

Neural Architecture Search for Deep Image Prior

^[1] Kary Ho, ^[1] Andrew Gilbert, ^[2] Hailin Jin, ^[1,2] John Collomosse

^[1] Centre for Vision Speech and Signal Processing,
University of Surrey

^[2] Creative Intelligence Lab, Adobe Research

Abstract

We present a neural architecture search (NAS) technique to enhance the performance of unsupervised image denoising, in-painting and super-resolution under the recently proposed Deep Image Prior (DIP). We show that evolutionary search can automatically optimize the encoder-decoder (E-D) structure and meta-parameters of the DIP network, which serves as a content-specific prior to regularize these single image restoration tasks. Our binary representation encodes the design space for an asymmetric E-D network that typically converges to yield a content-specific DIP within 10-20 generations using a population size of 500. The optimized architectures consistently improve upon the visual quality of classical DIP for a diverse range of photographic and artistic content.

1. Introduction

Many common image restoration tasks require the estimation of missing pixel data: de-noising and artefact removal; in-painting; and super-resolution. Usually, this missing data is estimated from surrounding pixel data, under a smoothness prior. Recently it was shown that the architecture of a randomly initialized convolutional neural network (CNN) could serve as an effective prior, regularizing estimates for missing pixel data to fall within the manifold of natural images. This regularization technique, referred to as the *Deep Image Prior* (DIP) [34] exploits both texture self-similarity within an image and the translation equivariance of CNNs to produce competitive results for the image restoration tasks mentioned above. However, the efficacy of DIP is dependent on the architecture of the CNN used; different content requires different CNN architectures for excellent performance and care over meta-parameter choices *e.g.* filter sizes, channel depths, epoch count, etc.

This paper contributes an evolutionary strategy to automatically search for the optimal CNN architecture and associated meta-parameters given a single input image. The core technical contribution is a genetic algorithm (GA) [8] for representing and optimizing a content-specific network architecture for use as DIP regularizer in unsupervised im-

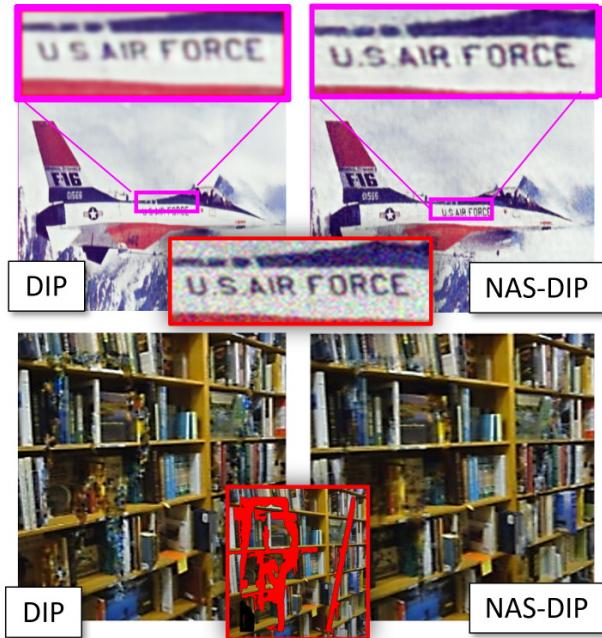


Figure 1. Neural Architecture Search yields a content-specific deep image prior (NAS-DIP, right) that enhances performance over classical DIP (left) for unsupervised image restoration tasks including de-noising (top) and in-painting (bottom). Source inset (red)

age restoration tasks. We show that superior results can be achieved through architecture search versus standard DIP backbones or random architectures. We demonstrate these performance improvements for image de-noising, Content-Aware Fill (otherwise known as inpainting), and content upscaling for static images; Fig. 1 contrasts the output of classic DIP [34] with our neural architecture search (NAS-DIP).

Unlike image classification, to which NAS has been extensively applied [29, 28, 23, 11], there is no ground truth available in the DIP formulation. An encoder-decoder network (whose architecture we seek) is overfitted to reconstruct the input image from a random noise field, so acquiring a generative model of that image's structure. Parallel work in GANs [40] has shown that architectural design, particularly of the decoder, is critical to learning a sufficient gen-

erative model, and moreover, that this architecture is content specific. Our work is the first to both automate the design of encoder-decoder architectures and to do so without supervision, leveraging a state-of-the-art perceptual metric to guide the optimization [44]. Further, we demonstrate that the optimized architectures are content specific, and enable clustering on visual style without supervision.

2. Related Work

Neural architecture search (NAS) seeks to automate the design of deep neural network architectures through data-driven optimization [12], most commonly for image classification [47, 28], but recently also object detection [47] and semantic segmentation [6]. NAS addresses a facet of the automated machine learning (AutoML) [20] problem, which more generally addresses hyper-parameter optimization and tuning of training meta-parameters.

Early NAS leveraged Bayesian optimization for MLP networks [4], and was extended to CNNs [9] for CIFAR-10. In their seminal work, Zoph and Le [46] applied reinforcement learning (RL) to construct image classification networks via an action space, tokenizing actions into an RNN-synthesised string, with reward driven by validation data accuracy. The initially high computational overhead (800GPUs/4 weeks) was reduced while further enhancing accuracy. For example, by exploring alternative policies such as proximal policy optimization [31] or Q-learning [2], RL approaches over RNN now scale to contemporary datasets *e.g.* NASNet for ImageNet classification [26, 47] and was recently explored for GAN over CIFAR-10 [16]. Cai *et al.* similarly tokenizes the architecture, but explore the solution space via sequential transformation of the string via function-preserving mutation operations [5] on an LSTM-learned embedding of the string. Our work also encodes architecture as a (in our case, binary) string, but optimizes via an evolutionary strategy rather than training a sequential model under RL.

Evolutionary strategies for network architecture optimization were first explored for MLPs in the early nineties [25]. Genetic algorithms (GAs) were used to optimize both the architecture and weights [1, 33], rather than rely upon back-prop, in order to reduce the GA evaluation bottleneck; however, this is not practical for contemporary CNNs. While selection and population culling strategies [11, 29, 28] have been explored to develop high performing image classification networks over ImageNet [23, 28]. Our work is unique in that we explore image restoration encoder-decoder network architectures via GA optimization, and as such, our architecture representation differs from prior work.

