Journal of Artificial Intelligence and Data Mining (JAIDM), Vol. 9, No. 1, 2021, 129-139.

Shahrood University of Technology

Journal of Artificial Intelligence and Data Mining (JAIDM)
Journal homepage: http://jad.shahroodut.ac.ir

Research paper

An Image Restoration Architecture using Abstract Features and Generative Models

A. Fakhari and K. Kiani*

Electrical and Computer Engineering Faculty, Semnan University, Semnan, Iran.

Article Info

Article History:
Received 15 May 2020
Revised 20 July 2020
Accepted 23 August 2020

DOI:10.22044/jadm.2020.9691.2101

Keywords:
Deep Learning, Generative Model, Image Restoration, Perceptual Features, GAN, RBM.

**Corresponding author: kourosh.kiani@semnan.ac.ir (K. Kiani).*

Abstract

Image restoration and its different variations are important topics in the low-level image processing. One of the main challenges in image restoration is the dependency of the current methods to the corruption characteristics. In this paper, we propose an image restoration architecture that enables us to address different types of corruption regardless of the type, amount, and location. The main intuition behind our approach is to restore original images from the abstracted perceptual features. Using an encoder-decoder architecture, image restoration can be defined as an image transformation task. Abstraction of the perceptual features is done in the encoder part of the model, and determines the sampling point within the Probability Density Function (PDF) of the original images. PDF of the original images is learned in the decoder section using a Generative Adversarial Network (GAN) that receives the sampling point from the encoder part. The pre-trained network extracts the perceptual features, and the Restricted Boltzmann Machine (RBM) makes the abstraction over them in the encoder section. By developing a new algorithm for training RBM, the features of the corrupted images are refined. In the decoder, the generator network restores the original images from the abstracted perceptual features, while the discriminator determines how good the restoration result is. The proposed approach is compared with both traditional approaches like BM3D and with modern deep models like IRCNN and NCSR. We also consider three different categories of corruption including denoising, inpainting, and deblurring. The experimental results confirm the performance of the model.

1. Introduction

Image restoration is one of the main topics in low-level digital image processing. Restoring the corrupted images as a way to obtain higher quality ones has its own importance. At the same time, it can be considered as the first step for other high-level image processing tasks such as object detection. Image degradation is generally an irreversible problem. The task of image restoration is to make a good estimation of the original image according to the corrupted image using the learned models [1].

The researchers have categorized image restoration into different separated tasks including denoising [2-4], inpainting [5-8], image super resolution [9-11], colorizing monochrome images [12-14], deblurring [15], and so on; however, all of them struggle with a common challenge that can be defined as receiving a corrupted image, eliminating the corruption, and restoring the original image.

Generally speaking, image restoration can be considered as an image transformation task [16] in which the model receives a corrupted image as an input and transforms it into a desired output. Mathematically, this transformation can be modeled using a corruption function and an added noise [2]. Every corrupted image can be modeled using Equation 1:

$$C(x, y) = H(R(x, y)) + m \quad (1)$$

in which C is the corrupted image, H is the corruption function, m is an arbitrary noise that can possibly exist depending on the problem, and R is the original image. By this definition, all the image restoration challenges are considered as image transformation tasks to remove H and m in Equation 1 and generate an image I that has the least difference with R. From the statistical viewpoint and by considering the generative models for this task, the probability distribution of the restored image (I) and the original image (R) should have a minimum distance.

For two random variables, the probability distribution that defines their simultaneous behavior is their joint probability. Assuming the original and corrupted images together, we can define their joint probability as follows [17]:

$$P(R, C) = P(R) \times P(C \mid R) \quad (2)$$

in which $P(R)$ is the probability of the original image, $P(C)$ is the probability of the corrupted image, and $P(C|R)$ is the conditional probability of the corrupted image conditioned to the original image. Concretely, the proposed model should transform the input image C into I so that the difference between $P(R)$ and $P(I)$ is minimized. As $P(R)$ is intractable, we may train a model in order to generate a synthesized image (I) to minimize the following mean square objective:

$$W^* = \underset{w}{\operatorname{argmin}} \sum \left\| R_i, f\ (w, C_i) \right\|_2 \quad (3)$$

in which W is the parameter set of the model and f is the function for mapping C to I using W. Although different approaches can be considered to define f, according to the previous descriptions, a probability-based approach that considers the joint probability between the original image and its corrupted version is selected.

The generative models consider the joint probability of the input and output, $P(X,Y)$. There is no need for an explicit supervision to train the generative models, and they map input to output using an unsupervised or semi-supervised manner. Therefore, the generative models completely fit in the image transformation task [17].

Many different architectures and models have been proposed for addressing the image restoration problems [18-23]. Among the neural network-based approaches, the deep models have recently achieved a great performance for this task [2, 3, 14, 16, 17, 24-27]; however, a common challenge still exists. Most of them are used for only one purpose and handle one image restoration problem at a time. Furthermore, they are dependent on the corruption type, amount, and location. In another work [17], the authors have called them the fixated models and have indicated that even the deep models suffer from this problem. In the fixated models, H in Equation 1 may be a deterministic function. However, in most practical cases, H is unknown for the model and a reversing transformation is impossible.

One of the main approaches for image transformation in the deep models is the encoding-decoding architecture, in which the input is firstly mapped into some latent variables. Afterwards, another model generates the desired output from the encoder's one. For the image processing related tasks, it is common to use the deep convolutional models [3, 10, 14, 24]; however, other deep models have also been observed [2, 11].

