0.9434	0.9430
	flip	14.00	0.9412	0.9416	0.9412	0.9412
	gb	13.00	0.9352	0.9349	0.9352	0.9346
	mixup	25.00	0.9459	0.9462	0.9459	0.9459
	none	13.00	0.9378	0.9379	0.9378	0.9374
	randaugment	19.00	0.9488	0.9493	0.9488	0.9490
	re	13.00	0.9358	0.9363	0.9358	0.9357
	rp	15.00	0.9452	0.9453	0.9452	0.9447
	rrc	24.00	0.9283	0.9321	0.9283	0.9291
	rotate	16.00	0.9434	0.9439	0.9434	0.9435
	trivialaugment	18.00	0.9458	0.9461	0.9458	0.9458
resnet18	affine	20.00	0.9421	0.9420	0.9421	0.9419
	augmix	18.00	0.9450	0.9450	0.9450	0.9449
	autoaugment	20.00	0.9459	0.9457	0.9459	0.9455
	cj	15.00	0.9400	0.9400	0.9400	0.9399
	cutmix	48.00	0.9468	0.9468	0.9468	0.9461
	cutout	17.00	0.9442	0.9439	0.9442	0.9438
	flip	15.00	0.9402	0.9399	0.9402	0.9400
	gb	16.00	0.9418	0.9421	0.9418	0.9419
	mixup	25.00	0.9447	0.9446	0.9447	0.9445
	none	15.00	0.9377	0.9381	0.9377	0.9379
	randaugment	25.00	0.9454	0.9453	0.9454	0.9451
	re	19.00	0.9436	0.9436	0.9436	0.9435
	rp	20.00	0.9424	0.9434	0.9424	0.9427
	rrc	32.00	0.9408	0.9405	0.9408	0.9403
	rotate	18.00	0.9404	0.9410	0.9404	0.9405
	trivialaugment	24.00	0.9427	0.9428	0.9427	0.9425
swin	affine	16.00	0.9520	0.9521	0.9520	0.9520
	augmix	15.00	0.9551	0.9550	0.9551	0.9550
	autoaugment	14.00	0.9496	0.9504	0.9496	0.9492
	cj	14.00	0.9539	0.9540	0.9539	0.9539
	cutmix	18.00	0.9589	0.9589	0.9589	0.9589
	cutout	14.00	0.9552	0.9552	0.9552	0.9551
	flip	14.00	0.9506	0.9508	0.9506	0.9504
	gb	13.00	0.9533	0.9538	0.9533	0.9534
	mixup	15.00	0.9579	0.9578	0.9579	0.9577
	none	14.00	0.9487	0.9489	0.9487	0.9487
	randaugment	14.00	0.9531	0.9533	0.9531	0.9530
	re	14.00	0.9556	0.9560	0.9556	0.9557
	rp	14.00	0.9546	0.9547	0.9546	0.9544
	rrc	23.00	0.9459	0.9464	0.9459	0.9460
	rotate	15.00	0.9530	0.9529	0.9530	0.9527
	trivialaugment	15.00	0.9544	0.9548	0.9544	0.9545
vit	affine	15.00	0.9418	0.9417	0.9418	0.9415
	augmix	17.00	0.9454	0.9456	0.9454	0.9446
	autoaugment	20.00	0.9503	0.9502	0.9503	0.9501
	cj	14.00	0.9432	0.9432	0.9432	0.9426
	cutmix	27.00	0.9530	0.9532	0.9530	0.9530
	cutout	15.00	0.9449	0.9457	0.9449	0.9450
	flip	13.00	0.9380	0.9379	0.9380	0.9378
	gb	13.00	0.9403	0.9402	0.9403	0.9400
	mixup	19.00	0.9471	0.9473	0.9471	0.9470
	none	14.00	0.9421	0.9425	0.9421	0.9412
	randaugment	17.00	0.9488	0.9491	0.9488	0.9489
	re	16.00	0.9447	0.9449	0.9447	0.9446
	rp	19.00	0.9438	0.9457	0.9438	0.9444
	rrc	21.00	0.9454	0.9454	0.9454	0.9451
	rotate	16.00	0.9409	0.9419	0.9409	0.9411
	trivialaugment	17.00	0.9494	0.9492	0.9494	0.9492

much information: e.g., Random Resized Crop dropped R50 to 92.83% (a -0.95% drop, the biggest negative effect noted). Cropping out a random part of a clothing item (and resizing) likely sometimes removes important parts (like cropping out the top of a shoe or half of a dress) making the image ambiguous. This indicates Random cropping is not advisable for Fashion-MNIST [11], unlike for CIFAR [9] where objects are centered. In Fashion-MNIST [11], the items are centered and relatively tight; cropping them can easily cut off distinguishing features (sleeves, straps, etc.), confusing the model. The results reflect that: ResNet-50 [3] and Swin [5] both saw dips with Random Resized Crop (R50: 93.78%→92.83%, Swin [5]: 94.87%→94.59%, a smaller drop). ViT [4] wasn't hurt (94.21%→94.54%, slight improvement), possibly because the ViT [4] might handle partial information better due to global context or because it was pretrained on many images where cropping is standard.

