7. Appendix

7.1. Continuation of Universality of PFs and GFMN Convergence

We summarize here the main definitions and theorems from [28] regarding universality of kernels and feature maps.

Universal Kernels. The following defines a universal kernel

Definition 1 (Universal Kernel)**.** *Given a kernel K defined on $\mathcal{X} \times \mathcal{X}$. Let $\mathcal{Z}$ be any compact subset of $\mathcal{X}$. Define the space of kernel sections:*

$$\mathcal{K}(\mathcal{Z}) = \overline{span}\{K_y, y \in \mathcal{Z}\},$$

*where $K_y : \mathcal{X} \to \mathbb{R}$, $K_y(x) = K(x, y)$. Let $\mathcal{C}(Z)$ be the space of all continuous real valued functions defined on $\mathcal{Z}$. A kernel is said **universal** if for any choice of $\mathcal{Z}$ (compact subset of $\mathcal{X}$) $\mathcal{K}(\mathcal{Z})$ is dense in $\mathcal{C}(\mathcal{Z})$.*

In other words a kernel is universal if $\mathcal{C}(\mathcal{Z}) = K(\mathcal{Z})$. Meaning if any continuous function can be expressed in the span of K_y.

Universal Feature Maps. We turn now for kernels defined by feature maps and how to characterize their universality. Consider a continuous feature map $\Phi : \mathcal{X} \to \mathcal{W}$, where $(\mathcal{W}, \langle, \rangle_{\mathcal{W}})$ is a Hilbert space; the kernel K has the following form:

$$K(x, y) = \langle \Phi(x), \Phi(y) \rangle_{\mathcal{W}}. \tag{6}$$

Let $\mathcal{Y}$ be an orthonormal basis of $\mathcal{W}$ define the following continuous function $F_y \in \mathcal{C}(\mathcal{Z})$ defined at $x \in \mathcal{Z}$:

$$F_y(x) = \langle \Phi(x), y \rangle_{\mathcal{W}},$$

and let:

$$\Phi(\mathcal{Y}) = \overline{\text{span}}\{F_y, y \in \mathcal{Y}\}$$

Definition 2 (Universal feature Map)**.** *A feature map is universal if $\Phi(\mathcal{Y})$ is dense in $\mathcal{C}(\mathcal{Z})$, for all $\mathcal{Z}$ compact subsets of $\mathcal{X}$.i.e A feature map is universal if $\Phi(\mathcal{Y}) = \mathcal{C}(\mathcal{Z})$.*

The following Theorem shows the relation between universality of a kernel defined by feature map and the universality of the feature map:

Theorem 2 ([28], Thm 4, Relation between $\mathcal{K}(\mathcal{Z})$ and $\Phi(\mathcal{Y})$)**.** *For kernel defined by feature maps in (6) we have $\mathcal{K}(\mathcal{Z}) = \Phi(\mathcal{Y})$. A kernel of form (6) is universal if and only if its feature map is universal.*

Hence the following Theorem 7 from [28]:

Theorem 3 ([28])**.** *Let $S = \{\phi_j, j \in I\}$, where I is a countable set and $\phi_j : \mathcal{X} \to \mathbb{R}$ continuous function. Define the following kernel*

$$K(x, y) = \sum_{j \in I} \phi_j(x)\phi_j(y).$$

K is universal if and only if the set of features S is universal.

7.2. Discussion of AMA versus MA

As we already discussed the moving average of v of the difference of features means

$$\Delta_t = \frac{1}{N}\sum_{i=1}^{N} E(x_i) - \frac{1}{N}\sum_{i=1}^{N} E(G(z_i, \theta_t))$$

between real and generated data at each time step t in the gradient descent up to time T, can be seen as a gradient descent in an online setting on the following cost :

