BEYOND CROPPING AND ROTATION: AUTOMATED EVOLUTION OF POWERFUL TASK-SPECIFIC AUGMENTATIONS WITH GENERATIVE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Data augmentation has long been a cornerstone for reducing overfitting in vision models, with methods like AutoAugment automating the design of task-specific augmentations. Recent advances in generative models, such as conditional diffusion and few-shot NeRFs, offer a new paradigm for data augmentation by synthesizing data with significantly greater diversity and realism. However, unlike traditional augmentations like cropping or rotation, these methods introduce substantial changes that enhance robustness but also risk degrading performance if the augmentations are poorly matched to the task. In this work, we present EvoAug, an automated augmentation learning pipeline, which leverages these generative models alongside an efficient evolutionary algorithm to learn optimal task-specific augmentations. Our pipeline introduces a novel approach to image augmentation that learns stochastic augmentation trees that hierarchically compose augmentations, enabling more structured and adaptive transformations. We demonstrate strong performance across fine-grained classification and few-shot learning tasks. Notably, our pipeline discovers augmentations that align with domain knowledge, even in low-data settings. These results highlight the potential of learned generative augmentations, unlocking new possibilities for robust model training.

1 INTRODUCTION

Generative AI has rapidly advanced across multiple domains. In computer vision, diffusion models now surpass GANs in producing realistic images and videos from simple prompts (Dhariwal & Nichol, 2021). In language, models like GPT generate human-like text and code, achieving high scores on standardized tests (OpenAI et al., 2024). Similar breakthroughs extend to generative audio (Schneider, 2023) and 2D-to-3D shape generation (Karnewar et al., 2023). These advances raise an important question: to what extent can AI-generated content improve AI itself (Yang et al., 2023b)? While far from true self-improvement, generative models are increasingly influencing their own training processes.

A key challenge in leveraging synthetic data is the syn-to-real gap—the discrepancy between generated and real-world data. Poorly matched synthetic augmentations degrade performance rather than enhance it. For example, diffusion models still struggle with fine details such as realistic fingers (Narasimhaswamy et al., 2024). Thus, a model trained on data augmented by flawed synthetic images may reinforce errors. Similarly, a language model could amplify its own biases by training on text that it generated itself. This issue is particularly critical in tasks requiring fine-grained distinctions, such as image classification, or in low-data settings like few-shot learning. Addressing this gap is essential for generative augmentations to contribute meaningfully to AI training.

Hence, methods that use synthetic or simulated data must balance the tradeoff between data variability and fidelity. This can be achieved by constraining data generation to closely match the real-world distribution, thereby reducing its variability while improving its fidelity. This approach has been successful in fields like robotics (Lu et al., 2024) and autonomous vehicles (Song et al., 2024). However, it has only seen limited application in synthetic image generation for computer vision. This work tackles the challenge of fine-grained few-shot classification. Due to the lack of real samples, synthetic data provides an attractive option for boosting performance. Since fine-grained distinctions between classes can be easily missed, a carefully designed image generation pipeline is required.

We propose using generative AI not for data creation, but for data augmentation—a paradigm shift. Instead of generating data from scratch, we condition the process on real data, thereby ensuring that it preserves the semantic priors and underlying structure of the original distribution while introducing meaningful and novel variations. While this approach constrains synthetic data to resemble real data, it also provides stronger guarantees of its validity, effectively overcoming the syn-to-real gap.

Motivated by this vision, we design EvoAug, a pipeline that automatically learns a powerful augmentation strategy. Our work makes use of evolutionary algorithms, which have been shown to work in a variety of domains and still remain more sample-efficient and straightforward than other methods (Ho et al., 2019; Wang et al., 2023). This is especially important when dealing with complex augmentation operators like conditional diffusion and NeRF models, where evaluation is expensive, gradients are very difficult to approximate, and sample efficiency is paramount.

As part of our pipeline, we construct an augmentation tree—a binary tree that applies a series of augmentation operators in accordance with learned branching probabilities. The augmentation tree can then be used to produce synthetic or augmented variations of the images in the dataset by stochastically following root-to-leaf paths. Our trees include nodes that perform either classical or generative augmentations. To produce accurate synthetic data, we condition the diffusion models on existing structural and appearance-based information rather than solely relying on prompt-based image generation. Our approach is powerful enough to work even with very small datasets and provides promising results on fine-grained and few-shot classification tasks across multiple datasets.

Our main contributions are the following:

1. The first automated augmentation strategy to leverage both modern augmentation operators like controlled diffusion and NeRFs, along with traditional augmentation operators like cropping and rotation
2. Strong results on fine-grained few-shot learning, a challenging domain where prior work has failed to preserve the minor semantic details that distinguish the classes
3. Novel unsupervised strategies that scale as low as the one-shot setting, where no supervision to evaluate augmentations is available
4. Constructing an augmentation pipeline from only open-source, pre-trained diffusion models, without requiring domain-specific fine-tuning

2 Related Work

Data augmentation reduces model overfitting by applying image transformations that preserve the original semantics while introducing controlled diversity into the training set. Traditional augmentations include rotations, random cropping, mirroring, scaling, and other basic transformations. These straightforward techniques remain fundamental in state-of-the-art image augmentation pipelines. More advanced methods—such as erasing (Zhong et al., 2020; Chen et al., 2020; Li et al., 2020; DeVries & Taylor, 2017), copy-pasting (Ghiasi et al., 2021), image mixing (Zhang et al., 2017; Yun et al., 2019), and data-driven augmentations like AutoAugment (Cubuk et al., 2018) and its simplified variant RandAugment (Cubuk et al., 2020)—have expanded the augmentation toolbox.

Another approach involves generating synthetic data using generative models (Figueira & Vaz, 2022). Early work explored GANs (Besnier et al., 2020; Jahanian et al., 2021; Brock et al., 2018), VAEs (Razavi et al., 2019), and CLIP (Ramesh et al., 2022), achieving strong results (Engelsma et al., 2022; Skandarani et al., 2023). Recently, diffusion models, particularly for text-to-image synthesis, have surpassed GANs in producing photorealistic images (Nichol et al., 2021; Ramesh et al., 2022; Saharia et al., 2022b; Yang et al., 2025). Trained on large-scale internet data (Schuhmann et al., 2022), diffusion models have been used for augmentation (Azizi et al., 2023; Sarıyıldız et al., 2023; He et al., 2022; Shipard et al., 2023; Rombach et al., 2022; Islam et al., 2025; 2024), often relying on class names or simple class agnostics prompts to guide generation. Despite promising initial results, synthetic data remains inferior to real data, highlighting the persistent domain gap between the two (Yamaguchi & Fukuda, 2023).

To address this gap, recent approaches have incorporated conditioning the generative process on real data. Some popular methods involve projecting the original images to the diffusion latent space (Zhou et al., 2023), fine-tuning diffusion models on real data (Azizi et al., 2023), leveraging multi-modal LLMs to obtain detailed, custom image captions for high-quality text prompting(Yu et al., 2023), and employing image-to-image diffusion models that enable direct conditioning on a specific image (Saharia et al., 2022a; Meng et al., 2021; Zhang et al., 2023; He et al., 2022; Trabucco et al., 2025). Controlled diffusion, a subset of these methods, introduces a more powerful paradigm, furthering the efficient use of both text and image priors (Fang et al., 2024; Islam & Akhtar, 2025) with applications in segmentation (Trabucco et al., 2023) and classification (Goldfeder et al., 2024) problems.

Given such a wide range of augmentation operators, an important problem is knowing which augmentations to use for a specific task, without the use of domain knowledge. This task, of automatically learning augmentation policies, falls under the class of meta learning and bi-level optimization problems, where we seek to learn a component of the learning algorithm itself (Hospedales et al., 2021). These algorithms generally fall under one of the following categories: gradient-based optimization, RL-based optimization, Bayesian optimization, and evolution-based optimization.

In the context of learning augmentation policies, all these methods have seen success (Yang et al., 2023a). Differentiable methods often train a neural network to produce augmentations (Lemley et al., 2017), sometimes in a generative adversarial setup (Shrivastava et al., 2017; Tran et al., 2017). By far the most notable method, AutoAugment (Cubuk et al., 2018), employs reinforcement learning. While RL is traditionally sample inefficient, improvements upon vanilla RL strategies have leveraged Bayesian methods (Lim et al., 2019), evolutionary strategies (Ho et al., 2019; Wang et al., 2023), or approximate gradient estimation for first-order optimization (Hataya et al., 2020).

Learning augmentation policies is especially challenging in low data settings, as full data policies are usually not transferable to the few-shot case. Various approaches have been considered, including proposing K-fold validation as a method of retaining the data while still performing validation (Naghizadeh et al., 2021). However, this method does not scale to one-shot settings. Utilizing clustering as a label-efficient evaluation method, where augmentations are designed to stay within their corresponding class cluster, can address this limitation (Abavisani et al., 2020).