The ViT [4] baseline 94.21% improved to 95.30% with CutMix [8] (+1.09%). This mirrors ViT [4]'s behavior on CIFAR-100 [9] where CutMix [8] was among the top. ViT [4] also got 94.94% with TrivialAugment [16], 94.88% with RandAugment [15], $\sim$ 94.7% with Mixup [7], etc. So all these gave roughly +0.5 to +1.1%. The best being CutMix [8] suggests the ViT [4] also gains from seeing partial images combined—perhaps it helps the attention layers focus on salient garment parts. ViT [4] did not benefit from Flip (dropped to 93.80%, a slight -0.41%). Flipping a clothing item horizontally usually doesn't change its class (a shoe or shirt mirrored is still the same type of clothing), so theoretically Flip should be fine. The small drop for ViT [4] might be noise or an indication that perhaps some fashion items have subtle asymmetry (maybe left/right shoe, but the dataset does not distinguish left vs. right shoe; they are just "ankle boot", etc.). In any case, Flip is neutral for CNNs and slightly negative for ViT [4] here.

The Swin [5] Transformer baseline 94.87% went up to 95.89% with CutMix [8](+1.02%). This was the best result among all models on Fashion-MNIST [11]. Swin [5] also reached 95.79% with Mixup [7] and 95.56% with Random Erasing [13]. So for Swin [5], mixing strategies clearly yielded the highest accuracy (CutMix [8] $>$ Mixup [7] $>$ others). This aligns with the idea that even for transformers, mixing augmentations provide a form of training sample interpolation that enhances generalisation[6]. Swin [5] handled flips and rotations fine (slight improvements actually to 95.06% and 95.30%). As noted, Random Resized Crop hurt Swin [5] a bit (94.59%, -0.28%), echoing the caution on cropping out important content.

It should noted that unlike MNIST [10], none of the augmentations here are semantically wrong for the task. Horizontal Flip is actually reasonable for fashion items (most clothing looks similar mirrored), and indeed Flip gave small improvements to ResNet [3] and Swin [5], and negligible effect on R50 and ViT [4]. Thus, the augmentations that helped or hurt were more about losing information vs. adding variety. Cropping (losing info) hurt as discussed. Gaussian Blur and heavy Color Jitter could potentially remove detail: the results for R50 show slight drops with Blur (93.52%) and Jitter (93.67%), whereas Swin [5] saw slight gains (Blur 95.33%, Jitter 95.39%). These differences are small, but possibly the transformer, with its ability to integrate context, is less bothered by a slightly blurred image than the CNN, which might rely on fine local details.

The overall best augmentation across models for Fashion-MNIST [11] appears to be CutMix [8]—it is top for ResNet-18 [3], ViT [4], Swin [5], and only slightly behind RandAugment [15] for ResNet-50. Mixup [7] is consistently high as well. This suggests that for this moderate-complexity dataset, mixing augmentations are broadly effective for both CNN and transformer architectures. They likely enrich the training set by creating new combinations (e.g., half T-shirt and half coat), forcing the model to focus on distinguishing features for each class. Meanwhile, geometric transforms (rotate, affine) and photometric (color, blur) provide smaller benefits, probably because the dataset already has some variability and the models can handle minor appearance changes.

Precision/Recall differences are minimal, indicating augmentations did not cause any significant bias toward precision vs. recall. For instance, in the best Swin [5] result (CutMix [8]), Prec = Rec = 95.89%. That means it improved both false-positive and false-negative rates evenly across classes.

Finally, early stopping in these experiments showed that with augmentations like CutMix [8] or RandAugment [15], the models would train a few epochs longer before validation loss plateaued, compared to no augmentation. This indicates improved generalisation. For example, ResNet-50 [3] on Fashion-MNIST [11] with no augmentation stopped around epoch 10 (with $\sim 94\%$ validation accuracy), whereas with RandAugment [15] it continued to epoch $\sim$ 18 and achieved $\sim 95\%$ validation accuracy before stopping. This corroborates the $\sim$ 1% gain seen in test accuracy with augmentation.

