Synthetic Data Generation for Few-Shot Learning

Rajan Vivek
Stanford University
rvivek@stanford.edu

Vaishnavi Shrivastava
Stanford University
vaish1@stanford.edu

Ofure Ebhomielen
Stanford University
ofure@stanford.edu

Extended Abstract

Achieving algorithms that can learn in a few-shot manner as humans do is a tantalizing goal. This ability requires general knowledge about a domain in order to recognize what parts of an example are salient to the task at hand. The incredible success of large models including ResNets, Vision Transformers, and diffusion models across a variety of image tasks suggests that these models have attained a level of general image domain knowledge. Departing from the convention of using human-curated image augmentation types for synthetic data generation, we ask: Can large models generate synthetic training data in a few-shot manner such that training on the synthetic data improves the performance of a smaller downstream model? We propose a model-agnostic training recipe that optimizes image models for generating training data that is useful to a downstream classifier, specifically in the few-shot regime. This involves a MAML-style inner and outer loop scheme run over many few-shot image tasks. In the inner loop, a small classifier trains on data generated from a larger image augmenter model using a small support set. In the outer loop, the augmenter parameters are optimized by a gradient step to minimize the loss of the downstream classifier. We experiment with two frameworks for synthetic data generation: an Encoder-Decoder framework that adds noise to encoded image representations before decoding, and a latent-diffusion framework that learns a soft prompting network to condition Stable Diffusion on a small image support set. We evaluate these frameworks on MiniImagenet across a wide variety of experimental parameters including support set size, task difficulty, amount of synthetic images generated, and complexity of the classifier and augmenter models.

We see some positive results for our Encoder-Decoder experiments where learned augmentations using ResNets and ConvNets improve performance at small support set sizes. Additionally, we discover that learned augmentations boost performance more for larger, pre-trained image classifiers than for smaller classifiers trained from scratch. For many other configurations and settings we investigate, we find that our current implementations do not surpass our baselines, tending to harm performance relative to training only on the support examples. We rigorously investigate the nature of our current experiments and identify several factors that likely contributed to the lack of success of these frameworks. We recognize that the primary factor was likely the complexity of the models we experimented with (the largest model being an unfrozen ResNet50) being too low for the image-augmentation task at hand. We empirically find that the performance of our models vary strongly across the diverse MiniImagenet few-shot tasks, further suggesting that greater model complexity is needed.

We obtain inconclusive results for our diffusion experiments. We find that our pretrained Stable Diffusion model can only generate high-quality images of a size

similar to what it was trained on, but these sizes result in a model too large for backpropogation with our available computational resources. The trained soft-prompting network was not able to compensate for the misaligned frozen diffusion model. We outline a variety of next steps for investigating the ideas put forth here. Our code is publicly available at https://github.com/rvivek3/Automatic-Data-Augmentation/.

1 Motivation

The central challenge of learning from a small amount of image data is that spurious correlations exist; certain image qualities that are independent of image class tend to be overrepresented in specific classes, causing a trained model to be right for the wrong reasons (on the training data). For this reason, data augmentation has proven to be a useful tool for low-data regimes. These techniques leverage human-curated augmentations that change image qualities that are independent of class labels, encouraging a model to learn representations related to class-specific features. However, for a given downstream classification task, deciding what augmentations would lead to the greatest performance boost is a challenging problem.

The success of large image models (e.g. ResNets, vision transformers, latent diffusion models) across a wide variety of image tasks raises an important question: can such models be leveraged to generate augmented images such that training a downstream classifier on these synthetic images results in better performance than training on the original training data? This can be thought of as a form of knowledge distillation: perhaps a large pre-trained model with rich image representation capabilities can convey to a small model what image qualities are important for a given task. Such a framework would permit very low inference time computation: all inference calls would only require a forward pass through the small model. Given that inference accounts for up to 90% of operational costs for productionized machine learning systems Ama, this goal is tantalizing.

We aimed to demonstrate that a deep neural network-based synthetic data generator can improve the performance of a small classifier on a few-shot image classification task.

2 Related Work

There are various existing methods for learning image augmentations from data. One active line of research approaches the problem with reinforcement learning, defining a search space of magnitudes of various predefined augmentations (e.g. shearing, cropping, adjusting contrast) and learning a policy that maximizes validation classification accuracy Yang et al. (2022); Cubuk et al. (2019, 2020). Another prominent work finds that learning instance-specific augmentations– magnitudes of predefined augmentations unique to each input image– improves performance over global augmentation techniques Miao et al. (2022). To our knowledge, existing methods do not learn the augmentation types from data, instead aiming to instill inductive biases in the model from human-curated augmentation types. Notably, we aim to learn augmentation types directly from data.

Rather than directly learning optimal augmentations, another approach to synthetic data generation is to leverage image generation models. Latent text-to-image diffusion models Rombach et al. (2022b), in which denoising auto-encoders are applied in sequence to a noise input represented in a latent space and guided by a representation of the text input, have achieved incredible image quality results. While few-shot diffusion models have been explored Giannone et al. (2022), to our knowledge no works have explicitly optimized these models for the output to be used as training data.

