## Abstract

Denoising, and deblurring of images are the two most crucial restoration operations in the document image processing assignment. The quality of denoising and deblurring, which are preprocessing steps in the processing pipeline, has a big impact on the results of subsequent tasks like character detection and recognition. As the use of digital devices has grown, the majority of data is now created and shared in compressed form. On many devices, JPEG compression is the default setting. Directly recovering a picture in the compressed domain is an interesting research problem in terms of storage and computing costs. In this study, we present a general technique for doing Residual network inference and learning in the JPEG transform domain, which allows the network to accept compressed images as input. In this method, we constructed and trained the model on a benchmark dataset, which turned out to work rationally well. We produce a result that is mathematically equal to a spatial domain network. We show that omitting the time-consuming decompression phase allows for faster image processing with little to no loss in network accuracy. In both compressed and uncompressed domains, the experimental findings indicated state-of-the-art performance.

Index Terms — Image restoration; autoencoder and decoder; Discrete Cosine Transform News Paper; Magazines; JPEG algorithm; Compression;

## INTRODUCTION

Digital imaging techniques have been used in a wide range of application areas, including engineering, construction, and fields like medical and research. However, one of the most frequent issues faced by image scientists is the ”noise” that might appear in photos as a result of the low-quality typewriter that was used to create the original document. The issue arises because a variety of elements, including lighting, picture distortion, and even flaws in documents themselves, can affect an image’s contrast. Numerous handwritten or typewritten scientific articles, historical documents and artifacts, recipes, and novels are kept as papers. The paper/notes tend to gather noise/dirt over time due to fingerprints, weakened paper fibers, dirt, coffee/tea stains, abrasions, wrinkling, etc. Examples of typical degraded images are shown in Fig-1.

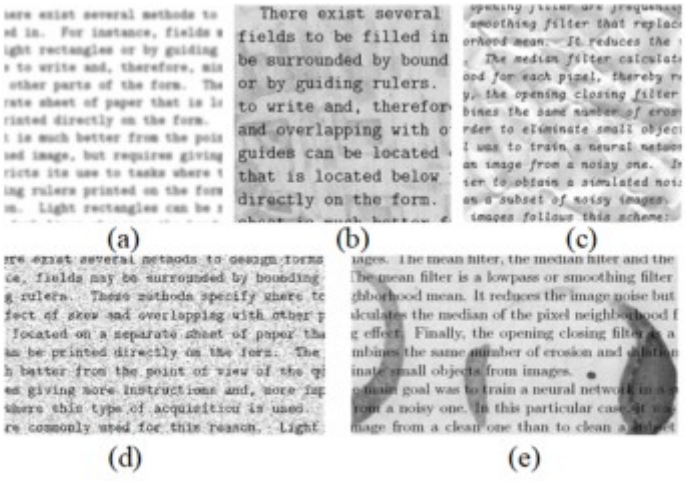



Fig. 1. Examples of Typical degraded images. (a)blurred. (b)poor paper. (c)wrinkles. (d)noising image (e)ink marks

Numerous surface cleaning techniques are employed for both cleaning and preservation, but they have limitations, the most important of which is the possibility that the original document may be changed. In order to remove distracting effects like

When reading these documents and reassembling them into digital format, some processing must be done. In recent decades, image restoration techniques have become increasingly popular. It tries to create a new image with less noise, less blur, and a closer resemblance to the original noise-free image.

The corrupted low-quality image can be expressed as:

$$x = F(y) + n,$$

where x represents the degraded image, y represents the original high-quality image, F represents the degradation function, and n represents noise, as in prior work.

Because the purpose of image restoration is to estimate an unknown original image from a (potentially noisy, blurry, or low-resolution) observation, it is also known as an inverse issue. That is, guessing the ‘x’ given the observation ‘y’ is the task of restoration.

Traditional restoration algorithms mostly focus on natural scene photos, and since the need for OCR grows rapidly, studies to tackle document restoration have been conducted in recent years.