4.5 Combined Augmentation Test Results on CIFAR-100 [9]

Table 5 presents the CIFAR-100 [9] results when using combined augmentation pipelines, each composed of multiple manual transforms (excluding automated policies). Five such multi-step pipelines were evaluated: (1) flip + rotate + color jitter (frtcj), (2) flip + random resized crop + Gaussian blur (frcgb), (3) affine + random perspective + Cutout [12] (arpco), (4) flip + rotate + color jitter + Mixup [7] (frtcjmx), and (5) flip + random resized crop + CutMix [8] (frccm). These compound strategies aim to cover diverse transformations in one pipeline, and their performance is compared to the no augmentation baseline on CIFAR-100 [9]. All four models (ResNet-18 [3], ResNet-50 [3], ViT [4], Swin [5]) are considered, with accuracy, precision, recall, and F1-score reported for each. The question is whether combining augmentations yields added benefits beyond the best single augmentations observed in Section 4.2, or if diminishing returns occur.

4.5.1 *ResNet-18*

The baseline (no augmentation) accuracy for ResNet–18 [3] on CIFAR–100 [9] was 77.62%. All five combined augmentation pipelines substantially improved this result, underscoring the value of diverse data transforms for this 100-class task. The flip+rotate+color-jitter pipeline (`frtcj`) reached

TABLE 5
CIFAR-100 results for **combined** augmentation pipelines (**ESE** = early-stopping epoch). Abbreviations for multi-step pipelines: **arpco**=affine + random perspective + cutout, **frccm**=flip + random resized crop + cutmix, **frcgb**=flip + random resized crop + gaussian blur, **frtcj**=flip + rotate + color jitter, **frtcjmx**=flip + rotate + color jitter + mixup, **gb**=Gaussian Blur, **cj**=Color Jitter, **rp**=Random Perspective, **rrc**=Random Resized Crop.

Mod.	Aug.	ESE	Acc	Prec	Rec	F1
resnet50	arpco	25.00	0.8376	0.8418	0.8376	0.8378
	frccm	51.00	0.8445	0.8489	0.8445	0.8446
	frcgb	33.00	0.8326	0.8355	0.8326	0.8327
	frtcj	20.00	0.8364	0.8386	0.8364	0.8361
	frtcjmx	35.00	0.8468	0.8477	0.8468	0.8460
	none	16.00	0.8061	0.8114	0.8061	0.8068
resnet18	arpco	44.00	0.8079	0.8109	0.8079	0.8078
	frccm	110.0	0.8109	0.8146	0.8109	0.8109
	frcgb	61.00	0.8004	0.8044	0.8004	0.8007
	frtcj	31.00	0.8142	0.8161	0.8142	0.8139
	frtcjmx	38.00	0.8169	0.8184	0.8169	0.8160
	none	22.00	0.7762	0.7799	0.7762	0.7764
swin	arpco	17.00	0.9225	0.9248	0.9225	0.9222
	frccm	28.00	0.9307	0.9317	0.9307	0.9305
	frcgb	18.00	0.9243	0.9257	0.9243	0.9243
	frtcj	13.00	0.9164	0.9178	0.9164	0.9164
	frtcjmx	16.00	0.9240	0.9250	0.9240	0.9239
	none	13.00	0.9180	0.9202	0.9180	0.9179
vit	arpco	30.00	0.9168	0.9195	0.9168	0.9166
	frccm	40.00	0.9229	0.9248	0.9229	0.9232
	frcgb	25.00	0.9228	0.9241	0.9228	0.9229
	frtcj	16.00	0.9174	0.9195	0.9174	0.9175
	frtcjmx	31.00	0.9233	0.9243	0.9233	0.9234
	none	14.00	0.9058	0.9095	0.9058	0.9062

81.42% accuracy, and adding Mixup [7] on top of that (`frtcjmx`) further boosted accuracy to 81.69%, which is the highest among the combined tests for ResNet–18 [3]. This represents a $+4.07\%$ increase over the baseline and even slightly surpasses the best single-augmentation result of 81.16% (from TrivialAugment [16] in Section 4.2).

The flip+crop+CutMix [8] combination (`frccm`) also performed well at 81.09%, nearly on par with `frtcj` and `frtcjmx`. In general, all multi-step augmentations yielded higher precision, recall, and F1 than the baseline, with metrics in the $\sim 81\%$ range, indicating improved classification across the board. Notably, combining multiple simple transforms (flip, rotate, color jitter) produced better results than any of those individual transforms alone—for example, `frtcj` (81.42%) outperformed flip-only ($\sim$ 79.8%) or rotation-only ($\sim$ 79.5%) by about $+1.5$–2%. This suggests a complementary effect where geometric and color perturbations together provide a more robust training signal. Incorporating a Mixup [7] strategy on top of those (`frtcjmx`) gave a modest additional lift, implying that mixing-based augmentation can stack with traditional transforms to further improve CNN performance.

