ABSTRACT

In this paper, we propose a new control framework called the moving endpoint control to restore images corrupted by different degradation levels using a single model. The proposed control problem contains an image restoration dynamic which is modeled by a convolutional RNN. The moving endpoint, which is essentially the terminal time of the associated dynamic, is determined by a policy network. We call the proposed model the dynamically unfolding recurrent restorer (DURR). Numerical experiments show that DURR is able to achieve state-of-the-art performances on blind image denoising and JPEG image deblocking. Furthermore, DURR can well generalize to images with higher degradation levels that are not included in the training stage.

1 INTRODUCTION

Image restoration, including image denoising, deblurring, inpainting, etc., is one of the most important areas in imaging science. Its major purpose is to obtain high quality reconstructions of images corrupted in various ways during imaging, acquiring, and storing, and enable us to see crucial but subtle objects that reside in the images. Image restoration has been an active research area. Numerous models and algorithms have been developed for the past few decades. Before the uprise of deep learning methods, there were two classes of image restoration approaches that were widely adopted in the field: transformation based approach and PDE approach. The transformation based approach includes wavelet and wavelet frame based methods [Elad et al. 2005; Starck et al. 2005; Daubechies et al. 2007; Cai et al. 2009], dictionary learning based methods [Aharon et al. 2006], similarity based methods [Buades et al. 2005; Dabov et al. 2007], low-rank models [Ji et al. 2010; Gu et al. 2014], etc. The PDE approach includes variational models [Mumford & Shah 1989; Rudin et al. 1992; Bredies et al. 2010], nonlinear diffusions [Perona & Malik 1990; Catté et al. 1992; Weickert 1998], nonlinear hyperbolic equations [Osher & Rudin 1990], etc. More recently, deep connections

1 Equal contribution.

Supplementary materials and full code can be found at the project page.
between wavelet frame based methods and PDE approach were established \cite{Cai2012,Cai2016,Dong2017}.

One of the greatest challenge for image restoration is to properly handle image degradations of different levels. In the existing transformation based or PDE based methods, there is always at least one tuning parameter (e.g. the regularization parameter for variational models and terminal time for nonlinear diffusions) that needs to be manually selected. The choice of the parameter heavily relies on the degradation level.

Recent years, deep learning models for image restoration tasks have significantly advanced the state-of-the-art of the field. \cite{Jain2009} proposed a convolutional neural network (CNN) for image denoising which has better expressive power than the MRF models by \cite{Lan2006}. Inspired by nonlinear diffusions, \cite{Chen2017} designed a deep neural network for image denoising and \cite{Zhang2017a} improves the capacity by introducing a deeper neural network with residual connections. \cite{Chen2017} use the CNN to simulate a wide variety of image processing operators, achieving high efficiencies with little accuracy drop. However, these models cannot gracefully handle images with varied degradation levels. Although one may train different models for images with different levels, this may limit the application of these models in practice due to lack of flexibility.

Taking blind image denoising for example. \cite{Zhang2017a} designed a 20-layer neural network for the task, called DnCNN-B, which had a huge number of parameters. To reduce number of parameters, \cite{Lefkimmiatis2017} proposed the UNLNet5, by unrolling a projection gradient algorithm for a constrained optimization model. However, \cite{Lefkimmiatis2017} also observed a drop in PSNR comparing to DnCNN. Therefore, the design of a light-weighted and yet effective model for blind image denoising remains a challenge. Moreover, deep learning based models trained on simulated gaussian noise images usually fail to handle real world noise, as will be illustrated in later sections.

Another example is JPEG image deblocking. JPEG is the most commonly used lossy image compression method. However, this method tend to introduce undesired artifacts as the compression rate increases. JPEG image deblocking aims to eliminate the artifacts and improve the image quality. Recently, deep learning based methods were proposed for JPEG deblocking \cite{Dong2015,Zhang2017a,Zhang2018}. However, most of their models are trained and evaluated on a given quality factor. Thus it would be hard for these methods to apply to Internet images, where the quality factors are usually unknown.

