DeFlow: Learning Complex Image Degradations from Unpaired Data with Conditional Flows

Valentin Wolf   Andreas Lugmayr   Martin Danelljan   Luc Van Gool   Radu Timofte

vawolf@ethz.ch   {andreas.lugmayr, martin.danelljan, vangool, radu.timofte}@vision.ee.ethz.ch

Computer Vision Lab, ETH Zurich, Switzerland

Abstract

The difficulty of obtaining paired data remains a major bottleneck for learning image restoration and enhancement models for real-world applications. Current strategies aim to synthesize realistic training data by modeling noise and degradations that appear in real-world settings. We propose DeFlow, a method for learning stochastic image degradations from unpaired data. Our approach is based on a novel unpaired learning formulation for conditional normalizing flows. We model the degradation process in the latent space of a shared flow encoder-decoder network. This allows us to learn the conditional distribution of a noisy image given the clean input by solely minimizing the negative log-likelihood of the marginal distributions. We validate our DeFlow formulation on the task of joint image restoration and super-resolution. The models trained with the synthetic data generated by DeFlow outperform previous learnable approaches on three recent datasets. Code and trained models will be made available at: https://github.com/volflow/DeFlow

1. Introduction

Deep learning based methods have demonstrated astonishing performance for image restoration and enhancement when large quantities of paired training data are available. However, for many real-world applications, obtaining paired data remains a major bottleneck. For instance, in real-world super-resolution [23, 8, 9] and denoising [2, 3], collecting paired data is cumbersome and expensive, requiring careful setups and procedures that are difficult to scale. Moreover, such data is often limited to certain scenes and contains substantial misalignment issues. In many settings, including enhancement of existing image collections or restoration of historic photographs, the collection of paired data is even impossible.

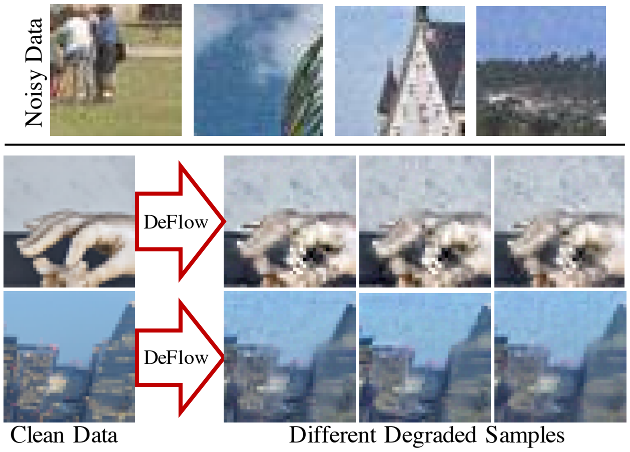


Figure 1. **DeFlow** is able to learn complex image degradation processes from unpaired training data. Our approach can sample different degraded versions of a clean input image (bottom) that faithfully resemble the noise of the real data (top).

To tackle this fundamental problem, one promising direction is to generate paired training data by applying synthesized degradations and noise to high-quality images. The degraded image then has a high-quality ground-truth, allowing effective supervised learning techniques to be applied directly to the synthesized pairs. However, in most practical applications the degradation process is *unknown*. It generally constitutes a complex combination of sensor noise, compression, and post-processing artifacts. Modeling the degradation process by hand is therefore a highly challenging problem, calling for learnable alternatives.

Since paired data is unavailable, learning the degradation process requires *unpaired* or *unsupervised* techniques. Several approaches resort to hand-crafted strategies tailored to specific types of degradations [17]. Existing learnable solutions mostly adopt generative adversarial networks (GANs) with cycle-consistency constraints [39, 23, 7] or domain-aware adversarial objectives [12, 34, 6] for unpaired training. However, these approaches require careful tuning of several losses. Moreover, cycle-consistency is a weak constraint that easily leads to changes in color and content [10]. Importantly, the aforementioned works rely on fully deterministic mappings, completely ignoring the fundamental stochasticity of natural degradations and noise. In this work, we therefore take a radically different approach.

We propose *DeFlow*: a novel conditional normalizing flow based method for learning degradations from unpaired data. *DeFlow* models the conditional distribution $p(y|x)$ of a degraded image y given its clean counterpart x. As shown in Fig. 1, this allows us to sample multiple degraded versions y of any clean image x, which closely resemble the characteristics of the unknown degradations. However, conventional conditional flow models [35, 26, 5, 1] require sample pairs (x, y) for supervised training. We therefore propose a novel formulation for conditional flows, capable of unpaired learning. Specifically, we treat the unpaired setting as the problem of learning the conditional distribution $p(y|x)$ from observations of the marginals $p(x)$ and $p(y)$. By modeling both domains x and y in the latent space of a joint flow network, we ensure sufficient constraints for effective unpaired learning while preserving flexibility for accurate modeling of $p(y|x)$. We additionally introduce a method for conditioning the flow on domain invariant information derived from either x or y to further facilitate the learning problem.

We apply our DeFlow formulation to the problem of joint image restoration and super-resolution in the real-world setting. DeFlow is tasked with learning complex image degradations, which are then used to synthesize training data for a baseline super-resolution model. We perform comprehensive experiments and analysis on the AIM2019 [25] and NTIRE2020 [24] real-world super-resolution challenge datasets. Our approach sets a new state-of-the-art among learning-based approaches by outperforming GAN-based alternatives for generating image degradations from unpaired data on three datasets.

2. Related Work

Learning degradations from unpaired data Realistic noise modeling and generation is a long-standing problem in Computer Vision research. The direction of finding learning-based solutions capable of utilizing unpaired data has received growing interest. One line of research employs generative adversarial networks (GANs) [13]. To learn from unpaired data, either cycle-consistency losses [23, 7] or domain-based adversarial losses [12, 34, 6] are employed. Yet, these approaches suffer from convergence and mode collapse issues, requiring elaborate fine-tuning of their losses. Importantly, such methods learn a deterministic mapping, ignoring the stochasticity of degradations.