From an early-stopping perspective, the combined policies strongly regularized ResNet–18 [3]'s training. The most aggressive pipeline (`frccm`) did not trigger early stopping until epoch 110, compared to epoch 22 with no augmentation. Even the other multi-augmentation runs continued for 30–60+ epochs, all far longer than the baseline. This extended training indicates that stacking augmentations delayed overfitting significantly, allowing ResNet–18 [3] to epoch through much more data before convergence and resulting in a higher final model quality.

4.5.2 ResNet-50

A similar pattern is observed with the deeper ResNet–50 [3]. Its no augmentation baseline accuracy was 80.61%. All combined augmentation strategies markedly improved this baseline. The four step flip+rotate+color-jitter+Mixup [7] pipeline (`frtcjmx`) achieved 84.68% accuracy (Precision 84.77%, Recall 84.68%, F_1 84.60%), representing a $+4.1\%$ gain over the baseline and nearly $+0.9\%$ higher than ResNet–50 [3]'s previous best single augmentation (CutMix at 83.80%).

The three step flip+crop+CutMix [8] pipeline (`frccm`) was a close second at 84.45%, effectively matching the Mixup [7] based pipeline. Other combinations such as `arpco` (affine+perspective+Cutout [12]) and `frtcj` (flip+rotate+jitter) reached $\sim$ 83.6–83.8%, still a solid improvement of $\sim$ 3% above baseline. Overall, including a mixing strategy (CutMix [8] or Mixup [7]) yielded the highest gains for ResNet–50 [3], aligning with prior observations that sample composition augmentations are especially powerful on complex datasets. Notably, even the purely geometric/photometric pipeline `frcgb` (flip+crop+blur) achieved 83.26%, underscoring that combining simple transforms can collectively mimic some benefits of more advanced methods.

Precision, recall, and F_1 for ResNet–50 [3] closely tracked the accuracy improvements, all rising into the mid 84% range for the top pipelines (versus $\sim$ 81% with no augmentation). Training dynamics also reflected stronger regularization: with the CutMix [8] combo (`frccm`), ResNet–50 [3] trained until epoch 51 before early stopping, compared to only 16 epochs without augmentation. This more than tripling of training duration indicates that the model continued to learn new variations much longer, translating into better generalization.

4.5.3 Vision Transformer (ViT)

The ViT [4] model already started with a high baseline accuracy of 90.58% on CIFAR–100 [9], thanks to its capacity and pretraining. Using combined augmentations, ViT [4] did improve further, but the gains were relatively small—indicative of diminishing returns on an already well generalizing model. The top performing pipeline for ViT [4] was again `frtcjmx` (flip+rotate+jitter+Mixup [7]) at 92.33% accuracy, with the CutMix [8] based pipeline `frccm` extremely close behind at 92.29%. These results are effectively tied and represent about a $+1.7$–1.8% boost over ViT's baseline, which is on par with the best single augmentation ViT [4] achieved (TrivialAugment [16] yielded $\sim$ 92.3%). In other words, the best combined strategy for ViT [4] barely edged out its best single augmentation. Other multi-step combos (e.g., `frcgb` and `arpco`) yielded $\sim$ 92.1–92.2%, still an improvement over no augmentation (and over most weaker single augmentations such as flips or blur), but all within a narrow band. This suggests that once ViT [4] has benefited from one strong augmentation, additional transforms contribute only marginal extra variability.

All combined pipelines kept ViT [4]'s precision, recall, and F_1 in sync with accuracy (e.g., $F_1 \approx 92.3$ for the best pipeline, vs. 90.62 baseline F_1), indicating consistent performance gains without skewing the precision/recall balance.

ViT [4]'s early stopping epochs did increase under heavier augmentation—for instance, `frccm` ran to 40 epochs vs. 14 epochs at baseline—but the final accuracy plateaued near the same level as earlier single augmentation results. This plateau implies that multi step augmentation offers limited added value for ViT [4] once high performance is already achieved, unlike for the CNNs, which saw more notable jumps.

4.5.4 Swin Transformer

The Swin [5] Transformer exhibited trends akin to ViT [4]. Its baseline accuracy was 91.80%. The best combined augmentation result came from the flip+crop+CutMix [8] pipeline (`frccm`) at 93.07%. This is essentially identical to Swin [5]'s best single augmentation result (CutMix [8] alone yielded 93.04% in Section 4.2). Other pipelines, including `frtcjmx` (92.40%) and `frcgb` (92.43%), also landed around 92.4–92.5%, representing only a $\sim 0.6\%$ gain over baseline. Interestingly, the simple `frtcj` pipeline (with no Mixup [7] or CutMix [8]) gave Swin [5] 91.64%—slightly below the no augmentation accuracy. Although this -0.16% difference is negligible, it highlights that for a high-performing transformer, adding mild geometric/color jitter alone may not help and can even introduce insignificant degradation (potentially because Swin [5] already learns those invariances, or because such transforms may disrupt some learned positional features). In contrast, the pipelines that included mixing augmentations (Mixup [7] or CutMix [8]) did provide small positive bumps for Swin [5], consistent with the idea that creating new synthetic samples by mixing images can expose even a strong model to harder training examples.