The deblurring algorithm in [1] is based on document picture foreground segmentation and uses a desired intensity probability density function as the prior. [3] examines the differences between document and natural-scene picture qualities and incorporates these domain-specific properties into the optimization procedure.

The L0-regularized intensity and gradient prior is used in [4]. The approaches in [6], [7] are deep neural network-based. The two-tone before is used in [8].

The value of deep learning technologies has dramatically increased recently. A deep structure is advantageous to achieve good outcomes in computer vision problems because the deeper network has more parameters and it naturally has powerful feature representation learning capabilities, as demonstrated by many earlier studies. Additionally, deeper networks have wider network receptive fields, which allow the convolutional layer kernels to take in more contextual data. In order to rebuild the document image from a low quality observation, we constructed a deep convolutional auto encoder model in this paper. To train the network, we used the DCT coefficients of the compressed JPEG image. Convolutional neural networks have been successfully trained using the simple method of using an original RGB pixel image. However, we might be able to get better performance if we switch to a compressed input representation. We presented and looked at the idea of training the network using the frequency domain of the compressed JPEG image after applying various transformations to the DCT. The neural network receives dequantization coefficients
as input. We further developed and trained the model using a benchmark dataset, and it seems to work rationally and successfully.

The main idea of this paper is to improve the performance of deep convolutional auto encoder by using a compressed domain input, which reduces the memory requirements in training convolutional neural networks. The main work results prove that reconstructing an image up to 100% using a compressed domain has been achieved by using the proposed deep convolutional auto encoder model.

# RELATED LITERATURE

## A. DENOISING TECHNIQUES & IT'S LITERATURE REVIEW

Analytical methods for resolving inverse problems have long been researched. There is a tonne of literature on the subject of picture restoration issues; the most recent reviews are found in [2] and [3]. The process of restoring images generally involves a variety of techniques, including denoising [3], [4], [5], [6] deblurring [7], [8], [9], [10] and [11] , demosaicking [12], [13], and super-resolution [14]. Denoising and deblurring are the two topics of this study that are most pertinent

The dictionary learning techniques in [15] use sparse and redundant representations across taught dictionaries to remove zero-mean white and homogeneous Gaussian additive noise from the provided image. The Block-matching and 3D filtering (BM3D) proposed in [5] achieves excellent results for denoising images contaminated by additive white gaussian noise (AWGN). The foundation of BM3D is the locally sparse representation of a picture in the transform domain. It first organises related 2D picture fragments into 3D data arrays, applies collaborative filtering to these 3D groups, and then employs an inverse 3D transform to produce the denoised image [16].

It is important to note that in recent years, learning-based algorithms have become more prevalent in picture restoration approaches. Deep neural network-based techniques in particular have dominated this field over time. Similar to the original method for creating deep networks, Stacked Denoising Autoencoders (SDA) stacks layers of denoising autoencoders that have been locally trained to denoise damaged versions of their input images. The [17] get superior outcomes than BM3D by using a simple multilayer perceptron (MLP) to picture patches. The techniques in [18] suggest shrinkage fields, a random field-based architecture that unifies the picture model and optimization algorithm into a single unit and achieves the aims of both high computational efficiency and restoration quality. To capture the characteristics, the [19] presented a new deep convolutional network topology and trained two sub-modules under supervision for deconvolution and artefact removal. In [20], the clear high-resolution image is recovered from the hazy low-resolution input using the generative adversarial network (GAN). It creates a unique GAN with a generator but two discriminators in order to handle face and text images simultaneously. The [21] combines CNN and BM3D.

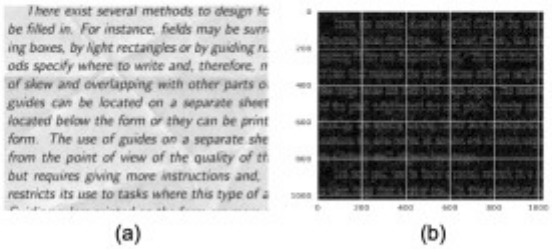
In this paper We have proposed a convolution neural network model for image restoration that is similar to the model proposed by [22], but here we make it more simple. After that Partial decoding on compressed images has been applied, and DCT coefficients obtained are fed as input to the network after some transformation [23] which removes noises in the compressed domain. The performance of this model has been evaluated on real images, and the obtained result is compared with that of other methods in section IV. Finally the conclusion is reported in section V.

