Understanding and Simplifying Perceptual Distances - Supplementary Material

Dan Amir and Yair Weiss
The Hebrew University of Jerusalem
{dan.amir,yair.weiss}@mail.huji.ac.il

1. Full Proof of Theorem

Theorem 1.1. *L2 distance over normalized representations in the post-pooling layer of an infinite random S-CNN with patch size P and pooling window size W is equal to the average $M\hat{M}D_K$ distance between the distributions of patches of size P within windows of size W in the two images. The kernel $K(p_1, p_2)$ of two patches is a robust, monotonically decreasing function of the L2 distance between the two patches.*

Proof. We first prove a Lemma on fully-connected networks.

Lemma: For a fully connected network with ReLU nonlinearity and *any depth* d, the kernel corresponding to an infinitely wide network with random weights (initialized with σ determined according to the He initialization scheme [2] and zero biases) can be written as:

$$K(x_1, x_2) = \|x_1\|\|x_2\|\rho^d(\frac{x_1}{\|x_1\|} - \frac{x_2}{\|x_2\|})$$

where $\rho(r)$ is a function that is monotonically decreasing as a function of the norm of its argument.

Proof. The proof is based on the recursive definition of the kernel, given by Cho and Saul [1]:

$$K^{(d+1)}(x_1, x_2) = \frac{1}{\pi}\left[K^{(d)}(x_1, x_1)K^{(d)}(x_2, x_2)\right]^{1/2} J_1(\theta^{(d)})$$

$$\theta^{(d)} = \cos^{-1}\left(\frac{K^{(d)}(x_1, x_2)}{\sqrt{K^{(d)}(x_1, x_1)}\sqrt{K^{(d)}(x_2, x_2)}}\right)$$

with:

$$J_1(\theta) = \sin\theta + (\pi - \theta)\cos(\theta)$$

Using these recursions, it is easy to see that for any d, $K^{(d)}(x_1, x_1) = \|x_1\|^2$, $K(x_2, x_2) = \|x_2\|^2$, and then we can prove by induction that for any depth the dependence on the angle between the two inputs remains monotonic, and this means that the dependence on the difference of two normalized inputs is also monotonic for any depth. From figure 1 it is clear that the shape of K becomes more robust as d grows. It can be shown that the normalized kernel is monotonically decreasing with depth for every value of $\langle x_1, x_2\rangle$.

□

If we consider the pre-pooling layer of an infinite random S-CNN, it is equivalent to many fully connected networks, each working on a single patch of the input image. From the lemma, this means that the NNGP kernel of each patch $K(p_1, p_2)$ corresponds to a robust, monotonically decreasing function of the L2 distance between the patches, weighted by the norm of the two patches. If we use normalized representations, the norm dependence disappears and $K(p_1, p_2)$ is a robust, monotonically decreasing function of the L2 distances between the patches.

As shown in [6], the effect of average pooling on the NNGP can be written as:

$$K(x_1, x_2) = \sum_{p_1 \in x_1, p_2 \in x_2} K(p_1, p_2) \quad (1)$$

where x_1, x_2 are the pooling windows in the two images and $p_1 \in x_1$ is a patch in the pooling window in the first image.

Now, if we take L2 distances between the representations in the post-pooling layers then:

$$d^2(x_1, x_2) = K(x_1, x_1) - 2K(x_1, x_2) + K(x_2, x_2) \quad (2)$$

Substituting equation 1 into equation 2 shows that the L2 distance is exactly the MMD between patch distributions in the pooling windows. □

Remark: In the proof above, we assume that all spatial representations in the S-CNN are normalized per location. This can be achieved by normalizing the outputs of the first convolution by dividing by the norm (taken over all channels at a single location). Due to the property of ReLU, this will guarantee that at any subsequent layer (after applying the 1×1 convolution and ReLU), the representations are still normalized per location. If we do not normalize the representations, an analogous result can be derived using an identical proof: the L2 distance over representations in the post-pooling layer will still be equal to the average

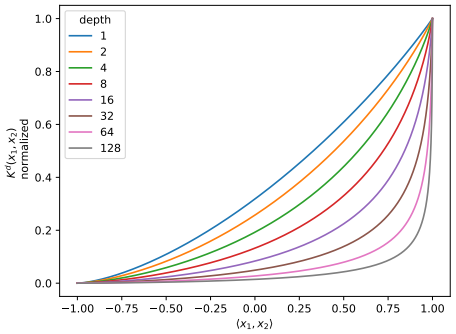

Figure 1. The equivalent kernel for the infinitely wide fully-connected network with ReLU activation. $K^d(x_1, x_2)$ values are normalized to the range $[0, 1]$. The MMD objective optimum doesn't depend on the scale of K.