Swin [5]'s precision and recall followed suit, with the best pipeline reaching $\sim 93.17\%$ precision / 93.05% recall, versus $\sim 92\%$ with weaker pipelines and 92.02% precision at baseline. The early stopping behavior for Swin [5] changed only modestly with combined augmentation: most pipelines led to stopping around 15–18 epochs, similar to the baseline 13 epochs, except the CutMix [8] combo (`frccm`) which extended training to 28 epochs. This smaller increase (compared to CNNs) reinforces that the transformer was less prone to overfitting from the start, so extra augmentation had a less dramatic regularization effect.

Overall,the combined augmentation strategies provided notable benefits for the CNN models on CIFAR–100 [9], while offering diminishing returns for the Transformer models. Every multi-step pipeline outperformed no augmentation by a wide margin in accuracy (e.g., $+3$ to $+5$ percentage points for ResNets [3], $+1$ to $+1.5$ for ViT/Swin [5]) and also improved the precision, recall, and F_1 scores accordingly. Compared with using a single augmentation, stacking multiple transforms yielded marginal yet consistent gains for ResNet–18/50 [3]—indicating that these models can exploit the additional diversity introduced by combined augmentations to generalize better. In practical terms, a combination of simple geometric and photometric tweaks can approximate or exceed the impact of more complex policies for CNNs.

For the more robust ViT [4] and Swin [5] Transformers, the best single augmentation already captured most of the possible benefit; adding more augmentations on top only inched performance slightly further, if at all. This suggests diminishing returns in the multi augmentation regime for models that are either pretrained or inherently resistant to overfitting. It is also noteworthy that the most effective pipelines across all models included a mixup [7] based component. In particular, the CutMix [8] based pipeline (`frccm`) was among the top two for every model—it achieved the highest accuracy on Swin [5] and was within 0.1–0.3% of the best on ResNet–18 [3], ResNet–50 [3], and ViT [4]. Likewise, the Mixup [7] enhanced pipeline (`frtcjmx`) edged out others on the CNNs and ViT. These results reinforce the dominance of mixed sample augmentations in boosting performance on complex datasets, echoing prior findings that CutMix [8] and similar techniques are especially potent for CNNs.

At the same time, the fact that `frtcjmx` slightly outperformed `frccm` for three of the four models suggests value in combining Mixup [7] with traditional transforms—possibly because Mixup [7] (which blends images at the pixel level) and CutMix [8] (which patches images) have different effects, and here Mixup [7] added variability without sacrificing the information provided by flips, rotations, and color changes.

In summary, experimenting with these combined augmentation pipelines on CIFAR–100 [9] shows that stacking multiple realistic augmentations can further improve model generalization—particularly for convolutional networks—by providing a broader range of augmented samples and delaying overfitting. The improvements over single augmentations are modest but consistent for CNNs (on the order of $+0.5$–1% in accuracy beyond the best single augmentation), indicating some added value from multi-step augmentation. For the Transformer based models, the benefit plateaus, implying they were already near saturation with one strong augmentation (hence additional transforms yield diminishing returns). Crucially, the flip+crop+CutMix [8] and flip+rotate+jitter+Mixup [7] pipelines stood out as the most effective, validating that CutMix [8] based strategies (and Mixup [7] variants) perform best overall in a combined augmentation setting on this challenging dataset.

5 EVALUATION

In evaluating model performance on CIFAR-10 [9], CIFAR-100 [9], MNIST [10], and Fashion-MNIST [11], several key observations emerge based on the test accuracy, precision, recall, and F1-score results for each model and augmentation strategy.

5.1 Impact of Augmentation vs. Dataset Complexity

The benefit of data augmentation was most pronounced on more complex or challenging tasks, and less so on simpler ones. On the hardest dataset (CIFAR-100 [9] with 100 classes), augmentation yielded significant gains in test accuracy and F1-score, especially for the CNN models (several percentage points improvement over the no-augmentation baseline). Fashion-MNIST [11] (grayscale clothing images with 10 classes) also saw notable performance boosts from augmentation, though somewhat smaller. In contrast, on easier or already well-solved tasks like CIFAR-10 (10 classes) and MNIST [10] (handwritten digits), most augmentations produced minimal to no improvement in accuracy or F1, since baseline performance was already very high. In the case of MNIST, certain transforms even harmed performance, underscoring that when the base accuracy is near ceiling, aggressive augmentations that distort the data's inherent structure can be detrimental. These results suggest that augmentation is most needed and effective when the model is at risk of overfitting or the dataset has high complexity/intra-class variability.