## B. JPEG Encode

The JPEG compression technique is depicted in Fig.2 as a series of steps. The Discrete Cosine Transform (DCT), which transforms images from the pixel domain to the frequency domain, lies at the heart of the JPEG algorithm. It can obtain a high compression ratio by deleting these frequencies. Lossy compression is achieved by eliminating data from images. We can't get the image back to its original quality once it's been converted to JPEG. Most photographs are stored in the JPEG domain because of the high compression ratio and minimal quality change. JPEG is the preferred format for storing photographs captured by modern devices.

$$C_{pq} = \alpha_p \alpha_q \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I_{mn} \cos \frac{\pi(2m+1)p}{2M} \cos \frac{\pi(2n+1)q}{2N} \quad (4)$$

$$\text{Where } c_u, c_v = \begin{cases} \frac{1}{\sqrt{2}} \text{ for u,v} = 0 \leq u, v \leq N-1 \\ 1 \text{ otherwise} \end{cases}$$



(a) (b)

Fig. 2. (a) image in RGB colour space, (b) image obtained after applying Discrete Cosine Transform

## C. Conversion of RGB image to YCbCr

The JPEG algorithm begins with the YCbCr conversion. It converts RGB channel space to YCbCr channel space, where the Y components represent the image's brightness and the Cb and Cr components constitute the colour space. The colour pixels are lowered by 2 or 3 since the human eye is more sensitive to brightness than colour. We can convert RGB space to YCbCr space using the equations 1,2 and 3.

Y = 0.257R + 0.504G + 0.098B + 16 (1)

Cb = −0.148R − 0.291G + 0.439B + 128 (2)

Cr = 0.439R − 0.368G − 0.071B + 128 (3)

## D. Discrete Cosine Transform

The image is divided into 8 x 8 or 4 x 4 blocks once it has been converted from RGB to YCbCr space. The majority of standers follow 8 × 8 blocks, which are then transformed into the frequency domain using the DCT equation 4. The DCT blocks are then subjected to a quantization filter. Because high-frequency DCT coefficients are less effective for the human eye, quantization filters remove them. The lossy part of the JPEG process occurs when most of the DCT coefficients are set to zero after quantization.The fig-3 shows the image after applying discrete cosine transform to it

## E. Denoising Autoencoder

An autoencoder is a type of unsupervised artificial neural network that is trained to replicate input to output. When dealing with picture data, the autoencoder encodes the image into a lower-dimensional representation before decoding the representation back to the image. The encoder down samples the data into lower dimensions, while the decoder reconstructs the original data from the representation in lower dimensional space. Autoencoders can only compress data that is identical to the data on which they were trained. They are also lossy, meaning that the output will worsen in comparison to the original input.

The autoencoders are closely related to principal component analysis (PCA). If the activation function used within the autoencoder is linear within each layer, the latent variables present at the bottleneck (the smallest layer in the network, aka. code) directly correspond to the principal components from PCA. This relationship implies that an autoencoder can be implemented as a feedforward artificial neural network by over pruning links and adding enough regularization to avoid overfitting.

Denoising Autoencoders are autoencoders that have been modified to prevent the network from learning the identity function. If the autoencoder is excessively large, it can only learn the data, resulting in an output that equals the input and no useful representation learning or dimensionality reduction.

Denoising autoencoders go around this by intentionally distorting the input data, adding noise, or obscuring some of the values.

