Meta-GAN: Mobile Adaptation of Few-Shot Generative Adversarial Networks

Steven Song Sungboo Yoon Kyuwook Chai
Seoul National University
{steve2972, yoonsb24, kwchai} @snu.ac.kr

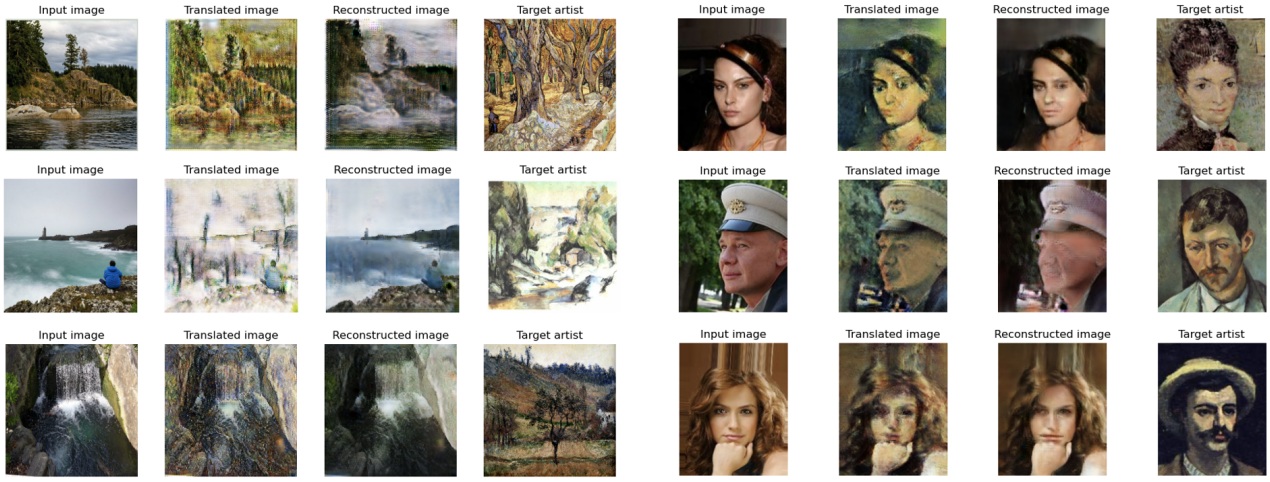

Figure 1: Given an input image domain X, a few-shot target domain Y and good initial model parameters θ, our algorithm learns to automatically translate an image from one into the other with only a few optimization steps (e.g. five gradient descent steps): (*top*) landscape photos from Kaggle and celebrity images from Celeb-A translated to a "Gogh" target image domain; (*center*) photos translated to a "Cezanne" target image domain; (*bottom*) photos translated into the style of "Monet".

Abstract

Although generative adversarial networks (GAN) have gained much attention in the research community, its use in mobile applications remain limited because training a GAN model to learn new tasks generally requires significant computing power and a large training dataset.

In this work, we propose `Meta-GAN`*, an algorithm that implements ideas from state-of-the-art meta-learning methods into the GAN training process. By utilizing meta learning, our algorithm allows a mobile version of GAN that meta-learns from a very small number of input images to synthesize images based on new tasks in a computationally efficient manner.* `Meta-GAN` *has a light meta-test step, which enables a model to learn a new task with resource-constrained mobile devices. We empirically demonstrate the learning performance of our algorithm on a mobile device using new datasets we built for few-shot image translation models,* ***Meta-Landscape*** *and* ***Meta-Portrait****, and quantitatively show that our model is comparable with state-of-the-art image translation models.*

1. Introduction

Since its conception in 2014, Generative Adversarial Network (GAN) [10] has gained much interest in the research community for its diverse applications. Due to its advanced image synthesis power, various GAN models are being developed to be widely used in real life applications such as photo blending [42], super high resolution image production [24], style transfer [17, 6], face aging [14], or expanding image datasets [20]. Furthermore, current research has improved GAN performance through novel architectures[18], loss functions[2], or training methods[30]. Nonetheless, to guarantee promising performance, training a GAN model requires state-of-the-art servers having large

computing power.

Meanwhile, applications for mobile devices, such as smartphones, are thriving more than ever. In particular, mobile devices, such as smartphones, tablets, and wearable devices, have increasingly become the primary computing device for most people [31]. Although recent advances in mobile technology made mobile phones more powerful, compared to servers, mobile devices are resource-constrained, lacking computational power and parallel computing ability utilizing GPUs. This limitation makes AI models installed in mobile devices difficult to learn new tasks. GAN models, requiring heavy computation for learning, cannot be free from this problem.

In this case, methods that enable models installed in mobile devices to learn new tasks without the help of a server would massively improve the flexibility of mobile applications.

Recent studies attempt to improve the learning efficiency of GAN models through few-shot learning. One approach is applying singular value decomposition (SVD) to the network weights of a pretrained generator and discriminator, and changing only the singular values for optimizing the weights to a new task in few-shots (providing less than 100 images) [36]. However, the algorithm in [36] is very slow, taking around 1 minute to learn how to translate an image to a previously unseen style, and generally requires more than five images (e.g. 5-100 images) for optimizing the parameters to a new task.

Another approach to improve learning efficiency is demonstrated by meta-learning. Meta-learning methods such as MAML [8] and Reptile [33] have been applied in supervised and reinforcement learning. A recent research adapts Reptile to GAN for generating few-shot gray-scale line images [7]. Though [7] shows successful results, the few-shot learning is limited to relatively simple image generation tasks such as drawing a digit or letter that was unseen during meta-training. For example, if the model is trained with digits of 0-8 in **MNIST** [23] during meta-training, the new task given to the model will be generating a monotone image of digit 9.

In short, previous studies have not shown an efficient few-shot learning algorithm that can meta-train a GAN model which is useful or practical enough to be used in real-life mobile device applications such as image style translation. Moreover, the studies have not shown that a GAN model installed in a mobile device can learn a new task without borrowing the computational resources of a server.

In this paper, we propose `Meta-GAN`, an algorithm that meta-learns from a very small number of input images (less than 5 images per domain) to synthesize images based on new tasks in a computationally efficient manner. `Meta-GAN` adopts Reptile [33] for meta-learning image translation, making the model efficient and lightweight so that they can learn new image synthesis tasks on a standard mobile device within a few seconds.

