Tunable Convolutions with Parametric Multi-Loss Optimization

Matteo Maggioni, Thomas Tanay, Francesca Babiloni, Steven McDonagh, Aleš Leonardis
Huawei Noah's Ark Lab
{matteo.maggioni, thomas.tanay, francesca.babiloni, steven.mcdonagh, ales.leonardis}@huawei.com

Abstract

Behavior of neural networks is irremediably determined by the specific loss and data used during training. However it is often desirable to tune the model at inference time based on external factors such as preferences of the user or dynamic characteristics of the data. This is especially important to balance the perception-distortion tradeoff of ill-posed image-to-image translation tasks. In this work, we propose to optimize a parametric tunable convolutional layer, which includes a number of different kernels, using a parametric multi-loss, which includes an equal number of objectives. Our key insight is to use a shared set of parameters to dynamically interpolate both the objectives and the kernels. During training, these parameters are sampled at random to explicitly optimize all possible combinations of objectives and consequently disentangle their effect into the corresponding kernels. During inference, these parameters become interactive inputs of the model hence enabling reliable and consistent control over the model behavior. Extensive experimental results demonstrate that our tunable convolutions effectively work as a drop-in replacement for traditional convolutions in existing neural networks at virtually no extra computational cost, outperforming state-of-the-art control strategies in a wide range of applications; including image denoising, deblurring, super-resolution, and style transfer.

1. Introduction

Neural networks are commonly trained by optimizing a set of learnable weights against a pre-defined loss function, often composed of multiple competing objectives which are delicately balanced together to capture complex behaviors from the data. Specifically, in vision, and in image restoration in particular, many problems are ill-posed, *i.e*. admit a potentially infinite number of valid solutions [15]. Thus, selecting an appropriate loss function is necessary to constrain neural networks to a specific inference behavior [39, 64]. However, any individual and fixed loss defined empirically before training is inherently incapable of generating optimal results for any possible input [43]. A classic example is the difficulty in finding a good balance for the perception-distortion trade-off [5, 64], as shown in the illustrative example of Fig. 1. The solution to this problem is to design a mechanism to reliably control (*i.e*. *tune*) neural networks at inference time. This comes with several advantages, namely providing a flexible behavior without the need to retrain the model, correcting failure cases on the fly, and balancing competing objectives according to user preference.

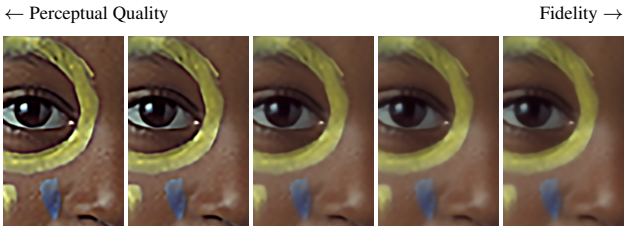

Figure 1. We propose a framework to build a single neural network that can be *tuned* at inference without retraining by interacting with controllable parameters, *e.g*. to balance the perception-distortion tradeoff in image restoration tasks.

Existing approaches to control neural networks, commonly based on weights [54, 55] or feature [53, 63] modulation, are fundamentally limited to consider only two objectives, and furthermore require the addition of a new set of layers or parameters for every additional loss considered. Different approaches, specific to image restoration tasks, first train a network conditioned on the true degradation parameter of the image, *e.g*. noise standard deviation or blur size, and then, at inference time, propose to interact with these parameters to modulate the effects of the restoration [17, 24, 50]. However this leads the network to an undefined state when asked to operate in regimes corresponding to combinations of input and parameters unseen during training [27].

In this work, we introduce a novel framework to reliably and consistently tune model behavior at inference time. We propose a parametric dynamic layer, called tunable convolution, consisting in p individual kernels (and biases) which we optimize using a parametric dynamic multi-loss, con-

sisting in p individual objectives. Different parameters can be used to obtain different combinations of kernels and objectives by linear interpolation. The key insight of our work is to establish an explicit link between the p kernels and objectives using a *shared* set of p parameters. Specifically, during training, these parameters are randomly sampled to explicitly optimize the complete loss landscape identified by all combinations of the p objectives. As a result, during inference, each individual objective is disentangled into a different kernel, and thus its influence can be controlled by interacting with the corresponding parameter of the tunable convolution. In contrast to previous approaches, our strategy is capable of handling an arbitrary number of objectives, and by explicitly optimizing all their intermediate combinations, it allows to tune the overall network behavior in a predictable and intuitive fashion. Furthermore, our tunable layer can be used as a drop-in replacement for standard layers in existing neural networks with negligible difference in computational cost.

In summary the main contributions of our work are:

- A novel plug-and-play tunable convolution capable to reliably control neural networks through the use of interactive parameters;
- A unique parametric multi-loss optimization strategy dictating how tunable convolution should be optimized to disentangle the different objectives into the different tunable kernels;
- Extensive experimental validation across several image-to-image translation tasks demonstrating state-of-the-art performance for tunable inference.

2. Related Works

In this section, we position our contribution and review relevant work. We put a particular emphasis on image-to-image translation and inverse imaging (*e.g.* denoising and super-resolution) as these problems are clear use-cases for dynamic and controllable networks due to their ill-posedness.

Dynamic Non-Interactive Networks. Contemporary examples of dynamic models [16] introduce dynamic traits by adjusting either model structure or parameters adaptively to the input at inference time. Common strategies to implement dynamic networks use attention or auxiliary learnable modules to modulate convolutional weights [9, 10, 49, 58, 63], recalibrate features [20, 31, 41], or even directly predicting weights specific to the given input [14, 23, 37, 57]. These methods strengthen representation power of the network through the construction of dynamic and discriminative features, however do not offer interactive controls over their dynamicity.

Parametric Non-Interactive Networks. The use of auxiliary input information is a strategy commonly used to condition the behavior of neural networks and to improve their performance [41]. In the context of image restoration, this information typically represents one or more degradation parameters in the input image, *e.g.* noise standard deviation, blur size, or JPEG compression level, which is then provided to the network as additional input channels [13, 37, 61], or as parameters to modulate convolutional weights [34]. This approach is proven to be an effective way to improve performance of a variety of image restoration tasks, including image denoising, super-resolution, joint denoising & demosaicing, and JPEG deblocking [13, 34], however it is not explicitly designed to control the underlying network at inference time.