Other works [21, 30, 22, 36] learn unsupervised denoising models based on the assumption of spatially uncorrelated (*i.e.* white) noise. However, this assumption does not apply to more complex degradations, which have substantial spatial correlation due to *e.g.* compression or post-processing artifacts. Our approach exploits fundamentally different constraints to allow for unpaired learning in this more challenging setting. Recently Abdelhamed *et al.* [1] proposed a conditional flow based architecture to learn noise models. Yet, their method relies on the availability of paired data for training. Moreover, the authors employ an architecture that is specifically designed to model low-level sensor noise. In contrast, we aim to model more general degradations with no available paired training data.

Unpaired Learning with Flows Whilst not for the application of learning image degradations, a few methods have investigated unpaired learning with flows. Grover *et al.* [14] trained two flow models with a shared latent space to obtain a model that adheres to exact cycle consistency. Their approach then requires an additional adversarial learning strategy based on CyCADA [15], to successfully perform domain translations. Further, Yamaguchi *et al.* [37] proposed domain-specific normalization layers for anomaly detection. As a byproduct, their approach can perform cross-domain translations on low-resolution images, by decoding an image of one domain with the normalization layer statistics of a different domain. Our proposed unpaired learning approach for flows is, however, fundamentally different from these methods. We do not rely on adversarial training nor normalization layers. Instead, we introduce a shared latent space formulation that allows unpaired learning soley by minimizing the marginal negative log-likelihood.

3. DeFlow

In this paper, we strive to develop a method for learning a mapping from samples of a source domain $x \sim p_x$ to a target domain $y \sim p_y$. While there are standard supervised learning techniques for addressing this problem, paired training datasets $\{(x_i, y_i)\}_{i=1}^n$ are not available in a variety of important real-world applications. Therefore, we tackle the *unpaired* learning scenario, where only unrelated sets of source $\mathcal{X} = \{x_i\}_{i=1}^n, x_i \sim p_x$ and target $\mathcal{Y} = \{y_i\}_{i=1}^m, y_i \sim p_y$ samples are available. While we formulate a more general approach for addressing this problem, we focus on the case where $x \sim p_x$ represent non-corrupted observations, while $y \sim p_y$ are observations affected by an unknown degradation process $x \mapsto y$. In particular, we are interested in image data.

Our aim is to capture *stochastic* degradation operations, which include noise and other random corruptions. The mapping $x \mapsto y$ therefore constitutes an unknown conditional distribution $p(y|x)$. The goal of this work is to learn a generative model $p(y|x; \theta)$ of this conditional distribution, without any paired samples (x_i, y_i).

3.1. Learning the Joint Distribution from Marginals

The unpaired learning problem defined above corresponds to the task of retrieving the conditional $p(y|x)$, or equivalently, the joint distribution $p(x, y) = p(y|x)p(x)$ given only observations from the marginals $p(x)$ and $p(y)$.

In general this is a highly ill-posed problem. However, under certain assumptions solutions can be inferred. As the most trivial case, assuming independence yields the solution $p(x, y) = p(x)p(y)$, which is not relevant since we are interested in finding correlations between x and y. Instead, we first present a simple univariate Gaussian model, which serves as an illustrative starting point for our approach. As we will see, this example forms the simplest special case of our general DeFlow formulation.

Let us assume a 1D Gaussian random variable $x \sim p_x = \mathcal{N}(\mu_x, \sigma_x^2)$ with unknown mean μ_x and variance σ_x^2. We additionally postulate that $y = x + u$, where $u \sim p_u = \mathcal{N}(\mu_u, \sigma_u^2)$ is a Gaussian random variable that is independent of x. As a sum of independent Gaussian random variables is again Gaussian, it follows that $y \sim p_y = \mathcal{N}(\mu_x + \mu_u, \sigma_x^2 + \sigma_u^2)$. Moreover, it is easy to see that $p(y|x) = \mathcal{N}(y; x + \mu_u, \sigma_u^2)$. Under these assumptions, we can estimate all unknown parameters $\theta = \{\mu_x, \sigma_x^2, \mu_u, \sigma_u^2\}$ in $p(x, y)$ by minimizing the combined negative log-likelihood of the marginal observations,

$$L(\theta) = -\frac{1}{n}\sum_{i=1}^{n} \ln p_x(x_i) - \frac{1}{m}\sum_{j=1}^{m} \ln p_y(y_j)\,. \quad (1)$$

The derivation and resulting analytic solution is given in Appendix A. This shows that inferring the full joint distribution $p(x, y)$ given only unpaired examples is possible in this simple case. Next, we generalize this example using normalizing flows to achieve a highly powerful class of models capable of likelihood-based unpaired learning.

3.2. Unpaired Learning of Conditional Flows

In this section, we introduce DeFlow, a normalizing flow based formulation capable of learning flexible conditional distributions from unpaired data. Its core idea is to model the relation between x and y in a Gaussian latent space. We then use a deep invertible encoder-decoder network to map latent variables to the output space. Our model is trained end-to-end by optimizing only the marginal log-likelihoods.

We first detail the proposed latent space formulation. Our model postulates that the random variables $x \sim p_x$ and $y \sim p_y$ are related through a shared latent space. Let z_x and z_y denote the latent variables corresponding to x and y respectively. In particular, we let $z_x \sim \mathcal{N}(0, I)$ follow a standard Normal distribution. The latent variable of y is modeled to depend on z_x, but perturbed by another Gaussian random variable $u \sim p_u = \mathcal{N}(\mu_u, \Sigma_u)$ such that $z_y = z_x + u$. The perturbation u is independent of x, and therefore also of z_x. The mean μ_u and covariance Σ_u of u are unknown. Note that, our latent space model is the multivariate generalization of the example presented in Sec. 3.1.