In this paper, we propose an encoding-decoding architecture that can be applied to different image restoration problems. The encoding part is made of a deep convolutional network and a RBM. The deep convolutional model maps the input into the perceptual features, and the RBM network adds an abstraction over them. This abstraction determines the sampling point in data PDF for the decoder part of the model. A Deep Convolution Generative Adversarial Network (DCGAN) [28] is in charge for generating an image from the sampling point. Finally, the proposed approach is compared with the traditional ones like BM3D and also with modern deep models like IRCNN and NCSR. We also consider three different categories of corruptions including denoising, inpainting, and deblurring.

The pre-trained networks are used because as a feature extractor, their extracted features are robust against changes. Concretely, both an image and its corrupted version have almost similar outputs in the intermediate convolution layers. Furthermore, according to the previously training applied to their filters, there is no need to retrain them. We remove the fully connected layers and call the output of the last convolution layer of the model as the perceptual features. The details will be discussed in Section 3.

The RBM model receives the perceptual features and then changes their dimension and distribution, creating some abstract latent variables. Such vectors determine the sampling points in data PDF for the decoder network. After extracting the corruption independent features, a GAN model is used to decode them. The original image is restored in the decoding phase. In fact, the GAN model learns the probability distribution of the original images and RBM determines the sampling point.

The rest of this paper is organized as what follows. In Section 2, the previous works in image restoration are reviewed. Section 3 describes the architecture of the model and its components. In Section 4, the experimental results are explained. Finally, in Section 5, a brief conclusion is presented.

2. Related Works

Although deep models are the mostly used approaches in these days, there are many other methods proposed in various architectures to address different types of challenges in image restoration. These methods can be categorized into two different categories: pixel-level approaches and encoding-decoding architectures.

The pixel-level traditional approaches such as total variation [29-32] and BM3D [33] that use collaborative filtering for image denoising usually work weak on the edges. The dictionary-based approaches and sparse coding [19, 20, 22, 23] have been proposed by considering a linear assumption for restoring images. Colorizing the B/W images using SVM have been proposed in [34] by assigning a color to each pixel and according to their color space.

Using Multi-Layer perceptron (MLP) was one of the first tries for using neural network in image restoration [23]. In [18], a patch-based MLP approach that achieved comparable results with BM3D was proposed. In [35], an auto-encoder (AE) has been used for image denoising, and in [2], AE has been combined with sparse coding for a better parameter optimization. Convolution neural networks (CNN) as the main approach for working on images has been employed for image denoising and has achieved a great performance both in the blind and non-blind situations [36]. In [37], CNN is used for decoding the JPEG blocks. By combining CNN and AE, the authors in [3] have proposed the Convolutional Auto-Encoder (CAE) for medical image denoising. Using the RBM models for image modelling and reconstruction has been discussed in [38, 39].

The encoding-decoding architecture has been considered for addressing different types of corruption simultaneously. CNN with the encoding-decoding architecture has been applied [24] to image denoising and inpainting. They have used the convolutional operators as the encoder and the deconvolutional operators as the decoder. In [13], the encoding-decoding architecture has been applied for colorizing the monochrome images. This architecture has also been employed for image super-resolution in [16].

The classic deep models struggle with only one corruption type and are also dependent on the corruption location and amount. In [17], this problem has been discussed and an independent model has been proposed. In [40], a new model based on CNN has been proposed, and the knowledge of these networks about generating a naturalized image, without training, for image restoration has been discussed.

In [41, 42], the deep generative models have been employed and analyzed for use in the image restoration task.

3. System Description

As mentioned earlier, the encoder-decoder architecture is the mostly used approach for the image transformation tasks in which a model, the encoder, is responsible for creating the latent variables, and another model, the decoder, is in charged for generating the images from those encoded latent variables.

Figure 1 demonstrates the proposed architecture and its components. In the encoder section, we consider two separated modules. At the first layer, the perceptual features are extracted from the image, and then a RBM model makes an abstraction over these features. The output of RBM will be used as the sampling point in the decoder model. Thus, RBM should generate a corruption independent output. From the probability viewpoint, by a good abstraction, an image and its corrupted version will reach a unique sampling point within the original image probability distribution that makes the restoration possible. The original image can be restored using a generative model that have previously learned the probability distribution of the original data.

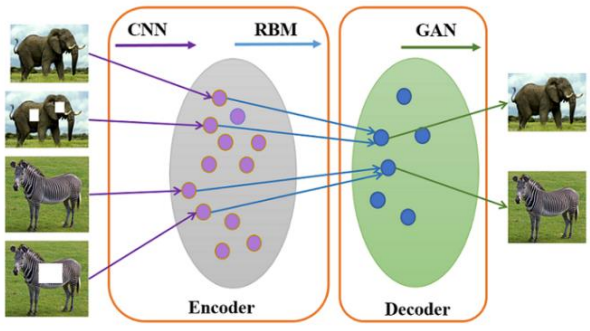

Figure 1. A schematic view of the proposed model: purple arrows: CNN; blue Arrows: RBM, and green arrows: GAN.

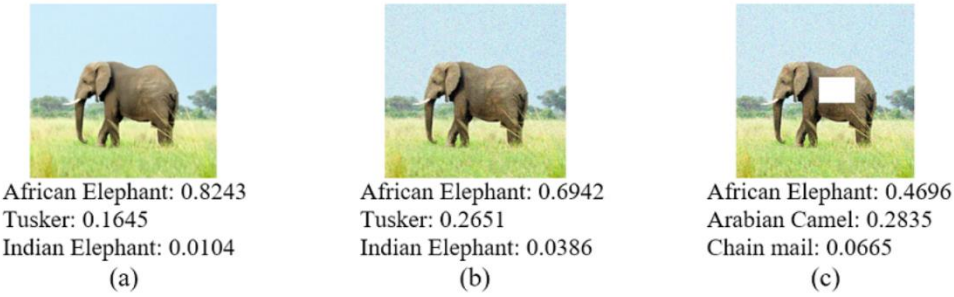

Figure 2. VGG16 model's robustness against different corruptions. (a) Original image, (b) noisy image, and (c) noisy image with one removed part.