The output of large generative models depends strongly on the nature of the (typically text) prompt input. While finding an ideal prompt to get the desired output is challenging, one useful approach is soft-prompting: training an additional model conditioned on the input to learn continuous vectors that, when concatenated to the existing input, improve the generative model output Bulat & Tzimiropoulos (2022).

3 Methods

3.1 Formulation

We propose two approaches for learning to generate synthetic training data for a few-shot image classification task. In both approaches, we aim to learn a neural network-based augmenter A that

accepts a small image support set S as input and outputs a synthetic image support set S' that is optimized for improving downstream classifier performance. Algorithm 1 shows our general model-agnostic training recipe. The recipe is analogous to MAML, consisting of an inner loop and an outer loop. In the inner loop, S is passed through our augmenter to generate S' and a classifier is trained to convergence on $S \cup S'$. In the outer loop, the loss of the classifier on the query set is backpropagated through all the classifier training steps to update the weights of the augmenter by one gradient step.

Algorithm 1 Augmenter Training Recipe for Optimizing Downstream Classifier Performance

$D_{train} \leftarrow \{$ image x_i, label $y_i\}_{i=1}^{B}$
$T \leftarrow$ number of training iterations
$I \leftarrow$ number of classifier training (inner loop) steps
$A \leftarrow$ augmenter architecture, possibly partially frozen
$C \leftarrow$ classifier architecture, possibly partially frozen
for $i \in range(T)$ **do** ▷ Outer Loop
  $S, Q \leftarrow$ support and query sets sampled from D_{train}
  $S' \leftarrow A(S)$, synthetic samples generated from augmenter
  $C \leftarrow C$ trained on $S \cup S'$ for I steps ▷ Inner Loop
  $L \leftarrow$ cross entropy loss of C evaluated on Q
  $A \leftarrow A$ optimized for one gradient step w.r.t. L
  $C \leftarrow C$ with unfrozen parameters reinitialized to random
end for

3.2 Approach 1: Encoder-Decoder Augmentations

In this approach, A consists of an encoder and decoder architecture. The encoder, which in our experiments is a convolutional neural network but can be any image-to-vector network, encodes each support image in a low-dimensional latent space. We optionally inject noise into each latent representation and then project the vector back to the original image size using a series of transposed convolutional layers. Intuitively, we expect this process to generate synthetic data that preserves class features but distorts other features. We experiment with various encoder-decoder architectures (see 1a)

3.3 Approach 2: Image-Conditioned Diffusion Augmentations

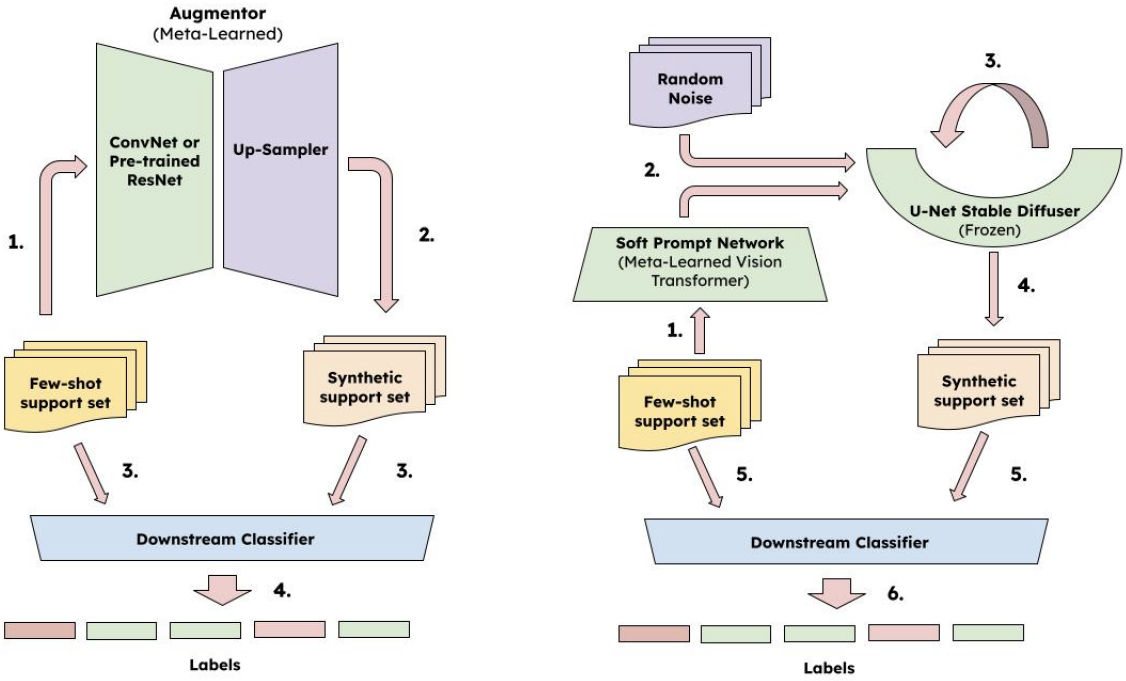
Taking inspiration from the success of soft prompting language models Bulat & Tzimiropoulos (2022); Zhao & Schütze (2021), we aim to learn a soft-prompting network to condition a text-to-image model (Stable Diffusion) on an image support set in order to generate new instances of each class. Specifically, our soft prompt network maps each class in a small image support set S to a tensor that is the same shape as an N-word text prompt but does not correspond to words in the vocabulary of the Stable Diffusion text encoder. In this approach, A consists of both the soft prompt network and the Stable Diffusion model, the latter of which is frozen (as typical for learning soft prompts). We use an (unfrozen) pre-trained Vision Transformer (see 1b) as our soft prompt network.