3 Methods

3.1 Augmentation Operators

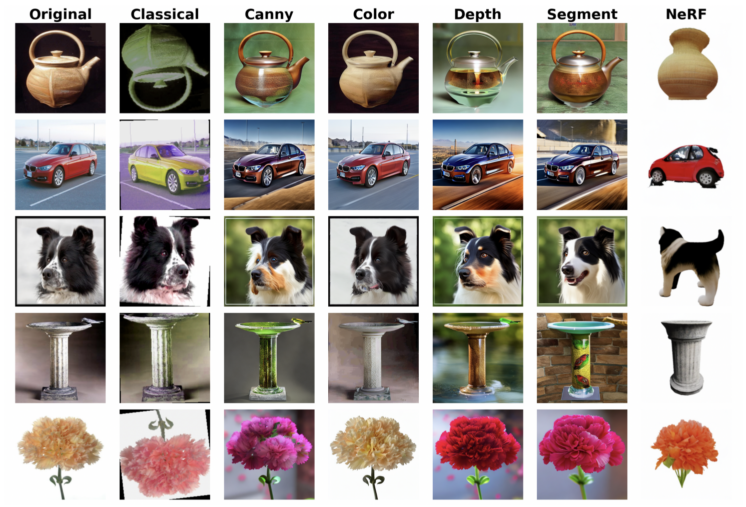

Figure 1: Example image augmentations using our pipeline. Classical augmentations include color jitter, rotation, and random cropping. Canny, color, depth, and segment use existing image information to steer a ControlNet diffusion model. NeRF uses a zero-shot NeRF to perform a 3D rotation.

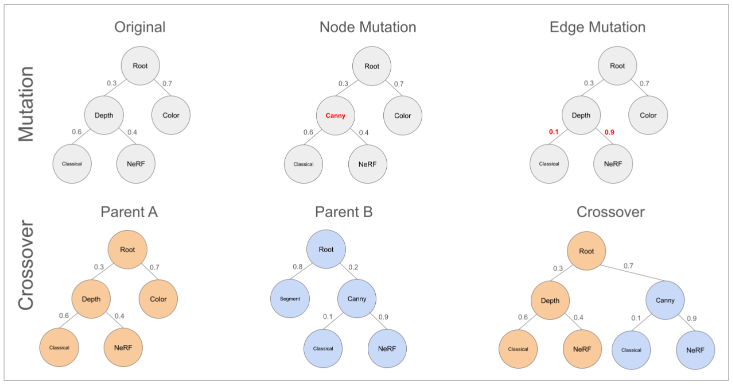

Figure 2: Mutation and Crossover for Augmentation Trees

The generative augmentation operators are based on both diffusion and NeRFs. For diffusion-based operators, we use ControlNet (Zhang et al., 2023), an architecture which allows rapid customization of diffusion models without fine-tuning. To condition the model, we extract edges using Canny edge detection (Canny, 1986), segmentations using Segment Anything (Kirillov et al., 2023), depth maps using MiDaS (Ranftl et al., 2020), and color palettes by simply downsampling the image. This gives four diffusion-based augmentation operators, termed ”Canny”, ”Segment”, ”Depth” and ”Color”. We use Zero123 (Liu et al., 2023b) for NeRF-based augmentation. This model creates a 3D reconstruction of an image from a single shot, allowing for 3D rotation. We then rotate 15 degrees left or right when performing an augmentation using this model. We term this operator ”NeRF”. Next, we include another augmentation operator, termed ”Classical.” This includes the full set of traditional augmentations: random crop, translation, scale, rotation, color jitter, and flip. This operator allows the evolution process to decide whether to include and build on the traditional classical augmentation pipeline or exclude it. Sometimes, all augmentations can be harmful, so we also included a ”NoOp” operator that simply duplicates the existing image. Figure 1 gives examples of these operators.

3.2 Evolutionary Strategy

For our augmentation policy learning pipeline, we choose an evolutionary approach. This choice is motivated by practical considerations: diffusion and NeRF based augmentation is considerably more expensive to evaluate than traditional augmentations, so pipeline efficiency is crucial. Population-based evolutionary strategies have been shown to be as effective as RL approaches, with less than one percent of the computational effort (Ho et al., 2019). While gradient approximation methods have been shown to be even more efficient in some cases(Hataya et al., 2020), those results are for approximating gradients of simpler transformations, and do not translate to our pipeline, which can handle arbitrary generative modules. Further, recent work has shown evolution to be effective for searching for augmentation polices even in very complex augmentation spaces (Wang et al., 2023).

We define an augmentation tree as a binary tree, where each node represents an augmentation operator. The edges of our tree represent transition probabilities to each child node, summing to 1. This structure is chosen as it serves as a common genome for evolutionary algorithms.

Mutation Illustrated in figure 2, mutation can occur at either the node level or the edge level. An edge mutation reassigns the probabilities of a transition between two child nodes. A node mutation switches the augmentation operator of that node (eg. Depth node becomes a Canny node).

Crossover Also illustrated in figure 2, crossover is the other basic evolutionary operator. Two parents are selected, a child is created by splicing the branches of the parents together.

We thus define a population P of size n, of initial trees. In each generation, we use mutation and crossover to generate c children P_{new}, that are appended to P. Finally, the population is evaluated with a fitness function f, and the top n are kept for the next generation. Mutation and crossover probability are parameterized by p_m and p_c respectively. Algorithm 1 describes this process.

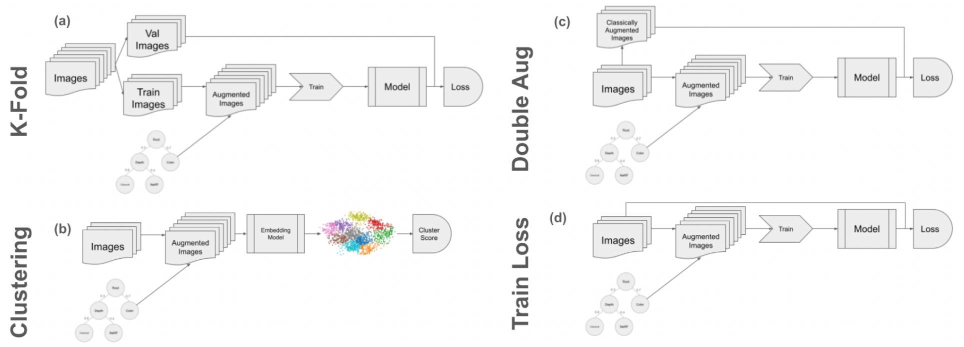

Figure 3: Tree Learning Pipelines. (a) K-Fold applies when there is more than one example per class. (b) We can measure cluster quality for the 1-shot case. (c) We can duplicate the image and assume the problem to be 2-shot instead of 1-shot (d) We can simply use training loss, though it is risky to assume that lower train loss equates to better performance.

3.3 Fitness Functions

The goal of our augmentation strategy is to improve downstream model robustness, and thus the fitness function we choose to evaluate augmentation trees should either directly reflect what we seek to achieve or be a strong proxy. Note that in a full data setting, training data can be split into a train and validation. An augmentation tree can be evaluated by simply training a model with generated augmentations on the training data and measuring performance on the previously unseen evaluation data. We divide our discussion into two, more difficult, settings.

3.3.1 Low Data Setting

In the low-data and few-shot case, the challenge becomes managing the noise of the evaluation function. We can no longer rely on a single train/val split to accurately measure the performance of a tree as low-data settings introduce high variability in splits. Thus, we use K-fold cross-validation.

In addition, directly using accuracy as our metric is no longer appropriate, as our validation set remains small enough that accuracy becomes coarse-grained and unstable. As a result, to align with the convention of higher fitness values corresponding to better candidates in the population, we use the negative validation loss as the fitness function in these settings. Algorithm 2 describes this process. The pipeline can be seen in figure 3a.