3.1. Encoder Model

The key idea for a good restoration process is that the latent variables should include a robust information about the original images. Concretely, the power of the encoder for extracting the encoded data has a major impact on the quality of restoration process. In this section, the components of the encoder are described.

3.1.1. Perceptual Features

The pre-trained convolutional networks have been applied to different image processing tasks. They may be applied directly to the image classification tasks, and in some cases, they have been used in the transfer learning or fine-tuning applications [43, 44]. We used a pre-trained network as the first part of the encoder section due to these reasons. (1) They are already trained, and there is no need to train them again. This feature makes the model faster. (2) The features they extract are discriminative and robust against some changes. In order to extract the perceptual features, we removed the fully connected layers from our pretrained network.

Figure 2 shows the output of the VGG16 model [45] for the image classification task. Due to the richness of the extracted features, the model makes a correct decision for a noisy image even when some parts are removed. Although the VGG model can classify its inputs correctly, the intermediate features are affected by the considered corruptions. We have to refine these perceptual features. The RBM model is considered for this task and to make them robust against corruptions.

Figure 3 shows the images and their corresponding feature maps of the last convolution layer of the pre-trained network. We aligned these feature maps and showed them in one single image. As it can be seen, these intermediate features are visually similar to each other due to the similar activated neurons but they are corruption-affected. In order to refine the corruption effects, we used RBM with a new training algorithm to make the feature vectors robust against different corruptions.

3.1.2. Feature Abstraction

In the next step, we used the RBM model to handle the impact of corruption on the perceptual features. RBM allows our latent variables to be robust to the corruptions. The goal, therefore, is to find a compromise between the suppressing corruption as much as possible and not losing too much image details [38]. For doing so, the key idea is that using a new training algorithm, RBM tries to reconstruct the original feature vector even when it receives a corrupted input.

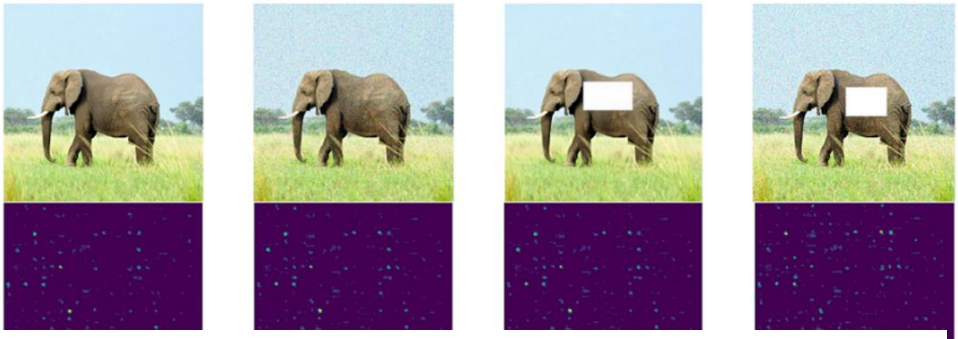

Figure 3. Perceptual features extracted from the last convolution layer of the VGG16 model for different corrupted versions of an image: (a) original, (b) noisy version, (c) blurry with removed part, and (d) noisy version with removed part.

Once the input data clamped to the visible units, the equilibrium distribution of the hidden units can be computed. There is a good reason to use RBMs for image modelling. Unlike the directed models, an RBM conditional distribution over the hidden nodes is very easy to compute [39].

RBM consists of two components, visible and hidden layers, and by forwarding data from the visible layer to the hidden layer and sending them backward, it tries to learn the important features within the data for the input reconstruction. If a binary vector, for example, an image, is taken to RBM, the model will be able to find how an energy state is compatible with that image. Although RBM receives a corrupted input, it should reconstruct the original images.

Let $v \in \{0,1\}^M$ be the visible layer and $h \in \{0,1\}^N$ be the hidden layer, where M and N are the natural numbers. Given the biases $b \in \mathbb{R}^M$, $c \in \mathbb{R}^N$, and weights $W \in \mathbb{R}^{M \times N}$, the probability, partition function, and energy function of an RBM are given by:

$$p(x|\theta) = \frac{1}{z}\exp\left(-E(x|\theta)\right) \tag{4}$$

$$p(x,h|\theta) = \frac{1}{z}\exp\left(-E(x,h|\theta)\right) \tag{5}$$

$$Z = \sum_{v}\sum_{h}\exp\left\{-E(v,h)\right\} \tag{6}$$

$$\begin{aligned} E_{RBM}(v,h,\theta) = &-\sum_{i}^{N_v} b_i v_i - \sum_{j}^{N_h} c_j h_j \\ &- \sum_{i,j}^{N_h,N_v} W_{i,j} v_i h_j \end{aligned} \tag{7}$$

in which E is the energy function, Z is the normalizing constant partition function, W is the weight matrix of the model, i refers to the ith visible, and j indicates the jth hidden node.

The goal of RBM training is to find the best probable reproduction. When the data in the visible and hidden units reach the same distribution, the learning will stop. In our model, RBM transforms the extracted perceptual features into a new space, while the distribution variation of its input and output (*KL Divergence*) is minimized. The goal for RBM training is to maximize the objective function, demonstrated in Equation 8:

$$\begin{aligned} \theta^* &= argmax_\theta \, \mathrm{E}\left[\log P(v)\right] \\ &= argmax_\theta E_{h\sim p(h|v;\theta)}[\log P(v,h;\theta)] \end{aligned} \tag{8}$$

in which θ, $P(v)$ and $P(h)$ are the parameter set of RBM, distribution of visible layer, and distribution of the hidden layer, respectively.