Interactive Single-Objective Networks. More recently, a number of works have explored the use of external degradation parameters not only as conditional information to the network, but also as a way to enable interactive image restoration. A recent example is based on feature modulation inspired by squeeze-and-excitation [20] where the excitation weights are generated by a fully-connected layer whose inputs are the external degradation parameters [17]. A similar modulation strategy is also employed in [24] for JPEG enhancement where a fully-connected layer controlling the quality factor is used to generate affine transformation weights to modulate features in the decoder stage of the network. In these works, the network is optimized using a fixed loss to maximize restoration performance conditioned on the *true* parameters. Then, at inference time, the authors propose to use of these parameters to change the model behavior, *e.g.* using a noise standard deviation lower than the real one to reduce the denoising strength. However, in these cases the network is requested to operate outside its training distribution, and –unsurprisingly– is likely to generate suboptimal results which often contain significant artifacts, as also observed in [27]. Finally, in a somewhat different spirit, alternative approaches use external parameters to modulate the noise standard deviation [29], or to build task-specific networks via channel pruning [27] or dynamic topology [43].

Interactive Multi-Objective Networks. A classical strategy towards building interactive networks for multiple objectives is based on weight/network interpolation [6, 54, 55]. The main idea consists in training a network with the same topology multiple times using different losses. After training, the authors propose to produce all intermediate behaviors by interpolating corresponding network weights using a convex linear combination driven by a set of interactive interpolation parameters. This approach allows to use an arbitrary number of objectives and has negligible computational cost, however the results are often prone to artifacts as the interpolated weights are not explicitly supervised. Other approaches use feature modulation [46, 53] to control the network behavior. These methods propose to use two separate branches connected at each layer via local

residual connections. Each branch is trained in a separate stage against a different objective, while keeping the other one frozen. Although effective, this approach has several drawbacks: it is limited to two objectives, the intermediate behaviors are again not optimized, and the complexity is effectively doubled.

Compared to all existing works, our strategy is more flexible and more robust, as we can handle an arbitrary number of objectives and we explicitly optimize all their combinations. On top of that, our strategy is also straightforward to train, agnostic to the model architecture, and has negligible computational overhead at inference time.

3. Method

In this section, we formally introduce our framework to enable tunable network behaviour. As a starting point for our discussion, we recall the general form of traditional and dynamic convolutions (Sec. 3.1). Next, we provide a formal definition for our tunable convolutions and discuss how to train these layers using the proposed parametric multi-loss optimization (Sec. 3.2).

3.1. Background

Traditional Convolutions. Let us define the basic form of a traditional convolutional layer as

$$\boldsymbol{y} = f_c(\boldsymbol{x}) = \boldsymbol{x} \circledast \boldsymbol{k} + \boldsymbol{b}, \tag{1}$$

where $\circledast$ is a convolution with kernel $\boldsymbol{k} \in \mathbb{R}^{k \times k \times c \times d}$ and bias $\boldsymbol{b} \in \mathbb{R}^d$ which transforms an input $\boldsymbol{x} \in \mathbb{R}^{h \times w \times c}$ into an output $\boldsymbol{y} \in \mathbb{R}^{h \times w \times d}$, being $h \times w$ the spatial resolution, k the spatial kernel support, c the input channels, and d the output channels.

Dynamic Convolutions. A dynamic convolution [9,63]

$$\boldsymbol{y} = f_d(\boldsymbol{x}) = \boldsymbol{x} \circledast \hat{\boldsymbol{k}}_{\boldsymbol{x}} + \hat{\boldsymbol{b}}_{\boldsymbol{x}}, \tag{2}$$

is parametrized by a dynamic kernel and bias generated as

$$\hat{\boldsymbol{k}}_{\boldsymbol{x}} = \sum_{i=1}^{p} \alpha_i \boldsymbol{k}_i \quad \text{and} \quad \hat{\boldsymbol{b}}_{\boldsymbol{x}} = \sum_{i=1}^{p} \alpha_i \boldsymbol{b}_i, \tag{3}$$

by aggregating an underlying set of fixed p kernels and biases $\{\boldsymbol{k}_i, \boldsymbol{b}_i\}_{i=1}^p$, using p aggregation weights $\boldsymbol{\alpha} = \{\alpha_i\}_{i=1}^p$ dynamically generated from the input. Note that this input-dependency is highlighted in (2) via the subscript $\boldsymbol{x}$. Formally the aggregation weights can be defined as

$$\boldsymbol{\alpha} = \phi_d(\boldsymbol{x}), \tag{4}$$

being $\phi_d : \mathbb{R}^{h \times w \times c} \rightarrow \mathbb{R}^p$ a function mapping the input $\boldsymbol{x}$ to the p aggregation weights. This function is typically implemented as a squeeze-and-excitation (SE) layer [20], *i.e.* a global pooling operation followed by a number of fully connected layers and a final softmax activation to ensure convexity (*i.e.* $\sum_{i=1}^p \alpha_i = 1$). While this layer is capable of dynamically adapting its response, it still lacks a mechanism to control its behaviour in a predictable fashion.

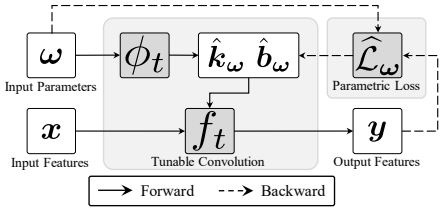

Figure 2. Illustration of the proposed framework to enable tunable networks, consisting of a tunable convolution taking as input a feature $\boldsymbol{x}$ and some external parameters $\boldsymbol{\omega}$ which also parametrize the loss used to update the tunable kernels and biases. See Sec. 3.2 for more details.

3.2. Tunable Networks

In this section we will discuss the two core components proposed in this work, namely a new parametric layer, called tunable convolution, and a parametric optimization strategy to enable the sought-after tunable behavior. A schematic of our framework can be found in Fig. 2.

Tunable Convolutions. The first building block of our framework consists of defining a special form of dynamic *and* tunable convolution

$$\boldsymbol{y} = f_t(\boldsymbol{x}, \boldsymbol{\omega}) = \boldsymbol{x} \circledast \hat{\boldsymbol{k}}_{\boldsymbol{\omega}} + \hat{\boldsymbol{b}}_{\boldsymbol{\omega}}, \tag{5}$$

which includes an additional input $\boldsymbol{\omega} = \{\omega_i\}_{i=1}^p$ consisting of p of interactive parameters which are used to control the effect of p different objectives. The kernels and biases are aggregated analogously to (3). However there is a key difference: here we propose to generate the aggregation weights not implicitly from the input $\boldsymbol{x}$, but rather explicitly from the interactive parameters as

$$\boldsymbol{\alpha} = \phi_t(\boldsymbol{\omega}), \tag{6}$$

where $\phi_t : \mathbb{R}^p \rightarrow \mathbb{R}^p$ is a function mapping the p interactive parameters into the p aggregation weights. Crucially, the kernel and bias of (5) now depends on the input parameters, as denoted by the subscript $\boldsymbol{\omega}$, thus highlighting the differences with respect to dynamic convolutions (2). Without loss of generality, $\boldsymbol{\omega}$ are all assumed to be in the range $[0, 1]$ but are not necessarily convex (*i.e.* $\sum_{i=1}^p \omega_i \geq 1$). Intuitively, the influence of the i-th objective is minimized by $\omega_i = 0$ and maximized by $\omega_i = 1$. There are multiple possibilities for the actual implementation of ϕ_t, ranging from SE [20] to a simple identity function. In our experiments we use a learnable affine transformation (*i.e.* an MLP) without any activation, as we empirically found that this leads to