$$f^* = \min_v \sum_{t=1}^{T} f_t(v) = \sum_{t=1}^{T} ||v - \Delta_t||_2^2$$

Note that we are in the online setting since Δ_t is only known when θ_t of the generator is updated. The sequence v_t generated by MA (moving average) and by AMA (ADAM moving average) is the SGD updates and ADAM updates respectively applied to the cost function f_t. Hence we can bound the regret of the sequence $\{v_t^{\text{MA}}\}$ and $\{v_t^{\text{AMA}}\}$ using known results on SGD and ADAM. Let d be the dimension of the encoding E. For MA, using classic regret bounds for gradient descents we obtain:

$$R_T^{\text{MA}} = \sum_{t=1}^{T} ||v_t^{\text{MA}} - \Delta_t||_2^2 - f^* \leq O(\sqrt{dT}).$$

For AMA, using ADAM regrets bounds from (Reddi et al., 2018). Let us define

$$R_T^{\text{AMA}} = \sum_{t=1}^{T} ||v_t^{\text{AMA}} - \Delta_t||_2^2 - f^*.$$

We have:

$$R_T^{\text{AMA}} \leq O(\sqrt{T}\sum_{i=1}^{d} \hat{u}_i^{T,\frac{1}{2}}) + \cdots$$
$$O\left(\sum_{i=1}^{d} \sqrt{\sum_{t=1}^{T} (\Delta_{t,i} - v_{t,i}^{\text{AMA}})^2}\right) + C$$

where $\hat{u}$ are defined in the ADAM updates as moving averages of second order moments of the gradients. The

regret bound of AMA is better than MA especially if $\sum_{i=1}^{d} \hat{u}_i^{T,\frac{1}{2}} \ll d$ and

$$\sum_{i=1}^{d} \sqrt{\sum_{t=1}^{T} (\Delta_{t,i} - v_{t,i}^{\text{AMA}})^2} \ll \sqrt{Td}.$$

7.3. Mean Matching vs. Mean + Covariance Matching in GFMN

In this Appendix, we present comparative results between GFMN with mean feature matching vs. GFMN with mean + covariance feature matching. Using the first and second moments to perform feature matching gives statistical advantage over using the first moment only. In Table 5, we can see that for different feature extractors, performing mean + covariance feature matching produces significantly better results in terms of both IS and FID. Mroueh *et al.* [31] have also demonstrated the advantages of using mean + covariance matching in the context of GANs.

7.4. Neural Network Architectures

In Tables 6 and 7, and Figure 6 we detail the neural net architectures used in our experiments. In both DCGAN-like generator and discriminator, an extra layer is added when using images of size 64×64. In VGG19 architecture, after each convolution, we apply batch normalization and ReLU. The Resnet generator is used for CelebA$_{128\times128}$ experiments and also for some experiments with CIFAR10 and STL10. For these two last datasets, the Resnet generator has 3 ResBlocks only, and the output size of the $DENSE$ layer is $4 \times 4 \times 512$.

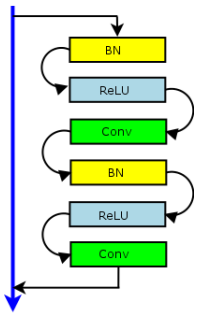


Figure 6: ResBlock

7.5. Pretraining of ImageNet Classifiers and Autoencoders

Both VGG19 and Resnet18 networks are trained with SGD with fixed 10^{-1} learning rate, 0.9 momentum term, and weight decay set to 5×10^{-4}. We pick models with best *top-1* accuracy on the validation set over 100 epochs of training; 29.14% for VGG19 (image size 32×32), and 39.63% for Resnet18 (image size 32×32). When training the classifiers we use random cropping and random horizontal flipping for data augmentation. When using VGG19 and Resnet18 as feature extractors in GFMN, we use features from the output of each ReLU that follows a conv. layer, for a total of 16 layers for VGG and 17 for Resnet18.