In this paper, we propose a single image restoration model that can robustly restore images with varied degradation levels even when the degradation level is well outside of that of the training set. Our proposed model for image restoration is inspired by the recent development on the relation between deep learning and optimal control. The relation between supervised deep learning methods and optimal control has been discovered and exploited by \cite{Weinan2017,Lu2018,Chang2017,Fang2017}. The key idea is to consider the residual block $x_{n+1} = x_n + f(x_n)$ as an approximation to the continuous dynamics $\dot{X} = f(X)$. In particular, \cite{Fang2017} demonstrated that the training process of a class of deep models (e.g. ResNet by \cite{He2016}, PolyNet by \cite{Zhang2017b}, etc.) can be understood as solving the following control problem:

\[
\min_w \left( L(X(T), y) + \int_0^T R(w(t), t)dt \right)
\]

s.t. $\dot{X} = f(X(t), w(t)), t \in (0, \tau)$

$X(0) = x_0$. \hfill (1)

Here $x_0$ is the input, $y$ is the regression target or label, $\dot{X} = f(X, w)$ is the deep neural network with parameter $w(t)$, $R$ is the regularization term and $L$ can be any loss function to measure the difference between the reconstructed images and the ground truths.

In the context of image restoration, the control dynamic $\dot{X} = f(X(t), \omega(t)), t \in (0, \tau)$ can be, for example, a diffusion process learned using a deep neural network. The terminal time $\tau$ of the diffusion corresponds to the depth of the neural network. Previous works simply fixed the depth of the network, i.e. the terminal time, as a fixed hyper-parameter. However \cite{Mrzeka2003} showed that

Published as a conference paper at ICLR 2019
the optimal terminal time of diffusion differs from image to image. Furthermore, when an image is corrupted by higher noise levels, the optimal terminal time for a typical noise removal diffusion should be greater than when a less noisy image is being processed. This is the main reason why current deep models are not robust enough to handle images with varied noise levels. In this paper, we no longer treat the terminal time as a hyper-parameter. Instead, we design a new architecture (see Fig. [3]) that contains both a deep diffusion-like network and another network that determines the optimal terminal time for each input image. We propose a novel moving endpoint control model to train the aforementioned architecture. We call the proposed architecture the dynamically unfolding recurrent restorer (DURR).

We first cast the model in the continuum setting. Let $x_0$ be an observed degraded image and $y$ be its corresponding damage-free counterpart. We want to learn a time-independent dynamic system $\dot{X}(t) = f(X(t), w)$ with parameters $w$ so that $X(0) = x$ and $X(\tau) \approx y$ for some $\tau > 0$. See Fig. [2] for an illustration of our idea. The reason that we do not require $X(\tau) = y$ is to avoid over-fitting. For varied degradation levels and different images, the optimal terminal time $\tau$ may vary. Therefore, we need to include the variable $\tau$ in the learning process as well. The learning of the dynamic system and the terminal time can be gracefully casted as the following moving endpoint control problem:

$$\min_{w, \tau(x)} L(X(\tau), y) + \int_{0}^{\tau(x)} R(w(t), t) dt$$

s.t. $\dot{X}(t) = f(X(t), w(t)), t \in (0, \tau(x))$

$$X(0) = x.$$ (2)

Different from the previous control problem, in our model the terminal time $\tau$ is also a parameter to be optimized and it depends on the data $x$. The dynamic system $\dot{X} = f(X(t), w)$ is modeled by a recurrent neural network (RNN) with a residual connection, which can be understood as a residual network with shared weights (Liao & Poggio, 2016). We shall refer to this RNN as the restoration unit. In order to learn the terminal time of the dynamics, we adopt a policy network to adaptively determine an optimal stopping time. Our learning framework is demonstrated in Fig. [3]. We note that the above moving endpoint control problem can be regarded as the penalized version of the well-known fixed endpoint control problem in optimal control (Evans, 2005), where instead of penalizing the difference between $X(\tau)$ and $y$, the constraint $X(\tau) = y$ is strictly enforced.