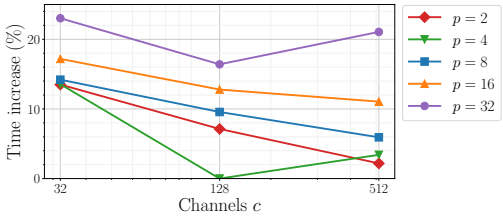

Figure 3. Average runtime increase (in percentage) between a traditional convolution with kernel size $k \in [3, 5, 7]$ and a tunable convolution with p parameters while processing input of size 128×128 with varying number of channels c.

better solutions in all our experiments, and also has a minimal impact on computational complexity. From a practical point of view, the proposed tunable convolutions are agnostic to input dimensions or convolution hyper-parameters. Thus, as we will show in our experiments, tunable variants of strided, transposed, point-wise, group-wise convolutions, or even attention layers, can be easily implemented.

There are several advantages that come with the proposed strategy: first, it increases the representation power of the model [9, 34]; second, it generates the aggregation weights in a predictable and interpretable fashion from the external parameters, as opposed to (4) which generates the non-interactive weights from the input; third, it is computationally efficient as the only additional cost over a traditional convolution consists in computing (6) and the corresponding kernel aggregation. Fig. 3 shows the average increase in runtime[1] required to process a tunable convolution compared to a traditional convolution with equivalent kernel size and number of channels. As may be observed from the figure the overhead is less than 20%, *i.e.* few fractions of a millisecond, and in the most common cases of $p \leq 4$ it is even lower than 5%.

Parametric Multi-Loss. Let us outline how to instill tunable behavior in (6) via the interactive parameters $\boldsymbol{\omega}$. The tunable convolution needs to be informed of the meaning of each parameter and supervised accordingly as each parameter is changed. We achieve this by explicitly linking these parameters to different behaviors through the use of a parametric multi-loss function composed of p different objectives $\mathcal{L}_i$. Specifically, the *same* parameters $\boldsymbol{\omega}$ used in (5) to aggregate the tunable kernels and biases are also used to aggregate the p objectives as

$$\widehat{\mathcal{L}}_{\boldsymbol{\omega}} = \sum_{i=1}^{p} \omega_i \cdot \lambda_i \cdot \mathcal{L}_i, \tag{7}$$

where $\lambda_i \geq 0$ is a fixed weight used to scale the relative contribution of the i-th objective to the overall loss. Note that each $\mathcal{L}_i$ can include multiple terms, so it is possible to encapsulate complex behaviors, such as a style transfer [25] or GAN loss [55], into a single controllable objective.

Our approach generalizes a vanilla training strategy. Specifically, if we keep $\boldsymbol{\omega}$ fixed during training, then the corresponding loss (7) will also be fixed and thus no tunable capability will be explicitly induced by $\boldsymbol{\omega}$. Differently, we propose to also optimize all intermediate objectives by sampling at each training step a different, random, set of parameters, which we use to generate a random combination of objectives in (7), and a corresponding combination of kernels in (5). Thus, the network is encouraged to disentangle the different objectives into the different tunable kernels (and biases) and, as a result, at inference time we can control the relative importance of the different objectives by interacting with the corresponding parameters. The critical advantage over prior methods [17, 46, 53, 54] is that our strategy is not restricted to a fixed number of objectives, and also actively optimizes all their intermediate combinations. In this work we use random uniform sampling, *i.e.* $\omega_i \sim \mathcal{U}(0, 1)$, for both its simplicity and excellent empirical performance, however different distributions could be explored to bias the sampling towards specific objectives [17].

4. Experiments

In this section we report experimental results of our tunable convolutions evaluated on image denoising, deblurring, super-resolution, and style transfer. In all tasks, we measure the ability of our method to tune inference behavior by interacting with external parameters designed to control various characteristics of the image translation process. To showcase the efficacy of our method, we compare performance to recent controllable networks, namely DNI [54], CFSNet [53], DyNet [46], and CResMD [17]. Further, we use our tunable convolutions as drop-in replacements in the state-of-the-art SwinIR [32] and NAFNet [8] networks and evaluate performance in standard benchmarks. A MindSpore [22] reference implementation is available[2].

4.1. Image Restoration

A general observation model for image restoration is

$$z = D(y) + \eta, \tag{8}$$

where $z \in \mathbb{R}^{h \times w \times 3}$ is a degraded image with resolution $h \times w$ and three RGB color channels, D is a degradation operator (*e.g.* downsampling or blurring) applied to the underlying (unknown) ground-truth image y, and η is a random noise realization which models the stochastic nature of the image acquisition process. Under this lens, a restoration network aims to provide a faithful estimate $\hat{y}$ of the underlying ground-truth image y, given the corresponding degraded observation z.

[1] Measured by averaging 5000 runs on a NVIDIA V100 GPU.

[2] https://github.com/mindspore-lab/mindediting

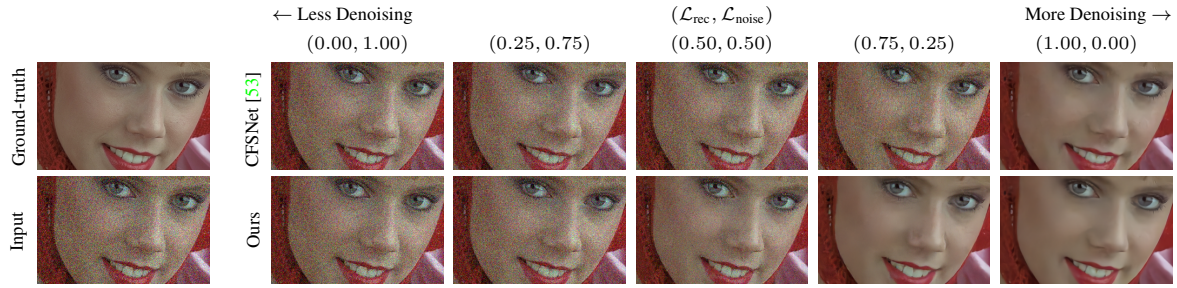

Figure 4. Tuning denoising strength on image 04 from the Kodak dataset [28] corrupted by Gaussian noise with standard deviation $\sigma = 30$