As the next step we use a powerful deep network, capable of disentangling complex patterns and correlations of *e.g.* images to the Gaussian latent space. In particular, we model this relation between the observations and the latent space with an invertible neural network f_θ. Our complete model is then summarized as,

$$x = f_\theta^{-1}(z_x)\,, \quad y = f_\theta^{-1}(z_y) = f_\theta^{-1}(z_x + u) \quad (2a)$$

$$z_x \sim \mathcal{N}(0, I)\,, \quad u \sim p_u = \mathcal{N}(\mu_u, \Sigma_u)\,, \quad z_x \perp u\,. \quad (2b)$$

Here, $\perp$ denotes stochastic independence. Note, that we can sample from the joint distribution by directly applying (2). More importantly, we can also easily sample from the conditional distribution $y_{|x} \sim p(y|x)$. The invertibility of f_θ implies $p(y|x) = p(y|z_x)$. From (2), we thus achieve,

$$y_{|x} = f_\theta^{-1}(f_\theta(x) + u) \sim p(y|x)\,, \quad u \sim \mathcal{N}(\mu_u, \Sigma_u)\,. \quad (3)$$

In words, $y_{|x}$ is obtained by first encoding $z_x = f_\theta(x)$ then sampling and adding u before decoding again.

To train DeFlow with the likelihood-based objective from (1), we employ the differentiable expressions of the marginal probability densities $p_x(x)$ and $p_y(y)$. The invertible normalizing flow f_θ allows us to apply the change of variables formula in order to achieve the expressions,

$$p_x(x) = \left|\det Df_\theta(x)\right| \cdot \mathcal{N}(f_\theta(x); 0, I) \quad (4a)$$

$$p_y(y) = \left|\det Df_\theta(y)\right| \cdot \mathcal{N}(f_\theta(y); \mu_u, I + \Sigma_u)\,. \quad (4b)$$

In both cases, the first factor is given by the determinant of the Jacobian Df_θ of the flow network. The second factors stem from the Gaussian latent space distribution of z_x and z_y, respectively. For an in depth explanation of this fundamental step of normalizing flows we refer the reader to Eq. (1) in [20]. It follows from (3), that $f_\theta(y_{|x}) = f_\theta(x) + u$. Therefore, we can derive the conditional density, again using change of variables, as

$$p(y|x) = \left|\det Df_\theta(y)\right| \cdot \mathcal{N}(f_\theta(y); f_\theta(x) + \mu_u, \Sigma_u)\,. \quad (5)$$

Using (4), our model can be trained by minimizing the negative log-likelihood of the marginals (1) in the unpaired setting. Furthermore, the conditional likelihood (5) also enables the use of paired samples, if available. Our approach can thus operate in both the paired and unpaired setting.

It is worth noting that the 1D Gaussian example presented in Sec. 3.1 is retrieved as a special case of our model by setting the flow f_θ to the affine map $x = f_\theta^{-1}(z) = \sigma_x z + \mu_x$. The deep flow f_θ thus generalizes our initial example beyond the Gaussian case such that complex correlations and dependencies in the data can be captured. In the case of modeling image degradations our formulation has a particularly intuitive interpretation. The degradation process $x \mapsto y$ can follow a complex and signal-dependent distribution in the image space. Our approach thus learns the bijection f_θ that maps the image to a space where this degradation can be modeled by additive Gaussian noise u.

This is most easily seen by studying (3), which implements the stochastic degradation $x \mapsto y$ for our model. The clean data x is first mapped to the latent space and then corrupted by the random Gaussian ‘noise’ u. Finally, the degraded image is reconstructed with the inverted mapping f_θ^{-1}.

Lastly, we note that our proposed model achieves conditioning through a very different mechanism compared to conventional conditional flows [35, 26, 5, 1]. These works learn a flow network that is directly conditioned on x as $z = f_\theta(y; x)$. Thus, a generative model of x is not learned. However, these methods rely on paired data since both x and y are simultaneously required to compute z and its likelihood. In contrast, our approach learns the full joint distribution $p(x, y)$ and uses an unconditional flow network. The conditioning is instead performed by our latent space model (2). However, we show next that our approach can further benefit from the conventional technique of conditional flows, without sacrificing the ability of unpaired learning.

3.3. Domain Invariant Conditioning

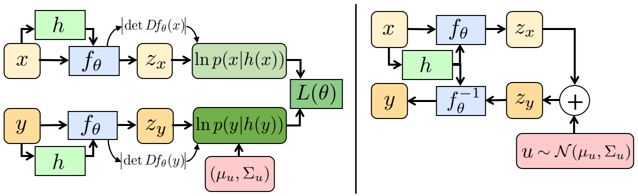
The formulation presented in Sec. 3.2 requires learning the marginal distributions p_x and p_y. For image data, this is a difficult task, requiring a large model capacity and big datasets. In this section, we therefore propose a further generalization of our formulation, which effectively circumvents the need for learning the full marginals and instead allows the network to focus on accurately learning the conditional distribution $p(y|x)$.

Our approach is based on conditioning the flow model on auxiliary information $h(x)$ or $h(y)$. Here, h represents a known mapping from the observation space to a conditional variable. We use the conventional technique for creating conditional flows [35, 26, 5] by explicitly inputting $h(x)$ into the individual layers of the flow network f_θ (as detailed in Sec. 4.1). The flow is thus a function $z_x = f_\theta(x; h(x))$ that is invertible only in the first argument. Instead of the marginal distributions in (4), our approach thus models the conditional densities $p(x|h(x))$. Since h is a known function, we can still learn $p(x|h(x))$ and $p(y|h(y))$ without paired data. Importantly, learning $p(x|h(x))$ is an *easier* problem since information in $h(x)$ does not need modeling.