Algorithm 1 Evolutionary Search for Augmentation Trees

```
Require: Population size $p$, number of generations $g$, fitness function $f$, number of children $c$,
    mutation probability $p_m$, crossover probability $p_c$
1: $P \leftarrow$ InitializePopulation($p$)
2: for $i = 1$ to $g$ do
3:     $P_{\text{new}} \leftarrow$ MutateAndCrossover($P, c, p_m, p_c$)
4:     $P \leftarrow P \cup P_{\text{new}}$
5:     Evaluate fitness $f(T)$ for each tree $T \in P$
6:     $P \leftarrow$ SelectBest($P, p$)                ▷ Keep top $p$ trees
7: end for
8: $T_{\text{best}} \leftarrow$ BestTree($P$)
9: return $T_{\text{best}}$
```

3.3.2 One-Shot Setting

In the most extreme case, we only have one image per class. Thus, proposed methods involving K-fold validation will not be able to span the full class range of the dataset (Naghizadeh et al., 2021). To address this problem, we devised the following strategies:

Algorithm 2 K-Fold Cross Validation Tree Fitness Function

```
Require: Dataset $D$, augmentation tree $T$, number of folds $k$
 1: Split $D$ into $k$ folds: $D_1, D_2, \ldots, D_k$
 2: Initialize $M \leftarrow 0$
 3: for $i = 1$ to $k$ do
 4:     $D_{\text{val}} \leftarrow D_i$
 5:     $D_{\text{train}} \leftarrow D \setminus D_i$
 6:     $D_{\text{aug}} \leftarrow$ ApplyAugmentationTree$(T, D_{\text{train}})$
 7:     Train model $M_i$ on $D_{\text{aug}}$
 8:     $m_i \leftarrow$ Evaluate$(M_i, D_{\text{val}})$
 9:     $M \leftarrow M + m_i$
10: end for
11: $\bar{m} \leftarrow \frac{M}{k}$
12: return $\bar{m}$
```

Label-Efficient Clustering Our goal is to find augmentations that preserve important class-specific characteristics while still providing novel data. Thus, when evaluating on a validation set is not possible, we can switch to a clustering approach. To find these novel, true-to-class augmentations, our intuition is to search for clusters that are wide, but still distinct from each other. Abavisani et al. proposed using this type of evaluation for augmentation pipelines in low-data and one-shot settings (Abavisani et al., 2020). They adopted Deep Subspace Clustering (Ji et al., 2017) and optimized the Silhouette coefficient as a measure of cluster quality. We improve upon this work in three ways:

1. We simplify the clustering process by using a pre-trained network to generate image embeddings which we then cluster, thus eliminating the need for a Deep Subspace Clustering network and requiring no training.
2. Prior work employed k-means to form clusters (Douzas et al., 2018), adding computational complexity. We simplify this by directly using known class labels as clusters. This allows us to evaluate explicitly whether augmentations form meaningful, class-based clusters rather than merely measuring separability.
3. When evaluating augmentation quality via clustering, traditional metrics like the Silhouette coefficient reward cohesion but do not penalize small or redundant clusters. This can cause the evolutionary algorithm to favor augmentation trees that produce minimal or trivial variations, which lack diversity and generalization potential. To avoid this pitfall, we introduce an additional penalty term based on average cluster radius, balancing cohesion with cluster size and separability. This modified metric thus encourages the formation of clusters that are both cohesive and sufficiently distinct, promoting better generalization. Experiments supporting these conclusions are presented in Appendix A.5.

This process is given in Algorithm 3. The pipeline can be seen in figure 3b.

Double Augmentation This strategy is simple yet effective. We apply classical augmentations—which reliably introduce meaningful variations—to expand the original one-shot dataset. The augmented dataset is then divided into k splits, and the negative validation losses are averaged across splits, as detailed in Algorithm 4 and illustrated in Figure 3c. This approach allows us to increase augmentations while minimizing the risk of degrading dataset quality or relevance through unintended variations introduced by generative models.

Algorithm 4 1-Shot Double Augmentation Fitness Function

```
Require: One-shot dataset $D$, augmentation tree $T$, number of folds $k$
1: $D' \leftarrow \emptyset$
2: for each image $x \in D$ do
3:     $A(x) = \{\text{ClassicAug}(x)_1, \ldots, \text{ClassicAug}(x)_k\}$
4:     $D' \leftarrow D' \cup A(x)$
5: end for
6: return KFoldFitness($D'$, $T$, $k$)                    ▷ Refer to Alg. 2
```

Algorithm 3 1-Shot Clustering Fitness Function

```
Require: Image dataset $D$, augmentation tree $T$, embedding model $E$
 1: $D_{\text{aug}} \leftarrow$ ApplyAugmentationTree$(T, D)$
 2: Initialize embedding list $L \leftarrow \emptyset$
 3: for each image $x \in D_{\text{aug}}$ do
 4:     $e \leftarrow E(x)$
 5:     Append $e$ to $L$
 6: end for
 7: $C \leftarrow$ Cluster$(L)$
 8: $S \leftarrow$ ComputeSilhouetteScore$(C)$
 9: $d \leftarrow$ ComputeMeanClusterDistance$(C)$
10: $s \leftarrow \alpha S - \frac{1-\alpha}{d}$
11: return $s$
```

Training Loss We can also simply use training loss as a proxy in the one-shot case. We augment all the images, and train a model. We then evaluate trees based on how low the training loss is after a fixed number of epochs. While this should encourage minor augmentations, and also makes use of train loss to estimate eval loss, a very erroneous assumption, it still works well in practice. The pipeline can be seen in figure 3d.

4 Results

4.1 Experiment Setup

We perform our experiments on six datasets: Caltech256 (Griffin et al., 2007), Oxford IIIT-Pets (Parkhi et al., 2012), Oxford 102 Flowers (Nilsback & Zisserman, 2008), Stanford Cars (Krause et al., 2013), Stanford Dogs (Khosla et al., 2011), and Food101 (Bossard et al., 2014). To highlight how powerful our method is, even in few-shot settings when the fine-grained semantic distinctions are minor, we *deliberately searched* for few-shot images and classes that were the most challenging.

For an n-way k-shot classification task, we do this as follows. First, we randomly select n classes from the original dataset. Then we randomly selected k images from each class. We fine-tune a pretrained Resnet50 model (He et al., 2016) on these images and record the accuracy. We repeat this procedure 10 times, gathering 10 different subsets of the classes with different images for each dataset. Afterwards, we note which subset of classes from the dataset had the lowest baseline test accuracy, and we choose this subset as the setting for our augmentation benchmarks.

For our genetic algorithm, we initialize a population of 14. For each of the seven augmentation operators, we initialize two trees whose root nodes use that operator, creating a balanced population. This broadens the solution space exploration and avoids the pitfalls of random initialization on a small population. We set the mutation probability to 10$\%$ and include 6 crossovers per generation. We restrict tree depth to 2, allowing the composition of at most 2 operations per augmentation. For each of the 10 generations, we generate 8 children. In the 2 and 5-shot cases, we use K-fold fitness, choosing folds such that the classes remained balanced. To evaluate augmentation trees, we train the models for 20 epochs and observe the corresponding loss. In the one-shot case, we examine the three other fitness functions (double augmentation, training loss, and clustering) proposed above.

Once the best tree is chosen, we generate augmentations and evaluate the downstream classification accuracy against several baselines:

1. Naive Baseline: We randomly apply classical augmentations (cropping, scaling, translation, horizontal/vertical flipping, color jitter, rotation)
2. RandAugment: We perform a grid search over the number of operations (num_ops) and magnitude parameters, selecting the configuration with the lowest validation loss using cross-validation on a train/validation split; this best-performing configuration is then evaluated on the full test set.
3. AutoAugment: We apply the ImageNet-learned AutoAugment policy to our datasets.

In all downstream classification tasks, training proceeds for 200 epochs. In the 1-shot setting, we augment each image in the original dataset 2 times, in the 2-shot setting 5 times, and in the 5-shot setting 2 times. We also evaluate our methods and baselines against augmentations generated from random trees in the ResNet experiments to ensure that our evolutionary search was an important part of creating true-to-class augmentations. Each experiment is performed at least three times with varying seeds, and the average and standard deviation are reported. We evaluate using a pre-trained ResNet50, ViT-Small (Dosovitskiy et al., 2021), or MobileNetV2 (Sandler et al., 2019) model. Models are fine-tuned using Adam (Kingma & Ba, 2017), with a learning rate of 1e-3. We use NVIDIA GeForce RTX 4090 chips with 24 GB of memory. Each experiment took between 2 and 24 hours to complete, depending on the number of ways and shots.

4.2 Few-Shot Results

Dataset	Model	Naive Baseline	Random Tree	RandAugment	AutoAugment	Learned
	ResNet50	79.78 ± 0.73	81.42 ± 7.64	84.71 ± 0.01	86.78 ± 0.00	**88.28 ± 1.75**
Caltech256	MobileNet	80.95 ± 1.08	-	**86.04 ± 0.01**	84.45 ± 0.01	84.18 ± 0.75
	ViT-Small	73.30 ± 6.61	-	81.85 ± 0.01	**81.74 ± 0.02**	79.83 ± 9.90
	ResNet50	70.49 ± 1.08	78.59 ± 2.11	83.65 ± 0.00	**86.60 ± 0.00**	73.73 ± 3.70
Flowers102	MobileNet	77.00 ± 1.83	-	**79.54 ± 0.01**	81.75 ± 0.01	73.21 ± 4.75
	ViT-Small	97.89 ± 1.43	-	**99.37 ± 0.00**	98.63 ± 0.00	94.30 ± 3.85
	ResNet50	78.44 ± 0.13	83.76 ± 6.42	80.76 ± 0.01	82.23 ± 0.00	**85.15 ± 2.73**
Stanford Dogs	MobileNet	75.34 ± 0.99	-	73.72 ± 0.02	75.26 ± 0.00	**77.51 ± 0.63**
	ViT-Small	83.67 ± 1.38	-	**86.95 ± 0.02**	83.97 ± 0.02	80.61 ± 2.01
	ResNet50	30.90 ± 1.68	36.94 ± 6.48	37.20 ± 0.01	34.68 ± 0.01	**40.40 ± 3.07**
Stanford Cars	MobileNet	35.35 ± 1.05	-	36.30 ± 0.01	36.95 ± 0.01	**37.36 ± 0.15**
	ViT-Small	40.64 ± 5.43	-	43.83 ± 0.01	42.74 ± 0.02	**46.32 ± 1.43**
	ResNet50	86.57 ± 0.60	84.97 ± 3.08	85.25 ± 0.01	86.57 ± 0.00	**88.34 ± 1.72**
Oxford-IIIT Pet	MobileNet	84.41 ± 0.53	-	87.27 ± 0.01	86.08 ± 0.00	**89.21 ± 2.93**
	ViT-Small	88.52 ± 0.55	-	91.28 ± 0.01	90.60 ± 0.01	**91.44 ± 2.06**
	ResNet50	47.82 ± 0.57	42.61 ± 4.56	46.23 ± 0.00	**51.32 ± 0.00**	49.78 ± 4.21
Food101	MobileNet	39.93 ± 1.46	-	**43.49 ± 0.00**	44.82 ± 0.00	42.97 ± 2.61
	ViT-Small	55.66 ± 3.22	-	62.20 ± 0.02	64.14 ± 0.02	**64.18 ± 2.13**