As computation of the joint distribution of the visible and hidden nodes in RBMs is not feasible, the Gibbs sampling is used for the approximation. The Contrastive Divergence (CD) algorithm [44] and its extensions use the Gibbs sampling for this approximation. Obtaining an unbiased estimation of the log-likelihood gradient using the MCMC methods typically requires many sampling steps. This leads to CD become a standard way to train RBMs. This algorithm has three main steps including positive phase, negative phase, and updating weights. The positive phase refers to transforming the input data into the hidden layer, while the negative phase refers to reconstructing the input data from the values obtained in the hidden nodes.

CD^N means that the positive and negative phases should be repeated for N times and then the weights are updated. In order to reach to the equilibrium, the algorithm starts with small and random values for weights and use CD^1, i.e. use one full step for updating weights. Once the weights grow, the Markov chain mixes more slowly, and the greater numbers are used for N. After repeating the Markov chain, i.e. positive and negative phases for k times, the weights and biases are updated using Equation 9.

$$\begin{aligned} \Delta W_{ji} &= \eta(\langle v_i h_j\rangle_o - \langle v_i h_j\rangle_k) \\ \Delta b_j &= \eta(\langle h_j\rangle_o - \langle h_j\rangle_k) \\ \Delta c_j &= \eta(\langle v_i\rangle_o - \langle v_i\rangle_k) \end{aligned} \tag{9}$$

in which W is the weight matrix, o refers to the training data index, and b and c are the biases of the visible and hidden layers, respectively [46].

We propose a new extension to the *CD* algorithm for training the RBM model. In the proposed algorithm, the positive phase is changed while the negative phase remains unchanged. In the new approach, the reconstruction error is calculated based on the output of the positive phase for the non-corrupted data and the output of the negative phase for the corrupted samples. In other words, both the original and corrupted inputs are clamped to the visible layer of the model, and the positive phases are calculated for both. The output of the clean data is stored temporarily and the negative phase continues for the corrupted input regularly.

In our approach, instead of using the clamped corrupted input in the positive phase, we consider the output of the positive phase for the original non-corrupted input in order to calculate the reconstruction error. Thus, the positive phase is done using both the corrupted input and its corresponding original version. Negative phase is done using only the corrupted input. Figure 4

demonstrates an overall diagram of the proposed method. By this change and according to figure 4 terminology, Equation 9 can be re-written as follows:

$$\begin{aligned}\Delta W_{ji} &= \eta \times \left[\langle I.h_0 I\rangle_o - \langle V_1 C.h_1 C\rangle_k\right] \\ \Delta b_j &= \eta \times \left[\langle h_0 I\rangle_o - \langle h_1 C\rangle_k\right] \\ \Delta c_j &= \eta \times \left[\langle I\rangle_o - \langle V_1 C\rangle_k\right]\end{aligned} \tag{10}$$

According to figure 4, *I* and *C* refer to the original and corrupted images, respectively. $\langle I.h_0 I\rangle_o$ refers to the positive phase for the clean data and $\langle V_1 C.h_1 C\rangle_k$ refers to the negative phase for the corrupted input obtained after repeating the reconstruction for *K* times. These abstracted features constitute the latent variables, and as a sampling point, they are fed into the decoder. RBM allows our latent variables to be robust to the corruptions.

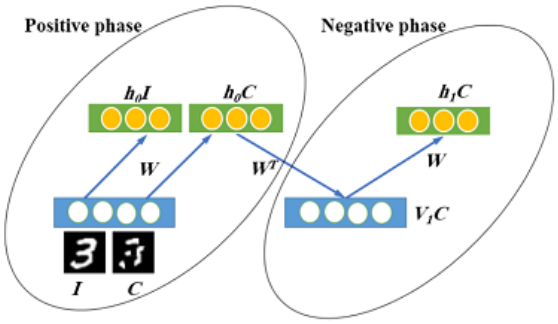

Figure 4. An overview diagram for the new algorithm.

3.2. Decoder Model

The decoder network is a model that receives a vector and generates the corresponding image according to the already learned distribution. The output of the encoder determines the sampling point within the probability distribution of the non-corrupted images. For doing so, we have considered a GAN model [28] for mapping the abstracted features to the original images. GAN uses two networks for generating images, i.e. the generator and discriminator networks. The generator network is responsible for generating images, and the discriminator determines the quality of the generator's results. The generator and discriminator networks play a MinMax game, in which the discriminator tries to maximize the generator's error by distinguishing between the real and fake (generated) images. On the other hand, the generator tries to minimize this function so that the difference between the real and generated images become indistinguishable. A well-trained GAN is able to fool its discriminator so that it cannot distinguish between the real and fake images. The mathematics and the cost function of this model are demonstrated in Equations 11 to 14. In these equations, *D* is the discriminator, *G* is the generator, and *z* indicates the latent variables that come from the encoder. *P(data(x))* refers to the distribution of the original images and *P(z)* indicates the distribution of the latent variables.

$$G^* = \min_G \max_D V(D,G) \tag{11}$$

$$\begin{aligned}V(D,G) = {} & E_{x\sim P_{data}(X)}\left[\log[D_x\right] \\ & + E_{z\sim P_z(z)}\left[\log(1-D(G(z))\right]\end{aligned} \tag{12}$$

$$\begin{aligned}J^D = {} & -\frac{1}{2}E_{x\sim P_{data}}\left[\log(D_x)\right] \\ & -\frac{1}{2}E_{z\sim P_z}\left[\log(1-D(G(z)))\right]\end{aligned} \tag{13}$$

$$J^G = -J^D \tag{14}$$

Despite the regular GANs that use random variables for *P(z)*, in our model, the latent variables come from the encoder. In our approach, the corrupted images are considered as fake in the GAN training process. Here are the required steps for training:

1- The discriminator is trained for *n* epochs using the prepared fake samples (both corrupted images and images generated by a not fully trained generator) and real ones. For doing so, beside the random generated samples, the corrupted images are also considered as fake samples to give the discriminator a better insight about the non-corrupted image probability distribution.
2- Train the generator network using the trained discriminator.
3- Repeat steps 1 and 2 for a few epochs. At this phase, consider a smaller number for the number of epochs in step 1.
4- Check the fake images. If the generated images are good-enough, stop the training, else go to step 1.