We use our algorithm and train a GAN model that is likely to be used in mobile applications–a model that translates photos to Impressionism style images *(cf. We also trained an image generating GAN model. For further details, please refer to Appendix A)*. In order train these tasks, we build new datasets **Meta-Landscape** and **Meta-Portrait** for training and evaluating few-shot image translation models. We compare the performance of a model that learned a new task (e.g. translate photo to Cezanne style image) after being trained with `Meta-GAN` and a model trained with the same task (e.g. translate photo to Cezanne style image) using `CycleGAN` [46]. Finally, we implement our method in an NVIDIA Jetson Nano (a standard mobile device), train a new task to the model using only the resources of the mobile device, and compare the results with learning a new task on a server.

Thus, the main contributions of our works are:

- We propose `Meta-GAN`, an efficient meta-learning algorithm for image translation GAN models.
- We quantitatively demonstrate that the performance of models that learned a new task after being trained with our algorithms are comparable with models trained with `CycleGAN` [46].
- We prove that model trained with our algorithm can learn a new task in a mobile device within a few seconds.
- We propose **Meta-Landscape** and **Meta-Portrait**, new datasets for training and evaluating few-shot image translation models.

2. Related Work

2.1. Generative Adversarial Networks (GANs)

GANs are generative models that learn to map random noise vector z to the distribution of an image dataset, allowing for sampling of novel images.

The training objective of the generator G is to make the divergence $Div(\cdot)$ between two probability distributions $P_G(x)$ and $P_{data}(x)$ as small as possible. The optimization formulation can be defined as:

$$G^* = \arg\min_G Div(P_G(x), P_{data}(x)) \tag{1}$$

Meanwhile, the discriminator D estimates the probability that a sample comes from the training data rather than G. The objective function can be formulated as:

$$D^* = \arg\max_D V(G, D) \tag{2}$$

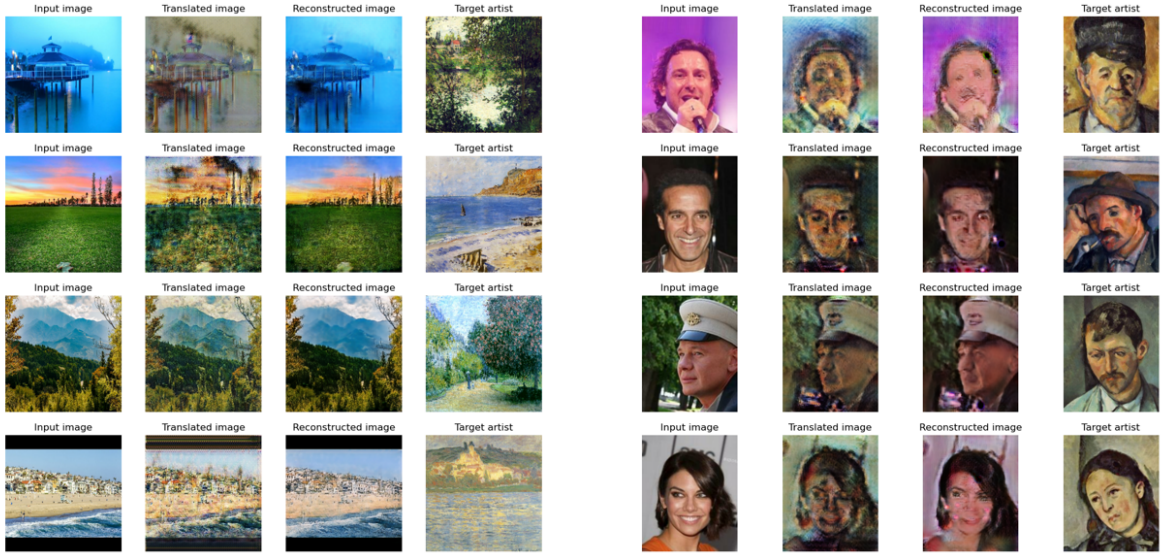

Figure 2: Examples of image translation tasks between two types of domains. (*left*) shows a photo → landscape task, with the meta-trained model optimizing on paintings created by Monet. (*right*) shows a photo → portrait task, with the meta-trained model optimizing on portrait paintings created by Gogh.

where $V(G, D)$ is defined as:

$$V(G,D) = \mathbb{E}_{x\sim P_{data}}[\log D(x)] + \mathbb{E}_{x\sim P_G}[\log(1 - D(x))] \tag{3}$$

Thus, the optimization formulation of G can be converted to:

$$G^* = \arg\min_G \max_D V(G,D) \tag{4}$$

In sum, GANs optimize a competitive objective where G maximizes the classification error of D trained to distinguish "real" images from "fake" images.

Recently, various adversarial losses have been proposed to stabilize the training or improve the convergence of the models. Neverthless, GANs still suffer from a lack of high-quality data. Training the GAN models on limited data is challenging, because the data scarcity leads to the problems such as unstable training dynamics, degraded fidelity of the generated images, and memorization of the training examples [38]. Lack of diversity is another issue. Generated images by GANs lack diversity even when large training sets are used, because the objective does not penalize the absence of outlier modes [34].

2.2. Meta-learning

In the research trend towards few-shot learning, meta-learning algorithms have shown promising results in few-shot learning and are widely adopted in creating GANs. The main idea of meta-learning is *learning to learn*. In other words, meta-learning aims to design an algorithm that trains a model to quickly adapt to new *task* [1]. This goal is achieved by improving outer learning algorithm over multiple inner learning episodes, sampled from a task family, and finally leads to a inner learning algorithm that performs well on new tasks sampled from this family [13]. In conventional supervised machine learning, the objective of the training process is to find parameters θ^*, given a training dataset $\mathcal{D} = \{(x_1, y_1), ..., (x_N, y_N)\}$ and a loss function $\mathcal{L}$:

$$\theta^* = \arg\min_\theta \mathcal{L}(\mathcal{D}; \theta, \omega) \tag{5}$$

The conditioning on ω denotes the dependence of this solution on assumptions about *how to learn*, such as the choice of optimizer for θ or function class for f. However in meta-learning, the training step of *learning to learn* is implemented to seek meta-knowledge ω^*, given a set of M source tasks $\mathcal{D}_{source} = \{(\mathcal{D}^{train}_{source}, \mathcal{D}^{val}_{source})^{(i)}\}_{i=1}^{M}$:

$$\omega^* = \arg\max_\omega \log p(w|\mathcal{D}_{source}) \tag{6}$$

In the meta-testing stage, the learned ω^* is used to train the base model on each new target task i, given a set of Q target tasks $\mathcal{D}_{target} = \{(\mathcal{D}^{train}_{target}, \mathcal{D}^{val)^{(i)}}_{target}\}_{i=1}^{Q}$:

$$\theta^{*^{(i)}} = \arg\max_\theta \log p(\theta|\omega^*, \mathcal{D}^{train^{(i)}}_{target}) \tag{7}$$