Single image restoration has been extensively studied in classical vision and deep learning, where priors are prescribed or learned from representative data. A common prior to texture synthesis is the Markov Random Field (MRF) in which the pair-wise term encodes spatial coherence. Several in-painting works exploit MRF formulations of this kind [22, 18, 24], including methods that source patches from the input [10] or multiple external [17, 14] images, or use random propagation of a few good matched patches [3].

Patch self-similarity within single images has also been exploited for single image super-resolution [15] and de-noising. The Deep Image Prior (DIP) [34] (and its video extension [42]) exploit translation equivariance of CNNs to learn and transfer patches within the receptive field. Very recently, single image GAN [32] has been proposed to learn a multi-resolution model of appearance and DIP has been applied to image layer decomposition [13]. Our work targets the use cases for DIP proposed within the original paper [34]; namely super-resolution, de-noising and region inpainting. All three of these use cases have been investigated substantially in the computer vision and graphics literature, and all have demonstrated significant performance benefits when solved using a deep neural network trained to perform that task on representative data. Generative Adversarial Networks (GANs) are more widely used for in-painting and super-resolution [43, 36] by learning structure and appearance from a representative corpus image data [41], in some cases explicitly maintaining both local and global consistency through independent models [21]. Our approach and that of DIP differs in that we do not train a network to perform a specific task. Instead, we use an untrained (randomly initialized) network to perform the tasks by overfitting such a network to a single image under a task-specific loss using neural architecture search.

3. Architecture Search for DIP

The core principle of DIP is to learn a generative CNN G_θ (where θ are the learned network parameters *e.g.* weights) to reconstruct x from a noise field \mathcal{N} of identical height and width to x , with pixels drawn from a *uniform* random distribution. Ulyanov *et al.* [34] propose a symmetric encoder-decoder network architecture with skip connections for G_θ , comprising five pairs of (up-)convolutional layers with varying architectures depending on the image restoration application (de-noising, in-painting or super-resolution) and the image content to be processed. A reconstruction loss is applied to learn G_θ for a single given image x :

$$\theta^* = \arg \min_{\theta} \|G_\theta(\mathcal{N}) - x\|_2^2 \quad (1)$$

Our core contribution is to optimize not only for θ but also for architecture G using a genetic algorithm (GA), guided via a perceptual quality metric [44], as now described.

3.1. Network Representation

We encode the space of encoder-decoder (E-D) architectures from which to sample G as a constant length binary sequence, representing N paired encoder decoder units $U = \{U_1, \dots, U_N\}$. Following [34], G is a fully convolutional E-D network and optimize for the connectivity and meta-parameters of the convolutional layers. A given unit U_n comprises encoder E_n and decoder D_n convolutional stages denoted E_n and D_n respectively each of which requires 7 bits to encode its parameter tuple. Unit U_n requires

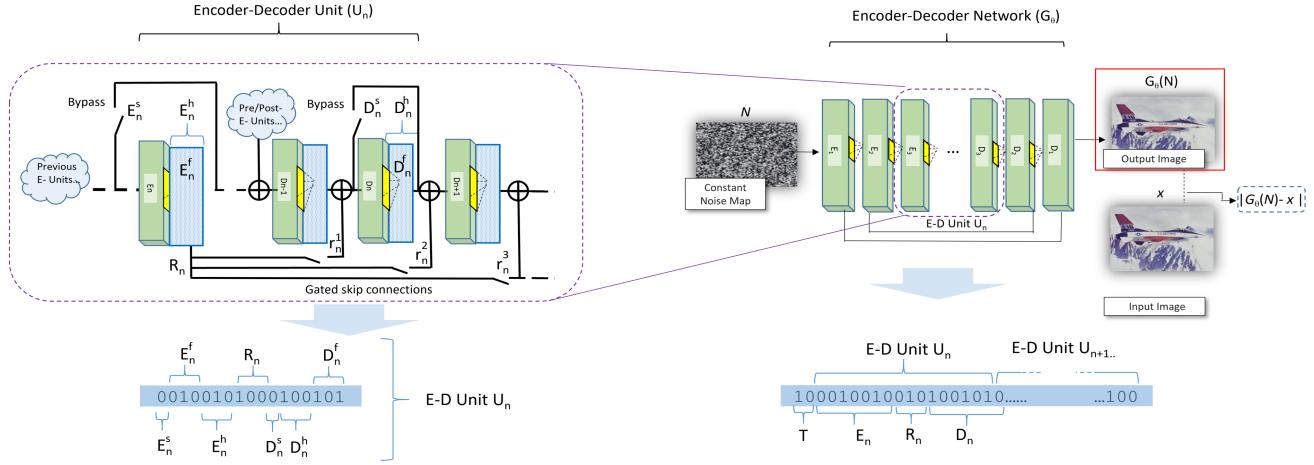


Figure 2. Architecture search space of NAS-DIP(-T). The Encoder-Decoder (E-D) network G (right) is formed of several E-D Units (U_n) each with an Encoder E_n and Decoder D_n paired stage (zoomed, left) represented each by 7 bits plus an additional $4N$ bits R_n to encode gated skip connections from E_n to other decoder blocks in the network. Optionally the training epoch count T is encoded. The binary representation for E-D units is discussed further in Sec. 3.1. Under DIP images are reconstructed from constant noise field N by optimizing to find weights θ thus overfitting the network to input image x under reconstruction loss e.g. here for de-noising (eq.3).

a total $14 + 4N$ bits to encode, as an additional $4N$ -bit block for the unit, encodes a tuple specifying the configuration of skip connections from its encoder stage to each of the decoder stages *i.e.* both within itself and other units. Thus, the total binary representation for an architecture in neural architecture search for DIP (**NAS-DIP**) is $N(14 + 4N)$. For our experiments, we use $N = 6$ but note that the effective number of encoder or decoder stages varies according to skip connections which may bypass the stage. Fig. 2 illustrates the organisation of the E-D units (right) and the detailed architecture of a single E-D unit (left). Each unit U_n comprises the following binary representation, where super-scripts indicate elements of the parameter tuple:

- $E_n^s \in [0, 1]$ (1 bit) a binary indicator of whether the encoder stage of unit U_n is skipped (bypassed).
- $E_n^f \in [0, 7]$ (3 bits) encoding filter size $f = 2E_n^f + 1$ learned by the convolutional encoder.
- $E_n^h \in [0, 7]$ (3 bits) encoding number of filters $h = 2^{E_n^h-1}$ and so channels output by the encoder stage.
- $D_n^s \in [0, 1]$ (1 bit) a binary indicator of whether the decoder stage of unit U_n is skipped (bypassed).
- $D_n^f \in [0, 7]$ (3 bits) encoding filter size $f = 2D_n^f + 1$ learned by the up-convolutional decoder stage.
- $D_n^h \in [0, 7]$ (3 bits) encoding number of filters $h = 2^{D_n^h-1}$ and so channels output by the decoder stage.
- $R_n \in \mathbb{B}^{4N}$ (4N bits) encodes gated skip connections as $[\rho_n^1, \dots, \rho_n^N]$; each 4-bit group $\rho_n^i \in [0, 15]$ determines whether gated skip path r_n^i connects from E_n to D_i and

if so, how many filters/channels (*i.e.* skip gate is open if $r_n^i = 0$).