In our experiments with autoencoders (AE) we pretrained them using either mean squared error (MSE) or the Laplacian pyramid loss [25, 4]. Let E and D be the encoder and the decoder networks with parameters ϕ and ψ, respectively.

$$\min_{\phi,\psi} \mathbb{E}_{p_{data}} ||x - D(E(x;\phi);\psi)||^2$$

or the Laplacian pyramid loss [25]

$$\text{Lap}_1(x, x') = \sum_j 2^{-2j} |L^j(x) - L^j(x')|_1$$

where $L^j(x)$ is the j-th level of the Laplacian pyramid representation of x. The Laplacian pyramid loss provides better signal for learning high frequencies of images and overcome some of the blurriness issue known from using a simple MSE loss. [4] recently demonstrated that the Lap$_1$ loss produces better results than L_2 loss for both autoencoders and generative models.

7.6. Quantitative Evaluation Metrics

We evaluate our models using two quantitative metrics: Inception Score (IS) [38] and Fréchet Inception Distance (FID) [15]. We followed the same procedure used in previous work to calculate IS [38, 29, 35]. For each trained generator, we calculate the IS for randomly generated 5000 images and repeat this procedure 10 times (for a total of 50K generated images) and report the average and the standard deviation of the IS.

We compute FID using two sample sizes of generated images: 5K and 50K. In order to be consistent with previous works [29, 35] and be able to directly compare our quantitative results with theirs, the FID is computed as follows:

- CIFAR10: the statistics for the real data are computed using the 50K training images. This (real data) statistics are used in the FID computation of both 5K and 50K samples of generated images. This is consistent with both Miyato *et al.* [29] and Ravuri *et al.* [35] procedure to compute FID for CIFAR10 experiments.

- STL10: when using 5K generated images, the statistics for the real data are computed using the set of 5K (labeled) training images. This is consistent with the FID

Table 5: CIFAR10 results for GFMN with Mean Feature Matching vs. GFMN with Mean + Covariance Feature Matching.

Feature Extractor	**Mean Matching**		**Mean + Covar. Matching**	
	IS	FID (5K / 50K)	IS	FID (5K / 50K)
DCGAN (Encoder)	3.76 ± 0.04	96.5 / 92.5	4.51 ± 0.06	82.8 / 78.3
Resnet18	7.03 ± 0.11	35.7 / 31.1	7.92 ± 0.10	29.1 / 24.3
VGG19	7.42 ± 0.09	27.5 / 22.8	7.88 ± 0.08	25.5 / 20.8

Table 6: DCGAN like Generator

$z \in \mathbb{R}^{100} \sim \mathcal{N}(0, I)$
DENSE $\rightarrow 4 \times 4 \times 512$
4×4, STRIDE=2 DECONV BN 256 RELU
4×4, STRIDE=2 DECONV BN 128 RELU
4×4, STRIDE=2 DECONV BN 64 RELU
3×3, STRIDE=1 CONV 3 BN 64 RELU
3×3, STRIDE=1 CONV 3 BN 64 RELU
3×3, STRIDE=1 CONV 3 TANH

Table 7: Resnet Generator

$z \in \mathbb{R}^{100} \sim \mathcal{N}(0, I)$
DENSE, $4 \times 4 \times 2048$
RESBLOCK UP 1024
RESBLOCK UP 512
RESBLOCK UP 256
RESBLOCK UP 128
RESBLOCK UP 164
BN, RELU, 3×3 CONV 3
TANH

computation of Miyato *et al*. [29]. When using 50K generated images, the statistics for the real data are computed using a set of 50K images randomly sampled from the unlabeled STL10 dataset.

FID computation is repeated 3 times and the average is reported. There is very small variance in the FID results.

7.7. Impact of the number of layers used for feature extraction

Figure 7 shows generated images from generators that were trained with a different number of layers employed to feature matching. In all the results in Fig.7, the VGG19 network was used to perform feature extraction. We can see a significant improvement in image quality when more layers are used. Better results are achieved when 11 or more layers are used, which corroborates the quantitative results in Sec. 5.2.