## III. PROPOSED MODEL

Before we get into the details of our proposed methodology, let's have a look at some design considerations.
The encoded JPEG images have two channels as chroma and luma channel.
Because the chroma channels' dimensions differ from those of the luma channels, i.e the luma channel has a greater dimension than the chroma channels, it is required to equalize their dimensions before the resultant of all three channels is concatenated and fed to the network. So, to match the dimension, we either perform downsampling on the luma channel or upsampling on the chroma channels, so we investigated both methods, namely downsampling the luma channel by a factor of two and upsampling the chroma channel by a factor of two, and then concatenating the three channels and feeding them as input to the proposed model. The proposed model architecture comprises many layers for processing images, which are divided into two categories: encoder and decoder.
The encoder reduces the size of the input data by compressing it into a lower-dimensional representation. The decoder reconstructs the representation in order to produce an output that is as near to the input as possible. The autoencoder learns the most important aspects of the input data in this way.

### A. Design of the Network

The encoder network is in charge of taking the input's features and mapping them to a latent space. Typically, the autoencoder reduces dimension by making the feature map in the bottleneck layer smaller. In our restoration work, we decrease the feature size to avoid losing the contextual information from the original image. Six residual blocks are placed after the first 3*3 convolution layer in the encoder. Each convolution layer is followed by a batch normalising layer and an activation of the Leaky ReLU. The main building component of the encoder, the residual block, was invented to address the vanishing deteriorated problem in image classification. Since then, a great deal of research has demonstrated that this kind of structure is successful in treating a range of low-level visual impairments, and a great deal of residual block variations have been created. We modified the leftover block's typical structure to make it suitable for document photo restoration.

## B. Encoder network

TABLE I
ARCHITECTURE OF THE ENCODING NETWORK

| Type | Kernel | Padding | Outputs | Activation |
|---|---|---|---|---|
| Conv | 3*3 | 1 | 48 | ReLU |
| Conv | 3*3 | 0 | 72 | Leaky ReLU |
| Conv | 3*3 | 1 | 144 | Leaky ReLU |
| Batch Normalization | | | | |
| MaxPooling | 2*2 | | | |
| Dropout | | | | |

## C. Decoder network

TABLE II
ARCHITECTURE OF THE DECODING NETWORK

| Type | Kernel | Padding | Outputs | Activation |
|---|---|---|---|---|
| Conv | 3*3 | 1 | 144 | ReLU |
| Conv | 3*3 | 0 | 72 | Leaky ReLU |
| Conv | 3*3 | 1 | 48 | Leaky ReLU |
| Batch Normalization | | | | |
| Upsampling | 2*2 | | | |
| conv | 3*3 | 0 | 1 | Leaky ReLU |

# IV. RESULTS & EXPERIMENTS

Experiments on our proposed autoencoder model in both RGB and compressed DCT input images are carried out in this section:

## A. Dataset

We employed a paper With 144 document images from 18 different fonts in its training set, the DataSet in [24]. Each font has an average of 8 pictures. Additionally, there are eight different types of sludge and two image sizes. The clean photos correlate to the training images. Furthermore, [24] has 72 document photographs, but no corresponding clean images are provided. Due to this, the test set can only be used to visually rather than statistically illustrate the model's performance. As a result, for the quantitative evaluation stage, we must divide the initial training set into a new training set and a test set. We trained our network using only the noisy photos as input and the corresponding clean images as labels in order to exclude genuine noise. The two quality metrics we used to assess our model are SNR, or signal-to-noise ratio, and PSNR, or peak signal-to-noise ratio (PSNR). A higher PSNR and SNR

often indicates a higher-quality reconstruction.

## B. Parameter setting

Although our denoising model is capable of processing input images of any size, we split the training image into small patches in order to test our model with various input image sizes, including 128, 256, 512, and 1024. The initial learning rate is 1e-2, decaying by 5 per 10 epochs. Our denoised model was trained on an Nvidia Tesla K40c GPU on the Kaggle framework.

### C. Data Augmentation

Since there are only 144 tagged photos in the data set [32], it is required to utilize various data amplification techniques. We expanded 4 times the number of photos and rescaled the original image to [0.8, 0.9, 1.1, 1.2] of size. Additionally, we had used a horizontal flip to increase their number by two. We refrain from using techniques (such as rotation) that would alter this data distribution because of the characteristics of the document image, such as the fact that the text is always horizontal

## D. Training procedure

We carried out our experiment in both RGB pixel images and the Compressed DCT domain. For different input sizes of the images we train our model and try to observe the performance. For DCT input, before feeding into the proposed model We transformed the picture RGB domain to YcbCr domain before passing it through 8x8 DCT and quantization it while training the DCT domain. After that, we'll store it on our hard disc. We directly entered the quantized value into the model when training it. For this, we utilised Adam as the optimizer, with mean absolute loss as the loss function and mean square error as the metric.