NAS-DIP-T. We explore a variant of the representation encoding maximum epoch count $T = 500 * (2^t - 1)$ via two additional bits coding for t and thus a representation length for NAS-DIP-T of $N(14 + 4N) + 2$.

Symmetric NAS-DIP. We also explore a compact variant of the representation that forces $E_n = D_n$ for all parameters, forcing a symmetric E-D architecture to be learned and requiring only 10 bits to encode U_n . However, we later show that such symmetric architectures are almost always underperformed by asymmetric architectures discovered via our search (subsec. 4.1).

3.2. Evolutionary Optimization

DIP provides an architecture specific prior that regularises the reconstruction of a given source image x from a uniform random noise field \mathcal{N} . Fixing N constant, we use a genetic algorithm (GA) to search architecture space for the optimal architecture G^* to recover a ‘restored’ *e.g.* denoised, inpainted or upscaled version \hat{x} of that source image:

$$\hat{x} = G_{\theta^*}^*(\mathcal{N}). \quad (2)$$

GANs simulate the process of natural selection by breeding successive generations of individuals through the processes of cross-over, fitness-proportionate reproduction and mutation. In our implementation, such individuals are network configurations encoded via our binary representation.

Individuals are evaluated by running a pre-specified DIP image restoration task using the encoded architecture. We consider unsupervised or ‘blind’ restoration tasks (de-noising, in-painting, super-resolution) in which an ideal \hat{x}

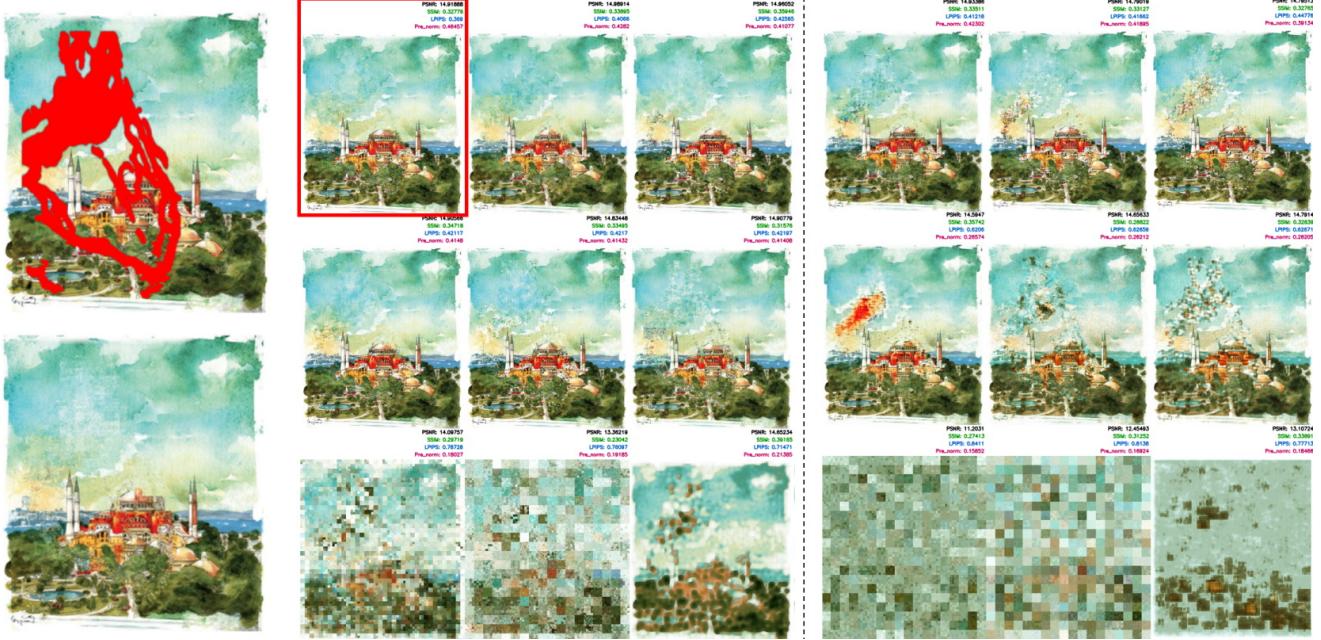


Figure 3. NAS-DIP convergence for in-painting task on BAM! [39]. Left: Input (top) and converged (bottom) result at generation 13. Right: Sampling the top, middle and bottom performing architectures (shown in respective rows) from a population of 500 architectures, at the final (1st, middle) and final (13th, rightmost) generations. Inset scores: Fitness selection is driven using LPIPS [44]; evaluation by PSNR/SSIM.

is unknown (is sought) and so cannot guide the GA search. In lieu of this ground truth we employ a perceptual measure (subsec. 3.2.1) to assess the visual quality generated by any candidate architecture by training $G_\theta^*(N)$ via backpropagation to minimize a task specific reconstruction loss:

$$\mathcal{L}_{de-noise}(x; G) = \min_\theta |G_\theta(\mathcal{N}) - x|. \quad (3)$$

$$\mathcal{L}_{in-paint}(x; G) = \min_\theta |M(G_\theta(\mathcal{N})) - M(x)|. \quad (4)$$

$$\mathcal{L}_{upscale}(x; G) = \min_\theta |D(G_\theta(\mathcal{N})) - x|. \quad (5)$$

where D is a downsampling operator reducing its target to the size of x via bi-linear interpolation, and $M(\cdot)$ is a masking operator that returns zero within the region to be in-painted.