Step 1 must not be done once because the discriminator could easily reach 100% accuracy with a few epochs, and an optimal discriminator with zero loss leads to the vanishing gradient issue, and hence, the gradient value becomes too small to make the generator trainable. As data is transformed from a low dimensionality to a space with a higher dimensionality, there are some spaces in there, $P_r(x) \to \infty$ and $P(z) \to 0$, that make the generator cost goes to infinity [45]. This

is the reason for using different corrupted images for training. Using this approach, we covered the latent space to prevent the *mode dropping* issue while training GAN. In order to make the discriminator more stable, we added noise to its input to make the input space continuous [47, 48]. The discriminator network is only applied to the training process. In the training phase, its output determines the quality of the restored image, and after training, it will be removed from the restoration pipeline.

4. Experimental Results

In this section, we evaluated the proposed model with different types, locations, and amounts of corruptions. The implementation details, corruption models, and practical results will be described in the following sub-sections.

4.1. Implementation Details

We implemented our proposed pipeline in Keras V.2. We also considered the VGG16 model as the pre-trained network in our model. We considered the following configuration for our model. The input images should have the 224×224×3 dimension. As we used the VGG16 model, the pre-trained network has thirteen convolution and five pooling layers. Each convolution layer includes already trained 3×3 filters. As we removed the fully connected layers from the model, the RBM input size was equal to the size of the Flatten layer of the VGG model. RBM receives a 25088 dimension input and maps it into a 1000 dimension vector. The generator network has five deconvolution layers, and the discriminator has a reverse architecture with generator to determine the quality of the generator's job. At the last layer of the discriminator, a Dense (1) layer is used. As the proposed method is not end-to-end, all the modules can be trained separately.

For training the model, we created some corrupted versions for each image. For doing so, we created different noisy and blurry images. We also considered different images with removed parts with different mask size placed in different locations randomly. None of these training images was used for testing the model.

4.2. Corruption Models

In order to test the proposed model, different types of corruptions were considered. In the first step, the proposed method with some noisy images was tested to evaluate it for image denoising. Consequently, the inpainting problem for evaluation was considered. Image deblurring was considered as the last restoration problem for testing the model. For a better demonstration, we trained the model with images from the ImageNet and BSD68 datasets. Furthermore, we considered some standard image processing test beds like Lena and CameraMan. In order to train the model, we added the corrupted version of each image in our dataset. No additional data such as the accompanying labels in this process. The results have been compared with MLP [18], CBM3D [49], NCSR [50], and IRCNN [25] regarding the problem type.

4.3. Restoration Results

In this section, the experimental results of the model are proposed. Each section will describe the results of different types of corruption used for evaluating the model.

4.3.1. Image Denoising

In order to test the model, we have considered Gaussian noise with several noise levels. Table 1 demonstrates the results obtained for the proposed method compared to the CBM3D and IRCNN methods for the BSD68 (color) dataset. While noise level increases from 5 to 35, the proposed model can overcome other approaches, whereas it performs weak on situations with a higher level of noises. This problem has been discussed in [51], demonstrating the performance of the VGG model against the distorted images.

Table 1. PSNR values for image denoising for the BSD68 (color) dataset.

Noise level	5	15	25	35	50
CBM3D	40.24	33.52	30.71	28.89	27.38
IRCNN	40.36	33.86	31.16	29.50	**27.86**
Proposed	**42.12**	**39.70**	**33.81**	**29.64**	25.46

Figure 5 demonstrates the output of our model in the image denoising problem for the Gaussian noise while the noise level is 10. As it can be seen, the original images have been restored successfully from their noisy versions.

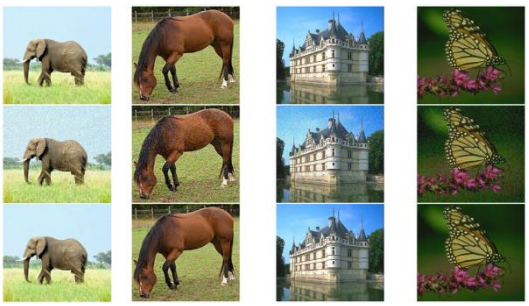

Figure 5. Image denoising results of the proposed model; top row: original images; second row: image with Gaussian noise; and third row: restored images.

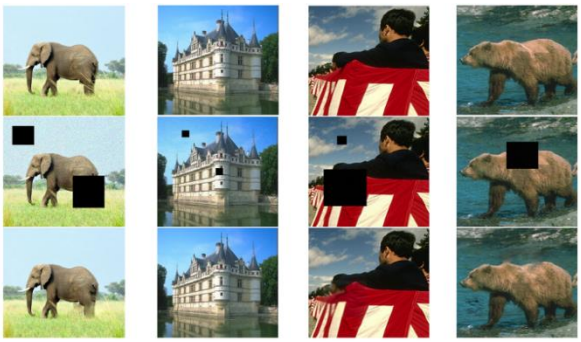

Figure 6. Image inpainting results of the proposed model; top row: original images; second row: image with Gaussian noise and different removed parts; and third row: restored images.

4.3.2. Image Inpainting

In order to test the model for image inpainting, we have considered different removed parts both in size and location. Furthermore, we have combined this corruption with a Gaussian noise to make it more challenging for the model. Patch size for the removed parts has been considered different while the noise level was 10. Figure 6 demonstrates the output of our model for image inpainting challenge. Different types and amounts of removed parts have been considered for testing the model. It should be noted that the considered corruptions are different in amount and location.