In short, we summarize our contribution as following:

- We are the first to use convolutional RNN for image restoration with unknown degradation levels, where the unfolding time of the RNN is determined dynamically at run-time by a policy unit (could be either handcrafted or RL-based).
- The proposed model achieves state-of-the-art performances with significantly less parameters and better running efficiencies than some of the state-of-the-art models.
- We reveal the relationship between the generalization power and unfolding time of the RNN by extensive experiments. The proposed model, DURR, has strong generalization to images with varied degradation levels and even to the degradation level that is unseen by the model during training (Fig. [1]).
- The DURR is able to well handle real image denoising without further modification. Qualitative results have shown that our processed images have better visual quality, especially sharper details compared to others.

2 Method

The proposed architecture, i.e. DURR, contains an RNN (called the restoration unit) imitating a nonlinear diffusion for image restoration, and a deep policy network (policy unit) to determine the terminal time of the RNN. In this section, we discuss the training of the two components based on our moving endpoint control formulation. As will be elaborated, we first train the restoration unit to determine $\omega$, and then train the policy unit to estimate $\tau(x)$.
2.1 Training the Restoration Unit

If the terminal time $\tau$ for every input $x_i$ is given (i.e. given a certain policy), the restoration unit can be optimized accordingly. We would like to show in this section that the policy used during training greatly influences the performance and the generalization ability of the restoration unit. More specifically, a restoration unit can be better trained by a good policy.

The simplest policy is to fix the loop time $\tau$ as a constant for every input. We name such policy as “naive policy”. A more reasonable policy is to manually assign an unfolding time for each degradation level during training. We shall call this policy the “refined policy”. Since we have not trained the policy unit yet, to evaluate the performance of the trained restoration units, we manually pick the output image with the highest PSNR (i.e. the peak PSNR).

We take denoising as an example here. The peak PSNRs of the restoration unit trained with different policies are listed in Table. Fig. illustrates the average loop times when the peak PSNRs appear. The training is done on both single noise level ($\sigma = 40$) and multiple noise levels ($\sigma = 35, 45$). For
the refined policy, the noise levels and the associated loop times are (35, 6), (45, 9). For the naive policy, we always fix the loop times to 8.

Table 1: Average peak PSNR on BSD68 with different training strategies.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Noise Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training Noise Policy</td>
<td>25</td>
</tr>
<tr>
<td>35, 45 Naive</td>
<td>27.74</td>
</tr>
<tr>
<td>35, 45 Refined</td>
<td>29.14</td>
</tr>
</tbody>
</table>

As we can see, the refined policy brings the best performance on all the noise levels including 40. The restoration unit trained for specific noise level (i.e. $\sigma = 40$) is only comparable to the one with refined policy on noise level 40. The restoration unit trained on multiple noise levels with naive policy has the worst performance.

These results indicate that the restoration unit has the potential to generalize on unseen degradation levels when trained with good policies. According to Fig. 4, the generalization reflects on the loop times of the restoration unit. It can be observed that the model with steeper slopes have stronger ability to generalize as well as better performances.

According to these results, the restoration unit we used in DURR is trained using the refined policy. More specifically, for image denoising, the noise level and the associated loop times are set to (25, 4), (35, 6), (45, 9), and (55, 12). For JPEG image deblocking, the quality factor (QF) and the associated loop times are set to (20, 6) and (30, 4).

2.2 Training the Policy Unit

We discuss two approaches that can be used as policy unit:

**Handcraft policy:** Previous work (Mrázek & Navara, 2003) has proposed a handcraft policy that selects a terminal time which optimizes the correlation of the signal and noise in the filtered image. This criterion can be used directly as our policy unit, but the independency of signal and noise may
not hold for some restoration tasks such as real image denoising, which has higher noise level in the low-light regions, and JPEG image deblocking, in which artifacts are highly related to the original image. Another potential stopping criterion of the diffusion is no-reference image quality assessment (Mittal et al. [2012]), which can provide quality assessment to a processed image without the ground truth image. However, to the best of our knowledge, the performances of these assessments are still far from satisfactory. Because of the limitations of the handcraft policies, we will not include them in our experiments.

