Learning to Learn Quickly: Few-shot image classification

Akash Patil
Virginia Tech
akashdnyaneshwar@vt.edu

Rajesh Kudupudi
Virginia Tech
rajeshk19@vt.edu

Abstract

Many of the deep learning algorithms which achieved great success in solving supervised learning problems, rely on huge labeled data. The approach of training such models require large amount of labelled data in order to perform well, due to this such models are not suited for tasks with less labelled data. This motivates us to look into few-shot learning, which aims to learn quickly from a few data samples. Meta learning was proposed as a framework to solve tasks in few-shot learning settings. The basic idea behind meta learning is to train a model with a variety of tasks, such that the learned model can perform well on new tasks only with few data samples. Model Agnostic Meta Learning (MAML) is a representative meta learning algorithm which can be applied to any Neural Network architecture. In MAML a meta learner learns a good initialization of parameters such that the base learner can adapt to any new task with just a few examples. we attempt to improve the accuracy of MAML by incorporating in which additional data is generated by a Generative network. We claim this method due to availability of more data can learn better decision boundaries, which leads to better performance. We have compared the performance of MAML model and with our GAN based model. We have evaluated our model on image recognition problems, for 1 shot and 5 shot learning setting on Omniglot and Mini Imagenet datasets. Our results show that our proposed model outperforms standard MAML in terms of training accuracy.

1. Introduction

Deep neural networks have achieved great success in the recent years. However, many of the deep learning algorithms rely on huge labeled data. These methods are data hungry and not well suited for tasks with less data. Also these algorithms struggle when they need to adapt to new tasks with few samples. Low data regime tasks like problems in medicine, robotics does not have huge labelled datasets available. Humans on the other hand are able to learn new concepts quickly, given just a few examples. The reason for this performance gap between human and such algorithms can be explained as that humans can effectively utilize prior experiences and knowledge when learning a new task, while deep learning methods usually overfit on such tasks due to lack of necessary prior knowledge.

Meta-learning addresses this shortcoming of deep learning algorithms by training a particular adaptation strategy to a distribution of similar tasks, trying to extract transferable patterns useful for many other tasks. The method of Meta-learning is developed as a framework to address the challenging few-shot learning setting. The key idea in these models is to leverage a large number of similar few-shot tasks in order to learn how to adapt a base-learner to a new task for which only a few labeled samples are available. Many different meta-learning and few-shot learning algorithms have been proposed.

In our project we have focused on meta learning methods for supervised setting. We use a convolutional Neural Network architecture coupled with a generative network. We hypothesize that images generated by the generative network can help our classifier improve its performance. We have experimented with different architectures and provide a comprehensive overview of results.

2. Related Works

In [2] they have used adversarial networks for few-shot learning tasks. They have used a GAN for data augmentation. Their approach takes an K-shot N-way classifier and adds an addition category to it (fake image category). They use a normal GAN with vanilla loss, like our N+1 approach discussed later in this project. They completely share weights between classifier and discriminator. [2] were the first introduce multi task learning objective for classifier and discriminator. But unfortunately they do not comment on the quality of generated images. We discuss in detail about quality of our generated images. We use a CGAN as opposed to normal GAN employed by [2]. In [2] they did not share their code or explained there model architecture clearly so we are unable to reproduce their results. So instead we compare our algorithm to standard baseline MAML.