$M\hat{M}D_K$ distance between the distributions of patches of size P within windows of size W in the two images. But the kernel $K(p_1, p_2)$ will depend not only on the distance between the two patches but also on their respective norms.

Corollary: Define the Mixture Patch Distribution for a set of possible outputs $Y = \{y_1...y_M\}$ and pooling window as the distribution of patches drawn uniformly from the set and assume the number of patches in a pooling window $N \to \infty$. If there exist an image y for which for all pooling windows its' patches are distributed according to the Mixture Patch Distribution of Y, then $\hat{y}$ using a perceptual loss with an infinite random S-CNN should have the same local distribution over patches as the Mixture Patch Distribution.

Proof: For every measurable kernel with bounded norm we can rewrite the MMD distance between two distributions P and Q as $\|\mu_P - \mu_Q\|^2$ where μ_P is the mean embedding of the distribution P with respect to the RKHS of K. When $N \to \infty$ we can replace the empirical estimate of $M\hat{M}D_K$ with the MMD distance, and the embedding $\hat{\mu}$ which minimizes the average MMD with respect to all images in Y is $\frac{1}{M}\sum \mu_{P_{y_i}}$. Assuming that y exists, its' patch distribution is clearly mapped to $\hat{\mu}$ for all pooling windows. Assuming K is characteristic, we get that the mapping of distributions to mean embeddings is injective and thus the patch distribution of $\hat{y}$ agrees with y.

2. Full Implementation Details for the Experimental Setup

2.1. Losses Implementation

For all random networks, filters were drawn from i.i.d Gaussian distribution with variance determined according to the He initialization scheme [2] at each optimization step and biases were fixed to 0. For the 2AFC experiments the mean of every filter (with respect to every input color channel) was reduced in order to ignore the DC component of each channel.

VGG loss - For all experiments we use post-activation features from layers $CONV_{1\_2}, CONV_{2\_2}, CONV_{3\_3}, CONV_{4\_3}, CONV_{5\_3}$. Input images are normalized using the RGB mean and standard deviations of ImageNet. For the GIM and super-resolution experiments, features are normalized for every layer and spatial location and then compared using L2 distance.

MMD loss - For all experiments we compute MMD distance on patches of size 3 over windows of size 32 with stride of 16 pixels. We varied σ for different applications but it was always in the range $0.54 - 0.72$. Similarly to the random networks, for the 2AFC task the DC component of each color channel was ignored. We use either 512, 1024 or 2048 random projections for the Random Fourier Features approximation depending on the available computational resources at each of the experiments, but preliminary results show that lower number of features could produce comparable results at least in some settings. For the extended MMD loss, we use $\sigma = 0.9$ for the local patch loss and ignore the channel DC value for the MMD term. The weight of the local term compared to the MMD term is 0.025.

2.2. Generalized Mean Optimization Task

For every problem instance of the described problem settings, we sample 50 target examples. For every loss evaluated, we initialize the image with the mean of the 50 examples (equivalent to pretraining with L2 loss). The image value is optimized with PyTorch [7] through SGD with the commonly used Adam optimizer [4] with default parameters and learning rate of 0.01, on random batches of 5 images for 400 epochs. Optimizing such losses without any additional regularization is doomed to fail, thus as in every implementation of CNN-based perceptual loss, we add a weighted pixel level L2 loss to the objective. The learning rate, as well as the L2 weight and value of σ for the MMD loss, were chosen over a small set of examples not used for the quantitative comparison. The predicted image is passed through $\tanh$ activation during optimization to ensure that all predicted pixels are within valid range. Since the sliced Wasserstein distance based metric introduced by [3] originally aims to compare samples of the same size out of the two distributions, we randomly sample balanced (one per problem instance) sets from the original images without replacement and average the distance across samples.