**Reinforcement learning based policy:** We start with a discretization of the moving endpoint problem (1) on the dataset \( \{ (x_i, y_i) \} \), where \( \{ x_i \} \) are degraded observations of the damage-free images \( \{ y_i \} \). The discrete moving endpoint control problem is given as follows:

\[
\min_{w, \{ N_i \}_{i=1}^d} r(w) + \sum_{i=1}^d L(X^i_{N_i}, y_i)
\]

\[
\text{s.t. } X^i_n = X^i_{n-1} + \Delta tf(X^i_{n-1}, w), n = 1, 2, \ldots, N_i, (i = 1, 2, \ldots, d) \quad (3)
\]

Here, \( X^i_n = X^i_{n-1} + \Delta tf(X^i_{n-1}, w) \) is the forward Euler approximation of the dynamics \( \dot{X} = f(X(t), w) \). The terminal time \( \{ N_i \} \) is determined by a policy network \( P(x, \theta) \), where \( x \) is the output of the restoration unit at each iteration and \( \theta \) the set of weights. In our experiment, we simply set \( r = 0 \), i.e. doesn’t introduce any regularization which might bring further benefit but is beyond this paper’s scope of discussion. In other words, the role of the policy network is to stop the iteration of the restoration unit when an ideal image restoration result is achieved. The reward function of the policy unit can be naturally defined by

\[
r(\{ X^i_n \}) = \begin{cases} 
\lambda (L(x_{n-1}, y_i) - L(x_n, y_i)) & \text{If choose to continue} \\
0 & \text{Otherwise} 
\end{cases} \quad (4)
\]

In order to solve the problem (2.2), we need to optimize two networks simultaneously, i.e. the restoration unit and the policy unit. The first is an restoration unit which approximates the controlled dynamics and the other is the policy unit to give the optimized terminating conditions. The objective function we use to optimize the policy network can be written as

\[
J = E_{X \sim \pi_\theta} \sum_{n}^{N_i} r(\{ X^i_n, w \}), \quad (5)
\]

where \( \pi_\theta \) denotes the distribution of the trajectories \( X = \{ X^i_n, n = 1, \ldots, N_i, i = 1, \ldots, d \} \) under the policy network \( P(\cdot, \theta) \). Thus, reinforcement learning techniques can be used here to learn a neural network to work as a policy unit. We utilize Deep Q-learning (Mnih et al. [2015]) as our learning strategy and denote this approach simply as DURR. However, different learning strategies can be used (e.g. the Policy Gradient).

### 3 Experiments

#### 3.1 Experiment Settings

In all denoising experiments, we follow the same settings as in Chen & Pock (2017); Zhang et al. (2017a); Lefkimmiatis (2017). All models are evaluated using the mean PSNR as the quantitative metric on the BSD68 (Martin et al., 2001). The training set and test set of BSD500 (400 images) are used for training. Six gaussian noise levels are evaluated, namely \( \sigma = 25, 35, 45, 55, 65 \) and 75. Additive noise are applied on the fly during training and testing. Both the training and evaluation process are done on gray-scale images.

The restoration unit is a simple U-Net (Ronneberger et al. [2015]) style fully convolutional neural network. For the training process of the restoration unit, the noise levels of 25, 35, 45 and 55 are
used. Images are cut into 64 × 64 patches, and the batch-size is set to 24. The Adam optimizer with the learning rate 1e-3 is adopted and the learning rate is scaled down by a factor of 10 on training plateaux.

The policy unit is composed of two ResUnit and an LSTM cell. For the policy unit training, we utilize the reward function in Eq. For training the policy unit, an RMSprop optimizer with learning rate 1e-4 is adopted. We’ve also tested other network structures, these networks and the detailed network structures of our model are demonstrated in the supplementary materials.

