On-Demand Learning for Deep Image Restoration

Ruohan Gao and Kristen Grauman
University of Texas at Austin
{rhgao,grauman}@cs.utexas.edu

Abstract

While machine learning approaches to image restoration offer great promise, current methods risk training models fixated on performing well only for image corruption of a particular level of difficulty—such as a certain level of noise or blur. First, we examine the weakness of conventional "fixated" models and demonstrate that training general models to handle arbitrary levels of corruption is indeed non-trivial. Then, we propose an on-demand *learning algorithm for training image restoration models with deep convolutional neural networks. The main idea is to exploit a feedback mechanism to self-generate training instances where they are needed most, thereby learning models that can generalize across difficulty levels. On four restoration tasks—image inpainting, pixel interpolation, image deblurring, and image denoising—and three diverse datasets, our approach consistently outperforms both the status quo training procedure and curriculum learning alternatives.*

1. Introduction

Deep convolutional networks [21, 36, 14] have swept the field of computer vision and have produced stellar results on various recognition benchmarks in the past several years. Recently, deep learning methods are also becoming a popular choice to solve low-level vision tasks in image restoration, with exciting results [8, 27, 23, 44, 6, 17, 31, 43]. Restoration tasks such as image super-resolution, inpainting, deconvolution, matting, and colorization have a wide range of compelling applications. For example, deblurring techniques can mitigate motion blur in photos, and denoising methods can recover images corrupted by sensor noise.

A learning-based approach to image restoration enjoys the convenience of being able to *self-generate* training instances purely based on the original real images. Whereas training an object recognition system entails collecting images manually labeled with object categories by human annotators, an image restoration system can be trained with arbitrary, synthetically corrupted images. The original image itself is the ground-truth the system learns to recover.

While existing methods take advantage of this convenience, they typically do so in a problematic way. Image corruption exists in various degrees of severity, and so in real-world applications the difficulty of restoring images will also vary significantly. For example, as shown in Fig. 1, an inpainter may face images with varying sizes of missing content, and a deblurring system may encounter varying levels of blur. Intuitively, the more missing pixels or the more severe the blur, the more difficult the restoration task.

Figure 1. Illustration of four image restoration tasks: image inpainting, pixel interpolation, image deblurring, and image denoising. Each task exhibits increasing difficulty based on size of inpainting area, percentage of deleted pixels, degree of blurriness, and severity of noise. Our work aims to train all-rounder models that perform well across the spectrum of difficulty for each task.

However, the norm in existing deep learning methods is to train a model that succeeds at restoring images exhibiting *a particular level of corruption difficulty*. In particular, existing systems self-generate training instances with a manually fixed hyper-parameter that controls the degree of corruption—a fixed inpainting size [31, 43], a fixed percentage of corrupted pixels [43, 27], or a fixed level of white Gaussian noise [27, 41, 16, 3]. The implicit assumption is that at test time, either i) corruption will be limited to that same difficulty, or ii) some other process, e.g., [26, 28, 4], will estimate the difficulty level before passing the image to the appropriate, separately trained restoration system. Unfortunately, these are strong assumptions that remain difficult to meet in practice. As a result, existing methods risk training *fixated models*: models that perform well only at a particular level of difficulty. Indeed, deep networks can severely overfit to a certain degree of corruption. Taking the inpainting task as an example, a well-trained deep network may be able to inpaint a 32×32 block out of a 64×64 image very well, then fails miserably at inpainting a (seemingly easier) 10×10 block (see Fig. 2 and Sec. 4). Furthermore, as we will show, simply pooling training instances across all difficulty levels makes the deep network struggle to adequately learn the concept.

How should we train an image restoration system to succeed across a spectrum of difficulty levels? In this work we explore ways to let a deep learning system take control and guide its own training. This includes i) a solution that simply pools training instances from across difficulty levels, ii) a solution that focuses on easy/hard examples, iii) curriculum learning solutions that intelligently order the training samples from easy to hard, and iv) a new *on-demand learning* solution for training general deep networks across difficulty levels. Our approach relies on a feedback mechanism that, at each epoch of training, lets the system guide its own learning towards the right proportion of sub-tasks per difficulty level. In this way, the system itself can discover which sub-tasks deserve more or less attention.

To implement our idea, we devise a general encoder-decoder network amenable to several restoration tasks. We evaluate the approach on four low-level tasks—inpainting, pixel interpolation, image deblurring, and denoising—and three diverse datasets, CelebFaces Attributes [29], SUN397 Scenes [40], and the Denoising Benchmark 11 (DB11) [7, 3]. Across all tasks and datasets, the results consistently demonstrate the advantage of our proposed method. On-demand learning helps avoid the common (but thus far neglected) pitfall of overly specializing deep networks to a narrow band of distortion difficulty.

2. Related Work

Deep Learning in Low-Level Vision: Deep learning for image restoration is on the rise. Vincent *et al.* [38] propose one of the most well-known models: the stacked denoising auto-encoder. A multi-layer perceptron (MLP) is applied to image denoising by Burger *et al.* [3] and post-deblurring denoising by Schuler *et al.* [35]. Convolutional neural networks are also applied to natural image denoising [16] and used to remove noisy patterns (e.g., dirt/rain) [9]. Apart from denoising, deep learning is gaining traction for various other low-level tasks: super-resolution [8, 17], inpainting [31, 43], deconvolution [42], matting [6], and colorization [23, 44]. While many models specialize the architecture towards one restoration task, recent work by Liu *et al.* presents a unified network for multiple tasks [27]. Our encoder-decoder pipeline also applies across tasks, and serves as a good testbed for our main contribution—the idea of on-demand learning. Our idea has the potential to benefit any existing method currently limited to training with a narrow band of difficulty [31, 43, 16, 3, 35, 27].