The accuracy of the meta-learner is evaluated by the performance of $\theta^{*^{(i)}}$ on the test split of each target task $\mathcal{D}^{test^{(i)}}_{target}$.

Few-shot image-to-image translation Generally, image-to-image translation can be categorized into two groups by learning methods: a supervised method [15, 45, 29], which uses a pairs of source and target domains, and an unsupervised method [19, 46, 9], which is an unpaired setting. Our work, as well as `CycleGAN`, is based on unsupervised image-to-image translation task.

We extend the range of tasks for mobile adapted GAN from the image generation to the image-to-image translation tasks. Compared to few-shot image generation, the task of learning image-to-image translator using few training examples is relatively newly-studied problem. Several recent works for few-shot image-to-image translation include using domain-specific feature distribution conditioned on domain attributes [27], using one image of the target domain as an exemplar to guide image translation [5], and applying semi-supervised learning via a noise-tolerant pseudo-labeling procedure [40].

Regarding few-shot image-to-image translation tasks, previous approaches have two main limitations:

1. Validation takes 5-100 images for image-to-image translation tasks, yet still suffer from memorization overload and instability, leading to a lack of diversity or poor visual quality, as well as limited applications for mobile adaption.

2. Learning new tasks is computationally and memory expensive and requires a large number of training steps. Because Mobile GAN requires a more lightweight model, our approach minimizes the required training steps and their overhead by implementing the inner loops with a Reptile algorithm.

3. Methodology

A key limitation in [7] is that the proposed algorithm disregards second-order optimizations and estimates the meta gradients by using an approximation technique proposed by [33]. Because this estimation negatively affects personalized accuracy in simple image classification tasks, we believe that approximating the meta-gradient can also affect a GAN model's ability to generalize to new tasks.

We propose a method of training the GAN model such that it can generalize to many new tasks with a high degree of precision by integrating ideas from state-of-the-art meta learning algorithms.

As the goal of meta learning is to design an algorithm that trains a model to quickly adapt to new tasks [1], the meta training objective first requires the models to be parametrized such that θ_G and θ_D represent easily adaptable parameters for the generator model and the discriminator model respectively. The model parameters are then updated from θ_{t-1} to θ_t at each round t through a meta-training process containing many different tasks.

Image-to-image translation In the image translation task, each task $\tau \sim P(T)$ represents a specific translation task such that there are two image domains X_τ and Y_τ. In this case, our goal is to learn a mapping $F : X_\tau \rightarrow Y_\tau$ that is derived from θ_G. Since there is no explicitly paired data during training, we observe there is naturally a secondary task which learns another mapping $H : Y\tau \rightarrow X_\tau$ in the reverse direction. Thus, we use these two translation tasks as a two-agent game for fine-tuning during the meta-training process. Specifically, two discriminators D_Y and D_X are used to determine the validity of the generated images for

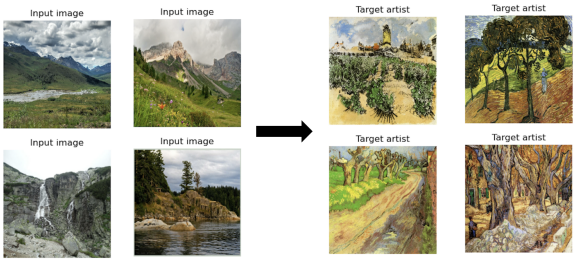

Figure 3: An example of a few-shot training task. In this case, photos from a "real-image" input domain are translated to a "Gogh" painting style given only four instances in the target domain.

Algorithm 1 Meta-GAN (image translation task)

Require: Hyperparameters α, β, λ_{cyc}, λ_{idt}
Initialize generator parameters $\theta_{G_{X\rightarrow Y},0}$, $\theta_{G_{Y\rightarrow X},0}$
Initialize discriminator parameters $\theta_{D_X,0}$, $\theta_{D_Y,0}$
for each round $t = 1, 2, ...$ **do**
    $S_t \leftarrow$ Random sample of m tasks ($1 \leq m \leq K$)
    for each task $\tau \in S_t$ **in parallel do**
        $\phi_{G,\tau}, \phi_{D,\tau} \leftarrow$ **InnerLoop**($\tau, \theta_{G,t-1}, \theta_{D,t-1}$)
        Calculate $v_{G,\tau}$, $v_{D,\tau}$ with Eqn. (10)
    end for
    $\theta_{G,t} \leftarrow \theta_{G,t-1} - \frac{\alpha}{\|S_t\|} \sum_{\tau \in S_t} v_{G,\tau}$
    $\theta_{D,t} \leftarrow \theta_{D,t-1} - \frac{\alpha}{\|S_t\|} \sum_{\tau \in S_t} v_{D,\tau}$
end for

InnerLoop$(\tau, \theta_G, \theta_D)$**:**
    $\phi_{G_{X\rightarrow Y},0}, \phi_{G_{Y\rightarrow X},0} \leftarrow \theta_G$
    $\phi_{D_X,0}, \phi_{D_Y,0} \leftarrow \theta_D$
    for each local epoch i from 1 to E **do**
        Generate fake images $Y' = \phi_{G_{X\rightarrow Y},i-1}(X_\tau)$
        Generate cycled images $\phi_{G_{Y\rightarrow X},i-1}(Y')$
        Generate identity images using real images X_τ and Y_τ
        Calculate real/fake discriminator outputs for all generated images
        Calculate $\mathcal{L}_\tau(\phi_G, \phi_D, \tau)$ with Eqn. (9)
        $\phi_{D_X,i} \leftarrow \phi_{D_X,i-1} - \beta \nabla_{\phi_{D_X,i-1}} \mathcal{L}_{X_\tau}(\cdot)$
        $\phi_{D_Y,i} \leftarrow \phi_{D_Y,i-1} - \beta \nabla_{\phi_{D_Y,i-1}} \mathcal{L}_{Y_\tau}(\cdot)$
        $\phi_{G_{X\rightarrow Y},i} \leftarrow \phi_{G,i-1} - \beta \nabla_{\phi_{G,i-1}} \mathcal{L}_{Y_\tau}(\cdot)$
        $\phi_{G_{Y\rightarrow X},i} \leftarrow \phi_{G,i-1} - \beta \nabla_{\phi_{G,i-1}} \mathcal{L}_{X_\tau}(\cdot)$
    end for
    Return $\phi_{k,E}$ to server

both tasks.