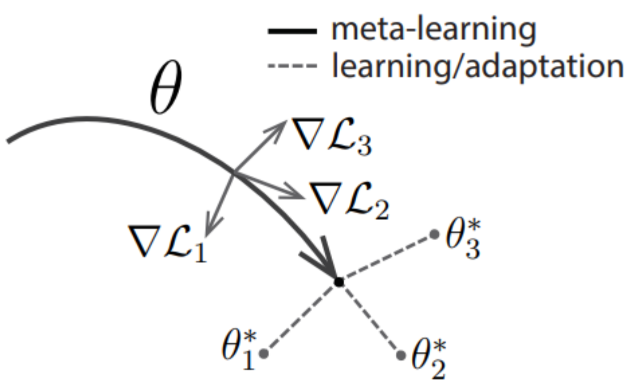

Figure 1. MAML update procedure

Algorithm MAML for Few-Shot Supervised Learning

```
Require: $p(\mathcal{T})$: distribution over tasks
Require: $\alpha$, $\beta$: step size hyperparameters
 1: randomly initialize $\theta$
 2: while not done do
 3:     Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
 4:     for all $\mathcal{T}_i$ do
 5:         Sample $K$ datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}_i$
 6:         Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ using $\mathcal{D}$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation (2)
            or (3)
 7:         Compute adapted parameters with gradient descent:
            $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
 8:         Sample datapoints $\mathcal{D}_i' = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}_i$ for the
            meta-update
 9:     end for
10:     Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$ using each $\mathcal{D}_i'$
        and $\mathcal{L}_{\mathcal{T}_i}$ in Equation 2 or 3
11: end while
```

Figure 2. MAML Pseudo code

[8] attempts to produce augmented version of images in the dataset using GAN. They then append these augmented images to original training datasets. The problem with [8] is that their model is not end to end. They again do not produce realistic images.

3. Technical Preview

Before diving deep into our algorithm there are a few background technical details which needs to be established. We will briefly introduce some of the details of our algorithm.

3.1. Model Agnostic meta learning (MAML)

MAML [1] excels at solving few shot problems. That is given only a few examples for a new task how to generalize to that task. MAML learns a good initialization of a model's parameters such that only a few gradient steps are required to adapt to a new task which has very less amount of data.

In the figure [1], θ are our generalized parameters and θ_i^* are optimized for specific tasks. We can see from the figure through meta updates we can get good generalized parameters from where using a few we can reach good generalized task specific parameters. MAML is often applied in the context of K-shot N-way classification problems, where our goal is to learn a classifier which can generalize to a new task given only K training examples for each of N class. MAML can be applied to any supervised learning problem and it works well with any architecture as it is model agnostic. MAML generalizes to new images by structuring the learning task in a way that seperates optimization into 2 steps: Outer loop and Inner loop Figure [2]. In inner loop say we are provided with few images one for each kind of class from this classifier has to predict how to predict the next images test images. In outer loop parameters are generalized between different tasks in inner loop so that classifier essentially learns to look at new test of few images and starts to predict new images. We use a GAN along with MAML to augment our images.

3.2. Conditional Generative Adversarial Networks

In Generative adversarial networks, there is no control over modes of the data to be generated. Conditional GANs [6] are a type of generative adversarial networks in which class conditioned vectors are appended to the input of generator and discriminator. In its core, CGANs train on a labeled data set and let you specify the label for each generated instance. For example, an unconditional MNIST GAN would produce random digits, while a conditional MNIST GAN would let you specify which digit the GAN should generate. We learned if mode collapse is avoided then CGAN tends to produce better realistic looking images. This ability of generating class conditioned images is very useful for few shot image generation.

In CGAN, we use conditional probability P(X / Y) for modeling, instead of the joint probability P(X, Y). Also, in CGAN labels act as an extension to the latent space z to generate and discriminate images better. These labels may give a significant head start to GAN for what to look for. The cost function for CGAN is the same as GAN.

3.3. Wasserstein Loss

Wasserstein loss [4] improves the stability of GAN model and provides a loss function that directly correlates with the quality of the image. We use this as opposed to binary cross entropy (BCE) loss to train the GAN. The main problem with BCE loss was, when the discriminator (whose values are restricted between 0 and 1) predicts a value close to 0 or 1 causes vanishing gradient problem for the generator. In Wasserstein loss the outputs of the discriminator (author refers to this as critic) has no restrictions on the values it can take. Wasserstein loss uses earth mover distance which