Table 1: 5-way, 2-shot classification accuracy (%) with standard deviation across 6 datasets and 3 downstream image classification architectures. Bolded values indicate the best performance per row.

Dataset	Model	Naive Baseline	Random Tree	RandAugment	AutoAugment	Learned
	ResNet50	88.15 ± 0.25	92.22 ± 0.93	92.55 ± 0.00	**93.31 ± 0.00**	91.48 ± 1.11
Caltech256	MobileNet	88.80 ± 0.19	-	89.51 ± 0.00	**91.41 ± 0.00**	90.05 ± 0.98
	ViT-Small	85.75 ± 1.04	-	90.32 ± 0.01	90.70 ± 0.01	**92.33 ± 1.27**
	ResNet50	82.61 ± 0.96	79.51 ± 2.69	89.04 ± 0.01	**89.92 ± 0.01**	84.95 ± 1.15
Flowers102	MobileNet	88.82 ± 0.69	-	**91.47 ± 0.00**	89.26 ± 0.01	86.60 ± 1.07
	ViT-Small	99.78 ± 0.19	-	99.89 ± 0.00	**99.89 ± 0.00**	99.34 ± 0.33
	ResNet50	88.69 ± 0.65	90.81 ± 1.08	89.21 ± 0.00	91.24 ± 0.00	**91.35 ± 0.72**
Stanford Dogs	MobileNet	82.88 ± 0.72	-	81.49 ± 0.00	83.10 ± 0.00	**83.40 ± 0.27**
	ViT-Small	88.73 ± 0.46	-	**89.73 ± 0.00**	87.99 ± 0.00	84.66 ± 0.85
	ResNet50	52.97 ± 0.80	53.75 ± 1.44	54.63 ± 0.00	51.48 ± 0.01	**57.98 ± 3.20**
Stanford Cars	MobileNet	50.79 ± 1.39	-	55.41 ± 0.01	**55.93 ± 0.01**	48.87 ± 1.71
	ViT-Small	58.12 ± 2.73	-	63.00 ± 0.02	**66.67 ± 0.01**	59.25 ± 2.23
	ResNet50	92.07 ± 0.44	93.12 ± 0.74	92.91 ± 0.01	93.61 ± 0.00	**93.63 ± 0.43**
Oxford-IIIT Pet	MobileNet	88.56 ± 0.85	-	90.32 ± 0.00	90.14 ± 0.00	**90.53 ± 0.76**
	ViT-Small	94.95 ± 0.56	-	**95.44 ± 0.01**	95.37 ± 0.01	93.54 ± 0.53
	ResNet50	54.09 ± 0.58	56.88 ± 2.60	56.72 ± 0.00	58.28 ± 0.00	**58.75 ± 1.60**
Food101	MobileNet	51.88 ± 0.59	-	52.00 ± 0.00	54.05 ± 0.00	**54.19 ± 1.36**
	ViT-Small	75.40 ± 2.22	-	**79.59 ± 0.00**	78.25 ± 0.00	76.49 ± 0.75

Table 2: 5-way, 5-shot classification accuracy (%) with standard deviation across 6 datasets and 3 downstream image classification architectures. Bolded values indicate the best performance per row.

The few-shot results are shown in Tables 1 and 2. We measure the accuracy on the test set for models trained using the baseline strategies, random augmentation trees, and the augmentation trees learned from our pipeline. While EvoAug consistently outperforms the Naive Baseline, results are mixed when evaluated against AutoAugment and RandAugment. Notably, EvoAug is much better on the Stanford Dogs and Oxford-IIIT Pets datasets, but marginally worse on Flowers102.

Dataset	Model	Naive Baseline	NoOp / Classical Tree	Random Tree	RandAugment	AutoAugment	Learned (Clustering)
Caltech256	ResNet50	65.77 ± 1.29	78.67 ± 2.00	81.57 ± 6.44	81.63 ± 0.01	82.92 ± 0.01	**83.65 ± 4.92**
	MobileNet	67.28 ± 2.56	-	-	71.97 ± 0.01	71.47 ± 0.01	**80.09 ± 4.73**
	ViT-Small	66.72 ± 3.53	-	-	75.60 ± 0.03	75.38 ± 0.03	**82.12 ± 3.64**
Flowers102	ResNet50	61.48 ± 0.78	66.15 ± 0.90	63.03 ± 3.95	63.97 ± 0.00	65.84 ± 0.01	**66.75 ± 2.34**
	MobileNet	57.11 ± 1.30	-	-	53.17 ± 0.01	58.46 ± 0.01	**60.78 ± 2.58**
	ViT-Small	94.60 ± 1.88	-	-	**96.26 ± 0.03**	93.98 ± 0.02	95.47 ± 2.19
Stanford Dogs	ResNet50	70.30 ± 0.58	75.79 ± 0.29	76.58 ± 3.84	75.86 ± 0.01	77.22 ± 0.02	**78.86 ± 3.21**
	MobileNet	60.58 ± 2.66	-	-	65.19 ± 0.02	67.73 ± 0.01	**69.70** ± 2.37
	ViT-Small	75.55 ± 1.61	-	-	78.47 ± 0.01	77.83 ± 0.02	**79.70 ± 3.12**
Stanford Cars	ResNet50	21.31 ± 0.80	28.11 ± 0.43	29.77 ± 1.83	**32.84 ± 0.01**	31.84 ± 0.03	29.66 ± 2.62
	MobileNet	30.43 ± 1.12	-	-	30.18 ± 0.01	**30.35 ± 0.02**	29.05 ± 3.18
	ViT-Small	31.10 ± 5.56	-	-	36.15 ± 0.02	34.66 ± 0.01	**37.35 ± 3.37**
Oxford-IIIT Pet	ResNet50	79.68 ± 1.50	82.44 ± 0.55	81.47 ± 6.34	78.17 ± 0.01	82.71 ± 0.00	**86.16 ± 1.19**
	MobileNet	72.18 ± 1.45	-	-	76.31 ± 0.00	74.17 ± 0.01	**80.43 ± 1.87**
	ViT-Small	76.10 ± 5.44	-	-	83.88 ± 0.02	79.61 ± 0.04	**84.58 ± 3.22**
Food101	ResNet50	30.90 ± 0.57	30.06 ± 0.31	32.83 ± 2.42	30.38 ± 0.00	30.46 ± 0.00	**34.28 ± 0.83**
	MobileNet	28.78 ± 0.63	-	-	25.52 ± 0.01	26.15 ± 0.01	**34.61 ± 1.74**
	ViT-Small	43.74 ± 2.51	-	-	**48.17 ± 0.02**	45.44 ± 0.01	44.89 ± 2.30

Table 3: 5-way, 1-shot classification accuracy (%) with standard deviation across 6 datasets and 3 downstream image classification architectures. Bolded values indicate the best performance per row.

4.3 One-Shot Results

Our 1-shot results are shown in Table 3. Here, we include our clustering-based fitness function learning strategy. Results for our double augmentation and training loss strategies are included in the appendix. EvoAug consistently outperforms the Naive Baseline, and often outperforms RandAugment and AutoAugment, achieving strong performance in scarce data settings. We also run our pipeline restricting nodes to just classical or NoOp transformations and find that these restricted trees perform worse than our normal trees. This supports the conclusion that generative augmentation operators are an important part of performance.

5 Conclusion

We present an automated augmentation strategy that leverages advanced generative models, specifically controlled diffusion and NeRF operators, in combination with classical augmentation techniques. By employing an evolutionary search framework, our method automatically discovers task-specific augmentation policies that significantly improve performance in fine-grained few-shot and one-shot classification tasks. Experimental results on a diverse set of datasets demonstrate that our approach not only outperforms standard baselines but also identifies augmentation strategies that effectively preserve subtle semantic details, which are crucial in low-data scenarios.