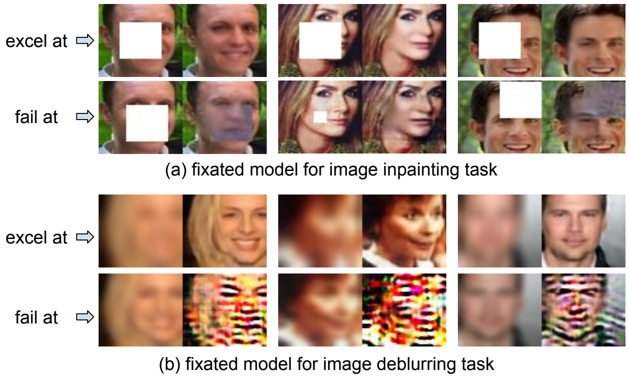
The fixation problem is also observed in recent denoising work, e.g., [3, 30], but without a dedicated and general solution. Burger *et al.* [3] attempt to train a network on patches corrupted by noise with different noise levels by giving the noise hyper-parameter as an additional input to the network. While the model can better denoise images at different noise levels, assuming the noise level is known at test time is problematic. Recently, Mao *et al.* [30] explore how the large capacity of a very deep network can help generalize across noise levels, but accuracy still declines noticeably from the fixated counterpart.

Curriculum and Self-Paced Learning: Training neural networks according to a *curriculum* can be traced back at least to Elman [11]. Prior work mainly focuses on supervised learning and a single task, like the seminal work of Bengio *et al.* [2]. Recently, Pentina *et al.* [32] pose curriculum learning in a multi-task learning setting, where sharing occurs only between subsequent tasks. Building on the curriculum concept, in *self-paced* learning, the system *automatically* chooses the order in which training examples are processed [22, 24]. We are not aware of any prior work in curriculum/self-paced learning that deals with image restoration. Like self-paced learning, our approach does not rely on human annotations to rank training examples from easiest to hardest. Unlike self-paced work, however, our on-demand approach self-generates training instances of a targeted difficulty.

Active Learning: Active learning is another way for a learner to steer its own learning. Active learning selects examples that seem most valuable for human labeling, and has been widely used in computer vision to mitigate manual annotation costs [19, 15, 10, 37, 25, 12, 18, 39]. Unlike active learning, our approach uses no human annotation, but instead actively synthesizes training instances of different corruption levels based on the progress of training. All our training data can be obtained for "free" and the ground-truth (original uncorrupted image) is always available.

3. Roadmap

We first examine the fixation problem, and provide concrete evidence that it hinders deep learning for image restoration (Sec. 4). Then we present a unified view of image restoration as a learning problem (Sec. 5.1) and describe inpainting, interpolation, deblurring, and denoising as instantiations (Sec. 5.2). Next we introduce the on-demand learning idea (Sec. 5.3) and our network architecture (Sec. 5.4). Finally, we present results (Sec. 6).



(a) fixated model for image inpainting task

(b) fixated model for image deblurring task

Figure 2. Illustration of the severity of overfitting for image inpainting and deblurring. The models overfit to a certain degree of corruption. They perform extremely well at that level of corruption, yet fail to produce satisfactory restoration results even for much easier sub-tasks. See Supp. for other tasks and details.

4. The Fixation Problem

The fixation problem arises when existing image restoration methods train a learning algorithm to restore images with a controlled degree of corruption [41, 43, 3, 35, 31, 27]. For example, Yeh *et al.* [43] train an image inpainter at a fixed size and location, and always delete 80% of pixels for pixel interpolation. Pathak *et al.* [31] mainly focus on a large central block for the inpainting task. Liu *et al.* [27] solve denoising, pixel interpolation, and color interpolation tasks all with a restricted degree of corruption. While such methods may fix the level of corruption in training as a proof of concept, they nonetheless do not offer a solution to make the model generally applicable.

Just how bad is the fixation problem in image restoration tasks? Fig. 2 helps illustrate. To get these results, we followed the current literature to train deep networks to target a certain degree of corruption for four applications (See Supp. for similar results of interpolation and denoising).[1]

Specifically, for the image inpainting task, following similar settings of prior work [31, 43], we train a model to inpaint a large central missing block of size 32×32. During testing, the resulting model can inpaint the central block of the same size at the same location very well (first row in Fig. 2-a). However, if we remove a block that is slightly shifted away from the central region, or remove a much *smaller* block, the model fails to inpaint satisfactorily (second row in Fig. 2-a). For the deblurring results in Fig. 2 (and interpolation & denoising results in Supp.), we attempt analogous trials, i.e., training for 80% missing pixels [43], a single width blur kernel or a single noise level, respectively, then observe poor performance by the fixated model on examples having different corruption levels.

The details of the deep networks used to generate the results in Fig. 2 are not identical to those in prior work. However, we stress that the limitation in their design that we wish to highlight is orthogonal to the particular architecture. To apply them satisfactorily in a general manner would require training a separate model for each hyper-parameter. Even if one could do so, it is difficult to gauge the corruption level in a novel image and decide which model to use. Finally, as we will see below, simply pooling training instances across all difficulty levels is also inadequate.

5. Approach

Next we present ideas to overcome the fixation problem.

5.1. Problem Formulation

While the problem of overfitting is certainly not limited to image restoration, both the issue we have exposed as well as our proposed solution are driven by its special ability to self-generate "free" training instances under specified corruption parameters. Recall that a real training image automatically serves as the ground-truth; the corrupted image is synthesized by applying a randomized corruption function.

We denote a real image as $\mathcal{R}$ and a corrupted image as $\mathcal{C}$ (e.g., a random block is missing). We model their joint probability distribution by $p(\mathcal{R},\mathcal{C}) = p(\mathcal{R})p(\mathcal{C}|\mathcal{R})$, where $p(\mathcal{R})$ is the distribution of real images and $p(\mathcal{C}|\mathcal{R})$ is the distribution of corrupted images given the original real image. In the case of a fixated model, $\mathcal{C}$ may be a deterministic function of $\mathcal{R}$ (e.g., specific blur kernel).