Different types and amounts of blurring filters have been considered for testing the model. We have used the Gaussian kernel with standard deviation 1.6 and two blur kernel from [52]. Furthermore, we have added the Gaussian noise with different σ values. Figure 7 demonstrates the comparable output of the model for the leaves image. The proposed method, figure 7 (e), achieved a higher PSNR value for the blurry and noisy image.

The PSNR values achieved for the CameraMan, House, Lena, Monar, Leaves, and Parrots images have been demonstrated in table 2. As it can be seen, except for the Lena image, the proposed approach achieved better PSNR values compared to the others. Perhaps changing the pre-trained network with VGG-Face may achieve a better performance.

5. Conclusion and Future Works

In this paper, we have proposed an encoder-decoder architecture for different image restoration problems. Using the proposed architecture, we addressed this problem. One of the main challenges in the image restoration task is that most of the current models and architectures are corruption-dependent, which means that most of the existing models have bias to the corruption type, amount, and location. We proposed an encoder-decoder architecture to map the corrupted image to its original version. The main idea behind our approach is that the original image and its corrupted version should return

Table 2. PSNR values for image debluring for different images.

Method	σ	*C.man*	*House*	*Lena*	*Monar*	*Leaves*	*Parrots*
Gaussian blur with standard deviation 1.6							
NCSR	2	27.99	33.38	30.99	28.32	27.50	30.42
MLP		27.84	33.43	31.10	28.87	28.91	31.24
IRCNN		28.12	**33.80**	31.17	30.00	29.78	32.07
Proposed		**30.11**	33.64	**31.22**	**32.49**	**29.84**	**34.80**
Kernel 1 (19×19) [57]							
EPLL	2.55	25.33	28.19	27.37	22.67	21.67	26.08
IRCNN		28.11	32.03	**29.51**	29.20	29.07	31.63
Proposed		**31.42**	**32.45**	28.91	**3170**	**29.14**	**33.16**
Kernel 2 (17×17) [57]							
EPLL	7.65	24.85	28.08	27.03	21.60	21.09	25.77
IRCNN		27.70	31.94	**29.77**	28.73	28.63	31.35
Proposed		**28.39**	**32.54**	28.63	**29.05**	**28.93**	**32.08**

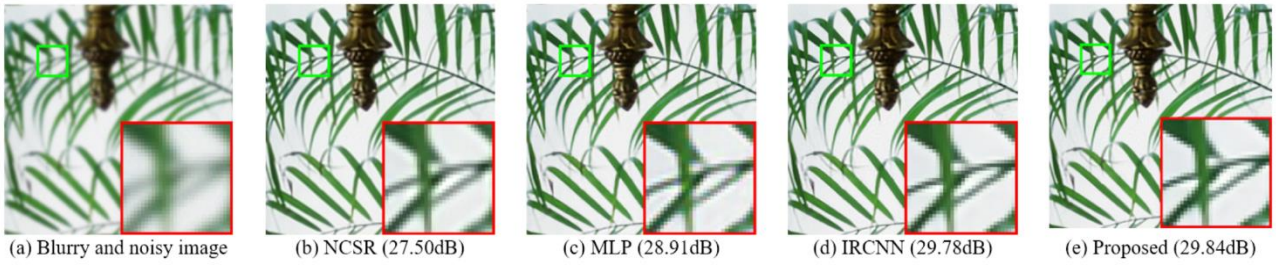

Figure 7. Performance of image debluring challenge for the leaves image with Gaussian kernel (standard deviation is 1.6 and σ is 2).

almost the same abstract features in the encoding phase that is the sampling point for the decoder network. The generative adversarial network was used in the decoding phase, where it was able to learn the probability distribution of the non-corrupted images. in the proposed architecture, a good sampling point leads to good restoration results.

As mentioned earlier, the restoration quality completely depends on the power of the encoder network. The richer the features, the better the result. The output of encoder should be enriched and include enough information about the original images. In this work, we used only one pre-trained model for feature extraction. In order to make the model more powerful, combining and merging different perceptual features came from different pre-trained networks may enrich the output of the encoder and lead to better results. By doing so, we can make sure that all the discriminative features are extracted for restoring an image. Furthermore, they can cover each other and make the result more robust against noisy inputs.

References

[1] Liu, D et al. (2020). Connecting image denoising and high-level vision tasks via deep learning. IEEE Transactions on Image Processing, vol. 29, pp. 3695-3706.

[2] Xie, J., Xu, L. & Chen, E. (2012). Image denoising and inpainting with deep neural networks. Advances in neural information processing systems, pp. 341-349.

[3] Tian, Ch., Xu, Y. Zuo, W. (2020). Image denoising using deep CNN with batch renormalization. Neural Networks, vol. 121, pp.461-473.

[4] Zhang, K., Zuo, W., Chen, Y., Meng, D., & Zhang, L. (2017). Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. IEEE Transactions on Image Processing, vol. 26, no. 7, pp. 3142-3155.

[5] Cai, W. & Wei., Zh. (2020). PiiGAN: Generative adversarial networks for pluralistic image inpainting. IEEE Access, vol. 8, pp. 48451-48463.

[6] Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T. & Efros, A.A. (2016). Context encoders: Feature learning by inpainting. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, pp. 2536-2544, 2016.

[7] Yang, C., Lu, X., Lin, Z., Shechtman, E., Wang, O. & Li, H. (2017). High-resolution image inpainting using multi-scale neural patch synthesis. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, pp. 6721-6729, 2017.

[8] Cai, N., Su, Z., Lin, Z., Hang, H., Yang, Z., & Ling, B.W.K. (2017). Blind inpainting using the fully convolutional neural network. The Visual Computer, vol. 33, no. 2, pp. 249-261.