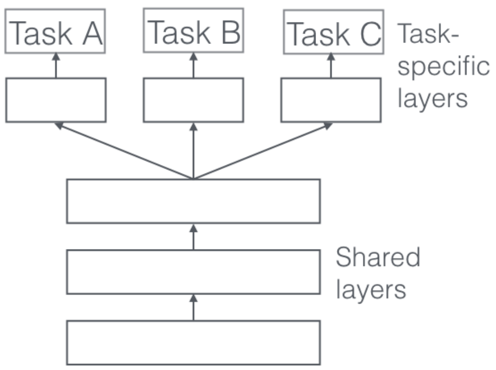

Figure 3. Example of neural network architecture for multi task learning which some features shared between different tasks

measure the distance between data generated by generated examples and training dataset distribution. This change is motivated by argument that training the generator must seek to minimize the distance between distribution of data observed to distribution and distribution of generated samples. The Wasserstein loss is given by:

$$\text{Critic loss} : E[C(x)] - E[C(G(z))]$$

$$\text{Generator loss} : E[C(G(z))]$$

Where C is the critic (discriminator), G is the generator, x is the real image and G(z) the generated image. Wasserstein loss should be used with weight clipping or gradient penalty. In this project we used weight clipping.

3.4. Multitask Learning

Multitask learning is a paradigm in machine learning which aims to leverage useful information across multiple related tasks to improve the generalized performance on any given task. Multitask learning exploits similarities across different tasks. Deep neural networks which perform multitask learning often share features between tasks. In [4] it has been shown that in many cases, multi task learning or feature sharing improves the performance in comparison to training on each task separately. A pictorial version of multitask learning architecture in Neural networks is shown in Fig[1].

Intuitively multi task learning is a way of taking experience from different tasks to do well on any given task, just like humans. Some also treat multitask learning as a form of regularization. Based on these insights we choose to do a multitask version, in which we share weights between our Neural network which performs classification and discrimination tasks. We will discuss more about this later sections of this paper.

Figure 4. Samples from : Left Omniglot, Right Mini-imagenet dataset

4. Datasets

Omniglot and Mini-Imagenet are the two benchmark datasets for few-shot learning tasks. We trained and evaluated our model on these datasets.

4.1. Omniglot

The Omniglot data set is created to develop few-shot learning algorithms. The dataset consists of handwritten character images from 50 languages. There are 1623 classes of characters with 20 examples within each class. For training and conducting experiments on our model we downsampled all images to 28 × 28 and randomly split the dataset into 1200 classes for training and 432 classes for testing. We also performed some data augmentation techniques on the image data.

4.2. Miniimagenet dataset

The Mini-Imagenet dataset is a revised subset of the well-known ImageNet dataset. It consists of 84 × 84 colored images with 100 classes and 600 random samples in each class. For our project we used the class split of 64, 16, and 20 for training, validation, and testing. Due to the complexity and size of the Miniimagenet dataset we mainly used Omniglot for our experiments.

5. Problem Statement

In our problem, given a new task T_i in which we have to predict the output the label Y given an input image X. For each task, we are given a few examples K pertaining to each class. We meta-train on many sets of labelled classification tasks to come up with good generalized parameters θ, (meta parameters) such that given a new task test time T_{test} we can learn a good mapping from $f : X_{test} \rightarrow Y_{test}$. (For every task meta training and test set split is 60:40). So, in meta-testing the neural network is given a new task with only a few examples per class, network should be able to optimize its meta-parameters θ such that it does well on the new task.