## E. Experiments

Here for the analysis of our proposed architecture, we have used NVIDIA Tesla T4 with a 12GB GPU setup provided by Kaggle. Here, we have followed a standard norm for training, validation, and testing like for training purposes we have used 70 % of the images, 15 % for the validation purpose, and 15% images for the Testing purpose

The proposed model is tested on a standard dataset given by "UCI machine learning repository" and the results are reported in respective tables. Table III shows the network behaviour in different input sizes on RGB images and Table IV shows the same with that of our proposed model working with JPEG compressed DCT coefficients.As input size is increasing our proposed model shows good result in terms of SNR value. Table V shows the training time between both RGB and DCT domains. It is noted that the compressed domain method used less time for training. Table VI shows the average use of GPU, memory allocation and memory accessing

time during training. It is noted that GPU usage, memory allocation is lower in the DCT domain as compared to RGB images. It is clear that our proposed model has some advantages over the RGB model in terms of training time, GPU usage and memory allocation. We also compared our PSNR results with a number of cutting-edge techniques, including RED30, BM3D, SSDA, and DCNP. The authors or other researchers have published the implementations of these methods online. For this experiment, We used the clean version image in the training set. The remaining 120 photos were used for training, and 24 were randomly chosen for testing. With zero mean and 10, 20, 30, 40, and 50 standard deviations, we test the additive Gaussian distribution.

TABLE III
THE AVERAGE PSNR(DB) RESULT OF $\sigma = 10, 20, 30, 40, 50$

| Methods | bm3d | ssda | red30 | dcnp | ours |
|---|---|---|---|---|---|
| $\sigma = 10$ | 34.28 | 34.23 | 34.41 | 34.91 | 35.14 |
| $\sigma = 20$ | 28.78 | 29.12 | 20.86 | 31.28 | 31.77 |
| $\sigma = 30$ | 25.40 | 26.61 | 27.87 | 28.04 | 28.04 |
| $\sigma = 40$ | 22.90 | 22.99 | 26.02 | 26.88 | 26.88 |
| $\sigma = 50$ | 20.20 | 20.68 | 25.44 | 26.25 | 26.25 |

The quantitative comparisons between our method and various earlier methods are shown in Fig. 5.

We have also calculated the SNR of every denoised image and the result is shown below. Along with that Visual comparisons of our model is given in Fig. 8. We can see that compared with Ground truth image, our result is more clear and with less noise.

TABLE IV
STUDYING THE BEHAVIOUR OF THE NETWORK WITH DIFFRENT INPUT SIZE IN RGB PIXEL DOMAIN

| Input size | Loss | MSE | SNR |
|---|---|---|---|
| 256 | 0.0011 | 0.0183 | 7.19 |
| 512 | 0.0688 | 0.038 | 6.09 |
| 1024 | 0.0924 | 0.0521 | 4.02 |

TABLE V
STUDYING THE BEHAVIOUR OF THE NETWORK WITH DIFFERENT INPUT SIZE IN DCT-COMP DOMAIN

| Input size | Loss | MSE | SNR |
|---|---|---|---|
| 256 | 0.0088 | 0.0646 | 8.19 |
| 512 | 0.0048 | 0.0438 | 6.09 |
| 1024 | 0.0021 | 0.0283 | 4.12 |

TABLE VI
COMPARISON OF TRAINING TIMEE BETWEEN RGB PIXEL AND DCT INPUT IMAGE

| INPUT SIZE | IN RGB(SEC) | IN DCT(SEC) |
|---|---|---|
| 256 | 2127.0004 | 2968 |
| 512 | 2314.16 | 594.57 |
| 1024 | 2516.5 | 1835.86 |