In order to ensure unpaired learning of the conditional distribution $p(y|x)$, the map h must satisfy an important criterion. Namely, that h only extracts *domain invariant* information about the sample. Formally, this is written as,

$$h(x) = h(y)\,, \quad (x, y) \sim p(x, y)\,. \tag{6}$$

It is easy to verify the existence of such a function h by taking $h(x) = 0$ for all x. This choice, where h carries no information about the input sample, retrieves the formulation presented in Sec. 3.2. Intuitively, we wish to find a function h that preserves the most information about the input, without violating the domain invariance condition (6). Since the joint distribution $p(x, y)$ is unknown, strictly ensuring (6) is a difficult problem. In practice, however, we only need h to satisfy domain invariance to the degree where it cannot be exploited by the flow network f_θ. The conditioning function h can thus be set empirically by gradually reducing its preserved information. We detail strategies for designing h for learning image degradations in Sec. 4.2.



(a) Training (b) Sampling

Figure 2. (a) DeFlow is trained to minimize the loss $L(\theta)$ in (7). Unpaired inputs x and y are separately encoded by the flow f_θ to evaluate the NLL conditioned on h. (b) We sample $y \sim p(y|x)$ using (8) by first encoding x, then adding the sampled noise u in the latent space and finally decoding it with the inverse flow f_θ^{-1}.

The formulation in Sec. 3.2 is easily generalized to the case that includes the domain invariant conditioning h by simply extending the flow network as $z_x = f_\theta(x; h(x))$ and $z_y = f_\theta(y; h(y))$. The training and inference stages of our resulting DeFlow formulation are visualized in Figure 2. The model is trained by minimizing the negative log-likelihood conditioned on h,

$$L(\theta) = -\frac{1}{n}\sum_{i=1}^{n} \ln p(x_i|h(x_i)) - \frac{1}{m}\sum_{j=1}^{m} \ln p(y_j|h(y_j)). \tag{7}$$

During inference, we sample from the conditional distribution $p(y|x)$ using,

$$y = f_\theta^{-1}\big(f_\theta(x; h(x)) + u;\ h(x)\big)\,, \quad u \sim \mathcal{N}(\mu_u, \Sigma_u)\,. \tag{8}$$

To avoid repetition, we include a detailed derivation of the generalized formulation in Appendix C.

4. Learning Image Degradations with DeFlow

In this section we discuss the application of our flow-based unpaired learning formulation to the problem of generating complex image degradations. We detail the model architecture used by DeFlow and explain our approach for obtaining domain invariant conditioning in this setting.

4.1. Model Architecture

Flow models are generally implemented as a composition of N invertible layers. Let f_θ^n denote the n-th layer. Then the model can be expressed recursively as

$$a^n = f_\theta^n(a^{n-1}; h(x)) \tag{9}$$

where $x = a^0$, $z = a^N$ and the remaining a^n represent intermediate feature maps. By the chain rule, (4) gives

$$p(x|h(x)) = p(z) \cdot \prod_{n=1}^{N} \left| \det D f_\theta^n(a^n; h(x)) \right| \tag{10}$$

allowing for efficient log-likelihood optimization.

We parametrize the distribution p_u in (2) with mean μ_u the weight matrix M, such that $u = M\tilde{u} + \mu_u$ where $\tilde{u} \sim \mathcal{N}(0, I)$ is a standard Gaussian. Consequently, the covariance is given by $\Sigma_u = MM^T$. To ensure spatial invariance, we use the same parameters μ_u and M at each spatial location in the latent space. We initialize both μ_u and M to zero, ensuring that $p(x)$ and $p(y)$ initially follow the same distribution.

Our DeFlow formulation for unsupervised conditional modeling can in principle be integrated into *any* (conditional) flow architecture f_θ. We start from the recent SRFlow [26] network architecture, which itself is based on the unconditional Glow [19] and RealNVP [11] models. We use an $L = 3$ level network. Each level starts with a *squeeze* operation that halves the resolution. It is followed by K flow steps, each consisting of four different layers. The level ends with a *split*, which removes a fraction of the activations as a latent variable. In our experiments we use $K = 16$ flow steps, unless specified otherwise. Next, we give a brief description of each layer in the architecture and discuss our modifications. Please, see [26, 19] for details.

Conditional Affine Coupling [26]: extends the affine coupling layer from [11] to the conditional setting. The input feature map a is split into two parts (a_1, a_2) along the channel dimension. From the subset a_1 and the conditional $h(x)$, a scaling and bias is computed using an arbitrary neural network. These are then applied to the other subset a_2 providing an invertible yet flexible transformation.

Affine injector [26]: computes an individual scaling and bias for each entry of the input feature map a from the conditional $h(x)$. The function computing the scaling and bias is not required to be invertible, enabling $h(x)$ to have direct influence on all channels.

Invertible 1x1 Convolution [19]: multiplies each spatial location with an invertible matrix. We found the LU-decomposed parametrization [19] to improve the stability and conditioning of the model.

Actnorm [19]: learns a channel-wise scaling and shift to normalize intermediate feature maps.

Flow Step: is the block of flow layers that is repeated throughout the network. Each flow step contains the above mentioned four layers. First, an Actnorm is applied, followed by the 1×1 convolution, Conditional Affine Coupling, and the Affine Injector. Note, that the last two layers are applied not only in reverse order but also in their inverted form compared to the Flow Step in SRFlow [26].

Feature extraction network: we encode the domain-invariant conditional information h using the low-resolution encoder employed by SRFlow. It consists of a modified Residual-in-Residual Dense Blocks (RRDB) model [32]. For our experiments, we initialize it with pretrained weights provided by the authors of [32]. Although this network was originally intended for super-resolution, it is here employed for an entirely different task, namely to encode domain-invariant information h for image degradation learning.

4.2. Domain-Invariant Mapping h

The goal of our domain-invariant conditioning h is to provide image information to the flow network, while hiding the domain of the input image. In our application, the domain invariance (6) implies that the mapping h needs to remove information that could reveal whether input is a clean x or a degraded y image. On the other hand, we want to preserve information about the underlying image content to simplify learning. We accomplish this by utilizing some prior assumptions that are valid for most stochastic degradations. Namely, that they mostly affect the high frequencies in the image, while preserving the low frequencies.