Our work introduces novel unsupervised evaluation metrics and proxy objectives to reliably guide augmentation policy search in settings where labeled data is scarce. While the computational overhead associated with evaluating complex generative augmentations remains a challenge, the substantial gains in classification accuracy validate the potential of our approach. Overall, our findings suggest that integrating generative models with automated policy learning can play a pivotal role in enhancing the robustness of vision systems, particularly in environments with limited data.

5.1 Limitations

A potential limitation of our method is its ability to extend to a full dataset recognition task, as directly scaling our pipeline to learn semantic priors from the full dataset is not efficient. Preliminary work, however, has shown that using a text conditioned process to augment images does improve the performance of models on image classification tasks against a classical augmentation baseline (discussion in Appendix A.6). We believe that a more careful augmentation learning strategy that efficiently learns augmentations that match the dataset may be able to further improve this accuracy.

Other avenues of interest are extending this framework to other vision tasks such as object detection and segmentation and further refining the balance between diversity and fidelity in generated augmentations. Preliminary work on these tasks has shown that our pipeline has the ability to improve model performance when compared to a baseline of classically augmented images (discussion in Appendix A.6).

REFERENCES

Mahdi Abavisani, Alireza Naghizadeh, Dimitris Metaxas, and Vishal Patel. Deep subspace clustering with data augmentation. *Advances in Neural Information Processing Systems*, 33:10360–10370, 2020.

Shekoofeh Azizi, Simon Kornblith, Chitwan Saharia, Mohammad Norouzi, and David J Fleet. Synthetic data from diffusion models improves imagenet classification. *arXiv preprint arXiv:2304.08466*, 2023.

Victor Besnier, Himalaya Jain, Andrei Bursuc, Matthieu Cord, and Patrick Pérez. This dataset does not exist: training models from generated images. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2020.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pp. 446–461. Springer, 2014.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Gridmask data augmentation. *arXiv preprint arXiv:2001.04086*, 2020.

Zhengsu Chen, Lingxi Xie, Jianwei Niu, Xuefeng Liu, Longhui Wei, and Qi Tian. Visformer: The vision-friendly transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 589–598, 2021.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 702–703, 2020.

David L. Davies and Donald W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979. doi: 10.1109/TPAMI.1979.4766909.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL `https://arxiv.org/abs/2010.11929`.

Georgios Douzas, Fernando Bacao, and Felix Last. Improving imbalanced learning through a heuristic oversampling method based on k-means and smote. *Information Sciences*, 465:1–20, 2018. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2018.06.056. URL `https://www.sciencedirect.com/science/article/pii/S0020025518304997`.

Joshua James Engelsma, Steven Grosz, and Anil K Jain. Printsgan: synthetic fingerprint generator. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):6111–6124, 2022.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88 (2):303–338, 2010.

Haoyang Fang, Boran Han, Shuai Zhang, Su Zhou, Cuixiong Hu, and Wen-Ming Ye. Data augmentation for object detection via controllable diffusion models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1257–1266, 2024.

Alvaro Figueira and Bruno Vaz. Survey on synthetic data generation, evaluation methods and gans. *Mathematics*, 10(15):2733, 2022.

Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2918–2928, 2021.

Judah A Goldfeder, Patrick Minwan Puma, Gabriel Guo, Gabriel Guerra Trigo, and Hod Lipson. Learning via imagination: Controlled diffusion image augmentation. In *NeurIPS 2024 Workshop on Compositional Learning: Perspectives, Methods, and Paths Forward*, 2024.

Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.

Ryuichiro Hataya, Jan Zdenek, Kazuki Yoshizoe, and Hideki Nakayama. Faster autoaugment: Learning augmentation strategies using backpropagation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*, pp. 1–16. Springer, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Ruifei He, Shuyang Sun, Xin Yu, Chuhui Xue, Wenqing Zhang, Philip Torr, Song Bai, and Xiaojuan Qi. Is synthetic data from generative models ready for image recognition? *arXiv preprint arXiv:2210.07574*, 2022.

Daniel Ho, Eric Liang, Xi Chen, Ion Stoica, and Pieter Abbeel. Population based augmentation: Efficient learning of augmentation policy schedules. In *International conference on machine learning*, pp. 2731–2741. PMLR, 2019.

Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9): 5149–5169, 2021.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Forrest Iandola, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell, and Kurt Keutzer. Densenet: Implementing efficient convnet descriptor pyramids. *arXiv preprint arXiv:1404.1869*, 2014.

Khawar Islam and Naveed Akhtar. Context-guided responsible data augmentation with diffusion models, 2025. URL `https://arxiv.org/abs/2503.10687`.

Khawar Islam, Muhammad Zaigham Zaheer, Arif Mahmood, and Karthik Nandakumar. Diffusemix: Label-preserving data augmentation with diffusion models, 2024. URL `https://arxiv.org/abs/2405.14881`.

Khawar Islam, Muhammad Zaigham Zaheer, Arif Mahmood, Karthik Nandakumar, and Naveed Akhtar. Genmix: Effective data augmentation with generative diffusion model image editing, 2025. URL `https://arxiv.org/abs/2412.02366`.

Ali Jahanian, Xavier Puig, Yonglong Tian, and Phillip Isola. Generative models as a data source for multiview representation learning. *arXiv preprint arXiv:2106.05258*, 2021.

Pan Ji, Tong Zhang, Hongdong Li, Mathieu Salzmann, and Ian Reid. Deep subspace clustering networks. *Advances in neural information processing systems*, 30, 2017.

Animesh Karnewar, Andrea Vedaldi, David Novotny, and Niloy J. Mitra. Holodiffusion: Training a 3d diffusion model using 2d images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18423–18433, June 2023.

Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR workshop on fine-grained visual categorization (FGVC)*, volume 2. Citeseer, 2011.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL `https://arxiv.org/abs/1412.6980`.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4015–4026, 2023.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561, 2013.

Joseph Lemley, Shabab Bazrafkan, and Peter Corcoran. Smart augmentation learning an optimal data augmentation strategy. *Ieee Access*, 5:5858–5869, 2017.

Pu Li, Xiangyang Li, and Xiang Long. Fencemask: a data augmentation approach for pre-extracted image features. *arXiv preprint arXiv:2006.07877*, 2020.

Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *Advances in neural information processing systems*, 32, 2019.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pp. 740–755. Springer, 2014.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2117–2125, 2017.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*, 2023a.

Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object, 2023b.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

Wenquan Lu, Yufei Xu, Jing Zhang, Chaoyue Wang, and Dacheng Tao. Handrefiner: Refining malformed hands in generated images by diffusion-based conditional inpainting, 2024. URL `https://arxiv.org/abs/2311.17957`.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

Alireza Naghizadeh, Dimitris N Metaxas, and Dongfang Liu. Greedy auto-augmentation for n-shot learning using deep neural networks. *Neural Networks*, 135:68–77, 2021.

Supreeth Narasimhaswamy, Uttaran Bhattacharya, Xiang Chen, Ishita Dasgupta, Saayan Mitra, and Minh Hoai. Handiffuser: Text-to-image generation with realistic hand appearances. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2468–2479. IEEE, June 2024. doi: 10.1109/cvpr52733.2024.00239. URL `http://dx.doi.org/10.1109/CVPR52733.2024.00239`.

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729. IEEE, 2008.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao

Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. URL `https://arxiv.org/abs/2303.08774`.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3498–3505. IEEE, 2012.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. arxiv 2022. *arXiv preprint arXiv:2204.06125*, 2022.

René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, 44(3):1623–1637, 2020.

Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, 2015.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 conference proceedings*, pp. 1–10, 2022a.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022b.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019. URL `https://arxiv.org/abs/1801.04381`.

Mert Bülent Sarıyıldız, Karteek Alahari, Diane Larlus, and Yannis Kalantidis. Fake it till you make it: Learning transferable representations from synthetic imagenet clones. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8011–8021, 2023.

Flavio Schneider. Archisound: Audio generation with diffusion, 2023. URL `https://arxiv.org/abs/2301.13267`.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022.

Jordan Shipard, Arnold Wiliem, Kien Nguyen Thanh, Wei Xiang, and Clinton Fookes. Diversity is definitely needed: Improving model-agnostic zero-shot classification via stable diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 769–778, 2023.

Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2107–2116, 2017.

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

Youssef Skandarani, Pierre-Marc Jodoin, and Alain Lalande. Gans for medical image synthesis: An empirical study. *Journal of Imaging*, 9(3):69, 2023.

Zhihang Song, Zimin He, Xingyu Li, Qiming Ma, Ruibo Ming, Zhiqi Mao, Huaxin Pei, Lihui Peng, Jianming Hu, Danya Yao, and Yi Zhang. Synthetic datasets for autonomous driving: A survey. *IEEE Transactions on Intelligent Vehicles*, 9(1):1847–1864, January 2024. ISSN 2379-8858. doi: 10.1109/tiv.2023.3331024. URL `http://dx.doi.org/10.1109/TIV.2023.3331024`.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pp. 6105–6114. PMLR, 2019.