4.1.1 Denoising

In this section we assess performance of our tunable convolutions used in different backbones applied to the classical problem of image denoising. Formally, if we refer to (8), D is the identity, and η is typically distributed as i.i.d. Gaussian noise. An interesting application of tunable models looks into modulating the denoising strength to balance the amount of noise removal and detail preservation in the predicted image; two objectives that, because of the inherent imperfection of any denoising process, are often in conflict with one another. Formally, we use a multi-loss $\widehat{\mathcal{L}}_{\text{rn}} = \omega_1 \cdot \mathcal{L}_{\text{rec}} + \omega_2 \cdot \mathcal{L}_{\text{noise}}$ which includes two terms: the first is a standard reconstruction objective measuring distortion with an L_1 distance, and the second is a novel noise preservation objective defined as

$$\mathcal{L}_{\text{noise}} = ||\hat{y} - y_\eta||_1, \tag{9}$$

where $y_\eta = y + \omega_2 \cdot \nu \cdot (z - y)$ is the target image containing an amount of residual noise proportional to the parameter ω_2 and a fixed pre-defined scalar $0 \leq \nu \leq 1$. Note that we set $\nu = 0.9$, so that even with maximum noise preservation $\omega_2 = 1$ we avoid convergence to a trivial solution (*i.e.* target image equal to the noisy input), and instead require 90% of the residual noise to be preserved. Let us recall that different combinations of (ω_1, ω_2) promote different inference behaviors, *e.g.* $(0.00, 1.00)$ maximizes noise preservation and $(1.00, 0.00)$ maximizes fidelity. Finally, in order to objectively measure the tunable ability of the compared methods, we use PSNR against the ground-truth y and PSNR_η against the target image y_η as clear performance indicators for $\mathcal{L}_{\text{rec}}$ and $\mathcal{L}_{\text{noise}}$, respectively.

Tunable Networks. Here we compare against the state-of-the-art in controllable networks, namely DNI [54], DyNet [46], and CFSNet [53] applied to synthetic image denoising. For fair comparison, we use the same ResNet [18] backbone as in [17, 46, 53] which includes a long residual connection before the output layer. Specifically, for DNI and our tunable network, we stack 16 residual blocks (Conv2d-ReLU-Conv2d-Skip) with 64 channels, whereas for DyNet and CFSNet we use 8 blocks for the main branch and 8 blocks for the tuning branch in order to maintain the same overall complexity. All methods are trained for $500,000$ iterations using Adam optimizer [47] with batch size 16 and fixed learning rate $1e{-}4$. For training we use patches of size 64×64, randomly extracted from the DIV2K [2] dataset, to which we add Gaussian noise $\eta \sim \mathcal{N}(0, \sigma^2)$ with standard deviation $\sigma \in [5, 30]$ as in (8).

		Kodak [28]					CBSD68 [35]					
	ω_1	0.00	0.25	0.50	0.75	1.00	0.00	0.25	0.50	0.75	1.00	$\mathcal{L}_{\text{rec}}$
	ω_2	1.00	0.75	0.50	0.25	0.00	1.00	0.75	0.50	0.25	0.00	$\mathcal{L}_{\text{noise}}$
PSNR		**26.74**	20.69	18.95	20.92	**35.56**	**26.78**	20.18	18.36	20.18	**34.55**	DNI [54]
		26.72	24.60	24.31	26.76	35.32	26.76	24.55	24.18	26.48	34.41	DyNet [46]
		26.73	26.83	27.95	29.89	35.32	26.77	26.88	27.95	29.66	34.41	CFSNet [53]
		26.74	**27.09**	**28.46**	**35.07**	35.54	**26.78**	**27.12**	**28.12**	**34.10**	34.53	Ours
PSNR_η		**52.21**	21.74	19.12	21.00	35.32	**51.55**	21.16	18.53	20.27	34.41	DNI [54]
		50.32	28.69	26.46	28.11	35.26	49.89	28.39	26.19	27.75	33.61	DyNet [46]
		51.83	38.49	34.45	32.88	35.32	51.33	38.45	34.33	32.59	34.41	CFSNet [53]
		51.97	**39.04**	**35.56**	**36.19**	**35.54**	51.36	**38.92**	**34.98**	**35.55**	**34.53**	Ours

Table 1. Tuning denoising strength. Increasing parameter ω_1 promotes data fidelity $\mathcal{L}_{\text{rec}}$, whereas increasing ω_2 promotes noise preservation $\mathcal{L}_{\text{noise}}$. Our strategy is the best in almost all cases.

In Table 1 we report performance averaged over noise levels $\sigma \in [5, 15, 30]$. It may be observed that the proposed tunable model outperforms the state of the art in almost all cases. Further, our method has almost identical accuracy to a network purely trained for fidelity, *i.e.* DNI with weights $(1.00, 0.00)$, whereas the others typically show a -0.2dB drop in PSNR. In general, DNI and DyNet do not generalize well in the intermediate cases, whereas CFSNet has competitive performance, especially in regimes where $\mathcal{L}_{\text{noise}}$ dominates. However the visual comparisons against CFSNet in Fig. 4, show that our method is characterized by a smoother and more consistent transition between objectives over the parameter range, and when denoising is maximal, our prediction also contains fewer artifacts.

Traditional Networks. In this section, we build tunable variants of traditional (fixed) state-of-the-art networks SwinIR [32] and NAFNet [8]. Note that these models contain a plethora of different layers such as MLPs, strided and depth-wise convolutions, as well as spatial, channel, and window attention, and all of these can be easily replaced by our tunable variants. We construct and train our tunable SwinIR and NAFNet models following the setup outlined

	IPT	DRUNet	SwinIR [32]			
σ	[7]	[60]	Fixed	$(1,0)$	$(0,1)$	
15	-	34.30	**34.42**	34.36	54.33	CBSD68 [35]
25	-	31.69	**31.78**	31.69	51.67	
50	28.39	28.51	**28.56**	28.43	48.42	
15	-	35.31	35.34	**35.41**	55.41	Kodak [28]
25	-	32.89	32.89	**32.94**	52.93	
50	29.64	29.86	29.79	**29.86**	49.84	
	PSNR	PSNR	PSNR	PSNR	PSNR_η	

(a) Color image denoising.

	NAFNet [8]		
	Fixed	$(1,0)$	$(0,1)$
PSNR	**39.96**	39.90	24.55
PSNR_η	-	39.90	59.90
SSIM	**0.960**	**0.960**	0.523
SSIM_η	-	0.960	0.999

SIDD [1]

(b) Real raw image denoising.

Table 2. SwinIR and NAFNet trained for tunable denoising strength ($\mathcal{L}_{\text{rec}}$, $\mathcal{L}_{\text{noise}}$). Tuning for data fidelity $(1,0)$ results in performance comparable to the fixed baselines.