5.2 CNN vs. Transformer Performance

Convolutional neural networks (CNNs) benefited more from augmentation than Transformer-based classifiers in relative terms. The ResNet-18 [3] and ResNet-50 [3] (CNN models) generally saw larger increases in accuracy and F1-score from augmented training. For example, on CIFAR-100 [9] the ResNet models gained up to a few percentage points in accuracy with certain augmentations, whereas the Vision Transformer (ViT [4]) and Swin [5] Transformer typically saw more modest improvements (on the order of $\sim 1\%$ or less on the same dataset). This trend was consistent on simpler datasets as well: CNNs accrued small but non-trivial boosts from augmentation on CIFAR-10 and Fashion-MNIST [11], while the Transformers, which started with higher baseline accuracy, improved only marginally on those tasks. The higher capacity and pretraining of the Transformers likely gave them an initial advantage (e.g. ViT [4] and Swin [5] achieved near-state-of-the-art accuracy even without augmentation), leaving less room for augmentation to help. In summary, data augmentation helped narrow the performance gap between the simpler CNNs and the more powerful Transformers by elevating CNN accuracy closer to Transformer levels. Quantitatively, while Transformers still attained higher absolute accuracy and precision/recall across all datasets, the relative gain from augmentation was clearly larger for CNN architectures.

5.3 Top-Performing Augmentation Techniques

The experiments revealed that certain augmentation methods consistently led to the highest test performance across different models and datasets. In particular, mixing-based augmentations proved extremely effective. Techniques like Mixup [7] and CutMix [8] yielded some of the largest improvements in accuracy and F1-score, especially on the complex datasets (CIFAR-100 [9] and Fashion-MNIST [11]). For instance, using CutMix [8] produced the best result for several model–dataset combinations (it was among the top improvements for ResNet [9] on CIFAR-100 [9] and for Swin [5] on Fashion-MNIST [11]), while Mixup [7] also frequently ranked near the top. These methods work by creating novel hybrid training samples (e.g. combining images or image patches with mixed labels), a form of regularization that encourages the models to learn more robust features. Additionally, automated augmentation policies – namely AutoAugment [14], RandAugment [15], and TrivialAugment [16] – performed consistently well without requiring manual tuning. Notably, TrivialAugment [16] often matched or exceeded the performance of more complex augmentation strategies: it achieved top-tier accuracy for ResNet-18 [3] on CIFAR-10 [9]/CIFAR-100 [9] and for ViT [4] on CIFAR-100 [9] in our results, despite its simplicity. Traditional augmentations (random horizontal flips, rotations, affine transformations, color jitter, blur, etc.) still provided baseline benefits and are confirmed to be useful components of the training pipeline – for example, horizontal flips significantly improved accuracy on natural image datasets like CIFAR-10 [9]. However, these basic methods on their own were generally less powerful than the advanced techniques mentioned above. Overall, the best results for both CNN and Transformer models were obtained with either the mixing-based augmentations or the automated, composite augmentation policies, indicating that diverse and information-rich transformations contribute most to improving model generalization.

5.4 Effect of combined pipelines on CIFAR-100 [9]

The five multi-step recipes boosted ResNet–18 [3] and ResNet–50 [3] by $\approx$ 0.8–1.1 pp beyond their best single augmentation scores, with the CutMix [8] centric pipeline (`frccm`) and the Mixup [7] enhanced pipeline (`frtcjmx`) performing best (e.g., ResNet–50: 84.68% vs. 83.80% with CutMix [8] alone). ViT [4] and Swin [5] improved only $\approx$ 0.3–0.5 pp, indicating diminishing returns once a strong single augmentation is in place. Early stopping patience more than doubled for ResNet–18/50 [3] under the most aggressive recipes (e.g., ResNet–18 [3] stopped at epoch 110 with `frccm` versus 22 with no augmentation), confirming stronger regularisation.