Meta-gradient computation We can then calculate the meta gradients for both the generator and the discriminator using these optimized parameters, resulting in $v_{G,\tau}$ and $v_{D,\tau}$. For every task τ there exists optimal discriminator and generator weights $\phi_{G,\tau}$ and $\phi_{D,\tau}$. Because the meta gradients are trained on multiple tasks, our model parameters θ_G and θ_D are trained to lie in a parameter space that minimizes the distance between θ_G, θ_D and $\phi_{G,\tau}$, $\phi_{D,\tau}$ for the generator model and the discriminator model respectively. Mathematically, this can be written as the following.

$$minimize \sum_{\tau} (\phi_D - \theta_D) + (\phi_G - \theta_G) \tag{8}$$

Specifically, in the image translation task, we solve the above equation such that we minimize the overall objective $\mathcal{L}_\tau$ for training the discriminator and generators where

$$\begin{aligned}\mathcal{L}_\tau(G, D, X_\tau, Y_\tau) = \mathcal{L}_{adv}(G, D, X_\tau, Y_\tau) \\ + \mathcal{L}_{adv}(wc(G), wc(D), Y_\tau, X_\tau) \\ + \lambda_{cyc}\mathcal{L}_{cyc}(G, wc(G), X_\tau, Y_\tau) \\ + \lambda_{idt}\mathcal{L}_{idt}(G, wc(G), X_\tau, Y_\tau)\end{aligned} \tag{9}$$

This can be optimized by minimizing the meta-task loss $\mathcal{L}$ upon the original task parameters θ such that we solve for $\nabla_\theta \mathcal{L}(\cdot)$. Whereas it is possible to calculate the gradient of $\mathcal{L}(G, D, X_\tau, Y_\tau)$ with respect to θ as in [8], this will entail very heavy computation and memory costs when propagating through the history of optimized ϕ in order to calculate the exact Hessian. Thus, this method is sub-optimal in a mobile environment with limited computation and memory resources. In other words, because training a GAN is computationally demanding by itself, calculating second-degree Hessians in this manner would be too difficult for a mobile device to execute. To mitigate this problem, we estimate these meta gradients by finding the direction of the gradient path through simply calculating the weight difference between ϕ and θ as proposed in [33].

$$v_\tau = \nabla_{\theta_\tau} \mathcal{L}(\phi_\tau) \approx \phi_\tau - \theta_\tau \tag{10}$$

Once we calculate the meta gradients for all tasks in S_t, we update the GAN model parameters using the average of all the meta-gradients. This method of training creates an adaptable GAN model with parameters θ_G and θ_D such that when exposed to a new task τ, the model may quickly converge to the optimal parameters $\phi_{G,\tau}$ and $\phi_{D,\tau}$ with only a few data points and E inner loops. Thus, meta learning allows the GAN model rapid and easy generalization to unseen tasks with few data points.

Figure 4: A sample of portrait images of a specific task (Renoir) from our *Meta-Portrait* dataset.

4. Experiments

4.1. Experimental Settings

All our experiments were carried out on a school server with four NVIDIA Titan RTX GPUs and two Intel Xeon Silver CPUs. For the mobile adaptation of our model, we use a Jetson Nano with a 128-core Maxwell GPU and 4 GB 64-bit LPDDR4 RAM. We include the list of hyperparameters we used to train our model in the appendix.

4.1.1 Model Architecture

Our experiments were carried out using a ResNet-based model architecture proposed by Johnson et al. [16] who have shown remarkable results for neural style transfer and super-resolution. Using this architecture, our generator model contains three convolutions, serveral residual blocks [11], two convolutions with a stride of $\frac{1}{2}$, and a final convolution layer that maps the features to a RGB image. We use nine residual blocks as we use images of higher quality - (256×256) for landscape images and (220×180) for portrait images. Here, we resize the images with bilinear interpolation with random horizontal flips as data augmentation for the training data. Similar to [16, 46] we use instance normalization [39] which is easily implemented with the TensorFlow addons library.

For the discriminator network, we follow in the footsteps of [46] and implement PatchGANs [25], which classifies the validity of 70×70 overlapping image *patches*.

4.1.2 Datasets [1]

The goal of our experiments is to see whether our method can quickly adapt to new image translation tasks. Thus, to do so, we had created our own datasets, **Meta-Landscape**

[1] datasets available at `https://drive.google.com/drive/folders/17cY59LAn-ocfDWll4bVJcJUdWKCsa0Rb?usp=sharing`

and **Meta-Portrait**, for our meta-tasks from a large collection of artwork provided by the WikiArt [41] database, landscape photos from Kaggle [22], and face photos from Celeb-A [28]. **Meta-Landscape** consists of landscape photos and landscape paintings. **Meta-Portrait** consists of portrait photos and portrait paintings.

To build the datasets, among the images in the WikiArt, we select the ones that fall into one of the categories: landscapes or portraits. We use the two categories separately to differentiate between two underlying task structures since a meta-model trained to optimized on landscape style translation tasks will often perform poorly on facial style translation tasks.

Next, we divide each category into their respective art styles, including everything from contemporary art (i.e. pop-art, contemporary realism, minimalism) to art dating back several centuries (i.e. romanticism, renaissance, etc.).