To restore the corrupted image, the most direct way is to find $p(\mathcal{R}|\mathcal{C})$ by applying Bayes' theorem. However, this is not feasible because $p(\mathcal{R})$ is intractable. Therefore, we resort to a point estimate $f(\mathcal{C},\mathbf{w})$ through an encoder-decoder style deep network **(details in Sec. 5.4)** by minimizing the following mean squared error objective:

$$\mathbb{E}_{\mathcal{R},\mathcal{C}}||\mathcal{R} - f(\mathcal{C},\mathbf{w})||_2^2. \tag{1}$$

Given a corrupted image $\mathcal{C}_0$, the minimizer of the above objective is the conditional expectation: $\mathbb{E}_{\mathcal{R}}[\mathcal{R}|\mathcal{C} = \mathcal{C}_0]$, which is the average of all possible real images that could have produced the given corrupted image $\mathcal{C}_0$.

Denote the set of real images $\{\mathcal{R}_i\}$. We synthesize corrupted images $\{\mathcal{C}_i\}$ correspondingly to produce training image pairs $\{\mathcal{R}_i, \mathcal{C}_i\}$. We train our deep network to learn its weights $\mathbf{w}$ by minimizing the following Monte-Carlo estimate of the mean squared error objective:

$$\hat{\mathbf{w}} = \operatorname*{argmin}_{\mathbf{w}} \sum_i ||\mathcal{R}_i - f(\mathcal{C}_i,\mathbf{w})||_2^2. \tag{2}$$

During testing, our trained deep network takes a corrupted image $\mathcal{C}$ as input and forwards it through the network to output $f(\mathcal{C},\mathbf{w})$ as the restored image.

5.2. Image Restoration Task Descriptions

Under this umbrella of a general image restoration solution, we consider four tasks.

[1] See Sec. 6 for quantitative results, and Sec. 5.4 for details about the encoder-decoder network used.

Image Inpainting The image inpainting task aims to refill a missing region and reconstruct the real image $\mathcal{R}$ of an incomplete corrupted image $\mathcal{C}$ (e.g., with a contiguous set of pixels removed). In applications, the "cut out" part of the image would represent an occlusion, cracks in photographs, or an object that should be removed from the photo. Unlike [31, 43], we make the missing square block randomized across the whole image in both position and scale.

Pixel Interpolation Related to image inpainting, pixel interpolation aims to refill non-contiguous deleted pixels. The network has to reason about the image structure and infer values of the deleted pixels by interpolating from neighboring pixels. Applications include more fine-grained inpainting tasks such as removing dust spots in film.

Image Deblurring The image deblurring task aims to remove the blurring effects of a corrupted image $\mathcal{C}$ to restore the corresponding real image $\mathcal{R}$. We use Gaussian smoothing to blur a real image to create training examples. The kernel's horizontal and vertical widths (σ_x and σ_y) control the degree of blurriness and hence the difficulty. Applications include removing motion blur or defocus aberration.

Image Denoising The image denoising task aims to remove additive white Gaussian (AWG) noise of a corrupted image $\mathcal{C}$ to restore the corresponding real image $\mathcal{R}$. We corrupt real images by adding noise drawn from a zero-mean normal distribution with variance σ (the noise level).

5.3. On-Demand Learning for Image Restoration

All four image restoration tasks offer a spectrum of difficulty. The larger the region to inpaint, the larger the percentage of deleted pixels, the more blurry the corrupted image, or larger the variance of the noise, the more difficult the corresponding task. To train a system that generalizes across task difficulty, a natural approach is to simply pool training instances across all levels of difficulty, insisting that the learner simultaneously tackle all degrees of corruption at once. Unfortunately, as we will see in our experiments, this approach can struggle to adequately learn the concept.

Instead, we present an *on-demand* learning approach in which the system dynamically adjusts its focus where it is most needed. First, we divide each restoration task into N sub-tasks of increasing difficulty. During training, we aim to jointly train the deep neural network restoration model (architecture details below) to accommodate all N sub-tasks. Initially, we generate the same number of training examples from each sub-task in every batch. At the end of every epoch, we validate on a small validation set and evaluate the performance of the current model on all sub-tasks. We compute the mean peak signal-to-noise ratio (PSNR) for all images in the validation set for each sub-task.[2] A lower PSNR indicates a more difficult sub-task, suggesting that the model needs more training on examples of this sub-task. Therefore, we generate more training examples for this sub-task in each batch in the next epoch. That is, we re-distribute the corruption levels allocated to the same set of training images. Specifically, we assign training examples in each batch for the next epoch inversely proportionally to the mean PSNR P_i of each sub-task T_i. Namely,

$$B_i = \frac{1/P_i}{\sum_{i=1}^{N} 1/P_i} \cdot \mathbb{B}, \tag{3}$$

where $\mathbb{B}$ is the batch size and B_i is the number of of training examples assigned to sub-task T_i for the next epoch. Please see Supp. for the pseudocode of our algorithm.

On-demand learning bears some resemblance to boosting and hard negative mining, in that the system refocuses its effort on examples that were handled unsatisfactorily by the model in previous iterations of learning. However, whereas they reweight the influence given to individual (static) training samples, our idea is to self-generate *new* training instances in specified difficulty levels based on the model's current performance. Moreover, the key is not simply generating more difficult samples, but to let the network steer its own training process, and decide how to schedule the *right proportions* of difficulty.

Our approach discretizes the difficulty space via its intrinsic continuity property for all tasks. However, it is the network itself that determines the difficulty level for each discretized bin based on the restoration quality (PSNR) from our algorithm, and steers its own training.