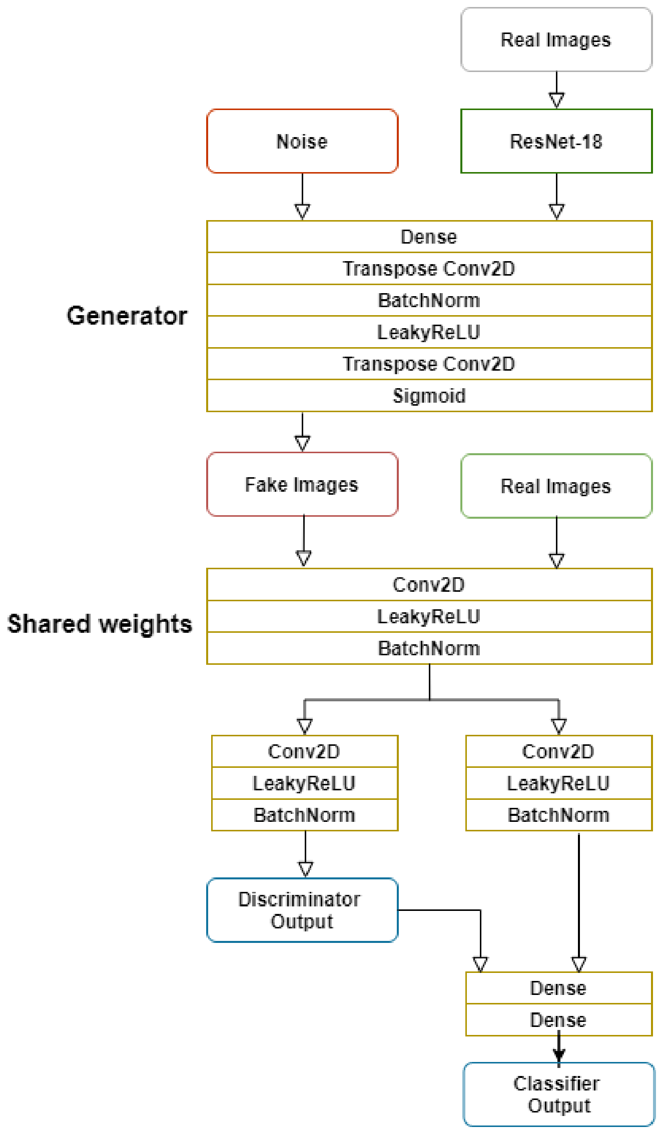

Figure 5. Our model Architecture

6. Methods

As we can see from the problem statement the main hurdle here is how do we reliably classify new examples of a completely new class given only a few images to train on. We claim that if we can augment these images, our algorithm will be able to perform better. That is by training on more images, our classifier can learn a better and sharper decision boundary [2]. But simple data augmentation techniques like translation, rotation do not work as neural networks easily ends-up over fitting them. So we propose a GAN based architecture which can generate new images which can be used during training time. So we propose 2 different architectures of our model and analyze each one of them. One partially shares weights for discriminator and classification networks and another completely shares all the weights between them. The latter was inspired from [2], where the combined classifier and discriminator network has N+1 units, N here corresponds to each class (classification logits) and the one for whether the image is real or fake. We experimented with different loss functions, inner and out loop update steps learning rates as well as generator architectures. The final architecture is given in the figure[5]. We will be expanding on the choice and effects of this architecture below.

6.1. Generator

We have experimented with many different types of conditioning for generator and finally settled on the one shown in Figure [5]. We tried to learn the latent embedding by a seperate untrained CNN for real images also, but later

```
Algorithm 1: Few shot image classification with GAN
Inputs: T number of classification tasks;
$\alpha$ : inner learning rate, $\beta$ : meta learning rate;
$f_\theta$: multi-task classifier N-way, real and fake discriminator
$G_\phi$ The Generator
while not done do:
    Sample a classification Task $T_i$
    for each of $T_i$ do:
        Sample K images $x^j, y^j$ from $T_i$
        Set $\phi^{'}, \theta^{'} = \phi, \theta$
        for all inner gradient steps do:
            From $G$ generate M fake examples using $D_i = x^k, y^k$
            Using fake examples along with $D_i$
            $\theta^{'}_{sh} \leftarrow \alpha \nabla_{\theta^{'}_{sh}} Loss_{sh}(f^{'}_{\theta})$
            $\theta^{'}_{classf} \leftarrow \alpha \nabla_{\theta^{'}_{classf}} Loss_{classf}(f^{'}_{\theta})$
            $\theta^{'}_{disc} \leftarrow \alpha \nabla_{\theta^{'}_{disc}} Loss_{disc}(f^{'}_{\theta})$
            $\phi^{'} \leftarrow \alpha \nabla_{\phi^{'}} Loss_{gen}(G^{'}_{\phi})$
        end
        Sample new images for meta updates $D_i = x^j, yj$ from $T_i$.
    end
    Update $\theta \leftarrow \beta \nabla_\theta \sum_{T_i} Loss_{classf}(f^{'}_{\theta i})$ using $D^{'}_i$
    $\phi \leftarrow \beta \nabla_\phi \sum_{T_i} Loss_{classf}(G_{\phi^{'}_i})$ using $D^{'}_i$
end
```

Figure 6. Our Algorithm Pseudo code.

dropped it because it generated inferior images. In Figure [5] we condition our generator on latent embeddings passed by pretrained ResNet-18 for every real image. This architecture has given us realistic looking images.