We construct h by down-sampling the image to a sufficient extent to remove the visible impact of the degradations. We found it beneficial to also add a small amount of noise to the resulting image to hide remaining traces of the original degradation. The domain invariant mapping is thus constructed as $h(x) = d_\downarrow(x) + n$, $n \sim \mathcal{N}(0, \sigma^2)$, where $d_\downarrow(x)$ denotes bicubic downsampling. Note that this operation is only performed to extract a domain-invariant representation, and is not related to the degradation $x \mapsto y$ learned by DeFlow. The purpose of h is to *remove* the original degradation, while preserving image content.

5. Experiments and Results

We validate the degradations learned by DeFlow by applying them to the problem of real-world super-resolution (RWSR). Here, the task is to train a joint image restoration and super-resolution model without paired data that is able to translate degraded low-resolution images to high-quality and high-resolution images. In particular, we employ DeFlow to learn the underlying degradation model and use it to generate paired training data for a supervised super-resolution model. Experiments are performed on three recent benchmarks designed for this setting. Detailed results with more visual examples are shown in Appendix D.

5.1. Datasets

AIM-RWSR: Track 2 of the AIM 2019 RWSR challenge [25] provides a dataset consisting of a source and a target domain. The former contains synthetically degraded images from the Flickr2k dataset [31] that feature some combination of noise and compression, while the latter contains the

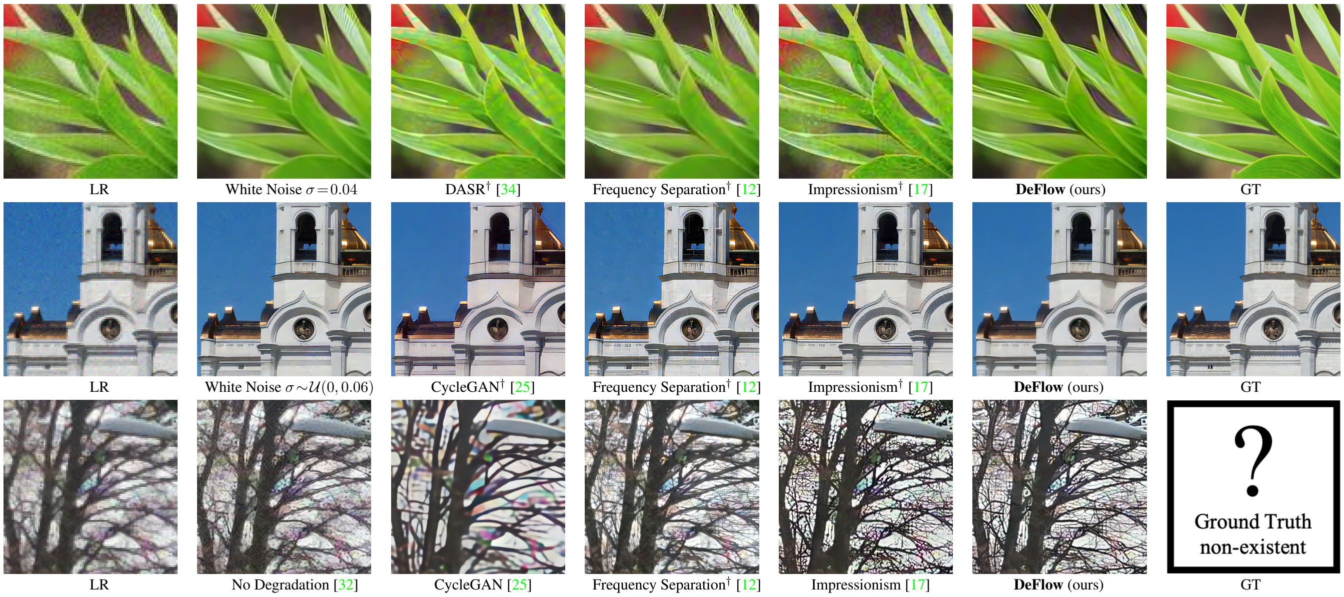


Figure 3. Super-resolved images from the AIM-RWSR (top), NTIRE-RWSR (mid) and DPED-RWSR (bottom) datasets. Top-5 methods are shown based on LPIPS score for the synthetic datasets and the visual judgement of the authors for the DPED-RWSR dataset.

high-quality non-degraded images of the DIV2k dataset [4]. The task is to $4\times$ super-resolve images from the source domain to high-quality images as featured in the target domain. Since the degradations were generated synthetically, there exists a validation set of 100 paired degraded low-resolution and high-quality ground-truth images, allowing the use of reference-based evaluation metrics.

NTIRE-RWSR: Track 1 of the NTIRE 2020 RWSR challenge [24] follows the same setting as AIM-RWSR. However, it features a completely different type of degradation, namely highly correlated high-frequency noise. As before, a validation set exists enabling a reference-based evaluation.

DPED-RWSR: Differing from the other two datasets, the source domain of Track 2 of the NTIRE 2020 RWSR challenge consists of real low-quality smartphone photos that are to be jointly restored and super-resolved. A high-quality target domain dataset is also provided. The source domain stems from the iPhone3 images of the DPED dataset [16], while the target domain corresponds to the DIV2k [4] training set. Because reference images do not exist evaluation is restricted to no-reference metrics and visual inspection.

5.2. Evaluation Metrics

For the synthetic datasets, we report the peak signal-to-noise ratio (PSNR) and the structural similarity index (SSIM) [33]. In addition we compute the Learned Perceptual Image Patch Similarity (LPIPS) [38] metric, a reference-based image quality metric based on feature distances in CNNs. As LPIPS has been shown to correlate well with human perceived image quality, we consider it the most important metric for the RWSR task. For the DPED-RWSR we report the NIQE [28], BRISQUE [27] and PIQE [29] no-reference metrics. We also conduct a user study comparing the best models with our DeFlow method. For each compared method, we show participants randomly selected crops super-resolved by both DeFlow and the compared method. Users are then asked to select the more realistic image. We report user preference as the percentage of images where the compared method was preferred over DeFlow. A User Preference $<50\%$ indicates that DeFlow obtains 'better' images than the comparison method. More details on the user study are provided in Appendix F.