We now describe a single iteration of the GA search, which is repeated until the improvements gained over the previous few generations are marginal (the change in both average and maximum population fitness over a sliding time window fall below a threshold).

3.2.1 Population Sampling and Fitness

We initialize a population of $K = 500$ solutions uniformly sampling $\mathbb{B}^{N(14+4N)}$ to seed initial architectures $\mathcal{G} = \{G_1, G_2, \dots, G_K\}$. The visual quality of \hat{x} under each architecture is assessed ‘blind’ using a learned perceptual measure (LPIPS score) [44] as a proxy for individual fitness:

$$f(G_i) = P(\arg\min_{\hat{x}} \mathcal{L}(\hat{x}; G_\theta)) \quad (6)$$

where \mathcal{L} is the task specific loss (Eq.3-5) and $P(\cdot)$ is the perceptual score [44] for a given individual G_i in the population. We explored several blind perceptual scores from the

GAN literature including Inception Score [30] and Frechet Inception Distance [19] but found LPIPS to better correlate with PSNR and SSIM during our evaluation, improving convergence for NAS-DIP.

Individuals are selected stochastically (and with replacement) for breeding, to populate the next generation of solutions. Architectures that produce higher quality outputs are more likely to be selected. However, population diversity is encouraged via the stochasticity of the process and the introduction of a random mutation into the offspring genome. We apply elitism; the bottom 5% of the population is culled, and the top 5% passes unperturbed to the next generation – the fittest individual in successive generations is thus at least as fit as those in the past. The middle 90% is used to produce the remainder of the next generation. Two individuals are selected stochastically with a bias to fitness $p(G_i) = f(G_i)/\sum_{j=1}^K f(G_j)$, and bred via cross-over and mutation (subsec. 3.2.2) to produce a novel offspring for the successive generation. This process repeats with replacement until the population count matches the prior generation.

3.2.2 Cross-over and Mutation

Individuals are bred via genetic crossover; given two constant length binary genomes of form $G\{(E_1, R_1, D_1), \dots, (E_N, R_N, D_N)\} \in \mathbb{B}^{N(14+4N)}$, a splice point $S = [1, N]$ is randomly selected such that units from a pair of parents A and B are combined via copying of units $U_{N < S}$ from A and $U_{N \geq S}$ from B . Such cross-over could generate syntactically invalid genomes e.g. due to

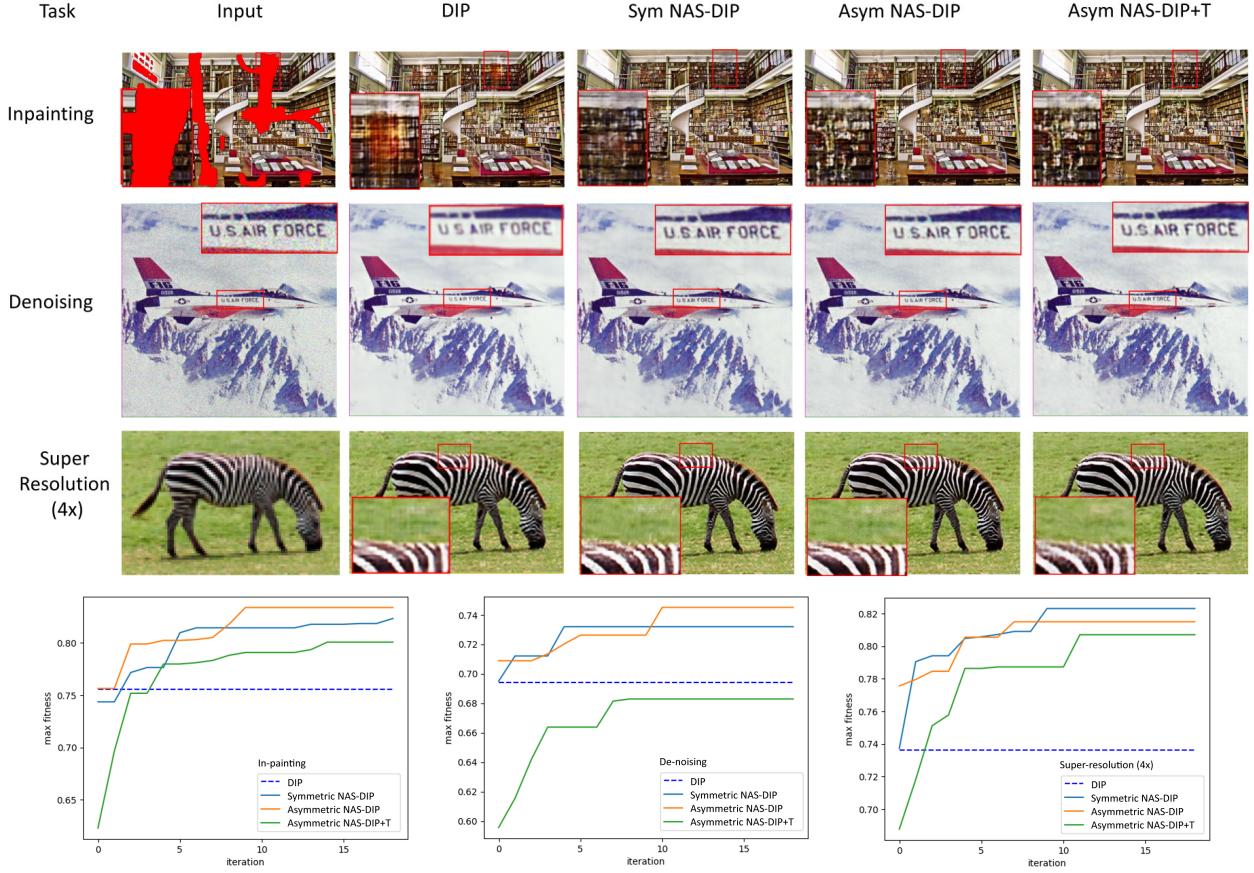


Figure 4. Evaluation of proposed NAS-DIP (and variants) vs. classical DIP [34] for in-painting, de-noising and 4× super-resolution on the Library, Plane and Zebra images of the *DIP* dataset. For each task, respectively, from left to right, the max population fitness graph is shown. We compare the proposed unconstrained (asymmetric) E-D network search of NAS-DIP (optionally with epoch count T freely optimized; NAS-DIP-T) with NAS-DIP constrained to symmetric E-D architectures only. The baseline (dotted blue) for classical DIP uses the architecture published in [34]. Examples further quantified in Table 2.

tensor size incompatibilities between units in the genome. During the evaluation, an invalid architecture evaluates to zero prohibiting its selection for subsequent generations.