6.2. Discriminator and Classifier

Our discriminator is tasked with predicting whether our image is real or fake and classifier is tasked to predicting what class a given image falls into. In our model architecture, classifier and discriminator have a few shared layers. In our model pipeline real and generated images would input to our combined network (shared network), and feature vectors would be passed to downstream N-way classifier and discriminator. We saw this as a natural way to incorporate generator network in MAML without disturbing the normal training procedure. We experimented with different layer sharing and from complete to partial. We hypothesized that discrimination and classification tasks have some similarities between them. Both of these tasks require basic understanding of the input image. So we shared some layers between these tasks. This would enable the network to learn good features for both the tasks. So this part of the network which is shared between 2 tasks (enabling multitask learning for those layers). During training we appended some of the samples generated by our GAN to our training data set. After sharing some layers, classifier and discriminator output their individual predictions (discriminator tells whether the image is real or fake and classifier which class the image belongs to) of the images and their individual losses are then computed. We used cross entropy loss for the classifier and Wasserstein loss for discriminator and generator. These losses are used to update our parameters. The parameters in shared layers would be affected by gradients from both classifier loss and discriminator loss. During the meta update (outer loop) phase we calculated classification loss on real test examples and unroll it to update the parameters in discriminator classifier and generator. We hypothesized that doing this would meta-optimize our generator such that it aids the classifier. This method gave us better generated images, which was very important for our classification. During final testing we can ignore the discriminator and our classifier would consist of shared layer and N-way classifier.

6.3. Loss functions

Instead of Vanialla gan loss we used Wasserstien GAN loss (or earth mover distance) for our discriminator and generator inside the inner loop. We used this loss in combination with gradient clipping. As discussed before Wasserstien GAN loss imporves the convergence of generator and it avoids mode collapse. During the meta training updates (outer loop) we used cross entropy loss. This choice of loss functions was fruitful as it led to better quality of generated images.

6.4. N+1 Model Architectural Exploration

We have explored different architectures for our classifier and discriminator as mentioned earlier. One of them is sharing the classification and discrimination networks completely, and this combined network outputs N+1 units (N corresponding to each class and the other corresponding to whether the given image is real or fake).

7. Contributions

- Rajesh implemented the CGAN with MAML while Akash helped in implementing N+1 classifier.
- We add a Generative Adversarial Network (GAN) to the meta learning framework.
- We propose an end to end framework for data augmentation with GAN and few shot image classification.
- We experiment with CGANs and MAML.
- We comment on about the quality of our generated images on performance of the classifier as opposed to [2].
- Evaluate on image classification tasks, for 1-shot learning (Omniglot).