4 Experiment Setup

4.1 Shared Setup

The specific setup we chose to evaluate our data augmentation framework was learning data augmentations to enable a classifier to more effectively perform an N-way classification task on the MiniImagenet dataset Vinyals et al. (2016), a dataset with 100 different object classes, with 60 images available for each. More precisely, an N-way, K-shot, Q query task is defined as performing image classification between N randomly selected object classes from Mini-Imagenet, with K support examples, and Q query examples sampled randomly from each class. As mentioned in section 3.1, the K support examples are used to tune the classifier in the inner loop, the loss of this tuned classifier is computed against the Q query examples, and then the gradient from the query examples is backpropagated into the augmenter network to learn augmentations that can help the classifier more effectively few-shot adapt to a new task.

We experiment with a few different baselines for our framework including training our classifier on:



(a) Encoder-Decoder Augmenter

(b) Image-Conditioned Diffusion Augmenter

Figure 1: Diagrams showing the augmenter architectures for our experiments

1. **No data augmentations**
2. **Duplicates of the original dataset**: duplicating each image in our support set
3. **Fixed data augmentations**: horizontal flips, cropping, adding gaussian noise, etc.
4. **Augmentations produced from a frozen randomly initialized ConvNet**: matching the ConvNet augmenter described below in architecture, but with random frozen weights

Our primary classifier was a standard convolutional neural network model with 3 convolutional layers with $kernel\_size = 3$, $stride = 1$, $padding = 1$ each, and 18, 36, and 72 filters respectively. Each convolutional layer was followed by a 2D max pool layer with $kernel\_size = 2$, $stride = 2$, $padding = 0$. The CNN and max pooling layers were then followed by two fully connected layers scaling the image representation down to first 64 features, and then to the number of classes we are classifying between (N). All experiments were done using this classifier, outside of those described in Section 5.4. In section 5.4, we also train a second more complex classifier which is a pre-trained ResNet-50 model He et al. (2016), where only the top few layers are tuned, followed by a few fully connected layers for classification.

When learning a data augmenter to produce good data augmentations, we experimented with different types of augmenters, including:

1. **Convolutional Neural Network-based Augmenter**: A simple CNN-based network that has a down convolution layer to compress the image into a dense representation followed by an up-convolution layer.
2. **Frozen Pre-trained ResNet-based Augmenter**: A frozen ResNet-50 model with the feedforward layers removed with 7 up-transpose layers with $kernel\_sizes$ varying from 3, to 5, to 7 and stride varying from 3, to 2, to 1, and padding staying consistent at 1. Only the up-convolution layers are trainable here.
3. **Unfrozen Pre-trained ResNet-based Augmenter**: Same as 2., but the ResNet-50 is unfrozen and trainable along with the up-convolution layers.
4. **Stable-Diffusion based Augmenter**: A pretrained (ViT-B 32) Vision Transformer Radford et al. (2021) for soft prompting and frozen Stable Diffusion model Rombach et al. (2022a) for image generation.

All of our models are trained for 500 iterations or until convergence, with a batch size of 4 (4 tasks sampled in each batch), with 20 inner loop steps. The models are all evaluated using their final checkpoints on 100 tasks that make up our test set for each configuration (N-way, K-shot, Q-query) and comparisons are done based on deltas in final accuracy compared to the baseline of incorporating no augmentations. All of the Mini-Imagenet images are resized to shape 128 x 128.

4.2 Varying Support Set Sizes

We would expect data augmentation to help the most when we have a small support set of examples since it would otherwise be difficult to use the small support set to learn a clear set of signals distinguishing the different classes. In this case, we would hope to learn data augmentations that reduce any spurious correlations between the classes and amplify the signal separating the classes, allowing the classifier to more easily perform the N-way classification task. In this set of experiments, we vary the support set sizes, while keeping the task difficulty (N-way classification) and number of query examples (Q) constant. Specifically, we consider a 3-way classification task and vary our support size between 1, 5, 10, and 20 examples, while keeping the query set size fixed at 5 examples. Our hypothesis is that our learned data augmentations will be more useful at smaller support set sizes than larger ones.