5.5 Data Appropriateness and Augmentation Pitfalls

The evaluation highlighted the importance of aligning augmentation strategies with the data domain. An augmentation that produces unrealistic or label-altering effects can undermine performance, as seen with certain cases. For example, applying horizontal flips to MNIST [10] digits introduced invalid samples (creating symbols that are no longer the original digit), which led to drops in precision and recall for those classes. Similarly, using an aggressive random crop on images where the subject occupies most of the frame (such as Fashion-MNIST [11]) sometimes removed critical parts of the object, hurting the classifier's ability to recognize the item. These negative outcomes were relatively rare and usually small in magnitude, but they serve as a caution. In contrast, augmentations that add plausible variation without altering the image's class label – such as slight rotations or shifts for digits, or color jitter for natural images – tended to either help or at least not hurt performance. The key insight is that domain knowledge should guide augmentation choices: one should avoid transformations that create out-of-distribution examples or change the semantics of the inputs. When augmentations were appropriate to the data (e.g. flipping animals or objects in CIFAR [9], which does not change their identity, or mixing images in ways that still produce valid hybrid classes), the models benefited by learning to handle that variability. Thus, addressing the research question isn't just about whether augmentation helps, but also which augmentations help for a given dataset – the key findings stress that the effectiveness of an augmentation method is context-dependent on the data characteristics.

5.6 Regularization and Early Stopping Effects

One clear pattern supporting the role of augmentation as a regularizer was observed through training dynamics and early stopping behavior. Models trained with heavier augmentation tended to continue improving for more epochs before the early stopping criterion was met, compared to models trained on unaugmented or lightly augmented data. In practice, with augmentation the validation loss kept decreasing for a longer period, indicating that the model was not overfitting as quickly. This allowed training to run for more epochs and reach a lower final validation loss (and higher final test accuracy) than the non-augmented case. Early stopping was triggered later for augmented models, which is evidence that augmentation expanded the effective training set and made the learning process more robust. For example, on CIFAR-100 [9], runs with strong augmentations like CutMix [8] or RandAugment [15] often converged several epochs later than the baseline, coinciding with higher achieved accuracy and F1 on the test set. In contrast, on MNIST [10] (where augmentations provided little new information), many models stopped very early, reflecting how quickly they converged to the near-optimal solution. Overall, the use of early stopping in these experiments confirmed that augmentation improves generalization: it yielded models that not only attained better test metrics but did so without overfitting. The combination of data augmentation and early stopping allowed the models to maximize performance safely – training continued as long as the model was learning useful variations from augmented data, and stopped automatically once those gains were exhausted.

In summary, the augmentation methods did address the research question by demonstrably improving model performance and generalization across a variety of datasets and architectures. The extent of improvement varied – greater on complex datasets and for CNNs, smaller on simple tasks and for Transformers – but overall the results validate that applying appropriate data augmentation can enhance both CNN and Transformer-based image classifiers. The evidence from test accuracy, precision/recall, F1-scores, and training behavior all support the conclusion that augmentation strategies are effective in boosting classification performance under the right conditions.

6 Conclusion

This comparative study evaluated a broad range of data augmentation techniques on two CNN architectures (ResNet-18 [3] and ResNet-50 [3]) and two Transformer-based vision models (Vision Transformer and Swin [5] Transformer) across four image classification datasets of varying complexity (MNIST [10], Fashion-MNIST [11], CIFAR-10 [9], CIFAR-100 [9]). Overall, the results indicate that appropriate data augmentation is a highly effective tool for improving model generalization and performance, though its impact varies with dataset difficulty and model type. Augmentation provided the most significant gains on the more complex tasks and for the less inherently flexible models: for instance, models trained on the 100-class CIFAR-100 [9] dataset saw substantial improvements with advanced augmentations, and the CNN classifiers derived greater relative benefit from augmentation than their Transformer counterparts. On simpler tasks like MNIST [10] , where baseline accuracy was already near ceiling, most augmentations had negligible effect (and could even degrade performance if they violated the nature of the data), whereas intermediate scenarios such as CIFAR-10 [9] and Fashion-MNIST [11] showed modest but consistent benefits. The Transformer-based models (ViT [4] and Swin [5]) achieved higher absolute accuracy than the CNNs under all conditions, reflecting their greater capacity and pretraining, but data augmentation still conferred modest improvements to these Transformers as well. Notably, the added regularization from augmentations helped narrow the performance gap between CNNs and Transformers in some cases by lifting the CNNs closer to the Transformer's level of accuracy.

With regard to augmentation strategies, the evidence draws a clear hierarchy. Modern single step composite methods—such as Mixup [7], CutMix [8], and automated policies (AutoAugment [14], RandAugment [15], TrivialAugment [16])—deliver the largest individual gains, supplying diverse and information rich training samples that outstrip the incremental improvements offered by basic flips, rotations, scaling, or colour jitter.