5.3. Baselines and other Methods

We compare DeFlow against Impressionism [17] the winner of the NTIRE 2020 RWSR challenge [24] and Frequency Separation [12] the winner of the AIM 2019 RWSR challenge [25]. Further, we compare with the very recent DASR [34] and the CycleGan based method introduced in [23]. All aforementioned methods apply the same two-stage approach, where first a degradation model is learned to generate synthetic training data that is then used to train a supervised ESRGAN [32] based super-resolution model. We also validate against simple baselines. Our *No Degradation* baseline is trained without any degradation model. The *White Noise* model adds zero mean Gaussian noise to the low-resolution patches during training. Here, we tested two variants, either fixing the standard deviation σ or sampling the standard deviation of the noise added to each image patch uniformly from $\mathcal{U}(0, \sigma_{max})$. For each dataset we tuned both variants with different choices of σ and σ_{max}, respectively, and only report the model that obtained the best LPIPS score.

	PSNR↑	SSIM↑	LPIPS↓	User Pref.
CycleGan [25]	21.19	0.53	0.476	-
Frequency Separation [12]	21.00	0.50	0.403	38.89%
DASR [34]	21.79	0.58	0.346	35.74%
No Degradation	21.82	0.56	0.514	-
Impressionism† [17]	**22.54**	0.63	0.420	27.58%
White Noise $\sigma = 0.04$	22.43	**0.65**	0.406	30.00%
Frequency Separation† [12]	20.47	0.52	0.394	39.37%
DASR† [34]	21.16	0.57	0.370	40.26%
DeFlow (ours)	22.25	0.62	**0.349**	reference

Table 1. AIM-RWSR results: Methods in the bottom segment use the same SR pipeline. User preferences are green if the method and red if DeFlow was preferred by the majority. Orange indicates a result within the 95% confidence interval.

	PSNR↑	SSIM↑	LPIPS↓	User Pref.
CycleGan[25]	24.75	0.70	0.417	35.78%
Impressionism [17]	24.77	0.67	0.227	54.11%
No Degradation	20.59	0.34	0.659	-
Frequency Separation† [12]	23.04	0.59	0.332	46.17%
CycleGan† [25]	22.62	0.60	0.314	44.72%
White Noise $\sigma \sim \mathcal{U}(0, 0.06)$	25.47	**0.71**	0.237	53.28%
Impressionism† [17]	25.03	0.70	0.226	56.44%
DeFlow (ours)	**25.87**	**0.71**	**0.218**	reference

Table 2. NTIRE-RWSR results: see caption in Tab. 1.

	NIQE ↓	BRISQUE↓	PIQE↓	User Pref.
CycleGAN [25]	5.47	49.19	86.83	-
Frequency Separation [12]	3.27	22.73	11.88	36.88%
Impressionism [17]	4.12	23.24	14.09	54.13%
No Degradation	3.55	24.56	**8.01**	-
KernelGAN† [6]	6.37	42.74	30.32	-
Frequency Separation† [12]	**3.39**	25.40	11.22	37.48%
Impressionism† [17]	3.85	21.49	12.84	50.72%
DeFlow (ours)	3.42	**21.13**	15.84	reference

Table 3. DPED-RWSR results: see caption in Tab. 1.

5.4. Training Details

We train all DeFlow models for 100k iterations using the Adam [18] optimizer. The initial learning rate is set to $5 \cdot 10^{-5}$ on the synthetic datasets and to $5 \cdot 10^{-6}$ on the DPED-RWSR dataset and is halved at 50k, 75k, 90k and 95k iterations. We use a batch size of 8 with random crops of size 160×160 on the AIM-RWSR and NTIRE-RWSR dataset. On DPED-RWSR we obtained better performance with a patch size of 80×80 and a batch size of 48. Batches are sampled randomly such that images of both domains are drawn equally often. Random flips are used as a data augmentation. We use 64 hidden channels in the affine injector layer for NTIRE-RWSR and DPED-RWSR and 128 on AIM-RWSR. Similar to [19, 26], we apply a 5-bit de-quantization by adding uniform noise to the input of the flow model. We train the DeFlow models using the 4× bicubic downsampled clean domain as $\mathcal{X}$ and the noisy domain as $\mathcal{Y}$. Given the large domain gap between the source and target images in DPED-RWSR we do not use the target images and instead use 4× and 8× bicubic downsampled noisy images as the clean domain $\mathcal{X}$. For DPED-RWSR we further follow the approach of [17] and estimate blur kernels of the degraded domain using KernelGAN [6]. These are then applied to any data from the clean domain, *i.e.* on the clean training data and before degrading images. On AIM-RWSR we normalize $\mathcal{X}$ and $\mathcal{Y}$ to the same channel-wise means and standard deviations. Degraded images are then de-normalized before employing them as training data for the super-resolution model. For the conditional $h(x)$ we used $\sigma = 0.03$ in conjunction with 4× bicubic downsampling on NTIRE-RWSR and DPED-RWSR and 8× bicubic downsampling on AIM-RWSR.

5.5. Super-Resolution Model

To fairly compare with existing approaches, we an ESRGAN [32] as the super-resolution model. Specifically, we employ the training code provided by the authors of Impressionism [17] that trains a standard ESRGAN for 60k iterations. For AIM-RWSR and NTIRE-RWSR the standard VGG discriminator is used while on DPED-RWSR a patch discriminator is applied. As in [17], we use the 2× downsampled smartphone images of the DPED-RWSR dataset as clean images and do not use the provided high-quality data. Unlike [17] however, we do not use any downsampled noisy images as additional clean training data. We evaluate the trained models after 10k, 20k, 40k and 60k iterations and report the model with the best LPIPS on the validation set. For DPED-RWSR we simply choose the final model. To better isolate the impact of the learned degradations, we further report the performance of other methods when using their degradation pipeline with our super-resolution model. We mark these models with the † symbol.