We arrived at this simple but effective approach after investigating several other schemes inspired by curriculum and multi-task learning, as we shall see below. In particular, we also developed a new curriculum approach that stages the training samples in order of their difficulty, starting with easier instances (less blur, smaller cut-outs) for the system to gain a basic representation, then moving onto harder ones (more blur, bigger cut-outs). Wary that what appears intuitively easier to us as algorithm designers need not be easier to the deep network, we also considered an "anti-curriculum" approach that reverses that ordering, e.g., starting with bigger missing regions for inpainting. More details are given in Sec. 6.3.

5.4. Deep Learning Network Architecture

Finally, we present the network architecture used for all tasks to implement our on-demand learning idea. Our image restoration network is a simple encoder-decoder pipeline. See Fig. 3. The encoder takes a corrupted image $\mathcal{C}$ of size 64×64 as input and encodes it in the latent feature space. The decoder takes the feature representation and outputs the

[2] PSNR is widely used as a good approximation to human perception of quality in image restoration tasks. We found PSNR to be superior to an L2 loss; because it is normalized by the max possible power and expressed in log scale, it is better than L2 at comparing across difficulty levels.

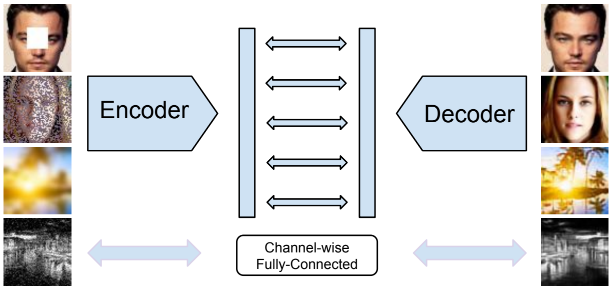


Figure 3. Network architecture for our image restoration framework, an encoder-decoder pipeline connected by a channel-wise fully-connected layer. See Supp. for details.

restored image $f(\mathcal{C}, \mathbf{w})$. Our encoder and decoder are connected through a channel-wise fully-connected layer. The loss function we use during training is L2 loss, which is the mean squared error between the restored image $f(\mathcal{C}, \mathbf{w})$ and the real image $\mathcal{R}$. We use a symmetric encoder-decoder pipeline that is efficient for training and effective for learning. It is a unified framework that can be used for all four image restoration tasks. Please see Supp. for the complete network architecture and detailed design choices.

6. Experiments

We compare with traditional "fixated" learners, hard negative mining, multi-task and curriculum methods, and several existing methods in the literature [31, 1, 7, 3, 13, 34, 5].

6.1. Datasets

We experiment with three datasets: CelebFaces Attributes (CelebA) [29], SUN397 Scenes [40], and the Denoising Benchmark 11 (DB11) [7, 3]. We do not use any of the accompanying labels. For CelebA, we use the first 100,000 images as the training set. Among the rest of the images, we hold out 1,000 images each for the validation and test sets. For SUN397, similarly, we use 100,000 images for training, and 1,000 each for validation and testing. DB11 consists of 11 standard benchmark images, such as "Lena" and "Barbara", that have been widely used to evaluate denoising algorithms [7, 3]. We only use this dataset to facilitate comparison with prior work.

6.2. Implementation Details

Our image restoration pipeline is implemented in Torch[3]. We use ADAM [20] as the stochastic gradient descent solver. We use the default solver hyper-parameters suggested in [33] and batch size $\mathbb{B} = 100$ in all experiments.

The number of sub-tasks N for on-demand learning controls a trade-off between precision and run-time. Larger values of N will allow the on-demand learning algorithm more fine-grained control on its sample generation, which could lead to better results. However, the time complexity for validating on all sub-tasks at the end of each epoch is $O(N)$. Therefore, a more fine-grained division of training examples among sub-tasks comes at the cost of longer running time during training. For consistency, we divide each of the image restoration tasks into $N = 5$ difficulty levels during training. We have not tried any other values, and it is possible other settings could improve our results further. We leave how to select the optimal value of N as future work. An extra level (level 6) is added during testing. The level 6 sub-task can be regarded as an "extra credit" task that strains the generalization ability of the obtained model.

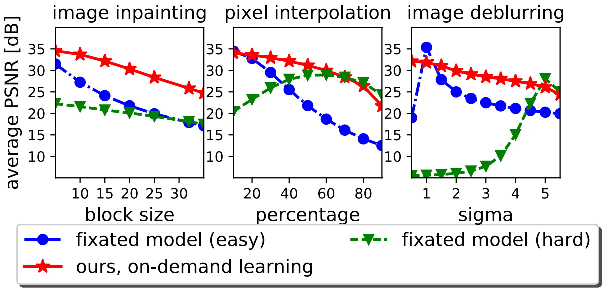


Figure 4. Our algorithm vs. fixated models on CelebA (See Supp. for results on SUN397 and denoising). Our algorithm performs well over the spectrum of difficulty, whereas fixated models perform well at only a certain level of corruption.

Image Inpainting: We focus on inpainting missing square blocks of size 1×1 to 30×30 at different locations across the image. We divide the range into the following five intervals, which define the five difficulty levels: $1 \times 1 - 6 \times 6$, $7 \times 7 - 12 \times 12$, $13 \times 13 - 18 \times 18$, $19 \times 19 - 24 \times 24$, $25 \times 25 - 30 \times 30$.

Pixel Interpolation: We train the pixel interpolation network with images corrupted by removing a random percentage of pixels. The percentage is sampled from the range $[0\%, 75\%]$. We divide the range into the following five difficulty levels: $0\% - 15\%$, $15\% - 30\%$, $30\% - 45\%$, $45\% - 60\%$, $60\% - 75\%$.