Finally, we combine one or more art style (or we may just use one art style). We split the combined art style (e.g. impressionism and post-impressionism) into tasks based upon the artist who created the paintings. By doing so, we are able to instill a unique style to each task, creating a non-i.i.d. dataset in the process and empirically showing that our model is robust to non-i.i.d. training data as well.

We split the combined art style into training and testing datasets with a random 90:10 split. Note that although the task split was done in a random manner, we ensured that the Monet, Cezanne, and van Gogh tasks were in the *testing* dataset in order to compare with other image-translation models [46]. By this, we get the training and testing datasets for landscape paintings and portrait paintings.

For landscape photos, we split the Kaggle dataset into training and testing datasets with a random 90:10 split. Combining landscape photos and landscape paintings, we get **Meta-Landscape**. Similarly, for portrait photos, we split the Celeb-A with a random 90:10 split. Combining the portrait photos and portrait paintings, we get **Meta-Portrait**. The following is summary of each dataset.

Meta-Landscape Consists of landscape paintings and photos. Landscape paintings are from the impressionism and post-impressionist era with a total of 258 artists and 6,030 paintings. There are 4,319 landscape photos in total.

Meta-Portrait Consists of portrait paintings and photos. Portrait paintings were cropped from artworks from the impressionism and post-impressionist era with a total of 188 artists and 2,378 paintings. There are 202,599 portrait photos in total.

A major obstacle we faced when collecting paintings of portraits was the discrepancy in size and position of the subject in relation to the whole image in the painting of WikiArt with the images in Celeb-A. To elaborate, subjects in Celeb-A usually display their face and small portion of their upper body. On the other hand, subjects in paintings displayed a diverse range of exposure, ranging from only their face to showing their entire body.

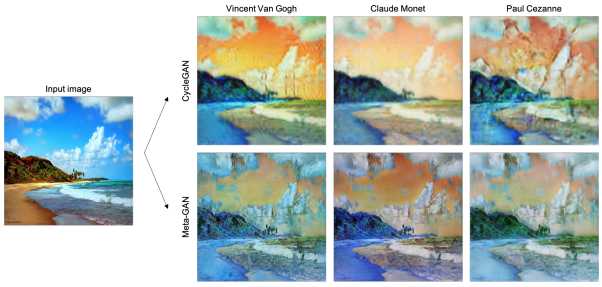

Figure 5: A comparison of our model with the original CycleGAN [46] model. Note that while the CycleGAN model was exclusively trained on each task, our model was fine-tuned with five gradient steps.

In order to solve this problem, we first ran a face detection algorithm using multi-task cascaded neural networks [43] to find the general location of the subject's face. Next, we mathematically calculated the offset between the detected face and the average bounding box points in the Celeb-A dataset. By resizing the bounding box detected on the portrait by this offset, we were able to extract a picture of the subject with the same proportions as the Celeb-A dataset. A sample of the images we created through this process is shown in Figure 4.

4.1.3 Evaluation Metrics

We used the following evaluation metrics for performance comparison.

- *Fréchet Inception Distance (FID).* We employed FID [12] to evaluate the quality of the translated images. FID measures the distance between the distribution of the generated and target samples through features extracted by Inception-V3 [37] classifiers.
- *Learned Perceptual Image Patch Similarity (LPIPS).* We employed LPIPS [44] to evaluate diversity of the translated images. LIPIS measures the average feature distances between generated samples.

4.2. Results

We conducted experiments on the *Meta-Landscape* and *Meta-Portrait* datasets with our model and the original `CycleGAN` [46] model, and compared the performance of the two models. As shown in Table 3, our few-shot image-to-image translation model outperforms the traditional unsupervised image-to-image translation model in majority of the tasks for both datasets. On the *Meta-Landscape* dataset, `Meta-GAN` achieves FID of 130.67 for Van Gogh task, 95.95 for Monet task, and 166.99 for Cezanne task. They

Table 1: A table summarizing the performance of our model in the server. Note that each task iteration takes approximately 0.2 seconds, meaning that fine-tuning with five inner loops on a particular task takes approximately 1 second to finish.

	$G_{X\to Y}$ loss	$G_{Y\to X}$ loss	D_X loss	D_Y loss	Time per iteration
Photo $\leftrightarrow$ Landscape	1.775	2.325	0.149	0.172	253 ± 16 ms
Photo $\leftrightarrow$ Portrait	1.834	2.313	0.126	0.170	213 ± 11 ms

Table 2: A table summarizing the performance of our model in a Jetson Nano. Note that each task iteration takes approximately 3 seconds, meaning that fine-tuning with five inner loops on a particular task takes approximately 15 seconds to finish.

	$G_{X\to Y}$ loss	$G_{Y\to X}$ loss	D_X loss	D_Y loss	Time per iteration
Photo $\leftrightarrow$ Landscape	2.117	2.468	0.138	0.207	3122 ± 45 ms
Photo $\leftrightarrow$ Portrait	2.100	2.227	0.131	0.203	2876 ± 37 ms

are all better than those achieved by the baseline model. Similar trends can be observed in LPIPS, which indicates that images translated through our model has both relatively high quality and diversity on the *Meta-Landscape* dataset, despite the fact that our model has much fewer training steps than the baseline model (see Figure 5). On the *Meta-Portrait* dataset, our model improves the diversity and fairly maintains the diversity of the baseline model in most of the tasks.

4.2.1 Experiment Architecture

Using the datasets defined in section 4.1.2, we define the meta-task as learning to map an image from an input domain X to a target Y_τ where τ represents a specific style or artist within the target domain space and $\|X\| = \|Y_\tau\| = 5$. In other words, a single training task consists of five input photos and five target photos with the model learning the mapping between these two. An example of this can be seen in Figure 3.

Each training round consists of sampling a task from the test dataset, fine-tuning on the task during meta-training, and updating the model with a meta-gradient calculated using task-specifically optimized weights. Such training rounds are repeated until convergence. In this manner, we trained the models with 50 epochs on the server, with each epoch consisting of approximately 1000 tasks.

We optimized the training process by training the model in *batches of tasks*, with each batch consisting of a single meta-task. The model is customized such that instead of performing gradient descent once with each batch, the model instead is optimized for E rounds, with the weights being updated after the final round in a manner consistent with algorithm (1).