StyleGAN - To generate a problem instance for the StyleGAN task, first an initial z is drawn from $z \sim \mathcal{N}(0, I)$. Then for every example we add noise from a smaller Gaussian, such that $z_i = z + \eta_i$ for $\eta \sim \mathcal{N}(0, 0.04I)$ and all the z_i's are normalized such that they will have the same norm

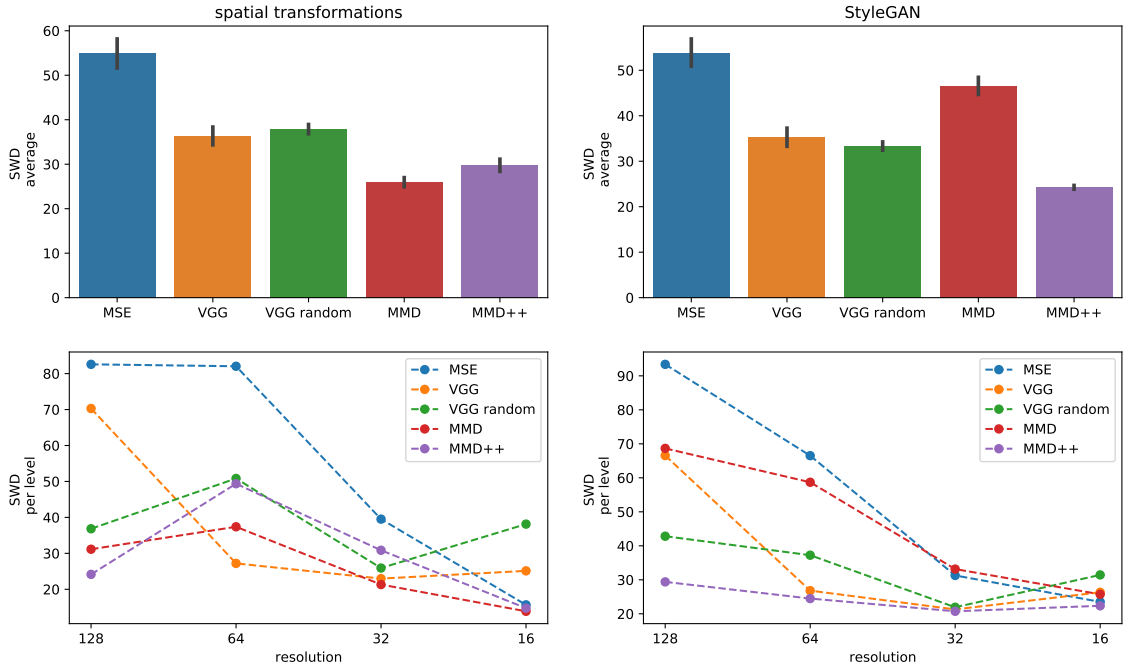

Figure 2. Sliced Wasserstein Distance over patches in different Laplacian Pyramid scales [3] on our generalized mean optimization tasks for all losses. MMD and VGG (either trained or randomly initialized) outperform the MSE. Adding local patch loss term in MMD++ improves greatly performance on StyleGAN image sets (where visual artifacts are noticable for the MMD loss) at the cost of slight degradation for the spatial transformations problem set.

as z. To obtain the set of images, all z_i's are fed to the official pretrained (on FFHQ data set) generator using truncation with $\psi = 0.75$, to avoid generator failures. We use the model trained on the maximal resolution (1024×1024) and then resize back to 128×128 to minimize the effect of possible GAN artifacts in high frequencies.

Spatial Transformations - For every problem instance, a random image was drawn from ImageNet and resized such that its short side is of 256 pixels and the aspect ratio is preserved. Then, for each example, a random square crop of size 128 is drawn within a $[-4, 4]$ range from the center crop at each direction.

2.3. Super Resolution

For the neural network architecture we use a modified version of the SRGAN architecture (that obtains close to state of the art results for the same task) presented in [5] where the last convolution kernel size is reduced from 9×9 to 3×3 to reduce the computational cost of training. As a training set, we draw 10,000 random examples from ImageNet. At training, the images are randomely cropped to a 128×128 image which is then down scaled by a factor of $\times 4$ using bicubic interpolation. All models are trained for 100 epochs with batch size of 8 which translates to 125,000 training iterations. All models are optimized with the Adam optimizer with the default parameters in PyTorch, an initial learning rate of $1e^{-3}$ which is then reduced by a factor of 2 three times during training. All hyper-parameters are chosen based on the performance on another random held-out set from ImageNet. We experimented with different configurations of VGG loss based on previous works on super-resolution (with / without normalization, using only $CONV_{2\_2}$, replacing the max-pool layers with average pooling, adding total variation regularization) and found either worse or similar performance on held-out set. Thus, for our final version, we stick with the same configuration as in the rest of the experiments. During evaluation, test images are cropped such that both their height and width will be divisible by a factor of 4 before resizing.