4.3 Varying Task Difficulty

As the classification task becomes more difficult (increase the N in N-way classification, to classify between more classes), the signal differentiating image classes becomes more challenging to extract from the support set. As a result, having data augmentations that minimize spurious correlations is likely to be more beneficial. In this set of experiments, we vary the task difficulty by considering tasks of classifying between varying numbers of object classes. Specifically, we consider 2-way, 3-way, and 5-way classification tasks, while keeping our support set size fixed at 10 examples and our query set size fixed at 5 examples. Our hypothesis is that the data augmentations we learn will be most beneficial as task difficulty increases.

4.4 Varying Augmentation Factor

We define the metric 'Augmentation Factor' as the number of augmentations generated per input image. For our encoder-decoder augmenters, adding noise to encoded image representations prior to decoding allowed us to generate an unbounded number of augmentations for a single input image. We chose to add Gaussian noise with zero mean and 0.01 variance for all encoder-decoder experiments. The variance was chosen to be as large as possible without harming the performance of an augmenter with augmentation factor 1 relative to a noiseless augmenter. For our diffusion experiments, the augmenter is stochastic by default and no additional noise was needed. We experimented with 3-way, 5-shot, 5-query image classification tasks with augmentation factors of 1, 2 and 5. We hypothesized that a greater augmentation factor would improve performance.

4.5 Varying Augmenter Complexity

Across all of these experiments, we vary the complexity of the augmenter. We first experiment with both a frozen and unfrozen ConvNet augmenter with one convolutional layer and one transposed convolutional layer. We then increase the complexity of the augmenter, using a frozen and unfrozen ResNet-50 He et al. (2016) with transposed convolutional layers as described in section 4.1 above. We hypothesized that the more complex models would learn better augmentations than the smaller or partially frozen models.

4.6 Impact of Augmentations on More Complex Classifiers

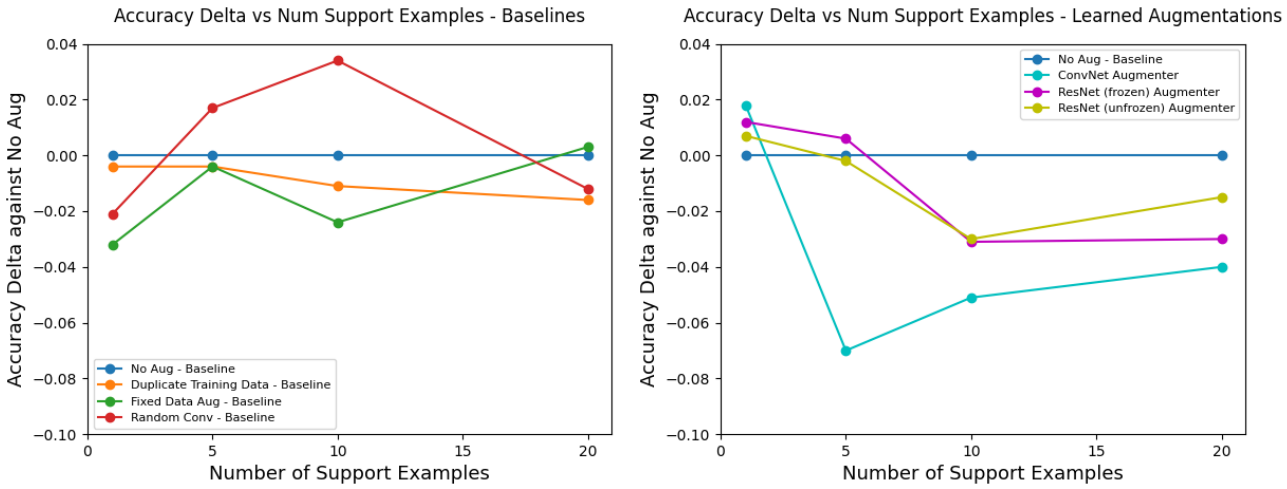
We compare the impact of learned augmentations on training a simple, ConvNet classifier vs finetuning a more complex, pre-trained ResNet classifier. We hypothesized that the more complex, pre-trained classifier would show more consistent performance gains of using augmented data, given our few-shot learning setup with small support set sizes. Whereas the simpler ConvNet classifier may not have the expressivity to learn these tasks well, given the small support sets, a larger pre-trained ResNet could perhaps have the complexity to better capture the augmented data and use it to perform more accurate classifications.

4.7 Image-Conditioned Diffusion Experiments

We experimented with various parameters for our image-conditioned diffusion set-up including various soft prompt sizes (i.e. how many "words" the soft prompt corresponds to), support set sizes, number of diffusion steps, learning rates, generated image sizes (which we would then resize to 128 x 128 for classifier training), number of synthetic images generated per class, and soft prompt architectures (vision transformer and 4-layer convolutional neural network with max pooling). In all settings, a soft prompt for each class was generated by averaging the soft prompt network output for all instances of a given class in a given batch.