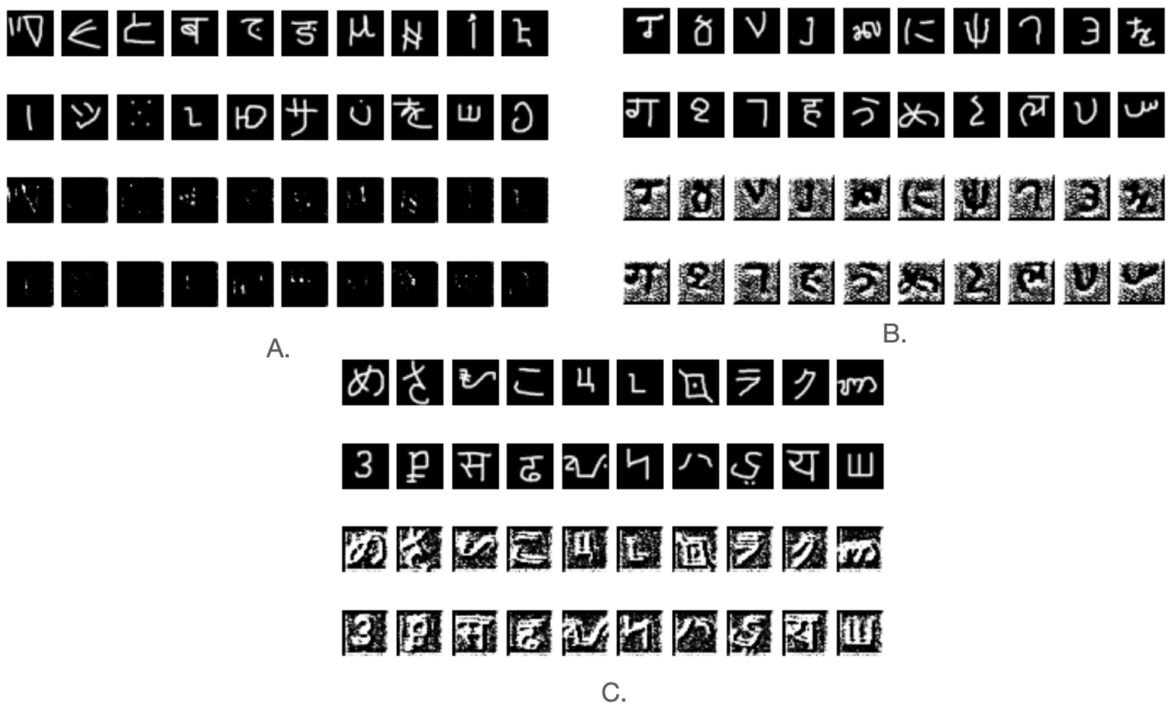

Figure 7. First 2 rows in every image are real images and rows 3, 4 are generated images. A: Generated Images after 500 steps. B:Generated images after 7000 steps. C: Generated images after 12500 steps. Generated image quality keeps improving as we train the our model.

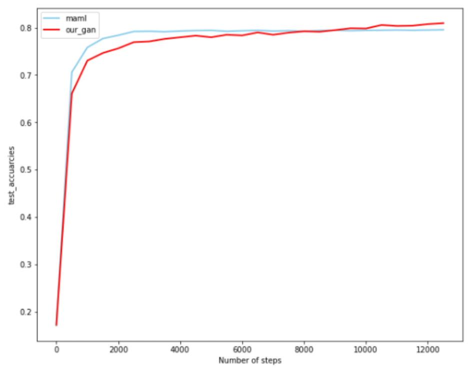

Figure 8. A Comparision between our model and standard maml model, test accuracy on Omniglot-20 way 1 shot

8. Results and Analysis

All of the experiments were run on Omniglot dataset-20 way 1 shot (20 classes, 1 image per class). We compared the performance of our model with standard MAML model. We can see from Figure[8] that our model converges faster and achieves better accuracy. We can also see that as the algorithm kept training the quality of generated images keeps getting better. We found that the classification accuracy was affected by the quality of generated images. That is the reason as the generator generates realistic looking images our classification accuracy also keeps getting better. So much so that it overtakes standard MAML model. Out of the many different architectures we explored the one in Figure[5] was working best.

On the other hand N+1 approach performs much worse than our normal model Figure [9]. The quality of generated images on almost all the steps is very bad for this N+1 model. In-spite of using Wasserstien loss to encourage the convergence of generator, and meta updating discriminator and generator the bad quality of generated images was negatively effecting the classifier Figure [10]. We hypothesized that this might be because, as discriminator and generator share all the parameters some of the gradients of classifier might be negatively effecting the discriminator and vice versa. This would happen when classifier and discriminator compete for same set of resources (parameters here). Our earlier model [5] has many layers which are decoupled between classifier and discriminator. So we conclude that completely sharing the parameters between classifier and discriminator does not yield fruitful results. Thus we had to stop working on this model and look into the model in Figure[10].
We tried to compare the performance of our model on mini-imagenet also which has RGB images as opposed to grayscale images. But the generated images were not realistic. In general we feel that generative adversarial networks are extremely difficult to train. Even when we have 25 im-