Population diversity is encouraged via the stochasticity of the selection process and the introduction of a random mutation into the offspring genome. Each bit within the offspring genome is subject to random flip with low probability p_m ; we explore the trade-off of this rate against convergence in subsec. 4.2.

4. Experiments and Discussion

We evaluate the proposed neural architecture search technique for DIP (NAS-DIP) for each of the three blind image restoration tasks proposed originally for DIP [34]: image in-painting, super-resolution and de-noising.

Datasets. We evaluate over three public datasets: 1) Places2 [45]; a dataset of photos commonly used for in-painting (test partition sampled as [21]); 2) Behance Artistic Media (BAM!) [38]; a dataset of artwork, using the test par-

tition of BAM! selected to evaluate in-painting in [14] comprising of 8 media styles; 3) DIP [34]; the dataset of images used to evaluate the original DIP algorithm of Ulyanov *et al.* Where baseline comparison is made to images from the latter, the specific network architecture published in [34] is replicated using according to [35] and the authors’ public implementation. Results are quantified via three objective metrics; PSNR (as in DIP) and structural similarity (SSIM) [37] are used to evaluate against ground truth, and we also report the perceptual metric (LPIPS) [44] used as fitness score in the GA optimization.

Training details. We implement NAS-DIP in Tensorflow, using ADAM and learning rate of 10^{-3} for all experiments. For NAS-DIP epoch count is 2000 otherwise this and other metaparameters are searched via the optimization. To alleviate the evaluation bottleneck in NAS (which takes, on average, 2-3 minutes per architecture proposal), we distribute the evaluation step over 16 Nvidia Titan-X GPUs capable of each evaluating two proposals concurrently for an average NAS-DIP search time of 3-6 hours total (for an average of

Dataset	Task	PSNR (higher better)		LPIPS (lower better)		SSIM (higher better)	
		DIP	Proposed	DIP	Proposed	DIP	Proposed
BAM! [39]	In-painting	16.8	19.76	0.42	0.25	0.32	0.62
Places2 [21]	In-painting	12.4	15.43	0.37	0.27	0.37	0.58
DIP [34]	In-painting	23.7	27.3	0.33	0.04	.76	0.92
	De-noising	17.6	18.95	0.13	0.09	0.73	0.85
	Super Resolution	18.9	19.3	0.38	0.16	0.48	0.62

Table 1. Per-dataset average performance of NAS-DIP (asymmetric representation) versus DIP over the dataset proposed in DIP, and for in-painting datasets BAM! and Places2.

Task	Image	PSNR (higher better)				LPIPS (lower better)				SSIM (higher better)			
		DIP	Sym NAS-DIP	Asym NAS-DIP	Asym NAS-DIP-T	DIP	Sym NAS-DIP	Asym NAS-DIP	Asym NAS-DIP-T	DIP	Sym NAS-DIP	Asym NAS-DIP	Asym NAS-DIP-T
In-painting	Vase	18.3	29.8	29.2	30.2	0.76	0.02	0.01	0.02	0.48	0.95	0.96	0.95
	Library	19.4	19.7	20.4	20.0	0.15	0.10	0.09	0.12	0.83	0.81	0.84	0.83
	Face	33.4	34.2	36.0	35.9	0.078	0.03	0.01	0.04	0.95	0.95	0.96	0.95
De-noising	Plane	23.1	25.8	25.7	25.9	0.15	0.096	0.09	0.07	0.85	0.90	0.92	0.94
	Snail	12.0	12.6	12.2	12.7	0.11	0.12	0.09	0.06	0.61	0.52	0.74	0.84
Super-Resolution	Zebra 4x	19.1	20.2	19.6	20.0	0.19	0.13	0.14	0.14	0.67	0.51	0.75	0.72
	Zebra 8x	18.7	19.1	18.96	19.1	0.57	0.29	0.19	0.20	0.28	0.34	0.48	0.47

Table 2. Detailed quantitative comparison of visual quality under three image restoration tasks, for the DIP dataset. Comparison between the architecture presented in the original DIP for each image, with the best architecture found under each of three variants of the proposed method: NAS-DIP, NAS-DIP-T and NAS-DIP constrained to symmetric E-D (subsec. 3.1). Quality metrics: LPIPS [44] (used in NAS objective); and both PSNR and SSIM [37] as external metrics.

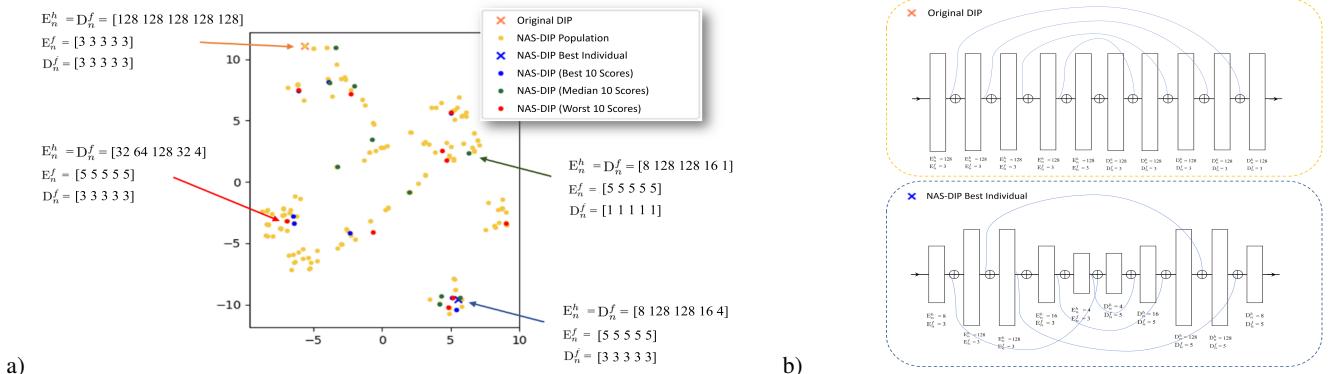


Figure 5. t-SNE visualizations in architecture space ($\mathbb{B}^{N(14+4N)}$) for NAS-DIP, $N = 5$ symmetric network: a) Population distribution at convergence; multi-modal distribution of similar best DIP architectures obtained for the super-resolution task (image inset); b) Visualizing best discovered architecture from the population (lower), and the classical DIP default architecture [35] (upper) for this task.