[9] Wang, Zh., Chen, J. & Hoi. S. (2020). Deep learning for image super-resolution: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 99, pp. 1-22.

[10] Kim, J., Lee, K. & Lee, M. (2016). Accurate image super-resolution using very deep convolutional networks. Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, NV, pp. 1646-1654, 2016.

[11] Zeng, K., Yu, J., Wang, R., Li, C., & Tao, D. (2017). Coupled deep auto-encoder for single image super-resolution. IEEE transactions on cybernetics, vol. 47, no. 1, pp. 27-37.

[12] Wan, Sh., Xia, Y., Qi, L., Yang, Y.H. & Atiquzzaman., M. (2020). Automated colorization of a grayscale image with seed points propagation. IEEE Transactions on Multimedia, pp. 1-10.

[13] Zhang, R., Isola, P., & Efros, A.A. (2016). Colorful image colorization. European Conference on Computer Vision, UK, pp. 649-666, 2016.

[14] Isola, P. Zhu, J.-Y., Zhou, T., & Efros, A.A. (2017). Image-to-image translation with conditional adversarial networks. arXiv preprint.

[15] Adam, T. & Paramesran., R. (2020). Hybrid non-convex second-order total variation with applications to non-blind image deblurring. Signal, Image and Video Processing, vol. 14, no. 1, pp. 115-123.

[16] Johnson, J., Alahi, A., & Fei-Fei, L. (2016). Perceptual losses for real-time style transfer and super-resolution. European Conference on Computer Vision, UK, pp. 694-711, 2016.

[17] Gao, R. & Grauman, K. (2017). On-demand learning for deep image restoration. Proc. IEEE Conf. Computer Vision and Pattern Recognition, Honolulu, HI, pp. 1086-1095.

[18] Burger, H.C., Schuler, C.J. & Harmeling, S. (2012). Image denoising: Can plain neural networks compete with bm3d? Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IR, USA, pp. 2392-2399, 2012.

[19] Yang, J., Wright, J., Huang, T.S., & Ma, Y. (2010). Image super-resolution via sparse representation. IEEE transactions on image processing, vol. 19, no. 11, pp. 2861-2873.

[20] Dong, W., Zhang, L., Shi, G. & Wu, X. (2011). Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization. IEEE Transactions on Image Processing, vol. 20, no. 7, pp. 1838-1857.

[21] Chang, Y., Yan, L., Zhao, X.L., Fang, H., Zhang, Zh., & Zhong, Sh. (2020). Weighted low-rank tensor recovery for hyperspectral image restoration. IEEE Transactions on Cybernetics, pp. 1-15.

[22] Elad, M. & Aharon, M. (2006). Image denoising via sparse and redundant representations over learned dictionaries. IEEE Transactions on Image processing, vol. 15, no. 12, pp. 3736-3745.

[23] Sivakumar, K. & Desai, U.B. (1993). Image restoration using a multilayer perceptron with a multilevel sigmoidal function. IEEE transactions on signal processing, vol. 41, no. 5, pp. 2018-2022.

[24] Mao, X.-J., Shen, C., & Yang, Y.-B. (2016). Image restoration using convolutional auto-encoders with symmetric skip connections. arXiv preprint arXiv:1606.08921.

[25] Zhang, K., Zuo, W., Gu, S., & Zhang, L. (2017). Learning deep cnn denoiser prior for image restoration. IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, 2017.

[26] Zhang, Y., Tian, Y., Kong, Y., Zhong, B., & Fu., Y. (2020). Residual dense network for image restoration. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[27] Li, J., Skinner, K.A., Eustice, R.M., & Johnson-Roberson, M. (2018). Watergan: Unsupervised generative network to enable real-time color correction of monocular underwater images. IEEE Robotics and Automation Letters, vol. 3, no. 1, pp. 387-394.

[28] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, pp. 2672-2680.

[29] Rudin, L.I., Osher, S., & Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. Physica D: nonlinear phenomena, vol. 60, no. 1-4, pp. 259-268.

[30] Chan, T., Esedoglu, S., Park, F., & Yip, A. (2005). Recent developments in total variation image restoration. Mathematical Models of Computer Vision, vol. 17, no. 2.

[31] Oliveira, J.P., Bioucas-Dias, J.M., & Figueiredo, M.A. (2009). Adaptive total variation image deblurring: a majorization-minimization approach. Signal processing, vol. 89, no. 9, pp. 1683-1693.

[32] Sahragard, E., Farsi, H., & Mohammadzadeh, S. (2018). Image restoration by variable splitting based on total variant regularizer. Journal of AI and Data Mining, vol. 6, no .1, pp. 13-33.

[33] Dabov, K., Foi, A., Katkovnik, V. & Egiazarian, K. (2007). Image denoising by sparse 3-d transform domain collaborative filtering. IEEE Transactions on image processing, vol. 16, no. 8, pp. 2080-2095.

[34] Charpiat, G., Bezrukov, I., Altun, Y., Hofmann, M. & SCH, B. (2009). Machine learning methods for automatic image colorization. Computational Photography: Methods and Applications, pp. 395-418.

[35] Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P.A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of machine learning research, vol. 11, pp. 3371-3408.

[36] Jain V. & Seung, S. (2009). Natural image denoising with convolutional networks. Advances in Neural Information Processing Systems, pp. 769-776.

[37] Dong, C., Deng, Y., Change Loy, C., & Tang, X. (2015). Compression artifacts reduction by a deep convolutional network. Proceedings of the IEEE International Conference on Computer Vision, Las Condes, Chile, pp. 576-584, 2015.

[38] Pires, R.G., Santos, D.F., Pereira, L.A., De Souza, G.B., Levada, L.A., & Papa, J.P. (2017). A robust restricted boltzmann machine for binary image denoising. Graphics, Patterns and Images (SIBGRAPI), 2017 30th SIBGRAPI Conference on, pp. 390-396.