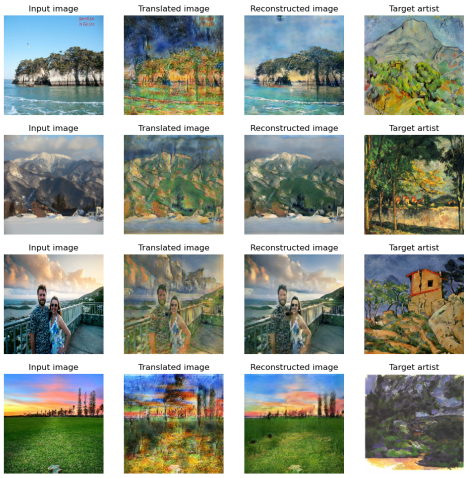

Figure 6: A set of photos fine-tuned on the server. First column: input image, second column: translated image, third column: reconstructed image, forth column: target artist.

4.2.2 Server Results

Fig. 6 shows the performance of our model when fine-tuning is run on the server.

The results of the server serve as a baseline for which we compare our model when fine-tuned on a mobile device. Note that the average time per iteration was about 253 milliseconds for a photo to landscape task, and 213 milliseconds for a photo to portrait task. This difference in time is to be expected since a single image in a portrait task is smaller (220×180) than a single image in a landscape task (256×256). An example of images created in this manner is shown in Figure 6

Table 3: Quantitative results on the *Meta-Landscape* and *Meta-Portrait* datasets. Lower FID and higher LPIPS are better.

Metrics	Models	Photo ↔ Landscape			Photo ↔ Portrait		
		Van Gogh	Monet	Cezanne	Van Gogh	Monet	Cezanne
FID	CycleGAN [46]	159.46	101.88	189.06	**184.4335**	236.5659	**183.3828**
	Meta-GAN (ours)	**130.67**	**95.95**	**166.99**	192.5191	**220.7521**	195.0791
LPIPS	CycleGAN [46]	0.6751 ± 0.0743	0.6728 ± 0.0603	0.6312 ± 0.0602	0.5404 ± 0.0605	0.5560 ± 0.0617	0.5310 ± 0.0583
	Meta-GAN (ours)	$\mathbf{0.8133 \pm 0.0680}$	$\mathbf{0.8003 \pm 0.0650}$	$\mathbf{0.7531 \pm 0.0663}$	$\mathbf{0.5489 \pm 0.0621}$	$\mathbf{0.5675 \pm 0.0523}$	$\mathbf{0.5688 \pm 0.0559}$

4.2.3 Mobile Results

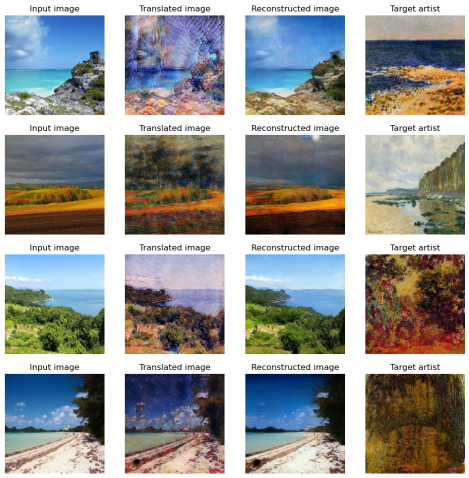

Figure 7: A set of photos fine-tuned on a mobile device. First column: input image, second column: translated image, third column: reconstructed image, forth column: target artist.

Fig. 7 shows the performance of our model when fine-tuning is run on a mobile device.

The results of the mobile device show that although there was not a noticeable difference in model performance (this is to be expected since the model is trained in the same Reptile manner as the one trained in the server), the training time per iteration increases by a factor of about 12. An example of the images created in this manner is shown in Figure 7, empirically showing that the performance of the model does not degrade when run on mobile.

However, it is important to note that this experiment empirically proves that training on a mobile device is possible, and that a mobile device is capable of fine-tuning a model that can generate images of an unseen class with the same level of quality as one that is trained on the server, albeit at a slower pace.

5. Discussion

5.1. Future Work

Although we show that `Meta-GAN` is comparable to the state-of-the-art when fine-tuned, the area of few-shot generative adversarial networks is still relatively unexplored and still has room for improvement. This is especially the case for few-shot GANs in a mobile environment. Methods of more mobile versions of `Meta-GAN` may be achieved through optimization techniques including, but not limited to, quantization for smaller weights and pruning for faster inference. This area can also be expanded upon by incorporating quantization-aware meta training in the server, allowing the server to create a meta-model that not only generalizes well to new tasks but also creates a model that is optimized for quantization.

Furthermore, since mobile GANs by nature run on privacy-sensitive user devices, measures of privacy may also need to be implemented while fine-tuning. In this case, ideas from federated learning and encryption alongside Meta-Federated Learning can be used to create a generalized initial model while preserving privacy.

Finally, the work we have done so far is fine-tuned on artists only in the impressionist era. Our future work may include creating a meta-model that is trained on different art styles instead of different artists, with the model capable of producing art of a certain style with only a few artist-agnostic paintings in the aforementioned art style.

5.2. Conclusion

In conclusion, we created a novel few-shot image translation framework `Meta-GAN` such that it can produce state-of-the-art generated images when fine-tuned even in mobile environments. In doing so, we first confirmed that meta-learning is empirically possible in image-translation tasks, with results exceeding models dedicated to producing images of a single style. In order to train our model in such a manner, we proposed two novel datasets **Meta-Landscape** and **Meta-Portrait** such that it provided few-shot image translation tasks to various art styles and artists. Finally, we trained our model in both the server and in a mobile environment, showing comparable performance in both.

References

[1] Succ a et al. "On the Optimization of a Synaptic Learning Rule". In: (Jan. 2002).

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. *Wasserstein GAN*. 2017. arXiv: 1701.07875 [stat.ML].

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. *Layer Normalization*. 2016. arXiv: 1607.06450 [stat.ML].

[4] Sergey Bartunov and Dmitry Vetrov. "Few-shot generative modelling with generative matching networks". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2018, pp. 670–678.

[5] Sagie Benaim and Lior Wolf. "One-shot unsupervised cross domain translation". In: *arXiv preprint arXiv:1806.06029* (2018).