3. Additional Results

Additional qualitative results for the different loss functions for the GIM task are available in figure 3 and figure 4. Figure 2 shows quantitative comparison of the extended MMD loss (MMD++) and the basic MMD loss on the same problem sets. Very strong periodic artifacts can be seen in the MMD results for some of the spatial transfor-

mations sets but combining the MMD loss with local patch loss (equivalent to pre-pooling layers of S-CNN) removes those artifacts. Super-resolution results for addition examples from BSD100 appear in figure 5.

Figure 6 shows how different modifications of the random VGG loss contribute to the final qualitative result. Figure 7 shows GIM experiments for which trained and random VGG behave differently. Using learned filters for the first convolution and random weights for the rest of the layers leads to generalized means mcuh more similar to the trained network, indicating that again the success of the trained VGG is not due to some semantic representation of the images.

References

[1] Youngmin Cho and Lawrence K Saul. Kernel methods for deep learning. In *Advances in neural information processing systems*, pages 342–350, 2009. 1

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 1, 2

[3] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 2, 3

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[5] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017. 3

[6] Roman Novak, Lechao Xiao, Jaehoon Lee, Yasaman Bahri, Greg Yang, Jiri Hron, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Bayesian deep convolutional networks with many channels are gaussian processes. *arXiv preprint arXiv:1810.05148*, 2018. 1