Brandon Trabucco, Kyle Doherty, Max Gurinas, and Ruslan Salakhutdinov. Effective data augmentation with diffusion models. *arXiv preprint arXiv:2302.07944*, 2023.

Brandon Trabucco, Kyle Doherty, Max Gurinas, and Ruslan Salakhutdinov. Effective data augmentation with diffusion models, 2025. URL `https://arxiv.org/abs/2302.07944`.

Toan Tran, Trung Pham, Gustavo Carneiro, Lyle Palmer, and Ian Reid. A bayesian data augmentation approach for learning deep models. *Advances in neural information processing systems*, 30, 2017.

Gary Wang, Ekin D Cubuk, Andrew Rosenberg, Shuyang Cheng, Ron J Weiss, Bhuvana Ramabhadran, Pedro J Moreno, Quoc V Le, and Daniel S Park. G-augment: Searching for the meta-structure of data augmentation policies for asr. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pp. 23–30. IEEE, 2023.

Shin'ya Yamaguchi and Takuma Fukuda. On the limitation of diffusion models for synthesizing training datasets. *arXiv preprint arXiv:2311.13090*, 2023.

Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications, 2025. URL `https://arxiv.org/abs/2209.00796`.

Zihan Yang, Richard O Sinnott, James Bailey, and Qiuhong Ke. A survey of automated data augmentation algorithms for deep learning-based image classification tasks. *Knowledge and Information Systems*, 65(7):2805–2861, 2023a.

Zuhao Yang, Fangneng Zhan, Kunhao Liu, Muyu Xu, and Shijian Lu. Ai-generated images as data source: The dawn of synthetic era. *arXiv preprint arXiv:2310.01830*, 2023b.

Zhuoran Yu, Chenchen Zhu, Sean Culatana, Raghuraman Krishnamoorthi, Fanyi Xiao, and Yong Jae Lee. Diversify, don't fine-tune: Scaling up visual recognition training with synthetic images. *arXiv preprint arXiv:2312.02253*, 2023.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847, 2023.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 13001–13008, 2020.

Yongchao Zhou, Hshmat Sahak, and Jimmy Ba. Training on thin air: Improve image classification with generated data. *arXiv preprint arXiv:2305.15316*, 2023.

A Appendix

We provide additional results for our method in the 5-way 1-shot setting, as well as a study on the one-shot clustering fitness function. We also examine how our method might scale to be used on full datasets and object detection/segmentation tasks.

A.1 FITNESS FUNCTION CHOICE IN ONE-SHOT SETTING

Carefully crafting a fitness function which can enable robust downstream classification is a difficult task. Our three main approaches were using augmented images themselves as a part of the validation set for models, using heuristics from the training loss to determine optimal learning, and an involved clustering approach which tried to capture the spread within a class and between classes. Our results are summarized in Table 4, which show that the clustering approach seemed to be a consistently good strategy for guiding our augmentation scoring.

Dataset	Model	Learned (Double Aug)	Learned (Train Loss)	Learned (Clustering)
Caltech256	ResNet50	80.27 ± 6.50	76.21 ± 5.58	**83.65 ± 4.92**
	MobileNet	75.04 ± 6.68	68.57 ± 2.85	**80.09 ± 4.73**
	ViT-Small	73.53 ± 4.31	68.57 ± 10.44	**82.12 ± 3.64**
Flowers102	ResNet50	55.95 ± 5.72	65.73 ± 6.54	**66.75 ± 2.34**
	MobileNet	57.70 ± 1.45	**64.38 ± 2.99**	60.78 ± 2.58
	ViT-Small	88.27 ± 2.42	86.11 ± 1.85	**95.47 ± 2.19**
Stanford Dogs	ResNet50	78.46 ± 2.27	68.26 ± 2.19	**78.86 ± 3.21**
	MobileNet	66.40 ± 2.56	67.25 ± 2.49	**69.70 ± 2.37**
	ViT-Small	74.73 ± 4.08	67.57 ± 1.55	**79.70 ± 3.12**
Stanford Cars	ResNet50	22.34 ± 2.92	28.36 ± 0.86	**29.66 ± 2.62**
	MobileNet	25.29 ± 4.40	26.70 ± 1.88	**29.05 ± 3.18**
	ViT-Small	24.79 ± 0.63	36.73 ± 0.80	**37.35 ± 3.37**
Oxford-IIIT Pet	ResNet50	76.07 ± 2.26	78.30 ± 1.44	**86.16 ± 1.19**
	MobileNet	75.76 ± 2.58	74.45 ± 4.54	**80.43 ± 1.87**
	ViT-Small	75.69 ± 4.36	80.58 ± 3.62	**84.58 ± 3.22**
Food101	ResNet50	30.40 ± 1.56	30.49 ± 3.22	**34.28 ± 0.83**
	MobileNet	28.44 ± 0.89	29.38 ± 2.32	**34.61 ± 1.74**
	ViT-Small	38.17 ± 1.53	39.53 ± 4.35	**44.89 ± 2.30**

Table 4: 5-way, 1-shot classification accuracy (%) with standard deviation across 6 datasets and 3 downstream architectures, showing only the three Learned methods. Bolded values indicate the best performance per row.

A.2 ENCODER PERFORMANCE COMPARISON

The one-shot clustering fitness function results only use a single image encoder, a pre-trained ResNet50. We begin this analysis by benchmarking various pre-trained image encoders—responsible for projecting augmented images into embedding space—for their effectiveness in the clustering-based fitness function. We explore two variants of Vision Transformers (Dosovitskiy et al., 2021) in addition to a ResNet50. Table 5 provides the results of the encoder performance comparison. The two vision transformer variants outperform the baseline on all datasets. Notably, however, there is no single best decoder that performs consistently the best across all datasets.

Table 5: Accuracy for 5-way 1-shot clustering fitness function across various image encoders

Dataset	Baseline	ResNet50	ViT-224	ViT-B/16
Caltech256	65.77 ± 1.29	$\mathbf{81.83 \pm 7.60}$	79.56 ± 1.10	72.28 ± 1.98
Flowers102	61.48 ± 0.78	56.70 ± 4.19	62.41 ± 2.38	$\mathbf{64.38 \pm 3.30}$
Stanford Dogs	70.30 ± 0.58	71.54 ± 2.70	$\mathbf{75.45 \pm 2.25}$	72.72 ± 2.42
Stanford Cars	21.31 ± 0.80	24.50 ± 3.00	$\mathbf{25.71 \pm 2.74}$	24.38 ± 1.32
Oxford-IIIT Pet	79.68 ± 1.50	85.21 ± 1.14	83.08 ± 2.81	$\mathbf{85.88 \pm 0.66}$
Food101	30.90 ± 0.57	$\mathbf{33.97 \pm 1.63}$	32.21 ± 0.41	32.62 ± 0.67