5 Results and Discussion

5.1 Varying Support Set Size



(a) Baseline

(b) Learned Augmentations

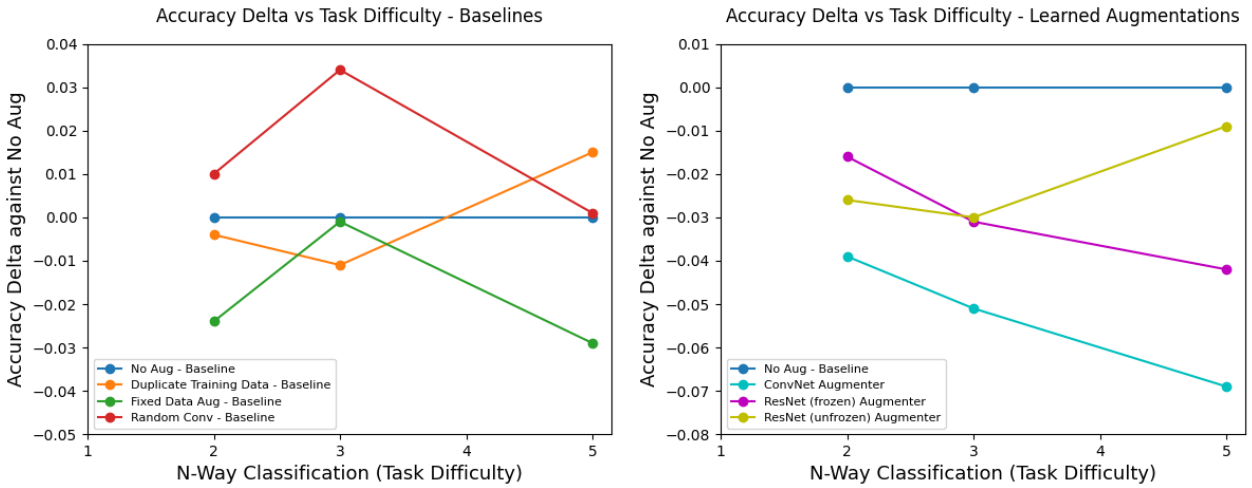
Figure 2: Plots of Accuracy Deltas against no Augmentation for varying support sizes

Figure 2a shows the deltas in percent accuracy for the baselines described in section 4.1 compared to the no augmentation baseline. The baselines of duplicating training and performing fixed augmentations consistently underperform the no augmentation baseline, with the fixed augmentation baseline providing slight gains at the largest support set size. This may be because duplicating the training data is not providing the model with any new information, and perhaps the set of fixed augmentations we perform are modifying the images enough to lose any useful inductive biases. The randomly initialized ConvNet baseline improves on the no augmentation baseline in the 5 and 10 support set size cases. Perhaps these weights were initialized in a way that allowed the ConvNet to preserve some initial structure from the images that was lost over the course of trying to train the ConvNet augmenter. Experimenting with additional random initializations of the ConvNet and seeing if the results remain consistent would be interesting future work.

Figure 2b shows the deltas in percent accuracy for each of our learned augmentations described in 4.1. As expected all learned augmentations improve performance at our smallest support set size of 1, and the frozen ResNet augmenter also improves performance at a support set size of 5. However for larger support set sizes all augmenters degrade performance compared to the baseline of training with no augmentations. We hypothesize that this is because for larger support set sizes, learning bad augmentations for any subset of the images can lead to performance degradations. Augmentations from both ResNet models outperformed augmentations from the ConvNet augmenter across the board, which makes sense given the increased complexity of the ResNet model. The unfrozen ResNet augmenter starts outperforming the frozen ResNet model at larger support set sizes, since then we are better able to tune the learnable parameters of the model.

5.2 Varying Task Difficulty

Figure 3a shows the performance of different baselines against the no augmentation baseline, when varying the task difficulty of N-way classification by increasing N. Figure 3b shows the same results



(a) Baseline

(b) Learned Augmentations

Figure 3: Accuracy Deltas against no Augmentation Baseline for Varying Task Difficulty

for each of our learned augmentations. The randomly initialized frozen ConvNet baseline is again able to improve performance across all task difficulties, while the duplicated training data baseline improves performance at higher difficulty tasks. This may be because with more difficult tasks, seeing the same examples for the classes can help reinforce the signals that distinguish them and help make the classification problem easier. The Fixed Augmentation baseline performs at or below the level of having no augmentations, which may be due to the set of augmentations corrupting the images beyond a point of being useful, or causing images of different classes to have more shared structure.

All of our learned augmentations underperformed the baseline of having no augmentations present during training. We further expand on why these learned augmentations may not be useful in Section 5.5. Both ResNet augmenters again outperformed the ConvNet augmenter, as expected based on augmenter complexity. The unfrozen ResNet augmenter improved over the frozen ResNet augmenter, since that is when the expressivity of tuning the ResNet layers would be most useful for discriminating between classes.