Image Deblurring: Blur kernel widths σ_x and σ_y, which are sampled from the range $[0, 5]$, control the level of difficulty. We consider the following five difficulty levels: $0 - 1$, $1 - 2$, $2 - 3$, $3 - 4$, $4 - 5$.

Image Denoising: We use gray-scale images for denoising. The variance σ of additive white Gaussian noise is sampled from the range [0,100]. We use the following five difficulty levels: $0 - 20$, $20 - 40$, $40 - 60$, $60 - 80$, $80 - 100$.

6.3. Baselines

For fair comparisons, all baseline models and our method are trained for the same amount of time (1500 epochs). Therefore, while our algorithm shifts the distribution of training instances it demands on the fly, it *never receives more training instances than the baselines*.

Fixated Model (Hard): The image restoration network is trained only on one level of severely corrupted images.

[3] `https://github.com/rhgao/on-demand-learning`

	CelebA						SUN397					
	Image Deblurring		Pixel Interpolation		Image Inpainting		Image Deblurring		Pixel Interpolation		Image Inpainting	
	L2 Loss	PSNR	L2 Loss	PSNR	L2 Loss	PSNR	L2 Loss	PSNR	L2 loss	PSNR	L2 Loss	PSNR
Rigid Joint Learning	1.58	29.40 dB	1.02	31.86 dB	1.05	32.11 dB	2.32	28.53 dB	1.29	31.98 dB	1.80	31.13 dB
Cumulative Curriculum	1.85	28.70 dB	1.11	31.68 dB	1.28	31.47 dB	2.64	27.86 dB	1.36	31.70 dB	1.94	30.75 dB
Cumulative Anti-Curriculum	1.49	29.31 dB	1.01	31.96 dB	1.04	31.90 dB	2.39	28.34 dB	1.25	32.02 dB	1.90	30.44 dB
Staged Curriculum	125	15.59 dB	2.10	28.51 dB	1.18	31.30 dB	133	14.44 dB	2.36	28.13 dB	1.87	30.42 dB
Staged Anti-Curriculum	5.54	25.43 dB	7.76	27.82 dB	4.80	28.10 dB	6.27	25.17 dB	7.05	27.76 dB	4.35	28.42 dB
Hard Mining	2.98	27.33 dB	1.85	29.15 dB	3.31	29.47 dB	3.98	26.35 dB	1.82	29.01 dB	2.61	29.83 dB
On-Demand Learning	**1.41**	**29.58 dB**	**0.95**	**32.09 dB**	**0.99**	**32.30 dB**	**2.11**	**28.70 dB**	**1.19**	**32.21 dB**	**1.69**	**31.38 dB**

Table 1. Summary of the overall performance of all algorithms for three image restoration tasks on the CelebA and SUN397 datasets. (See Supp. for similar results on denoising). Overall performance is measured by the mean L2 loss (in ‰, lower is better) and mean PSNR (higher is better) averaged over all sub-tasks. Numbers are obtained over 20 trials with standard error (SE) approximately 5×10^{-6} for L2 loss and 3×10^{-3} for PSNR on average. A paired t-test shows the results are significant with p-value 5×10^{-30}.

Fixated Model (Easy): The image restoration network is trained only on one level of lightly corrupted images.

Rigid Joint Learning: The image restoration network is trained on all sub-tasks of different difficulty levels (level 1-*N*) jointly. We allocate the same number of training examples for each sub-task per batch.

Staged Curriculum Learning: The network starts at the easiest sub-task (level 1) and gradually switches to more difficult sub-tasks. At any time, the network trains on only one sub-task. It trains on each sub-task for 300 epochs.

Staged Anti-Curriculum Learning: The network performs as the above, but reverses the curriculum to start with the most difficult task (level *N*).

Cumulative Curriculum Learning: The network starts at the easiest sub-task (level 1) and gradually adds more difficult sub-tasks and learns them jointly. More specifically, the baseline model is first trained on level 1 sub-task for 300 epochs, and then performs rigid joint learning on sub-tasks of level 1 and 2 for 300 epochs, followed by performing rigid joint learning on sub-tasks of level 1,2,3 for another 300 epochs, and so on.

Cumulative Anti-Curriculum Learning: The network performs as the above, but reverses the curriculum.

Hard Mining: For each task, we create a dataset of 1M images with various corruptions. We directly train on the dataset for 50 epochs, then continue training with hard mining until convergence. To select hard examples, we identify those with the largest reconstruction loss and use them to compute and back propagate gradients. Specifically, in each batch, we select the 10 with highest loss.

As far as source training data, the fixated model baselines represent the status quo in using deep learning for image restoration tasks [27, 31, 43, 41, 16, 3, 35], while the rigid joint learning baseline represents the natural solution of pooling all training data [16, 30]. The curriculum methods are of our own design. The hard mining baseline is designed to best mimic traditional hard negative mining strategies. Our system never receives more training images than any baseline; only the distribution of distortions among those images evolves over epochs. We test all algorithms across the whole spectrum of difficulty (sub-task 1-*N* and an extra level), and synthesize corresponding testing instances randomly over 20 trials. No methods have prior knowledge of the test distribution, thus none are able to benefit from better representing the expected test distribution during training.

6.4. Fixated Model vs. Our Model

We first show that our on-demand algorithm successfully addresses the fixation problem, where the fixated models employ an identical network architecture to ours. For inpainting, the fixated model (hard/easy) is only trained to inpaint 32×32 or 5×5 central blocks, respectively; for pixel interpolation, 80% (hard) or 10% (easy) pixels are deleted; for deblurring, $\sigma_x = \sigma_y = 5$ (hard) or $\sigma_x = \sigma_y = 1$ (easy); for denoising, $\sigma = 90$ (hard) or $\sigma = 10$ (easy).