10-20 generations to convergence). Our DIP implementation extends the authors' public code for DIP [34]. Reproducibility is critical for NAS-DIP optimization; the same genome must yield the same perceptual score for a given source image. In addition to fixing all random seeds (*e.g.* for batch sampling and fixing the initial noise field), we take additional technical steps to avoid non-determinism in cuDNN *e.g.* avoidance of atomic add operations in image padding present in original DIP code.

4.1. Network Representation

We evaluate the proposed NAS method over three variants of the architecture representation proposed in subsec.3.1: NAS-DIP, NAS-DIP-T (in which epoch count is also optimized), and a constrained version of NAS-DIP forcing a symmetric E-D network. For all experiments, $N = 5$ enabling E-D networks of up to 10 (up-)convolutional layers

with gated skip connections. Performance is evaluated using PSNR for comparison with original DIP [34] as well as SSIM, and the LPIPS score used to guide NAS is also reported. Table 2 provides direct comparison on images from Ulyanov *et al.* [34]; visual output and convergence graphs are shown in Fig. 4. Following random initialization relative fitness gains of up to 30% are observed after 10-20 generations after which performance converges to values above the DIP baseline [34] in all cases.

For both in-painting and de-noising tasks, asymmetric networks are found that outperform any symmetric network, including the networks published in the original DIP for those images and tasks. For super-resolution, a symmetric network is found to outperform asymmetric networks and constraining the genome in this way aids in discovering a performant network. In all cases, it was unhelpful to optimize for epoch count T , despite prior observations on the

Dataset	Image	PSNR (higher better)				LPIPS (lower better)				SSIM (higher better)			
		DIP $p_m = 0.01$	DIP $p_m = 0.02$	DIP $p_m = 0.05$	DIP $p_m = 0.10$	DIP $p_m = 0.01$	DIP $p_m = 0.02$	DIP $p_m = 0.05$	DIP $p_m = 0.10$	DIP $p_m = 0.01$	DIP $p_m = 0.02$	DIP $p_m = 0.05$	DIP $p_m = 0.10$
In-painting	Vase Library	18.3	24.5	28.5	30.2	29.4	0.76	0.12	0.10	0.01	0.07	0.48	0.57
	Face	19.4	20.0	19.5	20.4	19.8	0.15	0.12	0.11	0.09	0.09	0.83	0.81
	Plane Snail	33.4	34.5	35.6	35.9	35.4	0.01	0.01	0.01	0.01	0.01	0.95	0.95
De-noising	Plane	23.1	22.9	23.7	25.9	23.6	0.15	0.11	0.01	0.09	0.09	0.85	0.85
	Snail	12.0	12.0	12.3	12.7	12.2	0.11	0.10	0.09	0.00	0.02	0.61	0.78
Super-Resolution	Zebra 4x	19.1	19.6	19.7	20.0	19.6	0.19	0.15	0.15	0.14	0.16	0.67	0.70
	Zebra 8x	18.7	18.9	19.0	19.1	19.0	0.57	0.34	0.24	0.19	0.28	0.32	0.35

Table 3. Quantitative results for varying mutation rates; chance of bit flip in offspring $p(r)$. Comparing each of three variants of the proposed method: NAS-DIP, NAS-DIP-T and NAS-DIP constrained to symmetric E-D (subsec. 3.1) against original DIP. Quality metrics: LPIPS [44] (used in NAS objective); and both PSNR and SSIM as external metrics.

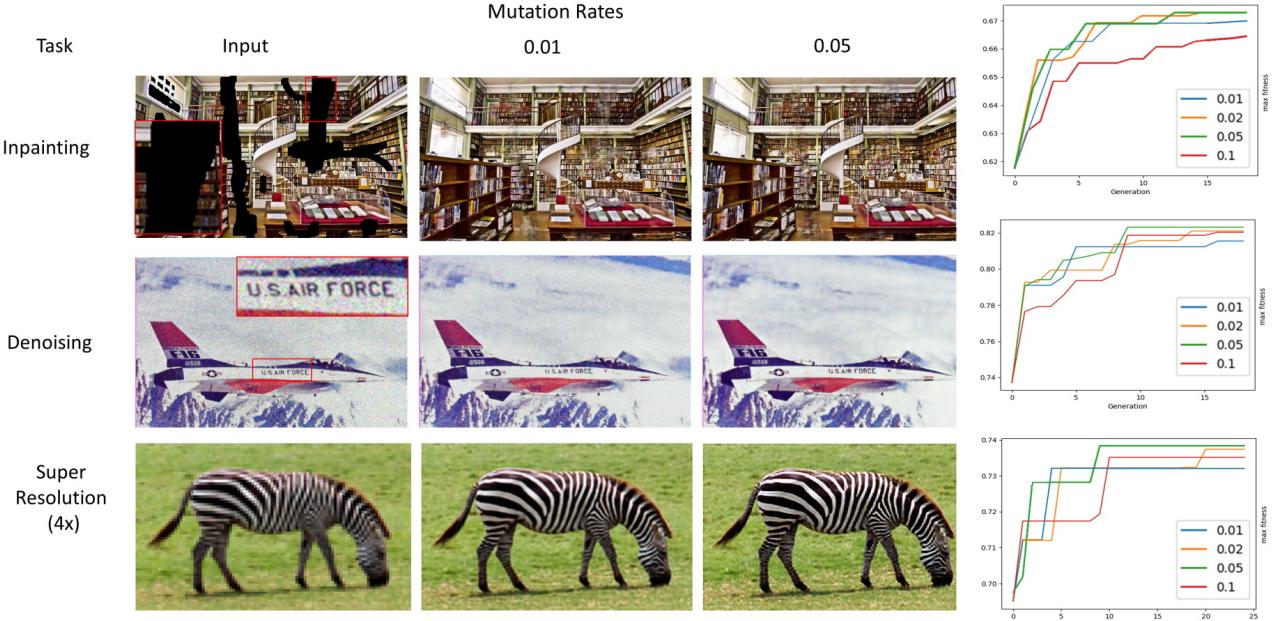


Figure 6. Result of varying mutation (bit flip) rate $p(r)$; improved visual quality (left, zoom recommended) is obtained with a mutation rate of $p(r) = 0.05$ equating to an average of 4 bit flips per offspring. Convergence graph for each visual example (right).

importance of T in obtained good reconstructions under DIP [34]. Table 1 broadens the experiment for NAS-DIP, averaging performance for three datasets: BAM! with 80 randomly sampled images, 10 from each 8 styles; Places2 with a 50 image subset included in [21]; DIP with 7 images used in [34]. For all three tasks and all three datasets, NAS-DIP discovers networks outperforming those of DIP [34].