[39] Tang, Y., Salakhutdinov, R. & Hinton, G. (2012). Robust Boltzmann Machines for recognition and denoising. Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, RI, 2012, pp. 2264-2271, 2012.

[40] Ulyanov, D., Vedaldi, A. & Lempitsky, V. (2017). Deep image prior. arXiv preprint arXiv:1711.10925.

[41] Basioti, K., & Moustakides, G. V. (2020). Image Restoration from Parametric Transformations using Generative Models. arXiv preprint arXiv:2005.14036.

[42] Pan, X., Zhan, X., Dai, B., Lin, D., Loy, C. C., & Luo, P. (2020). Exploiting Deep Generative Prior for Versatile Image Restoration and Manipulation. arXiv preprint arXiv:2003.13659.

[43] Esteva, A., Kuprel, B., Novoa, R.A., Ko, J., Swetter, S.M., Blau, H.M., & Thrun., S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. Nature, vol. 542, no. 7639, p. 115.

[44] Nogueira, K., Penatti, O.A. & dos Santos, J.A. (2017). Towards better exploiting convolutional neural networks for remote sensing scene classification. Pattern Recognition, vol. 61, pp. 539-556.

[45] Krizhevsky, A., Sutskever, I. & Hinton,G. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, pp. 1097-1105.

[46] Hinton, G. (2012). A practical guide to training restricted boltzmann machines. Neural networks: Tricks of the trade, pp. 599-619.

[47] Arjovsky M. & Bottou, L. (2017). Towards principled methods for training generative adversarial networks. arXiv preprint arXiv:1701.04862.

[48] Theis, L., Oord, A., & Bethge, M. (2015). A note on the evaluation of generative models. arXiv preprint arXiv:1511.01844.

[49] Dabov, K., Foi, A., Katkovnik, V., & Egiazarian, K. (2007). Color image denoising via sparse 3d collaborative filtering with grouping constraint in luminance-chrominance space. Image Processing, 2007. ICIP 2007. IEEE International Conference on, Texas, USA, pp. 313, 2007.

[50] Dong, W., Zhang, L., Shi, G., & Li, X. (2013). Nonlocally centralized sparse representation for image restoration. IEEE Transactions on Image Processing, vol. 22, no. 4, pp. 1620-1630.

[51] Dodge, S. & Karam, L. (2017). A study and comparison of human and deep learning recognition performance under visual distortions. Computer Communication and Networks (ICCCN), 2017 26th International Conference on, Canada, pp. 1-7, 2017.

[52] Levin, A., Weiss, Y., Durand, F., & Freeman, W.T. (2009). Understanding and evaluating blind deconvolution algorithms. 2009 IEEE Conference on Computer Vision and Pattern Recognition, FL, USA, pp. 1964-1971, 2009.

معماری جدید برای بازگردانی تصاویر با استفاده از ویژگی‌های انتزاعی و مدل‌های زایشی

علی فخاری و کوروش کیانی*

گروه هوش مصنوعی و رباتیکز ، دانشکده برق و کامپیوتر، دانشگاه سمنان، سمنان، ایران.

ارسال ۲۰۲۰/۰۵/۱۵؛ بازنگری ۲۰۲۰/۰۷/۲۰؛ پذیرش ۲۰۲۰/۰۸/۲۳

چکیده:

بازگردانی تصویر و انواع مختلف آن، یکی از موضوعات مهم در پردازش تصویر سطح پیکسل هستند. یکی از چالش‌های مهم در موضوع بازگردانی تصویر، وابستگی متدها به نوع تخریب است. در این مقاله معماری جدیدی برای بازگردانی تصویر ارائه شده است به نحوی که بتوان به انواع تخریب صرف نظر از نوع، میزان و محل آن در تصویر رسیدگی کرد. ایده اصلی در این مقاله، بازگردانی تصاویر از ویژگی‌های مهفومی-انتزاعی است. با استفاده از یک معماری کدگذار/کدگشا، بازگردانی تصویر به یک موضوع انتقال تصویر تبدیل می‌شود. ویژگی‌های انتزاعی در بخش کدگذار مدل ساخته می‌شوند و محل نمونه‌گیری را در تابع توزیع احتمال تصاویر اصلی تعیین می‌کنند. تابع توزیع احتمال در بخش دوم و توسط یک شبکه زایشی خصمانه (GAN) آموخته می‌شود که محل نمونه‌گیری را از بخش کدگذار دریافت می‌کنند. در نهایت، نمونه‌برداری صحیح از تابع توزیع احتمال تصاویر اصلی، به بازگردانی تصویر از تصویر تخریب‌شده منجر می‌گردد. در بخش کدگذار، یک شبکه آموزش‌دیده ویژگی‌های مفهومی را استخراج کرده و ماشین بولتزمان محدود (RBM) با یک الگوریتم جدید آن‌ها را پالایش می‌کند. در بخش کدگشا نیز، بخش اول شبکه GAN تصاوی اصلی را با توجه به خروجی بخش کدگذار تولید و بخش دوم آن، کیفیت یا خطای نتیجه را تعیین می‌کند. رویکرد پیشنهادی هم با روش‌های سنتی نظیر BM3D و هم با مدل‌های عمیق مشابه نظیر IRCNN و NCSR مقایسه شده است. همچنین سه نوع تخریب مشتمل بر حذف نویز، تکمیل تصویر و رفع تاری در نظر گرفته شده اند. نتایج پیاده سازی نشان از برتری روش پیشنهادی دارد.

کلمات کلیدی: مدل‌های زایشی، بازگردانی تصویر، ویژگی‌های انتزاعی، ماشین بولتزمان محدود، شبکه زایشی خصمانه.