Fig. 4 summarizes the test results on images of various corruption levels on CelebA (See Supp. for all). The fixated model overfits to a specific corruption level (easy or hard). It succeeds beautifully for images within its specialty (e.g., the sudden spike in Fig. 4 (right)), but performs poorly when forced to attempt instances outside its specialty. For inpainting, the fixated models also overfit to the central location, and thus cannot perform well over the whole spectrum. In contrast, models trained using our algorithm perform well across the spectrum of difficulty.

6.5. Comparison to Existing Inpainter

We also compare our image inpainter against a state-of-the-art inpainter from Pathak *et al.* [31]. We adapt their provided code[4] and follow the same procedures as in [31] to train two variants on CelebA: one is only trained to inpaint central square blocks, and the other is trained to inpaint regions of arbitrary shapes using random region dropout. Table 2 compares both variants to our model on the held out CelebA test set. Their first inpainter performs very well when testing on central square blocks (left cols), but it is unable to produce satisfactory results when tested on square blocks located anywhere in the image (right cols). Their second model uses random region dropout during training, but our inpainter still performs much better. The "all-rounder" inpainter trained under our on-demand learn-

[4] https://github.com/pathak22/context-encoder

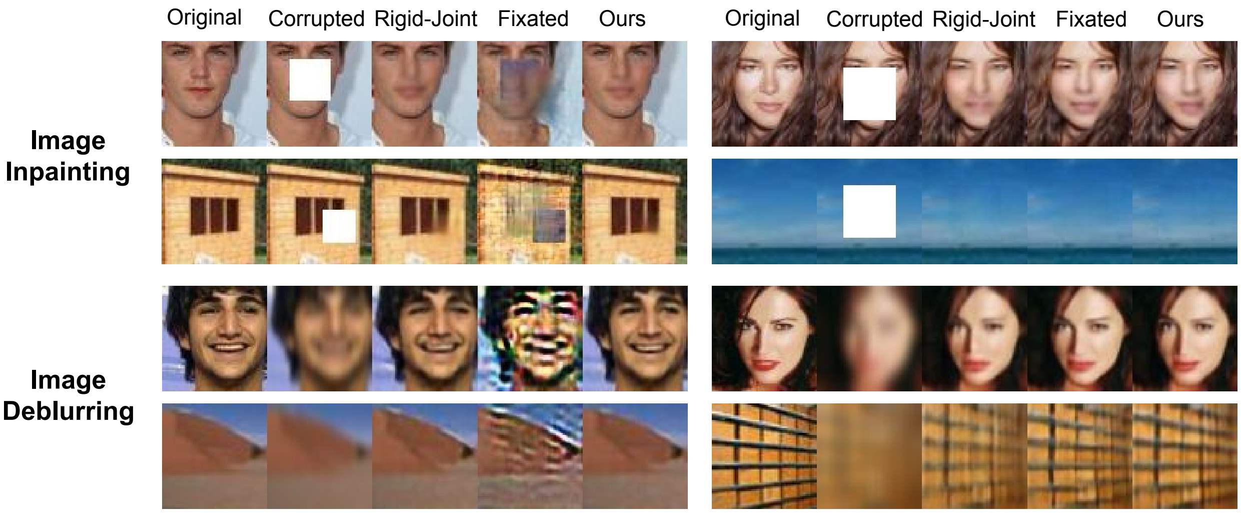


Figure 5. For each task, the first row shows testing examples of CelebA dataset, and the second row shows examples of SUN397 dataset. While the fixated model can only perform well at one level of difficulty (right col), the all-rounder models trained using our proposed algorithm perform well on images with various corruption levels. See Supp. for similar results on pixel interpolation and image denoising.

ing framework does similarly well in both cases. It is competitive—and stronger on the more difficult task—even without the use of adversarial loss as used in their framework during training. Please also see Supp. for some real-world applications (e.g., object removal in photos).

Method	Central Square Block		Arbitrary Square Block	
	L2 Loss	PSNR	L2 Loss	PSNR
Pathak *et al.* [31] Center	**0.83%**	**22.16 dB**	6.84%	11.80 dB
Pathak *et al.* [31] +Rand drop	2.47%	16.18 dB	2.51%	16.20 dB
Ours	0.93%	20.74 dB	**1.04%**	**20.31 dB**

Table 2. Image inpainting accuracy for CelebA on two test sets.

6.6. On-Demand Learning vs. Alternative Models

We next compare our method to the hard mining, curriculum and multi-task baselines. Table 1 shows the results (Please see Supp. for similar results on image denoising). We report average L2 loss and PSNR over all test images. Our proposed algorithm consistently outperforms the well-designed baselines. Hard mining overfits to the hard examples in the static pool of images, and the Staged (Anti-)Curriculum Learning algorithms overfit to the last sub-task they are trained on, yielding inferior overall performance. The Cumulative (Anti-)Curriculum Learning algorithms and Rigid Joint Learning are more competitive, because they learn sub-tasks jointly and try to perform well on sub-tasks across all difficulty levels. However, the higher noise levels dominate their training procedure by providing stronger gradients. As training goes on, these methods cannot provide the optimal distribution of gradients across corruption levels for effective learning. By automatically guiding the balance among sub-tasks, our algorithm obtains the best all-around performance. Especially, we observe our approach generalizes better to difficulty levels never seen before, and performs better on the "extra credit" sub-task.

Fig. 5 shows qualitative examples output by our method for inpainting and deblurring. See Supp. for similar results of interpolation and denoising. These illustrate that models trained using our proposed on-demand approach perform well on images of different degrees of corruption. *With a single model*, we inpaint blocks of different sizes at arbitrary locations, restore corrupted images with different percentage of deleted pixels, deblur images at various degrees of blurriness, and denoise images of various noise levels. In contrast, the fixated models can only perform well at one level of difficulty that they specialize in. Even though we experiment with images of small scale (64×64) for efficiency, qualitative results of our method are still visually superior to other baselines including rigid-joint learning.