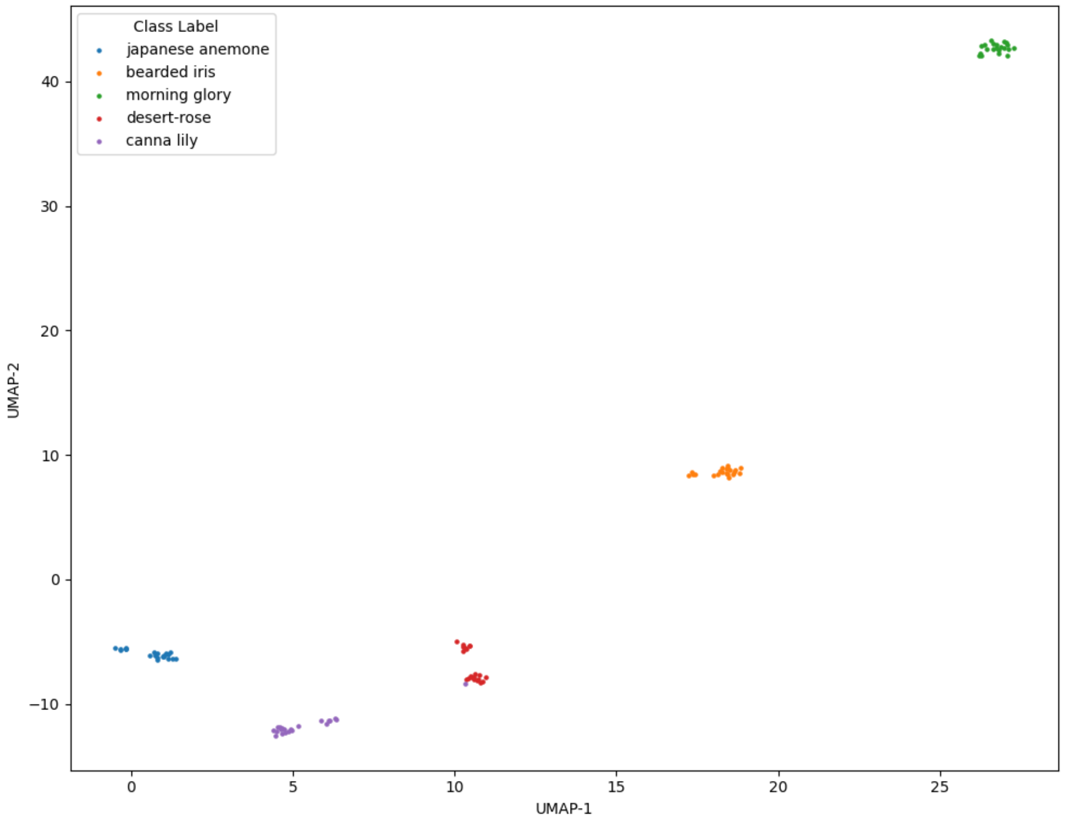

Figure 4: Success Case: ViT-B/16-Flowers

A.3 Success and Failure Analysis

We look at low-dimensional cluster visualizations of encoded augmentations from strong and weak-performing learned trees for success and failure cases using UMAP (McInnes et al., 2018). This motivates the desired and non-desired qualities of clusters. We examine the clusters of the embeddings of two encoders on the Flower dataset, shown in Figure 4 and Figure 5. We select a single dataset to establish domain consistency when comparing success and failure cases, as well as against the handcrafted tree study in the following section. The Flowers102 dataset is particularly interesting as it is the most fine-grained among those benchmarked. Unlike other datasets, where shape or size may be primary distinguishing features between classes, flowers are primarily defined by their color. As a result, applying augmentations that alter color can significantly degrade model performance.

For the success case – ViT-B/16 on Flowers102 – which performed 3% better than baseline, there are distinct clusters for all five classes, all of which are very tight. Clusters are also very well separated. For the failure case – ResNet50 on Flowers102 – which performed 5% worse than baseline, the classes are not clustered very accurately, with augmentations overlapping heavily between classes.

A.4 Handcrafted Augmentation Trees

We handcraft an ”ideal” augmentation tree for the Flowers102 dataset, shown in Table 6, to compare to the clusters of the EvoAug learned trees in the success and failure cases. The hypothesized ideal augmentation tree is structured as follows: the head node as Color, the left node as NeRF, and the right node as no augmentation, with a 0.5 probability of moving to either child node. We guarantee a Color node, as it uses Color ControlNet to preserve the color palette in augmentations. We also use a NeRF node, which performs a 3D rotation for an augmentation, yet not affecting color.

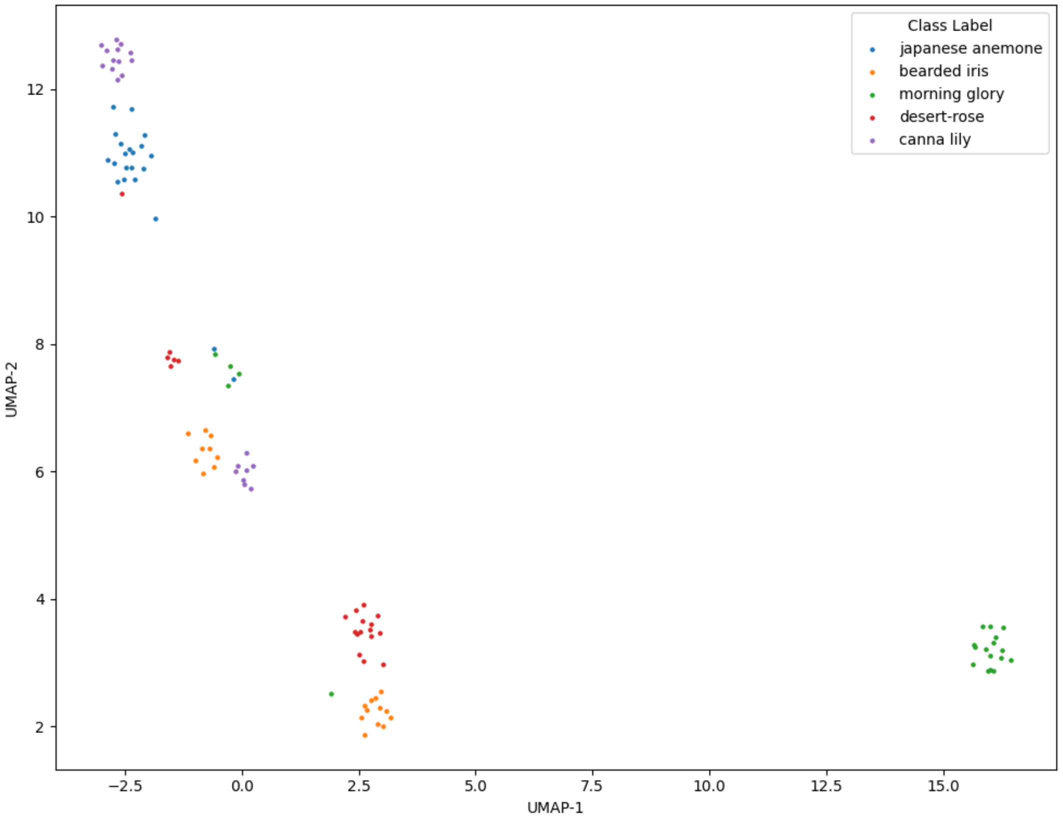

Figure 5: Failure Case: ResNet50-Flowers

Table 6: Handcrafted tree performance on the Flowers102 dataset. Tree structure format: (Head, p_L, Left, p_R, Right).

Name	Tree Structure	Accuracy (%)
Ideal	(Color, 0.5, NeRF, 0.5, None)	66.98 ± 6.56
Inferior	(Depth, 0.5, Depth, 0.5, Segmentation)	60.85 ± 2.30

We also handcraft an "inferior" augmentation tree as a sanity check and counterexample, allowing us to compare clusters and better isolate critical features to reward when designing the clustering fitness score. We use Depth and Segmentation nodes for augmentations, as neither augmentation operation preserves color, which we hypothesize to be the most important feature for flower classification.

The handcrafted ideal augmentation tree performs better than all other augmentation trees learned from any image encoder, suggesting that the EvoAug pipeline is not learning the best augmentation tree through the clustering score fitness function. The ideal handcrafted tree in Figure 6 and the learned tree success case in Figure 4 both display very well-separated clusters for each class. However, the clusters for the success case are noticeably tighter than those of the handcrafted tree

Image Encoder	$\mathbf{S - \frac{1}{d}}$	$\mathbf{S - \frac{2}{d}}$	**S**	$\mathbf{\frac{1}{DB}}$
ViT-B/16	64.382 ± 3.302	67.497 ± 2.827	61.059 ± 0.44	61.059 ± 0.44

Table 7: One-shot clustering results across different fitness functions for Flowers102 subset 50

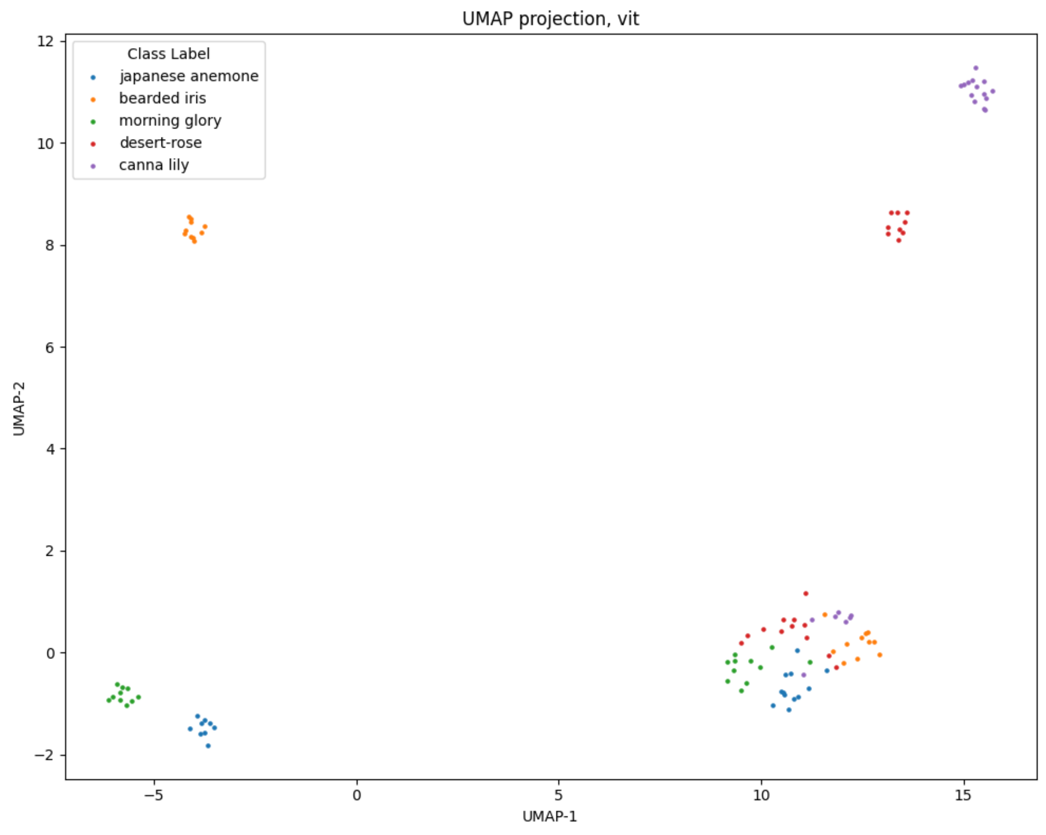

Figure 6: Handcrafted Ideal Tree

clusters. If we compare this to Figure 5 or 7, we can see larger clusters formed from a variety of different classes, with fewer clusters that distinctly correspond to a single class.