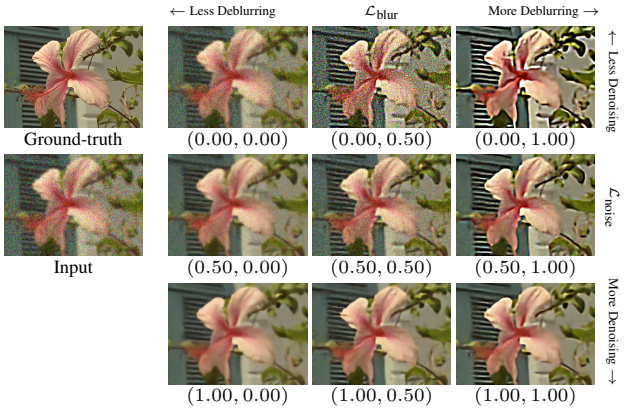

Figure 5. Image 07 from the Kodak dataset [28] corrupted by Gaussian noise $\sigma = 30$ and Gaussian blur with size $\rho = 2$. Our method optimizes denoising and deblurring objectives for all combinations of parameters (ω_1, ω_2).

in the original papers, which we also summarize in the supplementary materials.

In Table 2a, we report PSNR of our tunable SwinIR for color image denoising at noise levels $\sigma \in [15, 25, 50]$ against state-of-the-art IPT [7], DRUNet [60], and fixed SwinIR. In Table 2b, we report PSNR and SSIM [56] of our tunable NAFNet against the fixed NAFNet for real raw image denoising on SIDD [1]. In both tables, we show results for only two combinations of tuning parameters, one maximizing data fidelity $(0,1)$ and the other maximizing noise preservation $(1,0)$. Results largely concur with those in Table 1, and show that our tunable networks have similar, and often even better, performance when compared to the corresponding fixed baselines trained purely for data fidelity.

4.1.2 Joint Denoising & Deblurring

In this experiment we assess the ability of the proposed method to enable interactive image restoration across multiple degradations, specifically we consider joint denoising and deblurring. This is a task explored in CResMD [17] and as such we will use it here as baseline comparison.

For training, we add synthetic Gaussian noise with standard deviation $\sigma \in [5, 30]$ as explained in the previous section, and, following [17], we also add Gaussian blur using a 21×21 kernel with standard deviation in $\rho \in [0, 4]$. Recalling (8), D represents the blurring operation which is applied to the ground-truth y before adding the noise η. We define a multi-loss to simultaneously control the amount of denoising and deblurring as $\widehat{\mathcal{L}}_{\text{nb}} = \omega_1 \cdot \mathcal{L}_{\text{noise}} + \omega_2 \cdot \mathcal{L}_{\text{blur}}$, which includes the same noise preservation objective of (9) plus a deblurring and sharpening objective $\mathcal{L}_{\text{blur}} = ||\hat{y} - \tilde{y}_\eta||_1$ where

$$\tilde{y}_\eta = y_\eta + \omega_2 \cdot \gamma \cdot (y_\eta - g \circledast y_\eta)$$

is obtained via an "unsharp mask filter" [42] using a Gaussian kernel g with support 9×9 and standard deviation 2.5 to extract the high-pass information from y_η which is then scaled and added back to the same image to enhance edges and contrast. We introduce a fixed scalar $\gamma = 8$ to define the maximum amount of sharpening that can be applied to the image. Note that when $\gamma = 0$, the objective $\mathcal{L}_{\text{blur}}$ will be equivalent to $\mathcal{L}_{\text{noise}}$, as only deblurring but no sharpening will be applied to the input image.

In this experiment, the tuning parameters (ω_1, ω_2) are non-convex, and individually control the amount of noise and blur in the prediction. As can be seen from Fig. 5, in our method these parameters explicitly control the influence of the noise and blur objectives; differently, in CResMD the parameters represent true degradation in the input, which are modified at inference time to change the restoration behaviour[3]. Thus, for the same values of parameters, the two methods generate outputs with different characteristics. In Table 3 we report performance in terms of PNSR, LPIPS [62], and NIQE [38] averaged over all levels of noise $[5, 15, 30]$ and blur $[0, 1, 2, 3, 4]$. These results clearly show that our tunable strategy outperforms CResMD almost everywhere, and often by a significant margin. Furthermore, as also noted in [27], CResMD generates severe artifacts whenever asked to process combinations of input image and tuning parameters outside its training distribution (refer to the supplementary materials for some examples).

	Kodak [28]				CBSD68 [35]				
ω_1	0.00	0.00	1.00	1.00	0.00	0.00	1.00	1.00	Noise η
ω_2	0.00	1.00	0.00	1.00	0.00	1.00	0.00	1.00	Blur D
PSNR	**23.68**	11.75	25.92	20.65	**23.14**	11.84	24.84	19.97	CResMD [17]
	23.27	**18.15**	**28.05**	**24.84**	22.17	**17.20**	**27.13**	**23.86**	Ours
LPIPS	0.190	0.580	0.127	0.218	**0.195**	0.586	0.144	0.248	CResMD [17]
	0.172	**0.381**	**0.124**	**0.205**	0.203	**0.428**	**0.136**	**0.233**	Ours
NIQE	**13.54**	17.90	14.53	**13.62**	**15.83**	21.03	18.27	**17.64**	CResMD [17]
	16.18	**11.08**	**13.98**	14.13	19.02	**14.31**	**18.08**	18.87	Ours

Table 3. Tuning denoising and deblurring. In CResMD, the parameters (ω_1, ω_2) are considered as the amount of noise and blur present in the input; in our methods they are used to explicitly control the amount of noise and blur to remove.

[3] For instance, assuming true noise level in the input is σ, using $\beta \cdot \sigma$ for some $\beta < 1$ would implicitly reduce the denoising strength.

		Kodak [28]					CBSD68 [35]					
ω_1		0.00	0.25	0.50	0.75	1.00	0.00	0.25	0.50	0.75	1.00	$\mathcal{L}_{\text{rec}}$
ω_2		1.00	0.75	0.50	0.25	0.00	1.00	0.75	0.50	0.25	0.00	$\mathcal{L}_{\text{gan}}$
PSNR		24.34	17.89	15.63	17.10	27.25	22.88	17.17	14.83	16.27	**26.25**	DNI [54]
		25.34	26.54	27.08	27.22	27.31	23.99	25.24	25.82	26.00	26.08	DyNet [46]
		25.54	26.31	26.87	27.20	27.31	**24.08**	24.98	25.62	25.98	26.08	CFSNet [53]
		24.96	**26.72**	**27.12**	**27.28**	**27.37**	23.61	**25.42**	**25.82**	**26.05**	26.10	Ours
LPIPS		0.113	0.114	0.114	0.114	0.106	0.130	0.141	0.141	0.141	0.132	DNI [54]
		0.103	0.096	0.102	0.105	0.106	0.117	0.118	0.128	0.131	0.130	DyNet [46]
		0.103	0.096	**0.098**	**0.102**	0.105	0.117	0.118	0.125	0.128	0.130	CFSNet [53]
		0.102	**0.092**	**0.098**	0.103	**0.104**	**0.116**	**0.111**	**0.115**	**0.123**	**0.129**	Ours

Table 4. Tuning $\times 4$ super-resolution. Increasing parameter ω_1 promotes data fidelity, whereas increasing ω_2 promotes perceptual quality. Our strategy is the best in almost all cases.