We argue that the gain of our algorithm does not rest on *more* training instances of certain sub-tasks, but rather a suitable combination of sub-tasks for effective training. Indeed, we never use more training instances than any baseline. To emphasize this point, we separately train a rigid-joint learning model using 200,000 training images (the original 100,000 and the *extra* 100,000) from CelebA. [5] We observe that the extra training instances do not help rigid joint training converge to a better local minimum. This result suggests on-demand learning's gains persist even if our method is put at the disadvantage of having access to 50% fewer training images.

How does the system focus its attention as it learns? To get a sense, we examine the learned allocation of sub-tasks during training. Initially, each sub-task is assigned the same number of training instances per batch. In all tasks, as training continues, the network tends to dynamically shift

[5] The other datasets lack sufficient data to run this test.

Image	[1]	[7]	[3]	[13]	[34]	[5]	Ours
Barbara	29.49	30.67	29.21	**31.24**	28.95	29.41	28.92 / 29.63
Boat	29.24	29.86	29.89	30.03	29.74	29.92	**30.11 / 30.15**
C.man	28.64	29.40	29.32	29.63	29.29	29.71	29.41 / **29.78**
Couple	28.87	29.68	29.70	29.82	29.42	29.71	**30.04 / 30.02**
F.print	27.24	27.72	27.50	**27.88**	27.02	27.32	27.81 / 27.77
Hill	29.20	29.81	29.82	29.95	29.61	29.80	**30.03 / 30.04**
House	32.08	32.92	32.50	**33.22**	32.16	32.54	33.14 / 33.03
Lena	31.30	32.04	32.12	32.24	31.64	32.01	**32.44 / 32.36**
Man	29.08	29.58	29.81	29.76	29.67	29.88	**29.92 / 29.96**
Montage	30.91	32.24	31.85	32.73	31.07	32.29	32.34 / **32.74**
Peppers	29.69	30.18	30.25	30.40	30.12	**30.55**	30.29 / 30.48

Table 3. PSNRs (in dB, higher is better) on standard test images, $\sigma = 25$. We show the performance of both our all-rounder model (left) and fixated model (right) of our image denoising system. Note that our on-demand learning model is the *only* one that *does not* exploit the noise level (σ) of test images.

its allocations to put more emphasis on the "harder" subtasks, while never abandoning the "easiest" ones. The right proportions of difficulty lead to the superior overall performance of our model.

6.7. Comparison to Existing Denoising Methods

In previous sections, we have compared our on-demand learning denoising model with alternative models. To facilitate comparison to prior work and demonstrate the competitiveness of our image restoration framework, in this section we perform a case study on the image denoising task using our denoising system. See Supp. for details about how we denoise images of arbitrary sizes.

We test our image denoising system on DB11 [7, 3]. We first compare our model with state-of-the-art denoising algorithms on images with a specific degree of corruption ($\sigma = 25$, commonly adopted to train fixated models in the literature). Table 3 summarizes the results[6]. Although using a simple encoder-decoder network, we still have very competitive performance. Our on-demand learning model outperforms all six existing denoising algorithms on 5 out of the 11 test images (7 out of 11 for the fixated version of our denoising system), and is competitive on the rest. Note that our on-demand learning model *does not need to know the noise level of test images*. However, all other compared algorithms either have to know the exact noise level (σ value), or train a separate model for this specific level of noise ($\sigma = 25$).

More importantly, the advantage of our method is more apparent when we test across the spectrum of difficulty levels. We corrupt the DB11 images with AWG noise of increasing magnitude and compare with the denoising algorithms BM3D [7] and MLP [3] based on the authors' public code[7][8] and reported results [3]. We compare with two MLP models: one is trained only on corrupted images of $\sigma = 25$, and the other is trained on images with various noise levels.

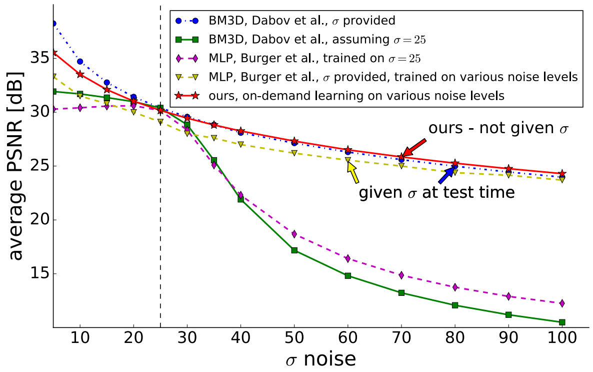


Figure 6. Comparisons of the performance of image denoising systems at different noise levels. Our system is competitive over the whole spectrum of noise levels without requiring knowledge of the corruption level of test images. Best viewed in color.

BM3D and MLP both need to be provided with the correct level of the noise (σ) during testing. We also run a variant of BM3D for different noise levels but fix the specified level of noise to $\sigma = 25$.

Fig. 6 shows the results. We see that the MLP model [3] trained on a single noise level only performs well at that specific level of corruption. Similarly, BM3D [7] needs the correct input of noise level in order to perform well across the spectrum of noise levels. In contrast, our image denoising system consistently performs well on all noise levels, yet we *do not* assume knowledge of σ during testing. This is an essential advantage for real-world applications.

7. Conclusion

We have addressed a common problem in existing work that leverages deep models to solve image restoration tasks: overfitting. We devise a symmetric encoder-decoder network amenable to all image restoration tasks, and propose a simple but novel *on-demand learning* algorithm that turns a fixated model into one that performs well on a task across the spectrum of difficulty. Experiments on four tasks on three diverse datasets demonstrate the effectiveness of our method. Our on-demand learning idea is a general concept not restricted to image restoration tasks, and may be applicable in other domains as well, e.g., self-supervised feature learning. As future work, we plan to design continuous subtasks to avoid discrete sub-task bins, and we will explore ways to make an image restoration task more self-paced by allowing the network to design the most desired sub-task on its own. Finally, another promising direction is to explore combinations of different types of distortions.