In all JPEG deblocking experiments, we follow the settings as in Zhang et al. (2017a; 2018). All models are evaluated using the mean PSNR as the quantitative metric on the LIVE1 dataset (Sheikh, 2005). Both the training and evaluation processes are done on the Y channel (the luminance channel) of the YCbCr color space. The PIL module of python is applied to generate JPEG-compressed images. The module produces numerically identical images as the commonly used MATLAB JPEG encoder after setting the quantization tables manually. The images with quality factors 20 and 30 are used during training. De-blocking performances are evaluated on four quality factors, namely QF = 10, 20, 30, and 40. All other parameter settings are the same as in the denoising experiments.

3.2 IMAGE DENOISING

We select DnCNN-B (Zhang et al., 2017a) and UNLNet (Lefkimmiatis, 2017) for comparisons since these models are designed for blind image denoising. Moreover, we also compare our model with non-learning-based algorithms BM3D (Dabov et al., 2007) and WNNM (Gu et al., 2014). The noise levels are assumed known for BM3D and WNNM due to their requirements. Comparison results are shown in Table 2.

Despite the fact that the parameters of our model (1.8 × 10^5 for the restoration unit and 1.0 × 10^5 for the policy unit) is less than the DnCNN (approximately 7.0 × 10^5), one can see that DURR outperforms DnCNN on most of the noise-levels. More interestingly, DURR does not degrade too much when the the noise level goes beyond the level we used during training. The noise level \( \sigma = 65, 75 \) is not included in the training set of both DnCNN and DURR. DnCNN reports notable drops of PSNR when evaluated on the images with such noise levels, while DURR only reports small drops of PSNR (see the last row of Table 2 and Fig. 6). Note that the reason we do not provide the results of UNLNet in Table 2 is because the authors of Lefkimmiatis (2017) has not released their codes yet, and they only reported the noise levels from 15 to 55 in their paper. We also want to emphasize that they trained two networks, one for the low noise level (\( 5 \leq \sigma \leq 29 \)) and one for higher noise level (\( 30 \leq \sigma \leq 55 \)). The reason is that due to the use of the constraint \( ||y - x||_2 \leq \epsilon \) by Lefkimmiatis (2017), we should not expect the model generalizes well to the noise levels surpasses the noise level of the training set.

For qualitative comparisons, some restored images of different models on the BSD68 dataset are presented in Fig. 5 and Fig. 6. As can be seen, more details are preserved in DURR than other models. It is worth noting that the noise level of the input image in Fig. 6 is 65, which is unseen by both DnCNN and DURR during training. Nonetheless, DURR achieves a significant gain of nearly 1 dB than DnCNN. Moreover, the texture on the cameo is very well restored by DURR. These results clearly indicate the strong generalization ability of our model.

More interestingly, due to the generalization ability in denoising, DURR is able to handle the problem of real image denoising without additional training. For testing, we test the images obtained from Lebrun et al. (2015). We present the representative results in Fig. 7 and more results are listed in the supplementary materials.

We also train our model for blind color image denoising, please refer to the supplementary materials for more details.

3.3 JPEG IMAGE DEBLOCKING

For deep learning based models, we select DnCNN-3 (Zhang et al., 2017a) for comparisons since it is the only known deep model for multiple QFs deblocking. As the AR-CNN (Dong et al., 2015) is a commonly used baseline, we re-train the AR-CNN on a training set with mixed QFs and denote
Table 2: Average PSNR (dB) results for gray image denoising on the BSD68 dataset. Values with * means the corresponding noise level is not present in the training data of the model. The best results are indicated in red.