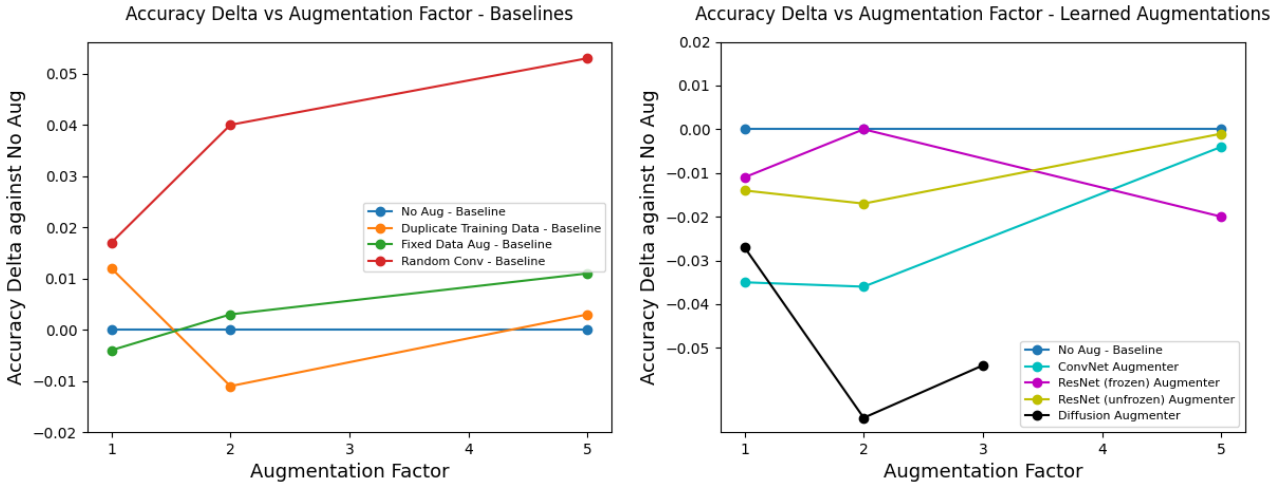
5.3 Varying Augmentation Factor

Figure 4a shows the deltas in percent accuracy of the baselines described in section 4.1 across various augmentation factors. The same results are shown for the learned augmenters in Figure 4b. We are again surprised to see that the random (untrained) ConvNet baseline is the strongest performer, improving as the amount of data it generates increases. All learned augmenters unfortunately only harm performance with no obvious trends. We elaborate on possible explanations for these results in Section 5.5.

5.4 Impact of Augmentations on More Complex Classifiers

After experimenting with using augmentations to improve the performance of ConvNet classifiers, we hypothesized that our learned augmentations might show more gains in the setting of using a large, pre-trained classifier like a ResNet, given the small number of support examples and difficulty of classification over the MiniImagenet dataset. Figure 5 shows the delta in percent accuracy of the ConvNet classifier + ConvNet augmenter model and the ResNet classifier + ConvNet augmenter model, relative to their respective baselines of using no data augmentations (ConvNet classifier + ConvNet augmenter is compared against training just a ConvNet classifier without data augmentations, while ResNet classifier + ConvNet augmenter is compared against training just a ResNet classifier without data augmentations).

When training with 1 support example, both the ConvNet and ResNet classifiers benefited from augmented data, leading to improvements over having no augmentations. This may be because



(a) Baseline (b) Learned Augmentations

Figure 4: Accuracy Deltas against no Augmentation Baseline for Varying Augmentation factor

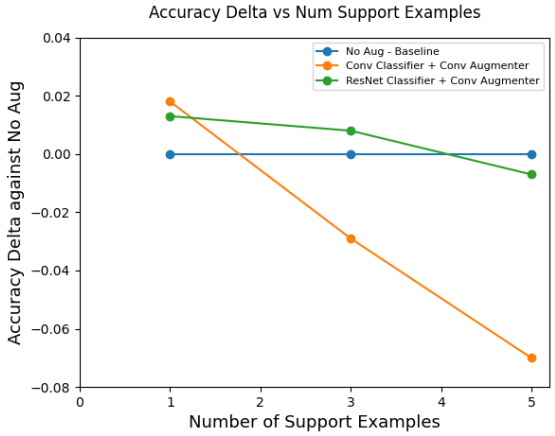


Figure 5: Effect of Augmentations on ResNet vs ConvNet Classifiers Across Support Set Sizes

a pre-trained model like a ResNet can already learn something from 1 support example, while a ConvNet would struggle to perform classification with just 1 support example per class and thus benefit more from addition augmented data. But upon increasing the support set size further, the ResNet classifier's performance improved more when tuned with augmented data than that of the ConvNet classifier. We suspect this is because when we increase the number of support examples per class, we are also proportionately increasing the number of augmented examples (one per training image), so it is easier to produce some augmented images harming our task performance. Since the ResNet classifier is pre-trained on ImageNet data, it is likely more robust to any bad augmentations produced and can still maintain relative performance.