[6] Yunjey Choi et al. "StarGAN: Unified Generative Adversarial Networks for Multi-domain Image-to-Image Translation". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8789–8797. DOI: 10.1109/CVPR.2018.00916.

[7] Louis Clouâtre and Marc Demers. *FIGR: Few-shot Image Generation with Reptile*. 2019. arXiv: 1901.02199 [cs.LG].

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. Vol. 70. ICML'17. Sydney, NSW, Australia: JMLR.org, 2017, pp. 1126–1135. URL: https://dl.acm.org/doi/10.5555/3305381.3305498.

[9] Zhe Gan et al. "Triangle generative adversarial networks". In: *arXiv preprint arXiv:1709.06548* (2017).

[10] Ian Goodfellow et al. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani et al. Vol. 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

[11] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[12] Martin Heusel et al. "Gans trained by a two time-scale update rule converge to a local nash equilibrium". In: *arXiv preprint arXiv:1706.08500* (2017).

[13] Timothy Hospedales et al. "Meta-learning in neural networks: A survey". In: *arXiv preprint arXiv:2004.05439* (2020).

[14] Zhizhong Huang et al. "PFA-GAN: Progressive Face Aging With Generative Adversarial Network". In: *IEEE Transactions on Information Forensics and Security* 16 (2021), pp. 2031–2045. ISSN: 1556-6021. DOI: 10.1109/tifs.2020.3047753. URL: http://dx.doi.org/10.1109/TIFS.2020.3047753.

[15] Phillip Isola et al. "Image-to-image translation with conditional adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.

[16] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. "Perceptual losses for real-time style transfer and super-resolution". In: *European Conference on Computer Vision*. 2016.

[17] Tero Karras, Samuli Laine, and Timo Aila. *A Style-Based Generator Architecture for Generative Adversarial Networks*. 2019. arXiv: 1812.04948 [cs.NE].

[18] Tero Karras, Samuli Laine, and Timo Aila. "A Style-Based Generator Architecture for Generative Adversarial Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[19] Taeksoo Kim et al. "Learning to discover cross-domain relations with generative adversarial networks". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 1857–1865.

[20] Dilip Krishnan et al. "Boundless: Generative Adversarial Networks for Image Extension". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 10520–10529. DOI: 10.1109/ICCV.2019.01062.

[21] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. "Human-level concept learning through probabilistic program induction". In: *Science* 350.6266 (2015), pp. 1332–1338. ISSN: 0036-8075. DOI: 10.1126/science.aab3050. eprint: https://science.sciencemag.org/content/350/6266/1332.full.pdf. URL: https://science.sciencemag.org/content/350/6266/1332.

[22] *Landscape Pictures: Datasets of pictures of natural landscapes*. https://www.kaggle.com/arnaud58/landscape-pictures/. Accessed: 2020-05-30.

[23] Yann LeCun and Corinna Cortes. "MNIST handwritten digit database". In: (2010). URL: http://yann.lecun.com/exdb/mnist/.

[24] Christian Ledig et al. *Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network*. 2017. arXiv: 1609.04802 [cs.CV].

[25] Chuan Li and Michael Wand. "Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks". In: *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III*. Ed. by Bastian Leibe et al. Vol. 9907. Lecture Notes in Computer Science. Springer, 2016, pp. 702–716. DOI: 10.1007/978-3-319-46487-9_43. URL: https://doi.org/10.1007/978-3-319-46487-9%5C_43.

[26] Weixin Liang, Zixuan Liu, and Can Liu. "DAWSON: A Domain Adaptive Few Shot Generation Framework". In: *arXiv preprint arXiv:2001.00576* (2020).

[27] Jianxin Lin et al. "Zstgan: An adversarial approach for unsupervised zero-shot image-to-image translation". In: *arXiv preprint arXiv:1906.00184* (2019).

[28] Ziwei Liu et al. "Deep Learning Face Attributes in the Wild". In: *Proceedings of International Conference on Computer Vision (ICCV)*. Dec. 2015.

[29] Qi Mao et al. "Mode seeking generative adversarial networks for diverse image synthesis". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1429–1437.

[30] Takeru Miyato et al. "Spectral Normalization for Generative Adversarial Networks". In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=B1QRgziT-.

[31] *Mobile Phone Ownership Over Time*. 2019. URL: https://www.pewresearch.org/internet/fact-sheet/mobile/.

[32] Arnab Kumar Mondal, Jose Dolz, and Christian Desrosiers. "Few-shot 3d multi-modal medical image segmentation using generative adversarial learning". In: *arXiv preprint arXiv:1810.12241* (2018).

[33] Alex Nichol, Joshua Achiam, and John Schulman. *On First-Order Meta-Learning Algorithms*. 2018. arXiv: 1803.02999 [cs.LG].

[34] Ben Poole et al. "Improved generator objectives for gans". In: *arXiv preprint arXiv:1612.02780* (2016).

[35] Alec Radford, Luke Metz, and Soumith Chintala. *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*. 2016. arXiv: 1511.06434 [cs.LG].

[36] Esther Robb et al. *Few-Shot Adaptation of Generative Adversarial Networks*. 2020. arXiv: 2010.11943 [cs.CV].

[37] Christian Szegedy et al. "Rethinking the inception architecture for computer vision". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2818–2826.

[38] Hung-Yu Tseng et al. "Regularizing Generative Adversarial Networks under Limited Data". In: *arXiv preprint arXiv:2104.03310* (2021).

[39] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. *Instance Normalization: The Missing Ingredient for Fast Stylization*. 2017. arXiv: 1607.08022 [cs.CV].

[40] Yaxing Wang et al. "Semi-supervised learning for few-shot image-to-image translation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 4453–4462.

[41] *WikiArt: Visual Art Encyclopedia*. https://www.wikiart.org/. Accessed: 2021-05-30.

[42] Huikai Wu et al. *GP-GAN: Towards Realistic High-Resolution Image Blending*. 2019. arXiv: 1703.07195 [cs.CV].

[43] Kaipeng Zhang et al. "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks". In: *IEEE Signal Processing Letters* 23.10 (Oct. 2016), pp. 1499–1503. ISSN: 1558-2361. DOI: 10.1109/lsp.2016.2603342. URL: http://dx.doi.org/10.1109/LSP.2016.2603342.

[44] Richard Zhang et al. "The unreasonable effectiveness of deep features as a perceptual metric". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 586–595.