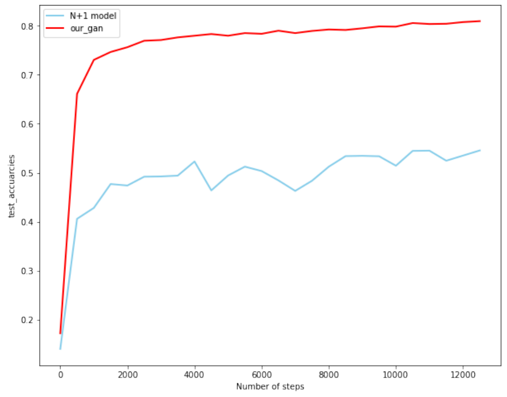

Figure 9. A Comparison between our model and n+1 approach, test accuracy on Omniglot-20 way 1 shot

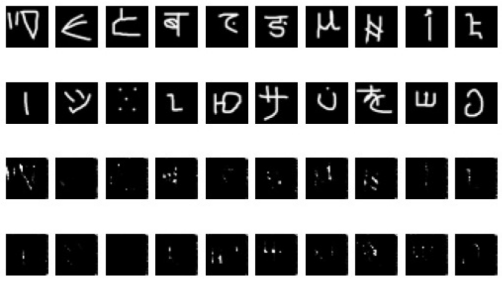

Figure 10. First 2 rows are real images, the next 2 rows are images generated by N+1 model. We can observe that quality of generated images is bad even after 8000 steps.

ages per each class, producing realistic images is still difficult in many cases. But when we are given only one image per class to train on and asked to generate realistic image for that class the task becomes even more difficult. We feel that quality of the images generated by our model on Omniglot dataset is decent this might be because of the simpler nature of images in Omniglot. But when we want to extend this to more complex realistic images (like mini-imagenet) then we need to look into state of the art few shot generators and then adapt them to few shot classification task. This can be a future extension for our project. We are extremely satisfied with our results as they outperform standard MAML model.

9. Conclusion

In this work we have explored a meta leaning framework for few-shot learning classification task. We implemented a generative network which assists the classifier by augmenting the images. We are the first to experiment with a conditional GAN coupled with MAML. We have compared the results of our model with the MAML model which is a representative meta learning model. Our model was able to achieve better performance than standard MAML. This augmentation of dataset helped our model to have better decision boundaries for this task. We have experimented with two different architectures, and found that that former architecture which shares the partially weights between generator and discriminator performs much better than N+1 model which completely shares weights. We found that the quality of generated images by our model was decent and kept increasing as we trained it. It had a significant impact on the performance of our model.

In future we hope to experiment with a model in which shares no weights and explore other few shot data generation algorithm as the state of the art models evolve.

References

[1] Finn, C., Abbeel, P., Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. 34th International Conference on Machine Learning, ICML 2017, 3.

[2] Zhang, R., Che, T., Bengio, Y., Ghahramani, Z., Song, Y. (2018). Metagan: An adversarial approach to few-shot learning. Advances in Neural Information Processing Systems, 2018-December(NeurIPS), 2365–2374.

[3] https://lilianweng.github.io/lil-log/2018/11/30/meta-learning.html

[4] https://jonathan-hui.medium.com/gan-wasserstein-gan-wgan-gp-6a1a2aa1b490

[5] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., … Bengio, Y. (2014). Generative adversarial nets. Advances in Neural Information Processing Systems, 3(January). https://doi.org/10.3156/jsoft.29.5.177.2

[6] Isola, P., Zhu, J. Y., Zhou, T., Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-January. https://doi.org/10.1109/CVPR.2017.632

[7] Wang, Y. X., Girshick, R., Hebert, M., Hariharan, B. (2018). Low-Shot Learning from Imaginary Data. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. https://doi.org/10.1109/CVPR.2018.00760