5.6. Comparison with State-of-the-Art

First, we discuss the results on the AIM-RWSR dataset shown in Tab. 1. The GAN-based Frequency Separation approach [12], the winner of this dataset's challenge, obtains an LPIPS similar to the White Noise baseline. DASR [34] obtains a highly competitive LPIPS, yet it is strongly outperformed by DeFlow in our user study. In fact, as shown in Fig. 3, DASR generates strong artifacts. This can be explained by overfitting, as DASR directly optimizes for LPIPS during training. When using the degradation model of DASR in conjunction with our super-resolution pipeline the resulting model DASR† performs slightly better in the user study while obtaining an LPIPS score of 0.370 compared to DeFlow's 0.349. Notably, DeFlow outperforms all previous methods by a large margin in the user study. It also obtains a higher PSNR and SSIM than all methods with learned, but GAN based degradation models.

On the NTIRE-RWSR dataset (see Tab. 2) DeFlow obtains the best scores among all reference metrics, making it the only model that consistently outperforms the White

Noise baseline. In the user study DeFlow is also preferred to all learned degradation models. Yet, the user study indicates better quality from the hand-crafted degradation models, namely Impressionism and the White Noise baseline, compared to the learned approach of DeFlow. However, as shown in the second row of Fig. 3, the White Noise baseline generates highly visible artifacts in smooth regions, *e.g.* sky, whereas DeFlow removes all noise from these areas.

Lastly, we compare the results on the DPED-RWSR dataset in Tab. 3. Similar to [24], we find that the no-reference metrics do not correlate well with the perceived quality of the images. As shown in Fig. 3, DeFlow obtains sharp images with pleasing details clearly outperforming all other learned approaches. Compared to Impressionism [17], we find that our method produces fewer artifacts and does not over-smooth textures. However, we notice that our images retain more noise and are sometimes less sharp. This is supported by the user study where DeFlow significantly outperforms the Frequency Separation method [12], while being head-to-head with Impressionism† [17].

Overall, DeFlow is the only method with consistently good performance across all three datasets, whereas the handcrafted approaches obtain the worst performance on the AIM-RWSR dataset and the other learned approaches are struggling to create artifact-free yet detailed images on the NTIRE-RWSR dataset. It is also noteworthy that CycleGAN [25], despite its immense popularity for unpaired learning, does not perform well on any of these datasets. This can be partly explained by the weak cycle consistency constraint and the use of a deterministic generator.

5.7. Ablation Study

In this section, we analyze DeFlow through an ablation study. We train a variety of models on the AIM-RWSR dataset and evaluate their downstream super-resolution performance. These models deviate only in the choice of a single hyper-parameter with all other training settings remaining as described in 5.4. In particular, we scrutinize on three core segments: the depth of the model, the choice of conditioning $h(x)$, and the method of learning the domain shift. For each segment we show the results of this study in a separate section of Tab. 4.

Network depth (Tab. 4, top): Increasing the number of Flow Steps K improves performance, showing that indeed powerful networks help to learn the complex degradations.

Conditioning (Tab. 4, middle): Next we analyze the impact of the domain invariant conditioning $h(x)$ (Sec. 3.3). Using $4\times$ downsampling in the conditional yields noticeable worse performance compared to larger factors. We conclude that larger downsampling factors are required to ensure the domain invariance of $h(x)$. Notably, $16\times$ downsampling yields only a slight performance reduction compared to $8\times$ downsampling. In contrast, no conditional information at all *i.e.* $h(x) = 0$ leads to a significantly worse performance where the translated images exhibits strong color shifts and blur. This highlights the importance of the conditional and shows that even little auxiliary information yields drastic performance improvements.

	PSNR↑	SSIM↑	LPIPS↓
$K = 4$ Flow Steps	22.18	0.61	0.362
$K = 8$ Flow Steps	22.20	0.63	0.355
$K = 16$ Flow Steps	22.25	0.62	0.349
$4\times$ downsampling in $h(x)$	22.44	0.61	0.429
$8\times$ downsampling in $h(x)$	22.25	0.62	0.349
$16\times$ downsampling in $h(x)$	21.52	0.61	0.352
No Conditional $h(x) = 0$	18.33	0.52	0.412
Non-learned Shift	22.04	0.62	0.405
Learned uncorrelated Shift	22.04	0.61	0.359
Learned correlated Shift	22.25	0.62	0.349

Table 4. Ablation study of DeFlow on the AIM-RWSR dataset. The final setting of the DeFlow model for AIM-RWSR are in **bold**.

Learned shift (Tab. 4, bottom): Last, we investigate our latent space formulation. We first restrict the added noise $u \sim p_u$ to be uncorrelated across the channels by constraining Σ_u to a diagonal covariance matrix. We notice a negative impact on performance. This demonstrates the effectiveness of our more general Gaussian latent space model. Further, we validate our choice of using domain dependent base distributions. We train a DeFlow model with a standard normal Gaussian as the base distribution for both domains (*i.e.* setting $u = 0$ in (2)). We then infer the domain shift after training by computing the channel-wise mean and covariance matrix in the latent space for each domain. The resulting empirical distributions of both domains become very similar and the inferred shift does no longer model the domain shift faithfully. This results in a substantially worse performance in the down-stream task and further shows the potential of our unpaired learning formulation.

6. Conclusion

We propose DeFlow, a method for learning conditional flow networks with unpaired data. Through a constrained latent space formulation, DeFlow learns the conditional distribution by minimizing the marginal negative log-likelihoods. We further generalize our approach by conditioning on domain invariant information. We apply DeFlow to the unsupervised learning of complex image degradations, where the resulting model is used for generating training data for the downstream task of real-world super-resolution. Our approach achieves state-of-the-art results on three challenging datasets.

Acknowledgements This work was partly supported by the ETH Zürich Fund (OK), a Huawei Technologies Oy (Finland) project, an Amazon AWS grant, a Microsoft Azure grant, and a Nvidia hardware grant.