[7] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 2

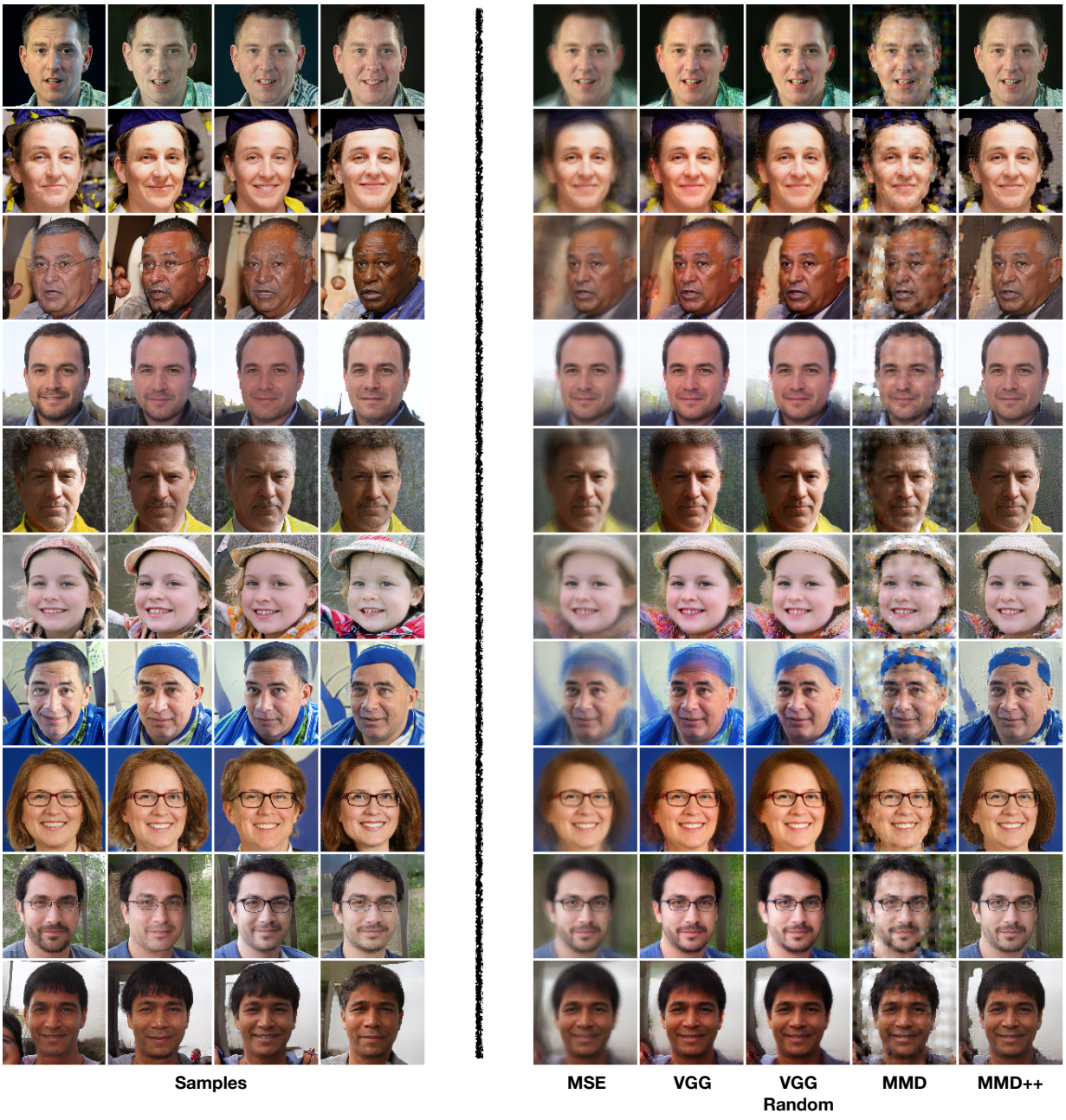

Figure 3. A random set of results for the different losses on the Generalized Image Mean optimization task for StyleGAN generated images.