5.5 Why Were Learned Encoder-Decoder Augmentations Not Always Useful?

Further analyzing our results in the context of our experimental setup, there are several reasons why the learned Encoder-Decoder augmentations may not have improved performance against our no-augmentation baseline. The first is simply model scale: these models were likely not complex enough. Our hypothesis rested on the idea of the image augmenters having seen enough labeled images to have a bias toward some image qualities being important to classification and others being spurious correlations. It is likely that this ability can only be acquired at a much greater scale. Additionally, our models were generally trained with a batch size of 4 due to computing constraints,

which may have been too small to learn effective gradient updates. The issue of small batch size may have amplified problems: upon manual inspection, there seems to be a lot of variance in the inherent task difficulty between sampled N-way classification tasks (e.g. distinguishing between a dog, bird, or lion is much easier than distinguishing between three kinds of birds), making our model performance much more stochastic. Furthermore, the MiniImagenet dataset, which consists of a wide variety of objects in different conditions, may have been too difficult a dataset for the size models were working with.

The initial goal of our project was to learn an augmenter that could transfer well across tasks in a zero-shot manner. However, given that the augmenter is learned in a manner similar to MAML, perhaps the ideal test setting for our work would be to few-shot adapt the augmenter at testing time before producing augmentations. Additionally, the general setup of learning an augmenter that can aid a classifier in better few-shot classification is probably more relevant for cases where we have a larger (and likely pretrained) classifier. While we tried a few experiments with using a ResNet-50 classifier instead of a simpler ConvNet classifier, we did not have the compute to experiment with several variations and do a thorough analysis of our methods on this setup.

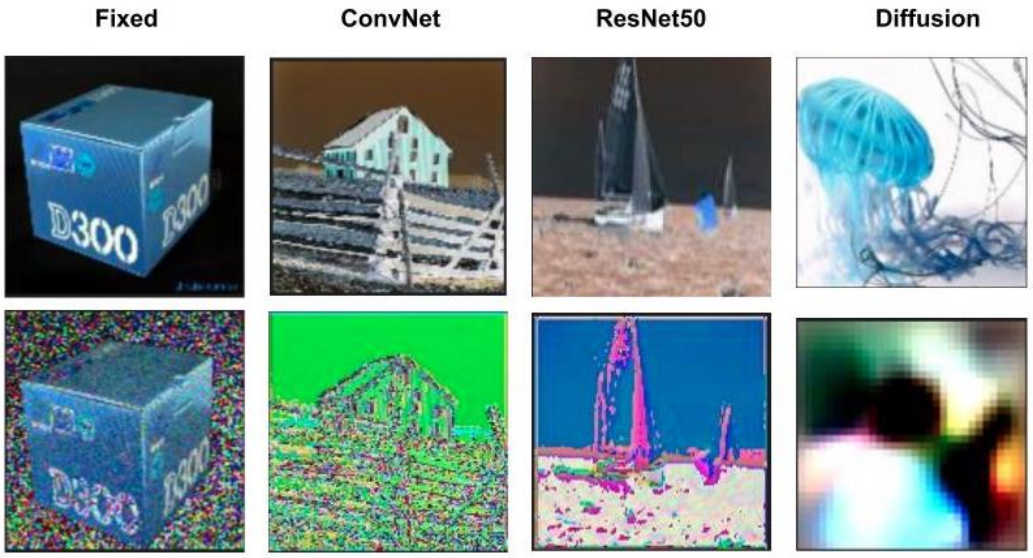


Figure 6: Example Original (Above) and Augmented (Below) Images for Each augmenter

5.6 Why was Image-Conditioned Diffusion Inconclusive?

Unfortunately, our image-conditioned diffusion framework was unable to generate synthetic data that was useful for our downstream classifier in all experimental settings we attempted. Due to the significant time taken to train and test the model in each setting, as well as the lack of sensational results, we only report quantitative results for varying the augmentation factor (Figure 4). We see that the generated images proved harmful to training. Additionally, observing the images generated by the model, we see that they do not appear to capture features unique to any class (Figure 6).

We suspect that freezing the diffusion model did not offer enough expressivity for the framework to generate useful examples. This issue was likely exacerbated by the fact that the Stable Diffusion weights we used were trained to generate 512 x 512 images, but using images of this size resulted in the model being too large for backpropagation (exceeding 37 GB VRAM). Thus, we had to reduce the image sizes generated to 128 by 128. Observing Stable Diffusion in inference mode using these image sizes (Figure 7), we see that the model is no longer able to generate images matching a text description, suggesting that the diffusion model weights need to be finetuned for this image size.

We additionally attempted to directly finetune our Stable Diffusion model implementation but found training to be very unstable. We ultimately did not have enough time to find hyperparameters which resulted in successful fine-tuning.

We became aware of DreamBooth, a technique for teaching new concepts to Stable Diffusion through specialized fine-tuning, late in the quarter. This technique allows the generation of novel images of a visual concept given only a small number of example images. We suspect incorporating this technique in our framework would be effective and hope to try this in the future.