<table>
<thead>
<tr>
<th>Noise Level</th>
<th>BM3D</th>
<th>WNNM</th>
<th>DnCNN-B</th>
<th>UNLNet</th>
<th>DURR</th>
</tr>
</thead>
<tbody>
<tr>
<td>σ = 25</td>
<td>28.55</td>
<td>28.73</td>
<td>29.16</td>
<td>28.96</td>
<td>29.16</td>
</tr>
<tr>
<td>σ = 35</td>
<td>27.07</td>
<td>27.28</td>
<td>27.66</td>
<td>27.50</td>
<td>27.72</td>
</tr>
<tr>
<td>σ = 55</td>
<td>25.26</td>
<td>25.49</td>
<td>25.80</td>
<td>25.64</td>
<td>25.91</td>
</tr>
<tr>
<td>σ = 65</td>
<td>24.69</td>
<td>24.51</td>
<td>23.40*</td>
<td>-</td>
<td>25.26*</td>
</tr>
<tr>
<td>σ = 75</td>
<td>22.63</td>
<td>22.71</td>
<td>18.73*</td>
<td>-</td>
<td>24.71*</td>
</tr>
</tbody>
</table>

Figure 5: Denoising results of an image from BSD68 with noise level 35.

Figure 6: Denoising results of an image from BSD68 with noise level 65 (unseen by both DnCNN and DURR in their training sets).

This model is called AR-CNN-B. Original AR-CNN as well as a non-learning-based method SA-DCT (Foi et al., 2007) are also tested. The quality factors are assumed known for these models.
Quantitative results are shown in Table 3. Though the number of parameters of DURR is significantly less than the DnCNN-3, the proposed DURR outperforms DnCNN-3 in most cases. Specifically, considerable gains can be observed for our model on seen QFs, and the performances are comparable on unseen QFs. A representative result on the LIVE1 dataset is presented in Fig. 8. Our model generates the most clean and accurate details. More experiment details are given in the supplementary materials.

Table 3: The average PSNR(dB) on the LIVE1 dataset. Values with "*" means the corresponding QF is not present in the training data of the model. The best results are indicated in red and the second best results are indicated in blue.

<table>
<thead>
<tr>
<th>QF</th>
<th>JPEG</th>
<th>SA-DCT</th>
<th>AR-CNN</th>
<th>AR-CNN-B</th>
<th>DnCNN-3</th>
<th>DURR</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>27.77</td>
<td>28.65</td>
<td>28.98</td>
<td>28.53</td>
<td>29.40</td>
<td>29.23*</td>
</tr>
<tr>
<td>20</td>
<td>30.07</td>
<td>30.81</td>
<td>31.29</td>
<td>30.88</td>
<td>31.59</td>
<td>31.68</td>
</tr>
<tr>
<td>30</td>
<td>31.41</td>
<td>32.08</td>
<td>32.69</td>
<td>32.31</td>
<td>32.98</td>
<td>33.05</td>
</tr>
<tr>
<td>40</td>
<td>32.45</td>
<td>32.99</td>
<td>33.63</td>
<td>33.39</td>
<td>33.96</td>
<td>34.01*</td>
</tr>
</tbody>
</table>

Figure 7: Denoising results on a real image from Lebrun et al. (2015).

3.4 OTHER APPLICATIONS

Our model can be easily extended to other applications such as deraining, dehazing and deblurring. In all these applications, there are images corrupted at different levels. Rainfall intensity, haze density and different blur kernels will all affect the image quality.

4 CONCLUSIONS

In this paper, we proposed a novel image restoration model based on the moving endpoint control in order to handle varied noise levels using a single model. The problem was solved by jointly optimizing two units: restoration unit and policy unit. The restoration unit used an RNN to realize the dynamics in the control problem. A policy unit was proposed for the policy unit to determine the loop times of the restoration unit for optimal results. Our model achieved the state-of-the-art results in blind image denoising and JPEG deblocking. Moreover, thanks to the flexibility of the given policy, DURR has shown strong abilities of generalization in our experiments.

ACKNOWLEDGMENTS

Bin Dong is supported in part by Beijing Natural Science Foundation (Z180001). Yiping Lu is supported by the Elite Undergraduate Training Program of the School of Mathematical Sciences at Peking University.

REFERENCES

Figure 8: JPEG deblocking results of an image from the LIVE1 dataset, compressed using QF 10.