4.2. Evaluating Mutation Rate

Population elitism requires a moderate mutation rate $p(r)$ to ensure population diversity and so convergence, yet raised too high convergence occurs at a lower quality level. We evaluate $p_m = \{0.01, 0.02, 0.05, 0.10\}$ in Table 3, observing that for all tasks a bit flip probability of 5% (*i.e.* an average of 4 bit flips per offspring for $N = 5$) encourages convergence the highest fitness value. This in turn correlates to highest visual quality according to both PSNR and SSIM external metrics. Fig. 6 provides representation visual output alongside convergence graphs. In general, convergence is achieved between 10-20 generations taking a few hours on single GPU for a constant population size $K = 500$.

4.3. Content aware in-painting

To provide a qualitative performance of the Content-Aware in-painting we compare our proposed approach against several contemporary baselines: PatchMatch [3] and Image Melding [7] which sample patches from within a single source image; and two methods that use millions or training images, the generative convnet approach 'Context encoder' [27] a recent GAN in-painting work [21], and the Style-aware in-painting [14]. We compare against the same baselines using scenic photographs in Places2, Fig 7 presents a visual comparison, and quantitative results are given in Tbl.1 over the image set included in [21] with the same mask regions published in that work.

4.4. Discovered Architectures

Architectures discovered by NAS-DIP are visualized in Fig. 5a, using t-SNE projection in the architecture space ($B^{N(14+4N)}$) for a solution population at convergence (generation 20) for a representative super-resolution task for which symmetric E-D network was sought (source image



Figure 7. Qualitative visual comparison of NAS-DIP in-painting quality against 5 baseline methods; examples from dataset Places2 [45].

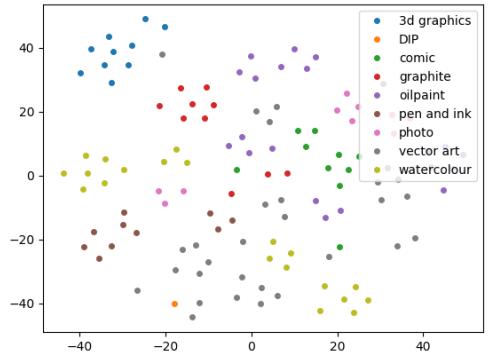


Figure 8. Content specific architecture discovery. NAS-DIP in-painted artwork results from BAM! [39] a) t-sne visualization of discovered architectures in $B^{N(14+4N)}$ for 80 in-painted artworks (7 BAM styles + DIP photograph); common visual styles yield a common best performing architecture under NAS-DIP. b) Sample in-painted results (source/mask left, output right).

inset). Distinct clusters of stronger candidate architectures have emerged each utilising all 10 convolutional stages and similar filter sizes but with distinct skip activations and chan-

nel widths; colour coding indicates fitness. Fig. 5b shows a schematic for the default architecture for this task published in [35] alongside the discovered best architecture for this task and image.

Fig. 8 visualizes 80 best performing networks for in-painting BAM! artworks of identical content (flowers) but exhibiting 8 different artistic styles. Each image has been run through NAS-DIP under loss eq. 4; representative output is included in Fig. 8. Visualizing the best performing architectures via t-SNE in $B^{N(14+4N)}$ reveals style-specific clusters; pen-and-ink, graphite sketches, 3D graphics renderings, comics, and vector artwork all form distinct groups while others e.g. watercolor form several clusters. We conclude that images that share a common visual style exhibit commonality in network architecture necessary perform image reconstruction well. This discovery suggests a potential application for NAS-DIP beyond architecture search to unsupervised clustering e.g. for visual aesthetics.

5. Conclusion

We reported the first neural architecture search (NAS) for image reconstruction under the recently proposed deep image prior (DIP) [34], learning a content-specific prior for a given source image in the form of an encoder-decoder (E-D) network architecture. Following the success of evolutionary search techniques for image classification networks [29, 28] we leveraged a genetic algorithm to search a binary space representing asymmetric E-D architectures and demonstrated its efficacy for image de-noising, in-painting and super-resolution. For the latter case, we observed a constrained version of our genome yielding symmetric networks exceeded that of asymmetric networks which benefited the other two tasks. All image restoration tasks were ‘blind’ and optimization was guided via a proxy measure for perceptual quality [44]. In all cases, we observed the performance of the discovered networks to significantly exceed classical DIP and the potential for content-specific architectures beyond image restoration to unsupervised style clustering. Future work could pursue the latter parallel application, as well as explore further generalizations of the architecture space beyond fully convolutional E-Ds to incorporate pooling and normalization strategies.