7.8. Pretrained Generator/Discriminator in WGAN-GP

The objective of the experiments presented in this section is to evaluate if WGAN-GP can benefit from DCNN classifiers pretrained on ImageNet. In the experiments, we used a WGAN-GP architecture where: (1) the discriminator is a VGG19 or a Resnet18; (2) the discriminator is pretrained on ImageNet; (3) the generator is pretrained on CIFAR10 through autoencoding. Although we tried different hyperparameter combinations, we were not able to successfully train WGAN-GP with VGG19 or Resnet18 discriminators. Indeed, the discriminator, being pretrained on ImageNet, can quickly learn to distinguish between real and fake images. This limits the reliability of the gradient information from the discriminator, which in turn renders the training of a proper generator extremely challenging or even impossible. This is a well-known issue with GAN training [10] where the training of the generator and discriminator must strike a balance. This phenomenon is covered in [2] Section 3 (illustrated in their Figure 2) as one motivation for work like Wassertein GANs. If a discriminator can distinguish perfectly between real and fake early on, the generator cannot learn properly and the min/max game becomes unbalanced, having no good discriminator gradients for the generator to learn from, producing degenerate models. Figure 8 shows some examples of images generated by the unsuccessfully trained models.

7.9. Impact of Adam Moving Average for VGG19 feature extractor.

In this appendix, we present a comparison between the simple moving average (MA) and ADAM moving average (AMA) for the case where VGG19 ImageNet classifier is used as a feature extractor. This experiment uses a mini-batch size of 64. We can see in Fig. 9 that AMA has a very positive effect in the quality of generated images. GFMN trained with MA produces various images with some sort of crossing line artifacts.

7.10. Visual Comparison between GFMN and GMMN Generated Images.

Figure 10 shows a visual comparison between images generated by GFMN (Figs. 10a and 10b) and Generative Moment Matching Networks (GMMN) (Figs. 10c and 10d).

(a) 1 Layer (b) 3 Layers (c) 5 Layers

(d) 7 Layers (e) 9 Layers (f) 11 Layers

(g) 13 Layers (h) 15 Layers (i) 16 Layers

Figure 7: Generated images from GFMN trained with a different number of VGG19 layers for feature extraction.

GMMN [24] generated images were obtained from Li *et al.* [22]. In this experiment, both GMMN and GFMN use a DCGAN-like architecture in the generator. Images generated by GFMN have significantly better quality compared to the ones generated by GMMN, which corroborates the quantitative results in Sec. 5.4.

7.11. Autoencoder features vs. VGG19 features for CelebA.

In this appendix, we present a comparison in image quality for autoencoder features vs. VGG19 features for the CelebA dataset. We show results for both simple moving

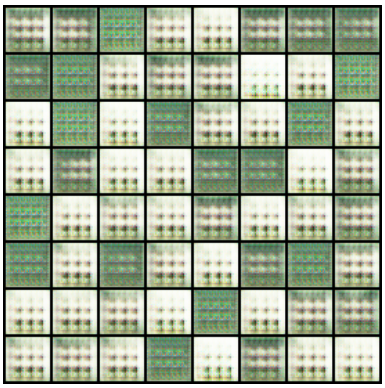
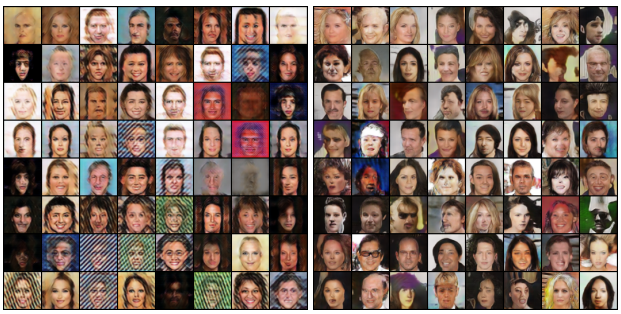