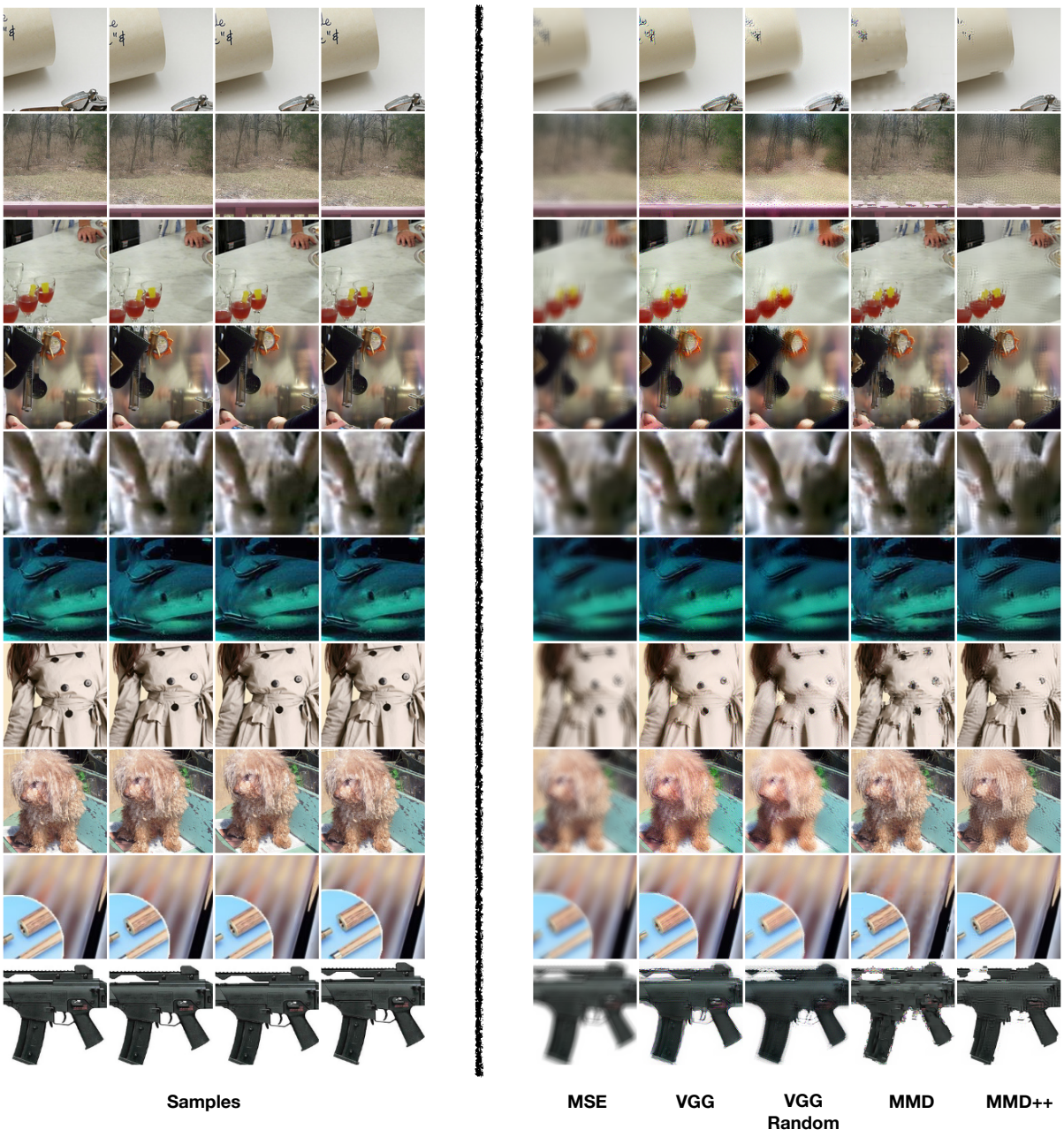

Figure 4. A random set of results for the different losses on the Generalized Image Mean optimization task for the spatial transformations problem sets.