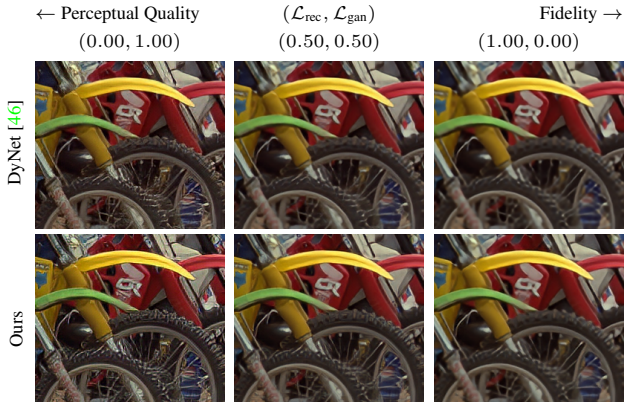

Figure 6. Tuning $\times 4$ super-resolution perceptual quality on image 04 from the Kodak dataset [28]. Our method generates more detailed and perceptually pleasing results.

4.1.3 Super-Resolution

For this experiment we focus on the scenario of $\times 4$ super-resolution, and we consider the degradation operator D in (8) to be bicubic downsampling, and $\eta = 0$ (*i.e.* no noise is added). We use the same setup as in the tunable denoising experiments, with the difference that this time for training we use patch size 48×48 (thus ground-truth patches are 192×192), and the backbone ResNet architecture includes a final pixel shuffling upsampling [33, 45].

Super-resolution is an ideal task to test the ability of our tunable model to balance the perception-distortion trade-off [5]. We design a multi-loss $\widehat{\mathcal{L}}_{\text{rg}} = \omega_1 \cdot \mathcal{L}_{\text{rec}} + \omega_2 \cdot \mathcal{L}_{\text{gan}}$ which includes a reconstruction objective $\mathcal{L}_{\text{rec}}$ to maximize accuracy, and an adversarial objective $\mathcal{L}_{\text{gan}}$ to maximize perceptual quality. Following [55], the latter is defined as $\mathcal{L}_{\text{gan}} = 0.01 \cdot \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{vgg}} + 0.005 \cdot \mathcal{L}_{\text{adv}}$, where $\mathcal{L}_{\text{vgg}}$ is a perceptual loss measuring L_1 distance of VGG-19 [48] features obtained at the "conv5_4" layer [25, 30], and $\mathcal{L}_{adv}$ is a relativistic adversarial loss [26] on the generator and discriminator. We use a discriminator similar to [55], with the difference that we use 4 scales and we add a pooling operator before the final fully-connected layer in order for the classifier to work with arbitrary input resolutions.

	SAN [11]	HAN [40]	NLSA [36]	SwinIR [32] Fixed	SwinIR [32] (1, 0)	SwinIR [32] (0, 1)	
PSNR	32.64	32.64	32.59	**32.72**	32.57	30.22	Set5 [4]
SSIM	0.900	0.900	0.900	**0.902**	0.895	0.838	
PSNR	28.92	28.90	28.87	**28.94**	28.83	26.27	Set14 [59]
SSIM	0.789	0.789	0.789	**0.791**	0.787	0.701	
PSNR	27.78	27.80	27.78	**27.83**	27.79	25.05	BSD100 [35]
SSIM	0.744	0.744	0.744	**0.746**	0.739	0.647	
PSNR	26.79	26.85	26.96	27.07	**27.10**	25.50	Urban100 [21]
SSIM	0.807	0.809	0.811	**0.816**	0.815	0.764	

Table 5. SwinIR trained for tunable $\times 4$ super-resolution ($\mathcal{L}_{\text{rec}}$, $\mathcal{L}_{\text{gan}}$). Tuning for data fidelity $(1, 0)$ results in performance comparable to the fixed baselines.

Tunable Networks. Objective results reported in Table 4 show that the proposed tunable strategy outperforms the compared methods almost everywhere in both fidelity and perceptual measures. Let us note that DyNet in this application exhibits better behavior and it is in general superior to CSFNet, whereas DNI still fails to correctly produce results corresponding to intermediate behaviors. Interestingly, our method is competitive, and even outperforms, a network solely trained with a reconstruction objective in terms of PSNR, *i.e.* DNI with weights $(1.00, 0.00)$. In Fig. 6 we show a visual example which demonstrate the smooth transition between the two objectives, and superior performance over DyNet in terms of quality of details and robustness to artifacts across the full parameter range.

Traditional Networks. As in our denoising experiments, we also show comparison against the state of the art in classical $\times 4$ image super-resolution [11, 32, 36]. In particular, we use SwinIR as backbone and we build a tunable version to optimize the same multi-loss $\widehat{\mathcal{L}}_{\text{rg}}$ described above; to ensure fair comparison, we use the training protocol outlined in [32] for input patch size 48×48. Results shown in Table 5 confirms that, even in this arguably more challenging application, our tunable SwinIR achieves comparable results to the fixed baselines.

4.2. Style Transfer

In this section, in order to test a different image-to-image translation task as well as the the ability of our method to deal with more than two objectives, we consider style transfer [12, 25]. Similarly to [25], we use a UNet [44] without skip connections consisting of two scales and 6 residual blocks (Conv2d-ReLU-Conv2d-Skip-ReLU) to process latent features. Downsampling and upsampling is performed as a strided convolution and nearest neighbor interpolation, respectively. The number of channels is 64 and then doubled at every scale. The first and last convolutions have kernel size 9×9. Instance normalization [51] and ReLU is applied after every convolution expect the last one. We train for $40,000$ iterations with weight decay $1e{-}5$, learning rate $1e{-}4$, batch size 4, and patch size 384×384.