References

[1] Abdelrahman Abdelhamed, Marcus A. Brubaker, and Michael S. Brown. Noise flow: Noise modeling with conditional normalizing flows. In *ICCV*, 2019. 2, 4

[2] Abdelrahman Abdelhamed, Stephen Lin, and Michael S. Brown. A high-quality denoising dataset for smartphone cameras. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1

[3] Abdelrahman Abdelhamed, Radu Timofte, Michael S. Brown, et al. Ntire 2019 challenge on real image denoising: Methods and results. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2019. 1

[4] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *CVPR Workshops*, 2017. 6

[5] Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Guided image generation with conditional invertible neural networks. *CoRR*, abs/1907.02392, 2019. 2, 4

[6] Sefi Bell-Kligler, Assaf Shocher, and Michal Irani. Blind super-resolution kernel estimation using an internal-gan. In *Advances in Neural Information Processing Systems*, pages 284–293, 2019. 1, 2, 7

[7] Adrian Bulat, Jing Yang, and Georgios Tzimiropoulos. To learn image super-resolution, use a gan to learn how to do image degradation first. *arXiv preprint arXiv:1807.11458*, 2018. 1, 2

[8] Jianrui Cai, Shuhang Gu, Radu Timofte, and Lei Zhang. Ntire 2019 challenge on real image super-resolution: Methods and results. In *CVPR Workshops*, June 2019. 1

[9] Jianrui Cai, Hui Zeng, Hongwei Yong, Zisheng Cao, and Lei Zhang. Toward real-world single image super-resolution: A new benchmark and a new model. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 1

[10] Casey Chu, Andrey Zhmoginov, and Mark Sandler. Cyclegan, a master of steganography. *arXiv preprint arXiv:1712.02950*, 2017. 1

[11] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. 5

[12] M. Fritsche, S. Gu, and R. Timofte. Frequency separation for real-world super-resolution. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3599–3608, 2019. 1, 2, 6, 7, 8

[13] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014. 2

[14] Aditya Grover, Christopher Chute, Rui Shu, Zhangjie Cao, and Stefano Ermon. Alignflow: Cycle consistent learning from multiple domains via normalizing flows, 2019. 2

[15] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213*, 2017. 2

[16] Andrey Ignatov, Nikolay Kobyshev, Radu Timofte, Kenneth Vanhoey, and Luc Van Gool. Dslr-quality photos on mobile devices with deep convolutional networks. In *ICCV*, 2017. 6

[17] Xiaozhong Ji, Yun Cao, Ying Tai, Chengjie Wang, Jilin Li, and Feiyue Huang. Real-world super-resolution via kernel estimation and noise injection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020. 1, 6, 7, 8

[18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 7

[19] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pages 10236–10245, 2018. 5, 7

[20] I. Kobyzev, S. Prince, and M. Brubaker. Normalizing flows: An introduction and review of current methods. *TPAMI*, 2020. 3

[21] Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2void - learning denoising from single noisy images. *CoRR*, abs/1811.10980, 2018. 2

[22] Samuli Laine, Tero Karras, Jaakko Lehtinen, and Timo Aila. High-quality self-supervised deep image denoising, 2019. 2

[23] Andreas Lugmayr, Martin Danelljan, and Radu Timofte. Unsupervised learning for real-world super-resolution. In *ICCV Workshops*, 2019. 1, 2, 6

[24] Andreas Lugmayr, Martin Danelljan, and Radu Timofte. Ntire 2020 challenge on real-world image super-resolution: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020. 2, 6, 8

[25] Andreas Lugmayr, Martin Danelljan, Radu Timofte, et al. Aim 2019 challenge on real-world image super-resolution: Methods and results. In *ICCV Workshops*, 2019. 2, 5, 6, 7, 8

[26] Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. Srflow: Learning the super-resolution space with normalizing flow. In *ECCV*, 2020. 2, 4, 5, 7

[27] A Mittal, AK Moorthy, and AC Bovik. Referenceless image spatial quality evaluation engine. In *45th Asilomar Conference on Signals, Systems and Computers*, volume 38, pages 53–54, 2011. 6

[28] Anish Mittal, Rajiv Soundararajan, and Alan C. Bovik. Making a ”completely blind” image quality analyzer. *IEEE Signal Process. Lett.*, 20(3):209–212, 2013. 6

[29] Venkatanath N., Praneeth D., Maruthi Chandrasekhar Bh., Sumohana S. Channappayya, and Swarup S. Medasani. Blind image quality evaluation using perception based features. In *NCC*, pages 1–6. IEEE, 2015. 6

[30] Mangal Prakash, Manan Lalit, Pavel Tomancak, Alexander Krull, and Florian Jug. Fully unsupervised probabilistic noise2void, 2020. 2

[31] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, Lei Zhang, Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, Kyoung Mu Lee, et al. Ntire 2017 challenge on single image super-resolution: Methods and results. *CVPR Workshops*, 2017. 5

[32] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Chen Change Loy, Yu Qiao, and Xiaoou Tang. Esrgan: Enhanced super-resolution generative adversarial networks. *ECCV*, 2018. 5, 6, 7

[33] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Processing*, 13(4):600–612, 2004. 6

[34] Yunxuan Wei, Shuhang Gu, Yawei Li, and Longcun Jin. Unsupervised real-world image super resolution via domain-distance aware training, 2020. 1, 2, 6, 7

[35] Christina Winkler, Daniel Worrall, Emiel Hoogeboom, and Max Welling. Learning likelihoods with conditional normalizing flows, 2019. 2, 4

[36] Xiaohe Wu, Ming Liu, Yue Cao, Dongwei Ren, and Wangmeng Zuo. Unpaired learning of deep image denoising, 2020. 2

[37] Masataka Yamaguchi, Yuma Koizumi, and Noboru Harada. Adaflow: Domain-adaptive density estimator with application to anomaly detection and unpaired cross-domain translation, 2019. 2

[38] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *CVPR*, 2018. 6

[39] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *ICCV*, 2017. 1