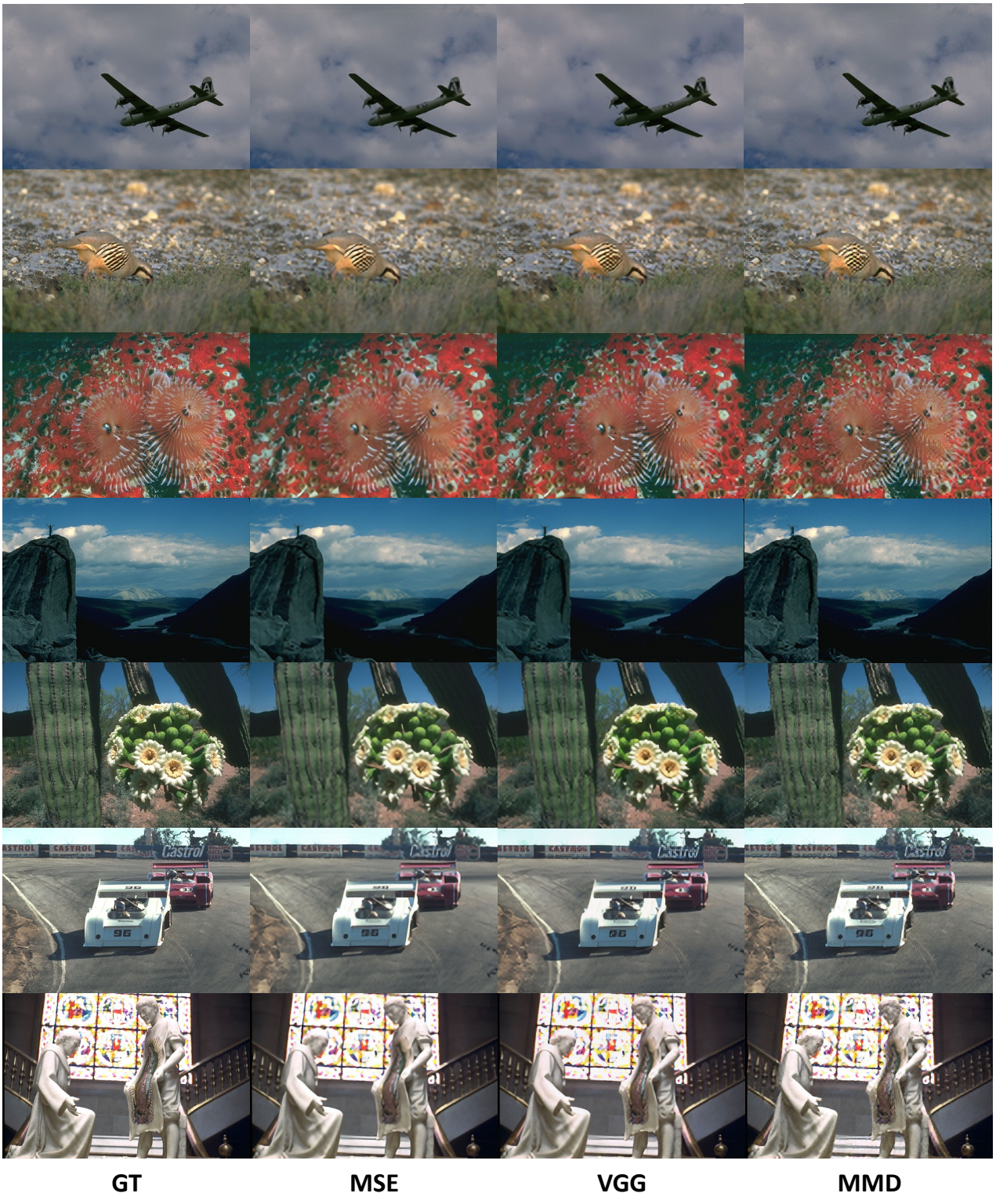

Figure 5. Additional qualitative results for the super-resolution models on the BSD100 data set.

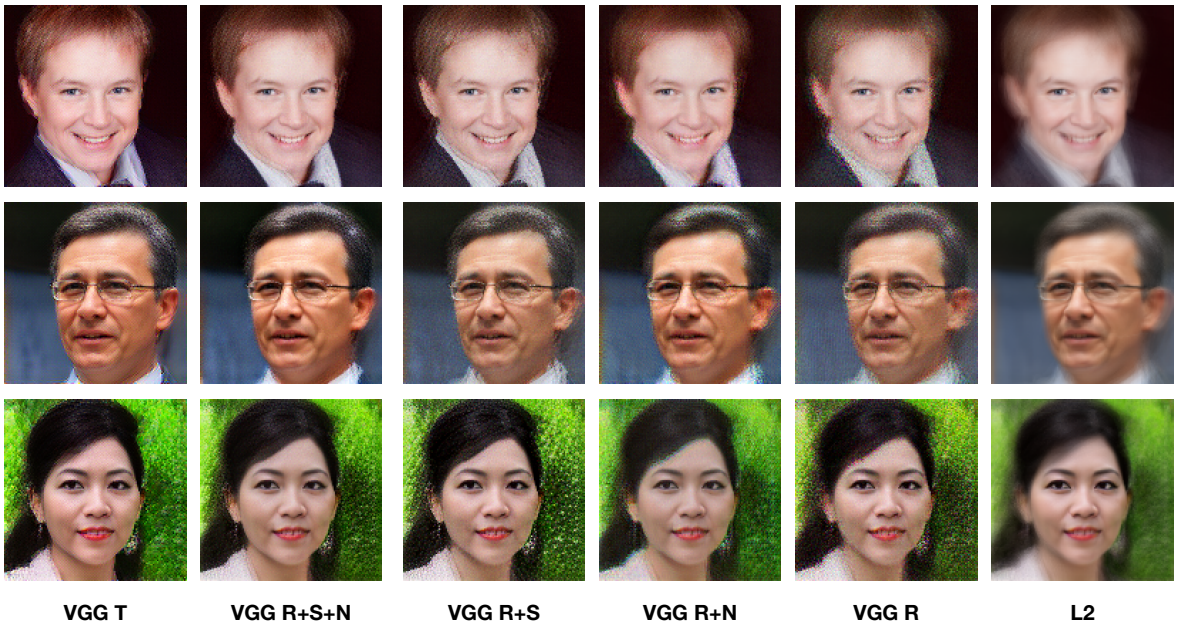

Figure 6. The effect of different loss modifications for random VGG. Results on the spatial transformations image optimization problem set. **T** stands for trained model, **R** for randomely initialized model, **S** for stochastic model (model re-initialized at each iteration) and **N** for normalization of the representation along the channel axis prior to loss computation). Clearly, both re-initialization and normalization are necessary to obtain results comparable to pre-trained VGG with a random network.