Acknowledgements: This research is supported in part by NSF IIS-1514118. We also gratefully acknowledge the support of the Texas Advanced Computing Center (TACC) and a GPU donation from Facebook.

[6] We take the reported numbers [3] or use the authors' public available code [13, 34, 5] to generate the results in Table 3.

[7] http://www.cs.tut.fi/~foi/GCF-BM3D/

[8] http://people.tuebingen.mpg.de/burger/neural_denoising/

References

[1] M. Aharon, M. Elad, and A. Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 2006. 5, 8

[2] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *ICML*, 2009. 2

[3] H. C. Burger, C. J. Schuler, and S. Harmeling. Image denoising: Can plain neural networks compete with bm3d? In *CVPR*, 2012. 1, 2, 3, 5, 6, 8

[4] G. Chen, F. Zhu, and P. Ann Heng. An efficient statistical method for image noise level estimation. In *ICCV*, 2015. 1

[5] Y. Chen and T. Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *TPAMI*, 2016. 5, 8

[6] D. Cho, Y.-W. Tai, and I. Kweon. Natural image matting using deep convolutional neural networks. In *ECCV*, 2016. 1, 2

[7] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, 2007. 2, 5, 8

[8] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *TPAMI*, 2016. 1, 2

[9] D. Eigen, D. Krishnan, and R. Fergus. Restoring an image taken through a window covered with dirt or rain. In *CVPR*, 2013. 2

[10] E. Elhamifar, G. Sapiro, A. Yang, and S. Shankar Sasrty. A convex optimization framework for active learning. In *ICCV*, 2013. 2

[11] J. L. Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 1993. 2

[12] A. Freytag, E. Rodner, and J. Denzler. Selecting influential examples: Active learning with expected model output changes. In *ECCV*, 2014. 2

[13] S. Gu, L. Zhang, W. Zuo, and X. Feng. Weighted nuclear norm minimization with application to image denoising. In *CVPR*, 2014. 5, 8

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[15] S.-J. Huang, R. Jin, and Z.-H. Zhou. Active learning by querying informative and representative examples. *TPAMI*, 2014. 2

[16] V. Jain and S. Seung. Natural image denoising with convolutional networks. In *NIPS*, 2009. 1, 2, 6

[17] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *ECCV*, 2016. 1, 2

[18] C. Käding, A. Freytag, E. Rodner, P. Bodesheim, and J. Denzler. Active learning and discovery of object categories in the presence of unnameable instances. In *CVPR*, 2015. 2

[19] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell. Gaussian processes for object categorization. *IJCV*, 2010. 2

[20] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 5

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[22] M. P. Kumar, B. Packer, and D. Koller. Self-paced learning for latent variable models. In *NIPS*, 2010. 2

[23] G. Larsson, M. Maire, and G. Shakhnarovich. Learning representations for automatic colorization. *ECCV*, 2016. 1, 2

[24] Y. J. Lee and K. Grauman. Learning the easy things first: Self-paced visual category discovery. In *CVPR*, 2011. 2

[25] X. Li and Y. Guo. Multi-level adaptive active learning for scene classification. In *ECCV*, 2014. 2

[26] C. Liu, W. T. Freeman, R. Szeliski, and S. B. Kang. Noise estimation from a single image. In *CVPR*, 2006. 1

[27] S. Liu, J. Pan, and M.-H. Yang. Learning recursive filters for low-level vision via a hybrid neural network. In *ECCV*, 2016. 1, 2, 3, 6

[28] X. Liu, M. Tanaka, and M. Okutomi. Single-image noise level estimation for blind denoising. *IEEE transactions on image processing*, 2013. 1

[29] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 2, 5

[30] X.-J. Mao, C. Shen, and Y.-B. Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. *NIPS*, 2016. 2, 6

[31] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 1, 2, 3, 4, 5, 6, 7

[32] A. Pentina, V. Sharmanska, and C. H. Lampert. Curriculum learning of multiple tasks. In *CVPR*, 2015. 2

[33] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 5

[34] U. Schmidt and S. Roth. Shrinkage fields for effective image restoration. In *CVPR*, 2014. 5, 8

[35] C. J. Schuler, H. Christopher Burger, S. Harmeling, and B. Scholkopf. A machine learning approach for non-blind image deconvolution. In *CVPR*, 2013. 2, 3, 6

[36] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 1

[37] S. Vijayanarasimhan and K. Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. *IJCV*, 2014. 2

[38] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008. 2

[39] Z. Wang, B. Du, L. Zhang, L. Zhang, M. Fang, and D. Tao. Multi-label active learning based on maximum correntropy criterion: Towards robust and discriminative labeling. In *ECCV*, 2016. 2

[40] J. Xiao, K. A. Ehinger, J. Hays, A. Torralba, and A. Oliva. Sun database: Exploring a large collection of scene categories. *IJCV*, 2014. 2, 5

[41] J. Xie, L. Xu, and E. Chen. Image denoising and inpainting with deep neural networks. In *NIPS*, 2012. 1, 3, 6

[42] L. Xu, J. S. Ren, C. Liu, and J. Jia. Deep convolutional neural network for image deconvolution. In *NIPS*, 2014. 2

[43] R. Yeh, C. Chen, T. Y. Lim, M. Hasegawa-Johnson, and M. N. Do. Semantic image inpainting with perceptual and contextual losses. *arXiv preprint arXiv:1607.07539*, 2016. 1, 2, 3, 4, 6

[44] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. *ECCV*, 2016. 1, 2