These observations give rise to two interpretations: (1) the original fitness function may have undervalued the importance of large clusters, because the better performing handcrafted ideal tree resulted in larger yet still distinct clusters and (2) that the original fitness function may have overvalued the importance of large clusters at the expense of cluster separability, as the failure case and handcrafted inferior tree demonstrate. This motivates an exploration of alternative fitness functions that may better capture cluster dynamics.

A.5 Clustering Fitness Function Modifications

Table 7 compares the performance of different clustering metrics as the fitness function in the EvoAug pipeline, where **S** is the Silhouette coefficient, **d** is the average cluster radius, **DB** is the Davies-Boudlin Index (Davies & Bouldin, 1979). We conduct experiments using the Flowers102 dataset and use ViT-B/16 encoder as it performs the best on this dataset.

We test a fitness function of just **S** as a baseline, but using only the Silhouette Coefficient results in a learned tree of None nodes, causing all generated augmentations to be exact copies of the original image. This is expected, as the Silhouette Coefficient scores clusters of the same embedding as a perfect score of 1, due to the small intra-cluster distances. The same result occurs with the $\frac{\mathbf{1}}{\mathbf{DB}}$ fitness function, confirming that Davies-Bouldin is functionally the same as the Silhouette Coefficient.

We modify the original proposed fitness function by doubling the penalty to small cluster sizes. Under this setting, the learned augmentation tree is $(\text{Head}, p_L, \text{Left}, p_R, \text{Right}) = (\text{None}, 0.51, \text{None}, 0.49, \text{NeRF})$. This tree results in the best downstream classification performance

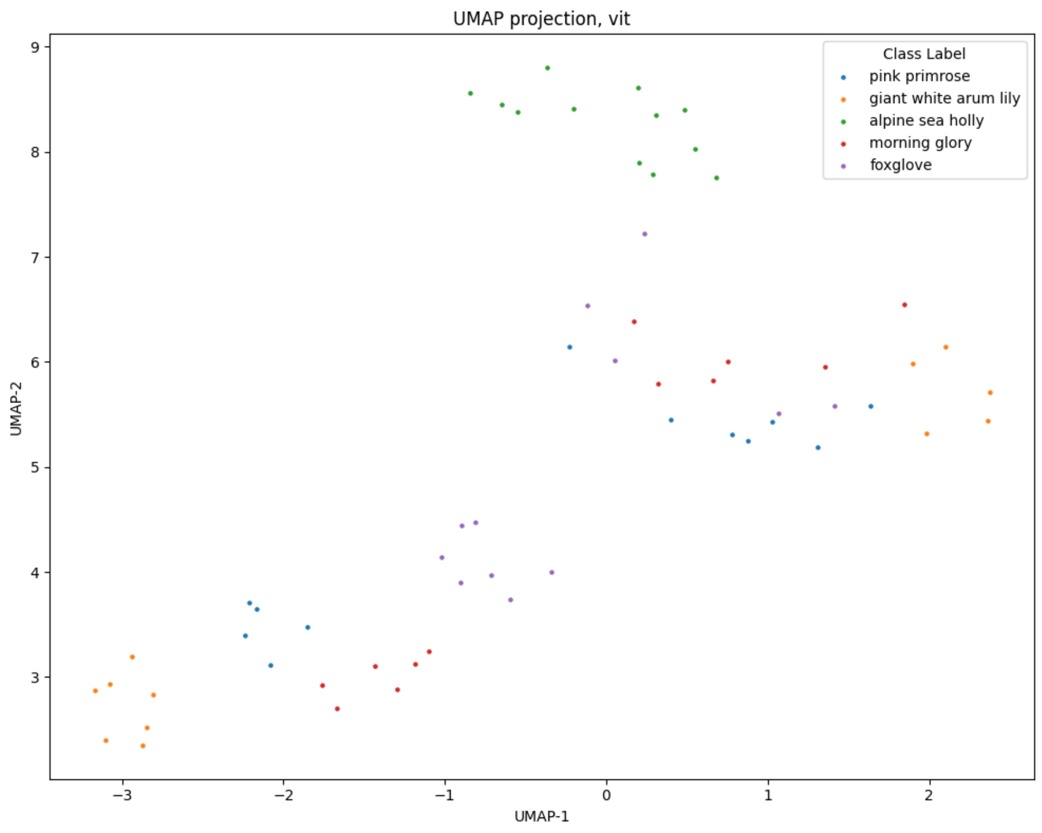

Figure 7: Handcrafted Inferior Tree

across all experiments, including those from handcrafted trees, demonstrating that this fitness function was able to learn better trees than human intuition. This learned tree was likely favored in the evolutionary algorithm, as NeRF preserves colors and edges, two features we believe are vital for classifying flowers. These results strengthen the interpretation that a large intra-cluster distance is important may help in model generalization. Future work will seek to substantiate this claim in other settings and datasets.

A.6 Generalization to Full Datasets, Detection, and Segmentation

While the main body of our work focuses on the few shot setting, there are also experiments done which have indicated that conditioned generation is beneficial in the full dataset setting (Anonymous, 2024). The method used in these experiments employs LLaVa2 (Liu et al., 2023a) generated captions to condition the augmentation of images in the dataset. We believe that with more intelligent conditioning (by learning augmentation trees which match the dataset), we can achieve better performance.

We reproduce the relevant summary statistics below in Table 8 for completeness. The results show that conditioned generation consistently achieves higher accuracy than a classically augmented baseline across six datasets: Caltech256 (Griffin et al., 2007), Stanford Cars (Krause et al., 2013), FGVC Aircraft (Maji et al., 2013), Stanford Dogs (Khosla et al., 2011), Oxford IIIT-Pets (Parkhi et al., 2012); and eight model architectures: ResNet (He et al., 2016), VGG (Simonyan & Zisserman, 2014), EfficientNet (Tan & Le, 2019), Visformer (Chen et al., 2021), Swin Transformer (Liu et al., 2021), MobileNet (Howard et al., 2017), DenseNet (Iandola et al., 2014), and ViT (Dosovitskiy et al., 2021).

We have also done some 5-way, 2-shot experiments on the PASCAL VOC dataset (Everingham et al., 2010). For these experiments, we fine-tuned a Faster R-CNN (Ren et al., 2015) with a ResNet-50-FPN backbone (Lin et al., 2017) pretrained on COCO (Lin et al., 2014). Our results show that a baseline strategy which only uses classical augmentations achieves a performance of 18.77 ± 5.95 percent, while our generative augmentation pipeline achieves a performance of 21.53 ± 7.20 percent. This indicates that our generative augmentation pipeline can also benefit dense prediction tasks.

Table 8: Accuracy on full datasets for various models

Dataset	Setting	RN50	RN101	VGG19	EN	Visformer	Swin	MN	DN
Caltech	Baseline	72.37	73.62	67.40	71.79	68.83	63.95	66.48	75.74
	Conditioned	**76.49**	**77.64**	**70.82**	**73.85**	**73.15**	**69.55**	**68.33**	**78.10**
Cars	Baseline	86.78	88.16	87.22	86.75	83.37	75.43	80.80	91.08
	Conditioned	**91.02**	**90.95**	**89.61**	**88.56**	**87.40**	**82.32**	**82.70**	**92.20**
Aircraft	Baseline	75.23	75.91	**88.80**	81.25	72.61	60.88	70.24	80.53
	Conditioned	**82.33**	**81.10**	88.20	**81.76**	**74.67**	**71.74**	**74.17**	**83.29**
Dogs	Baseline	66.49	70.15	**68.63**	**64.17**	**64.65**	52.10	**58.60**	**70.44**
	Conditioned	**68.74**	**70.40**	66.05	62.45	64.36	**56.50**	58.30	70.21
Pets	Baseline	69.22	70.72	**83.17**	73.59	73.02	58.54	67.35	**80.16**
	Conditioned	**71.07**	**74.03**	81.28	**74.41**	**76.24**	**61.00**	**68.46**	79.34