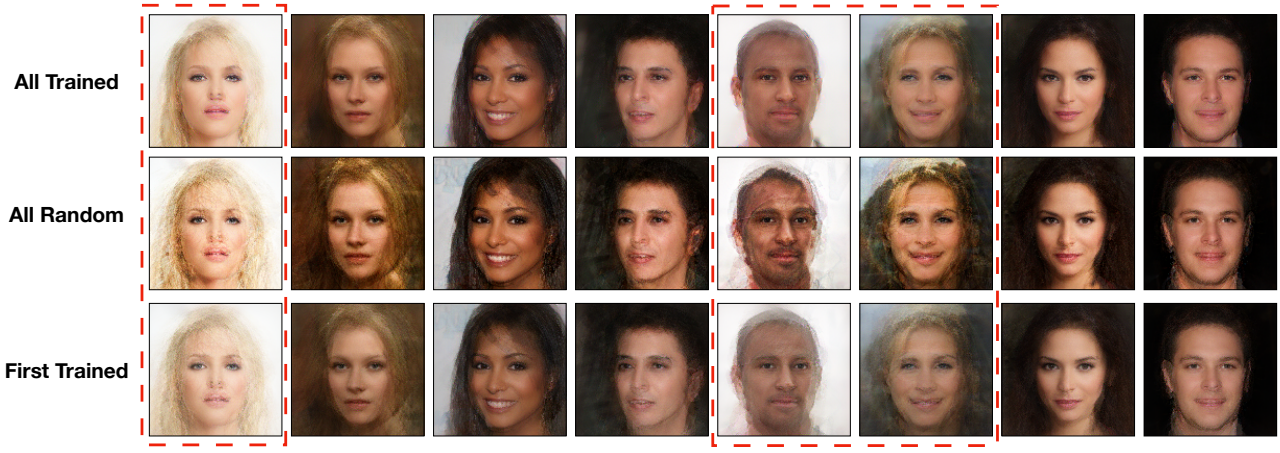

Figure 7. Comparison of VGG perceptual losses with different levels of supervision on random CelebA-HQ images and their 9 nearest neighbors. "All trained" and "All random" use the same configuration as in the rest of the paper. In "First trained" the weights of the first convolution are set to the ones of the trained model while the rest of the convolutions are randomly initialized at every iteration. Problem instances in which the difference between the trained and random are more pronounced are marked in red. Adding learned filters to the first layer results in generalized means similar to the trained model.

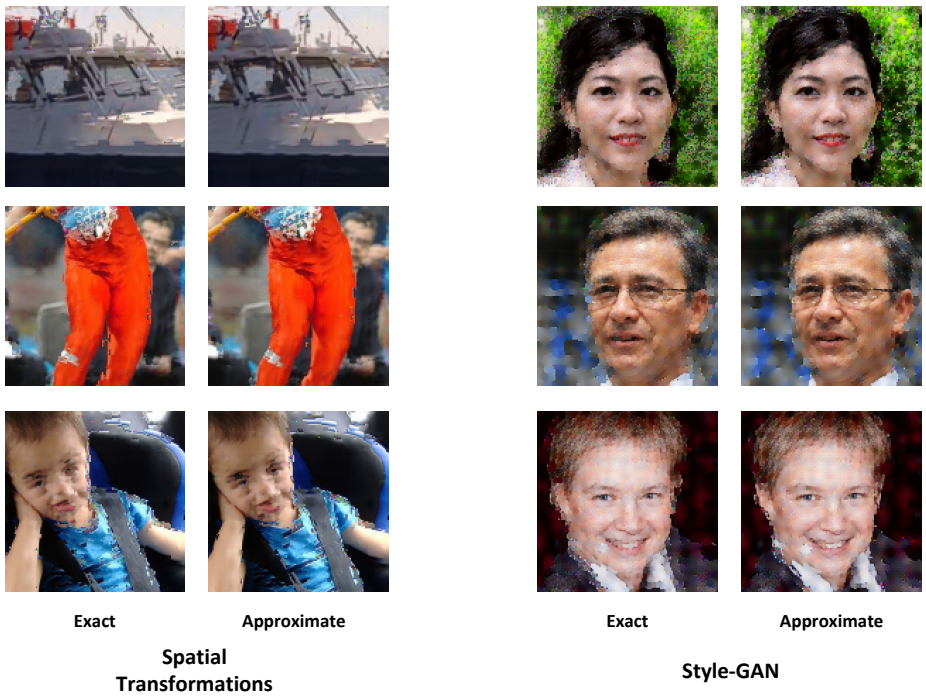

Figure 8. Qualitative comparison of the MMD loss using the exact objective and approximation with random Fourier features.