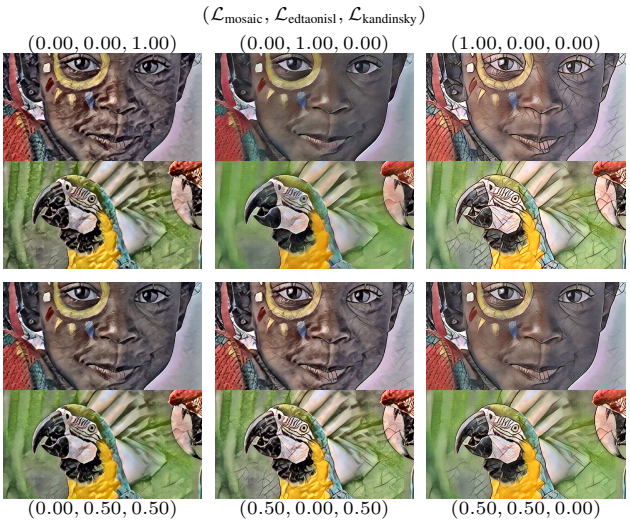

Figure 7. Our method is able to generate pleasing results for all style combinations by interacting with the parameters $(\omega_1, \omega_2, \omega_3)$ controlling *Mosaic*, *Edtaonisl*, and *Kandinsky* style transfer on image 15 and 32 of the Kodak [28] dataset.

We define the style transfer objective similarly to [25]: $\mathcal{L}_{\text{style}} = \lambda_{\text{gram}} \cdot \mathcal{L}_{\text{gram}} + \lambda_{\text{vgg}} \cdot \mathcal{L}_{\text{vgg}} + \lambda_{\text{tv}} \cdot \mathcal{L}_{\text{tv}} + \mathcal{L}_{\text{uv}}$, where $\mathcal{L}_{\text{gram}}$ is the squared Frobenius norm of the difference between the Gram matrices of the predicted and target style features extracted from the "relu3_3" layer of a pretrained VGG-19 [48], $\mathcal{L}_{\text{vgg}}$ is the L_2 distance between the predicted and target features extracted from the "relu3_3" layer, $\mathcal{L}_{\text{tv}}$ is a Total Variation regularization term used to promote smoothness [3], and $\mathcal{L}_{\text{uv}}$ is the L_2 distance between the YUV chrominance channels used to preserve the original colors. We use different λ weights for different styles; detailed settings can be found in the supplementary materials.

Fig. 7 demonstrates that our method is capable to smoothly transition across three different styles, *i.e. Mosaic*, *Edtaonisl*, and *Kandinsky*. In contrast to existing approaches, such as DyNet [46], our method is (the only one) able to reliably optimize more than two style objectives, including all their intermediate combinations.

5. Discussion

In this section, we shed some light onto the inner workings of our method by analyzing the relationship between the kernels in our tunable convolution and the corresponding objectives in the multi-loss. As illustrative examples, we take the denoising and super-resolution experiments of Table 1 and Table 4. In both cases we use the same ResNet backbone with 16 residual blocks and 64 channels. First, we tune these networks using five parameter combinations uniformly sampled with 0.25 step; then we extract the *tuned* kernel tensors from (both convolutions in) each residual block; and finally, we reduce the dimensionality via PCA

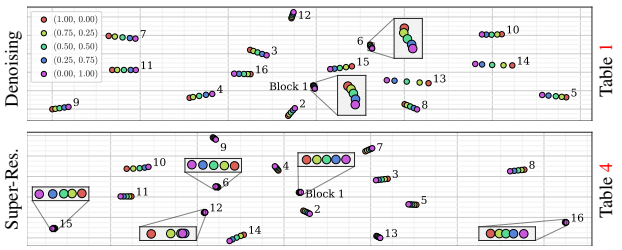

Figure 8. t-SNE of the tuned kernels in the 16 residual blocks of our tunable denoising and super-resolution networks. The kernels span quasi-linear manifolds of different lengths and orientations between the two objectives $(0.00, 1.00)$ and $(1.00, 0.00)$.

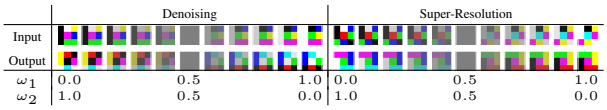

Figure 9. Input and output kernels tuned with uniform parameters reveal a linear behavior while transitioning between objectives.

followed by t-SNE [19, 52] to embed the tuned kernels into 2D points. In Fig. 8 we visualize scatter plots of these points for both denoising and super-resolution networks. Evidently, each block is well-separated into a different cluster, and, more interestingly, the tuned kernels span quasi-linear manifolds in the region identified by the tunable parameters with various lengths and orientations. This clearly demonstrates that the tuned kernels smoothly transition from one pure objective $(0.00, 1.00)$ to the other $(1.00, 0.00)$. The same linear transition can be also observed in Fig. 9, where we depict the first principal component of the input and output RGB kernels extracted from the first and last tunable convolution layer after tuning with 11 parameters uniformly sampled with step 0.1.

6. Conclusions

We have presented tunable convolutions: a novel dynamic layer enabling change of neural network inference behaviour via a set of interactive parameters. We associate each parameter with a desired behavior, or objective, in a multi-loss. During training, these parameters are randomly sampled, and all possible combinations of objectives are explicitly optimized. During inference, the different objectives are disentangled into the corresponding parameters, which thus offer a clear interpretation as to which behaviour they should promote or inhibit. In comparison with existing solutions our strategy achieves better performance, is not limited to a fixed number of objectives, explicitly optimizes for all possible linear combinations of objectives, and has negligible computational cost. Further, we have shown that our tunable convolutions can be used as a drop-in replacement in existing state-of-the-art architecture enabling tunable behavior at almost no loss in baseline performance.

References

[1] Abdelrahman Abdelhamed, Stephen Lin, and Michael S. Brown. A high-quality denoising dataset for smartphone cameras. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1692–1700, 2018. 6

[2] Eirikur Agustsson and Radu Timofte. NTIRE 2017 challenge on single image super-resolution: Dataset and study. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 126–135, 2017. 5

[3] Hussein-A. Aly and Eric Dubois. Image up-sampling using total-variation regularization with a new observation model. *IEEE Transactions on Image Processing*, 14(10):1647–1659, 2005. 8

[4] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie-Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In *British Machine Vision Conference (BMVC)*, pages 135.1–135.10, 2012. 7

[5] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR*, pages 6228–6237, 2018. 1, 7

[6] Haoming Cai, Jingwen He, Yu Qiao, and Chao Dong. Toward interactive modulation for photo-realistic image restoration. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 294–303, 2021. 2

[7] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12294–12305, 2021. 6

[8] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *European Conference on Computer Vision (ECCV)*, pages 17–33, 2022. 4, 5, 6

[9] Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11027–11036, 2020. 2, 3, 4

[10] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *IEEE International Conference on Computer Vision (ICCV)*, pages 764–773, 2017. 2

[11] Tao Dai, Jianrui Cai, Yongbing Zhang, Shu-Tao Xia, and Lei Zhang. Second-order attention network for single image super-resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11057–11066, 2019. 7

[12] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2414–2423, 2016. 7

[13] Michaël Gharbi, Gaurav Chaurasia, Sylvain Paris, and Frédo Durand. Deep joint demosaicking and denoising. *ACM Transactions on Graphics*, 35(6):1–12, 2016. 2