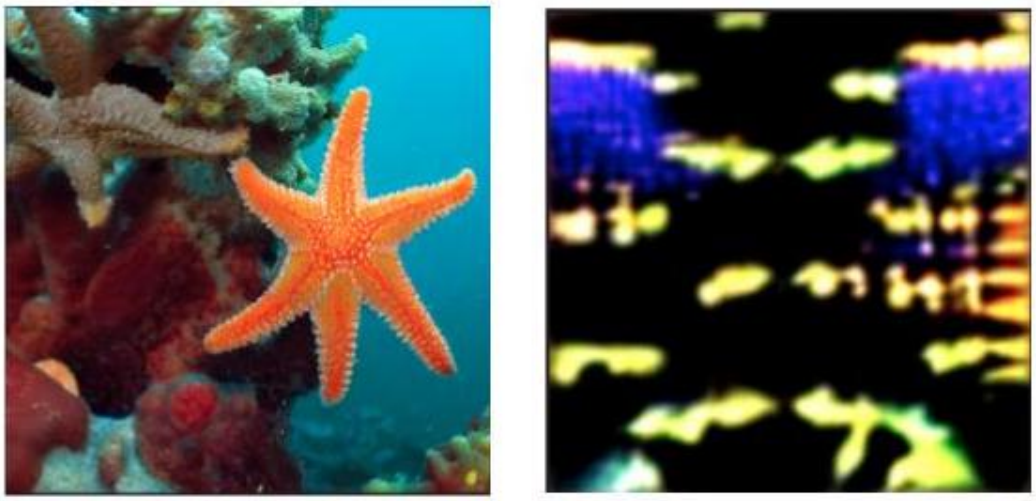


Figure 7: "Orange Starfish" Prompt to Stable Diffusion with 512 by 512 size argument (left) and 128 by 128 size argument (right). Clearly, this change caused the diffusion model to be unsuccessful at matching the text prompt.

6 Conclusion

We aimed to learn a deep neural network-based image augmenter that could map a small image support set to synthetic training images optimized for improving the performance of a downstream classifier in a zero-shot manner. We proposed a training recipe to achieve this and experimented with two augmentation architectures: an encoder-decoder and a soft-prompted latent diffusion model. The encoder-decoder augmentation architecture showed promising gains over the no augmentation baseline in small support set settings, as well as in cases using a large, pre-trained model for image classification, as compared to when learning a smaller classification model from scratch. However, in several other settings of varying task difficulty and number of augmentations produced, we found that neither approach surpassed the performance of our baselines. We suspect that the lack of success of the encoder-decoder augmenter in certain cases was primarily due to inadequate model complexity of the classifier and augmenter and an overly difficult dataset. The lack of success with the diffusion-based augmentations was primarily due to limited computational resources to finetune our diffusion models. We strongly recommend addressing these factors as next steps. We are excited by the promise of continuing this work with greater computational resources and model scale. We are particularly interested in adapting Dreambooth to our framework, and hope to explore this in the future.

7 Team Contributions

We detail the primary contributions of each team member below. These contributions varied greatly from our original intended roles. This was because our research goals were reprioritized over the course of our project, conditioned on the results of our investigations. So the experiments performed and how they were divided evolved accordingly.

1. **Rajan Vivek**: Rajan implemented the initial AutoAug codebase infrastructure to follow our training recipe, performed the initial experiments with our ResNet and ConvNet augmenters,

and led the diffusion-related efforts.

2. **Vaishnavi Shrivastava:** Vaishnavi implemented the initial ConvNet and ResNet classifiers and augmenters, and contributed to performing the ReNet and ConvNet augmenter/classifier experiments across different axes.

3. **Ofure Ebhomielen:** Ofure: Contributed to the randomly initialized ConvNet and the non-frozen ResNet augmenters and ran experiments on them. Also contributed to the fixed augmentations baseline.

References

Adrian Bulat and Georgios Tzimiropoulos. Language-aware soft prompting for vision & language foundation models. *arXiv preprint arXiv:2210.01115*, 2022.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 113–123, 2019.

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*, pp. 702–703, 2020.

Giorgio Giannone, Didrik Nielsen, and Ole Winther. Few-shot diffusion models. *arXiv preprint arXiv:2205.15463*, 2022.

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.

Ning Miao, Tom Rainforth, Emile Mathieu, Yann Dubois, Yee Whye Teh, Adam Foster, and Hyunjik Kim. Instance-specific augmentation: Capturing local invariances. *Arxiv*, 2022.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

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 (CVPR)*, pp. 10684–10695, June 2022a.

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, 2022b.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

Suorong Yang, Weikang Xiao, Mengcheng Zhang, Suhan Guo, Jian Zhao, and Furao Shen. Image data augmentation for deep learning: A survey. *arXiv preprint arXiv:2204.08610*, 2022.

Mengjie Zhao and Hinrich Schütze. Discrete and soft prompting for multilingual models. *arXiv preprint arXiv:2109.03630*, 2021.