[45] Jun-Yan Zhu et al. "Toward multimodal image-to-image translation". In: *arXiv preprint arXiv:1711.11586* (2017).

[46] Jun-Yan Zhu et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232.

A. Applying `Meta-GAN` to Image Generating GAN Models

Although our paper focuses on meta-learning for an image style translation GAN model, `Meta-GAN` can be applied to wide range of GAN models with slight modification in its algorithm. In this section, we show how to use our algorithm for image generating GAN models.

A.1. Related Work: Few-shot Image Generation

Few-shot adaption of the image generation task seeks good generalization on image generation function with only a few training examples of the target classes [32]. Several methods have been proposed for few-shot image generation.

[4] employs memory-assisted matching networks to achieve few-shot image generation. [7] uses Reptile to generate more advanced concepts of images with 8 samples from an unseen class. [26] integrates MAML with various GANs for both few-shot music and image generation. Despite the fact that excellent performance of the few-shot image generation models has been observed through numerical and theoretical analysis, mobile adaption of GAN still suffers from limited application.

A.2. Methodology: Image Generation from Latent Variables

The modification of `Meta-GAN` is given in Algortithm 2. Concerning an image generation task, the GAN model is optimized on each task τ with an inner loop consisting of E local epochs. The algorithm initializes a latent vector z and generates fake data using z and the current generator model. Then, we update the task specific discriminator parameters with cross-entropy loss using both the fake data and real data sampled by τ, producing an optimized discriminator model with parameters $\phi_{D,\tau}$. The generator is trained on the fine-tuned discriminator using cross-entropy loss with a newly created latent vector z, producing an optimized generator model with parameters $\phi_{G,\tau}$.

A.3. Experiments

A.3.1 Model Architecture

Our experiments use a DCGAN [35] with binary cross-entropy loss. Images from simpler datasets are resized to 32×32 or 64×64 with binlinear interpolation, while images from more complex datasets are reconfigured as necessary. No data augmentation layers were used. Results were sampled every 10 epochs, with each epoch containing about 500 meta-steps, while experiments took about 500 epochs to converge.

Algorithm 2 Meta-GAN (image generation task)

Require: learning rates α and β
Initialize generator parameters $\theta_{G,0}$
Initialize discriminator parameters $\theta_{D,0}$
for each round $t = 1, 2, ...$ **do**
  $S_t \leftarrow$ Random sample of m tasks ($1 \leq m \leq K$)
  Create a copy $W_G \leftarrow \theta_{G,t-1}$
  Create a copy $W_D \leftarrow \theta_{D,t-1}$
  for each task $\tau \in S_t$ **in parallel do**
    $\phi_{G,\tau}, \phi_{D,\tau} \leftarrow$ **InnerLoop**$(\tau, \theta_{G,t-1}, \theta_{D,t-1})$
    Calculate $v_{G,\tau}$, $v_{D,\tau}$ with Eqn. (10)
  end for
  $\theta_{G,t} \leftarrow \theta_{G,t-1} - \frac{\alpha}{\|S_t\|} \sum_{\tau \in S_t} v_{G,\tau}$
  $\theta_{D,t} \leftarrow \theta_{D,t-1} - \frac{\alpha}{\|S_t\|} \sum_{\tau \in S_t} v_{D,\tau}$
end for

InnerLoop$(\tau, \theta_G, \theta_D)$**:**
  $\phi_{G,0} \leftarrow \theta_G$
  $\phi_{D,0} \leftarrow \theta_D$
  for each local epoch i from 1 to E **do**
    Generate latent vector z
    Generate fake data y with z and $\phi_{G,i-1}$
    $\phi_{D,i} \leftarrow \phi_{D,i-1} - \beta \nabla_{\phi_{D,i-1}} \mathcal{L}(\tau, y)$
    Generate latent vector z
    $\phi_{G,i} \leftarrow \phi_{G,i-1} - \beta \nabla_{\phi_{G,i-1}} \mathcal{L}(\tau, y)$
  end for
  Return $\phi_{k,E}$ to server

A.3.2 Datasets

The goal of our experiments is to see whether our method can quickly adapt to new image generation tasks. The datasets that we use are as follows.

MNIST [23] is a dataset of handwritten digits containing 10 classes with 6,000 images of each class. In the context of few-shot image generation, each digit represents a task (i.e. τ_0 to τ_9) to solve. We choose tasks τ_0 to τ_8 as the training dataset to train the model to easily adapt to a new task (in this case τ_9) with only a few images. In our experiments, we chose $n = 5$, which means that our generator can be optimized to generating images of 9 with only 5 images sampled from τ_9.

Omniglot [21] represents a slightly more difficult image generation task, containing 1,623 unique characters from 50 different alphabets, each with 20 images of each class. We split the dataset by allocating 80% and 20% of the *characters* alongside its corresponding data to a training set and a validation set respectively. Contrary to MNIST, however, the larger dataset forces our model to test on a larger unseen sample set, testing its full ability to adapt to new tasks. We have found that the Omniglot dataset presented a much more difficult challenge, requiring a more complex model

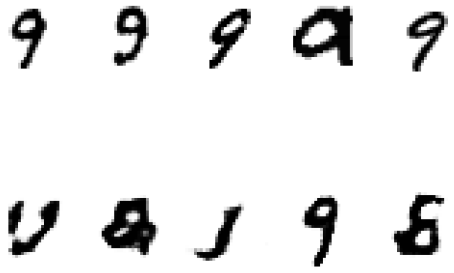

Figure 8: Validation of MNIST after 390 epochs. Top: MNIST Data, Bottom: Generated Image

Figure 9: Validation of Omniglot after 190 epochs trained with the DCGAN model. The generated images show less-than-optimal performance due to the simplistic nature of the DCGAN model.

and model optimization strategies such as layer normalization [3] and parametric ReLU.

A.3.3 Results

The results based on training with MNIST and Omniglot are given in Figs 8, 9 respectively. We find that the more the dataset is complex, the more advanced model architecture and delicate optimization strategies are required. Considering the results in Section 4, it is likely that if a model having a deeper structure (such as a ResNet-based structure) is trained with Algortithm 2, it will show better performance yet still be possible to learn new tasks using only the resources of a mobile device.