A supplementary test of five multi step "recipes" on CIFAR–100 [9] shows that further stacking traditional geometric or photometric transforms with a sample mixing component can add a smaller but still meaningful boost: convolutional networks gained roughly 0.8–1.1 percentage points beyond their best single augmentation, whereas Vision Transformers realised only marginal (< 0.5 pp) benefit. The most effective pipeline combined flip + random resized crop + CutMix [8], underscoring the continued dominance of mixed sample augmentations even in a compound setting.In practical terms, the results suggest applying a strong composite augmentation first and—especially for CNNs tackling complex, low data regimes—layering inexpensive classic transforms thereafter, while recognising that returns diminish once a high capacity Transformer already reaches near saturation.

In conclusion, data augmentation remains a powerful and indispensable technique in the deep learning toolbox for image classification. When applied judiciously, it can significantly boost the performance of both conventional CNNs and cutting-edge Vision Transformers, especially in scenarios with limited data or high complexity. The findings of this study provide empirical guidance on which augmentation methods are most effective for different model architectures and dataset types. Practitioners training CNN models are encouraged to employ rich augmentation pipelines (potentially leveraging mixing or automated augmentations) to close the gap with more complex models, while even Transformer models can benefit from a well-chosen augmentation strategy to further improve robustness. The overarching conclusion is that appropriate data augmentation enhances model generalization across the board – yielding more accurate and more robust image classifiers – and it should be considered a key component of model training strategy for vision tasks.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, pp. 84–90, 05 2012.

[2] M. Xu, S. Yoon, A. Fuentes, and D. S. Park, “A comprehensive survey of image augmentation techniques for deep learning,” *Pattern Recognition*, vol. 137, p. 109347, 05 2023.

[3] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 06 2016. [Online]. Available: https://ieeexplore.ieee.org/document/7780459

[4] “Dosovitskiy, a. (2021) an image is worth 16 16 words transformers for image recognition at scale. proceedings of the ieee/cvf conference on computer vision and pattern recognition, new york city, 23-26 june 2021, 45-67. - references - scientific research publishing,” Scirp.org, 2021. [Online]. Available: https://www.scirp.org/reference/referencespapers?referenceid=3709102

[5] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 10 2021.

[6] B. J. Kim and S. W. Kim, “Configuring data augmentations to reduce variance shift in positional embedding of vision transformers,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, pp. 17 840–17 849, 04 2025.

[7] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” *arXiv:1710.09412 [cs, stat]*, 04 2018. [Online]. Available: https://arxiv.org/abs/1710.09412

[8] S. Yun, D. Han, S. Oh, S. Chun, J. Choe, and Y. Yoo, “Cutmix: Regularization strategy to train strong classifiers with localizable features.” [Online]. Available: https://openaccess.thecvf.com/content$_{ICCV_2}019/papers/Yun_{C}utMix_{R}egularization_{S}trategy_{t}o_{T}rain_{S}trong_{C}lassifiers_{W}ith_{L}ocalizable_{F}eatures$

[9] A. Krizhevsky, “Learning multiple layers of features from tiny images,” 04 2009. [Online]. Available: https://www.cs.toronto.edu/ kriz/learning-features-2009-TR.pdf

[10] Y. LeCun, “Mnist handwritten digit database, yann lecun, corinna cortes and chris burges,” Lecun.com, 2009. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[11] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms,” *arXiv:1708.07747 [cs, stat]*, 09 2017. [Online]. Available: https://arxiv.org/abs/1708.07747

[12] T. DeVries and G. W. Taylor, “Improved regularization of convolutional neural networks with cutout,” *arXiv:1708.04552 [cs]*, 11 2017. [Online]. Available: https://arxiv.org/abs/1708.04552

[13] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, “Random erasing data augmentation,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, p. 13001–13008, 04 2020. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/7000

[14] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, “Autoaugment: Learning augmentation policies from data,” *arXiv:1805.09501 [cs, stat]*, 04 2019. [Online]. Available: https://arxiv.org/abs/1805.09501

[15] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, “Randaugment: Practical automated data augmentation with a reduced search space,” *arXiv:1909.13719 [cs]*, 11 2019. [Online]. Available: https://arxiv.org/abs/1909.13719

[16] S. G. Müller and F. Hutter, “Trivialaugment: Tuning-free yet state-of-the-art data augmentation,” arXiv.org, 2021. [Online]. Available: https://arxiv.org/abs/2103.10158

[17] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, “Augmix: A simple data processing method to improve robustness and uncertainty,” *arXiv:1912.02781 [cs, stat]*, 02 2020. [Online]. Available: https://arxiv.org/abs/1912.02781

[18] A. Gupta, “Revisiting the unreasonable effectiveness of data,” Research.google, 2017. [Online]. Available: https://research.google/blog/revisiting-the-unreasonable-effectiveness-of-data/

[19] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, “Training data-efficient image transformers distillation through attention,” *arXiv:2012.12877 [cs]*, 01 2021. [Online]. Available: https://arxiv.org/abs/2012.12877