[14] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. In *International Conference on Learning Representations (ICLR)*, 2017. 2

[15] Jacques Salomon Hadamard. *Lectures on Cauchy's problem in linear partial differential equations*, volume 18. Yale university press, 1923. 1

[16] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[17] Jingwen He, Chao Dong, and Yu Qiao. Interactive multi-dimension modulation with dynamic controllable residual learning for image restoration. In *European Conference on Computer Vision (ECCV)*, pages 53–68, 2020. 1, 2, 4, 5, 6

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 5

[19] Geoffrey Hinton and Sam Roweis. Stochastic neighbor embedding. *Advances in Neural Information Processing Systems (NeurIPS)*, 15:833–840, 2003. 8

[20] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7132–7141, 2018. 2, 3

[21] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5197–5206, 2015. 7

[22] Huawei. MindSpore. https://www.mindspore.cn/en, 2020. 4

[23] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 29, 2016. 2

[24] Jiaxi Jiang, Kai Zhang, and Radu Timofte. Towards flexible blind JPEG artifacts removal. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4977–4986, 2021. 1, 2

[25] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision (ECCV)*, pages 694–711, 2016. 4, 7, 8

[26] Alexia Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard GAN. In *International Conference on Learning Representations (ICLR)*, 2019. 7

[27] Heewon Kim, Sungyong Baik, Myungsub Choi, Janghoon Choi, and Kyoung Mu Lee. Searching for controllable image restoration networks. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14214–14223, 2021. 1, 2, 6

[28] Kodak Image Dataset. http://r0k.us/graphics/kodak/, 1999. 5, 6, 7, 8

[29] Kinam Kwon, Eunhee Kang, Sangwon Lee, Su-Jin Lee, Hyong-Euk Lee, ByungIn Yoo, and Jae-Joon Han. Controllable image restoration for under-display camera in smartphones. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2073–2082, 2021. 2

[30] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 105–114, 2017. 7

[31] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 510–519, 2019. 2

[32] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. SwinIR: Image restoration using swin transformer. In *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 1833–1844, 2021. 4, 5, 6, 7

[33] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1132–1140, 2017. 7

[34] Fangzhou Luo, Xiaolin Wu, and Yanhui Guo. Functional neural networks for parametric image restoration problems. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021. 2, 4

[35] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *IEEE International Conference on Computer Vision*, volume 2, pages 416–423, 2001. 5, 6, 7

[36] Yiqun Mei, Yuchen Fan, and Yuqian Zhou. Image super-resolution with non-local sparse attention. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3516–3525, 2021. 7

[37] Ben Mildenhall, Jonathan T. Barron, Jiawen Chen, Dillon Sharlet, Ren Ng, and Robert Carroll. Burst denoising with kernel prediction networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2502–2510, 2018. 2

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

[39] Aamir Mustafa, Aliaksei Mikhailiuk, Dan Andrei Iliescu, Varun Babbar, and Rafał K. Mantiuk. Training a task-specific image reconstruction loss. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 21–30, 2022. 1

[40] Ben Niu, Weilei Wen, Wenqi Ren, Xiangde Zhang, Lianping Yang, Shuzhen Wang, Kaihao Zhang, Xiaochun Cao, and Haifeng Shen. Single image super-resolution via a holistic attention network. In *European Conference on Computer Vision (ECCV)*, pages 191–207, 2020. 7

[41] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2332–2341, 2019. 2

[42] Andrea Polesel, Giovanni Ramponi, and John V. Mathews. Image enhancement via adaptive unsharp masking. *IEEE Transactions on Image Processing*, 9(3):505–510, 2000. 6

[43] Dripta S. Raychaudhuri, Yumin Suh, Samuel Schulter, Xiang Yu, Masoud Faraki, Amit K. Roy-Chowdhury, and Manmohan Chandraker. Controllable dynamic multi-task architectures. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10945–10954, 2022. 1, 2

[44] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 234–241, 2015. 7

[45] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1874–1883, 2016. 7

[46] Alon Shoshan, Roey Mechrez, and Lihi Zelnik-Manor. Dynamic-Net: Tuning the objective without re-training for synthesis tasks. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3214–3222, 2019. 2, 4, 5, 7, 8

[47] Karen Simonyan and Andrew Zisserman. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 5

[48] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015. 7, 8

[49] Hang Su, Varun Jampani, Deqing Sun, Orazio Gallo, Erik Learned-Miller, and Jan Kautz. Pixel-adaptive convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11158–11167, 2019. 2

[50] Ethan Tseng, Ali Mosleh, Fahim Mannan, Karl St-Arnaud, Avinash Sharma, Yifan Peng, Alexander Braun, Derek Nowrouzezahrai, Jean-Francois Lalonde, and Felix Heide. Differentiable compound optics and processing pipeline optimization for end-to-end camera design. *ACM Transactions on Graphics*, 40(2):1–19, 2021. 1

[51] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:2004.10694*, 2016. 7

[52] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. 8

[53] Wei Wang, Ruiming Guo, Yapeng Tian, and Wenming Yang. CFSNet: Toward a controllable feature space for image restoration. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4139–4148, 2019. 1, 2, 4, 5, 7

[54] Xintao Wang, Ke Yu, Chao Dong, Xiaoou Tang, and Chen Change Loy. Deep network interpolation for continuous imagery effect transition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1692–1701, 2019. 1, 2, 4, 5, 7

[55] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. ESRGAN: Enhanced super-resolution generative adversarial networks. In *European Conference on Computer Vision Workshops (ECCV)*, pages 63–79, 2018. 1, 2, 4, 7

[56] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 6

[57] Zhihao Xia, Federico Perazzi, Michaël Gharbi, Kalyan Sunkavalli, and Ayan Chakrabarti. Basis prediction networks for effective burst denoising with large kernels. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11844–11853, 2020. 2

[58] Brandon Yang, Gabriel Bender, Quoc V Le, and Jiquan Ngiam. CondConv: Conditionally parameterized convolutions for efficient inference. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019. 2

[59] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *Curves and Surfaces*, pages 711–730, 2012. 7

[60] Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool, and Radu Timofte. Plug-and-play image restoration with deep denoiser prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6360–6376, 2022. 6

[61] Kai Zhang, Wangmeng Zuo, and Lei Zhang. FFDNet: Toward a fast and flexible solution for CNN-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018. 2

[62] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, 2018. 6

[63] Yikang Zhang, Jian Zhang, Qiang Wang, and Zhao Zhong. DyNet: Dynamic convolution for accelerating convolutional neural networks. *arXiv preprint arXiv:2004.10694*, 2020. 1, 2, 3

[64] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on Computational Imaging*, 3(1):47–57, 2017. 1