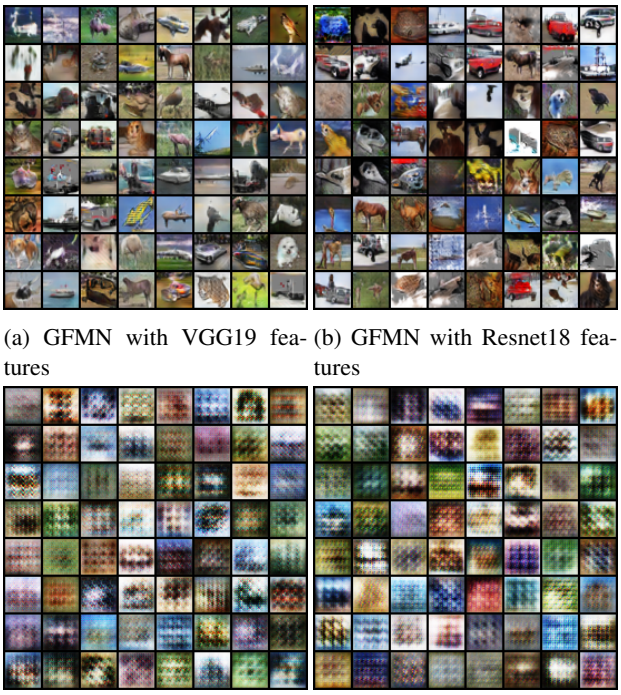
Figure 8: Generated images by WGAN-GP with pretrained VGG19 as a discriminator.



(a) MA (b) AMA

Figure 9: Generated images from GFMN trained with either simple moving average (MA) or Adam moving average (AMA). VGG19 ImageNet classifier is used as feature extractor.

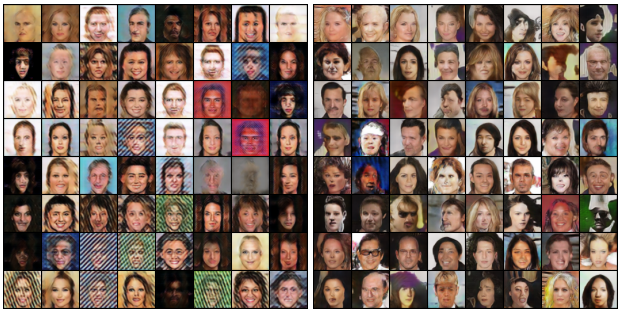
average (MA) and ADAM moving average (AMA), for both cases we use a minibatch size of 64. In Fig. 11, we show generated images from GFMN trained with either VGG19 features (top row) or autoencoder (AE) features (bottom row). We show images generated by GFMN models trained with simple moving average (MA) and Adam moving average (AMA). We can note in the images that, although VGG19 features are from a cross-domain classifier, they lead to much better generation quality than AE features, specially for the MA case.



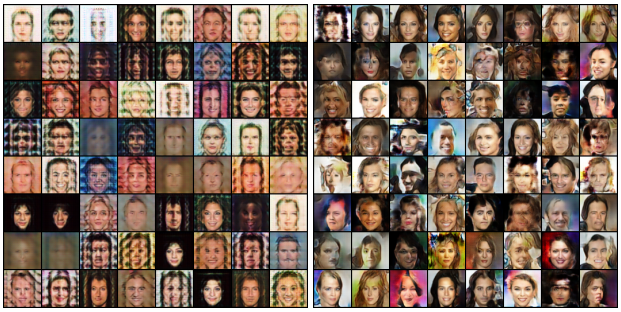
(a) GFMN with VGG19 features (b) GFMN with Resnet18 features

(c) GMMN - Matching on data space (d) GMMN+AE - Matching on AE space

Figure 10: Generated images from GFMN (10a and 10b) and GMMN (10c and 10d). GMMN images were obtained from Li *et al*. [22].



(a) MA - VGG19 Features (b) AMA - VGG19 Features



(c) MA - AE Features (d) AMA - AE Features

Figure 11: Generated images from GFMN trained with either VGG19 features (top row) or autoencoder (AE) features (bottom row). We show images generated by GFMN models trained with simple moving average (MA) and Adam moving average (AMA). Although VGG19 features are from a cross-domain classifier, they perform much better than AE features, specially for the MA case.