References

- [1] P. Angeline, G. Saunders, and J. Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE Trans. on Neural Networks*, 5(1):54–65, 1994. [2](#)
- [2] B. Baker, O. Gupta, N. Naik, and R. Raskar. Designing neural network architectures using reinforcement learning. In *Proc. ICLR*, 2017. [2](#)
- [3] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing,. In *Proc. ACM SIGGRAPH*, 2009. [2, 7, 8](#)
- [4] J. Bergstra, D. Yamins, and D. Cox. Making a science of model search: Hyper-parameter optimization in hundreds of dimensions for vision architectures. In *Proc. ICML*, 2013. [2](#)
- [5] H. Cai, T. Chen, W. Zhang, Y. Yu, and J. Wang. Efficient architecture search by network transformation. In *Proc. AAAI*, 2018. [2](#)
- [6] L. Chen, M. Collins, Y. Zhu, G. Papandreou, B. Zoph, F. Schroff, H. Adam, and J. Shlens. Searching for efficient multi-scale architectures for dense image prediction. In *Proc. NeurIPS*, pages 8713–8724, 2018. [2](#)
- [7] Soheil Darabi, Eli Shechtman, Connelly Barnes, Dan B Goldman, and Pradeep Sen. Image melding: combining inconsistent images using patch-based synthesis. In *ACM TOG*, 2012. [7, 8](#)
- [8] K. de Jong. Learning with genetic algorithms. *J. Machine Learning*, 3:121–138, 1988. [1](#)
- [9] T. Domhan, J. T. Springenberg, and F. Hutter. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In *Proc. IJCAI*, 2015. [2](#)
- [10] A. Efros and T. Leung. Texture Synthesis by non-parametric sampling. In *Proc. Intl. Conf. on Computer Vision (ICCV)*, 1999. [2](#)
- [11] T. Elsken, J. Metzen, and F. Hutter. Efficient multi-objective neural architecture search via lamarckian evolution. In *Proc. ICLR*, 2019. [1, 2](#)
- [12] T. Elsken, J. H. Metzen, and F. Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research (JMLR)*, 20:1–21, 2019. [2](#)
- [13] Y. Gandelsman, A. Shocher, and M. Irani. Double-dip: Unsupervised image decomposition via coupled deep-image-priors. In *Proc. CVPR*. IEEE, 2018. [2](#)
- [14] A. Gilbert, J. Collomosse, H. Jin, and B. Price. Disentangling structure and aesthetics for content-aware image completion. In *Proc. CVPR*, 2018. [2, 5, 7, 8](#)
- [15] D. Glasner, S. Bagon, and M. Irani. Super-resolution from a single image. In *Intl. Conference on Computer Vision (ICCV)*, 2009. [2](#)
- [16] Xinyu Gong, Shiyu Chang, Yifan Jiang, and Zhangyang Wang. Autogan: Neural architecture search for generative adversarial networks. In *Proc. ICCV*, 2019. [2](#)
- [17] James Hays and Alexei A Efros. Scene completion using millions of photographs. In *ACM Transactions on Graphics (TOG)*. ACM, 2007. [2](#)
- [18] K. He and J. Sun. Statistics of patch offsets for image completion. In *Euro. Conf. on Comp. Vision (ECCV)*, 2012. [2](#)
- [19] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Proc. NeurIPS*, pages 6629–6640, 2017. [4](#)
- [20] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors. *Automated Machine Learning: Methods, Systems, Challenges*. Springer, 2019. In press, available at <http://automl.org/book>. [2](#)
- [21] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and Locally Consistent Image Completion. *ACM Transactions on Graphics (Proc. of SIGGRAPH 2017)*, 36(4):107:1–107:14, 2017. [2, 5, 6, 7, 8](#)
- [22] V. Kwatra, A. Schodl, I. Essa, G. Turk, and A. Bobick. Graph-cut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics*, 3(22):277–286, 2003. [2](#)
- [23] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L. Li, F-F. Li, A. Yuille, J. Huang, and K. Murphy. Progressive neural architecture search. In *Proc. ECCV*, 2018. [1, 2](#)
- [24] Y. Liu and V. Caselles. Exemplar-based image inpainting using multiscale graph cuts. *IEEE Trans. on Image Processing*, pages 1699–1711, 2013. [2](#)
- [25] G. F. Miller, P. M. Todd, and S. U. Hedge. Designing neural networks using genetic algorithms. In *Proc. Intl. Conf. on Genetic Algorithms*, 1989. [2](#)
- [26] R. Negrinho and G. Gordon. Deeparchitect: Automatically designing and training deep architectures, 2017. arXiv:1704.08792. [2](#)
- [27] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. In *CVPR’16*, 2016. [7, 8](#)
- [28] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le. Aging evolution for image classifier architecture search. In *Proc. AAAI*, 2019. [1, 2, 8](#)
- [29] E. Real, S. Moore, A. Selle, S. Saxena, Y. Suematsu, Q. V. Le, and A. Kurakin. Large-scale evolution of image classifiers. In *Proc. ICLR*, 2017. [1, 2, 8](#)
- [30] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training GANs. In *Proc. NeurIPS*, pages 2234–2242, 2016. [4](#)
- [31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms, 2017. arXiv:1707.06347. [2](#)
- [32] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Proc. ICCV*, 2019. [2](#)
- [33] K. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10:99–127, 2002. [2](#)
- [34] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Deep image prior. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. [1, 2, 5, 6, 7, 8](#)
- [35] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Deep image prior. *Intl. Journal Computer Vision (IJCV)*, 2019. [5, 6, 8](#)
- [36] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proc. CVPR 2018*, 2018. [2](#)
- [37] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. [5, 6](#)
- [38] M. Wilber, C. Fang, H. Jin, A. Hertzmann, J. Collomosse, and S. Belongie. Bam! the behance artistic media dataset for recognition beyond photography. In *Proc. ICCV*, 2017. [5](#)

- [39] Michael J Wilber, Chen Fang, Hailin Jin, Aaron Hertzmann, John Collomosse, and Serge Belongie. Bam! the behance artistic media dataset for recognition beyond photography. *arXiv preprint arXiv:1704.08614*, 2017. [4](#), [6](#), [8](#)
- [40] Z. Wojna, V. Ferrari, S. Guadarrama, N. Silberman, L. Chen, A. Fathi, and J. Uiklings. The devil is in the decoder: Classification, regression and gans. *Intl. Journal of Computer Vision (IJCV)*, 127:1694–1706, December 2019. [1](#)
- [41] Raymond Yeh, Chen Chen, Teck Yian Lim, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with perceptual and contextual losses. *arXiv preprint arXiv:1607.07539*, 2016. [2](#)
- [42] H. Zhang, L. Mai, H. KJin, Z. Wang, N. Xu, and J. Collomosse. Video inpainting: An internal learning approach. In *Proc. ICCV*, 2019. [2](#)
- [43] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Sharon Huang, and Dimitris Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *IEEE Trans. PAMI*, 41(8):1947–1962, 2017. [2](#)
- [44] R. Zhang, P. Isola, A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. CVPR*, 2018. [2](#), [4](#), [5](#), [6](#), [7](#), [8](#)
- [45] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Antonio Torralba, and Aude Oliva. Places: An image database for deep scene understanding. *arXiv preprint arXiv:1610.02055*, 2016. [5](#), [8](#)
- [46] B. Zoph and Q. V. Le. Neural architecture search with reinforcement learning. In *Proc. ICLR*, 2017. [2](#)
- [47] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. Learning transferrable architectures for scalable image recognition. In *Proc. CVPR*, 2018. [2](#)