TABLE VII
STUDYING THE AVERAGE USAGE OF GPU AND MEMORY ALLOCATION DURING TRAINING

| HARDWARE USAGE | IN RGB | IN DCT |
|---|---|---|
| GPU power usage | 74% | 62% |
| Memory allocation | 95% | 82% |
| Memory accessing time | 26% | 20% |

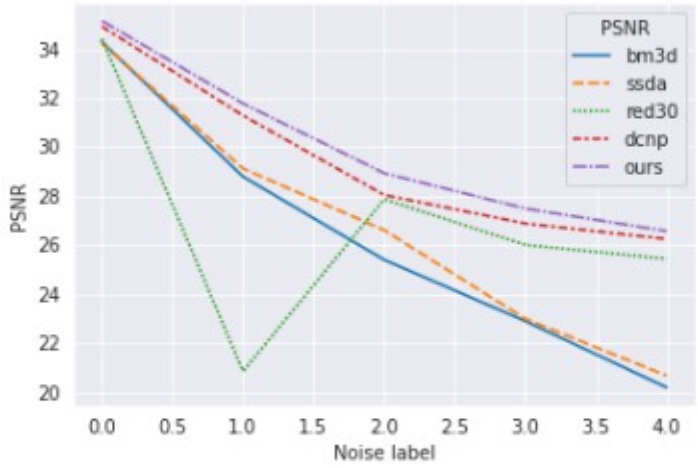



Fig. 5. Quantitative comparisons between our method and existing state of the art methods

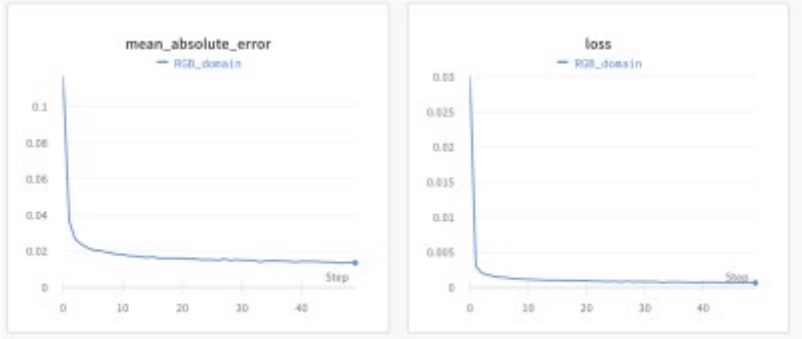



Fig. 6. shows MSE and loss during training in RGB domain

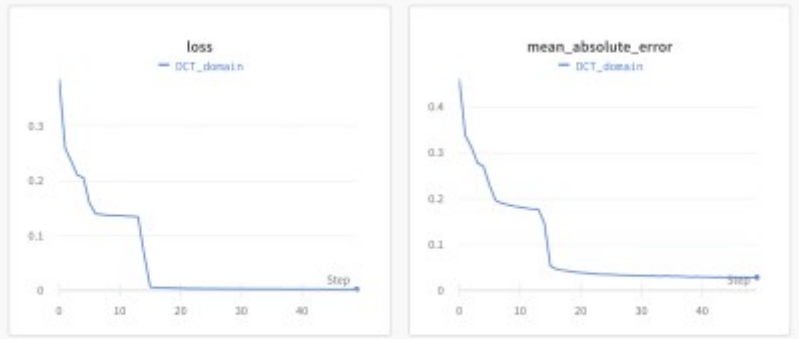



Fig. 7. Shows MSE and loss during training in DCT domain

## V. CONCLUSION AND FUTURE WORK

In this work, we looked at how to denoise dirty documents with inputs in the DCT domain. We described how the document picture JPEG compression process may appear in numerous forms as a first step toward a quick and inexpensive denoising approach, but there are still many hurdles to overcome. Because of its architecture, the suggested technique can handle (various sub-sampling ratios, encoding accuracy, etc.) which can be managed - but not assessed. Many methods exist in the spatial domain to enhance speed at the expense of accuracy; it would be fascinating to find new problems in the DCT domains.

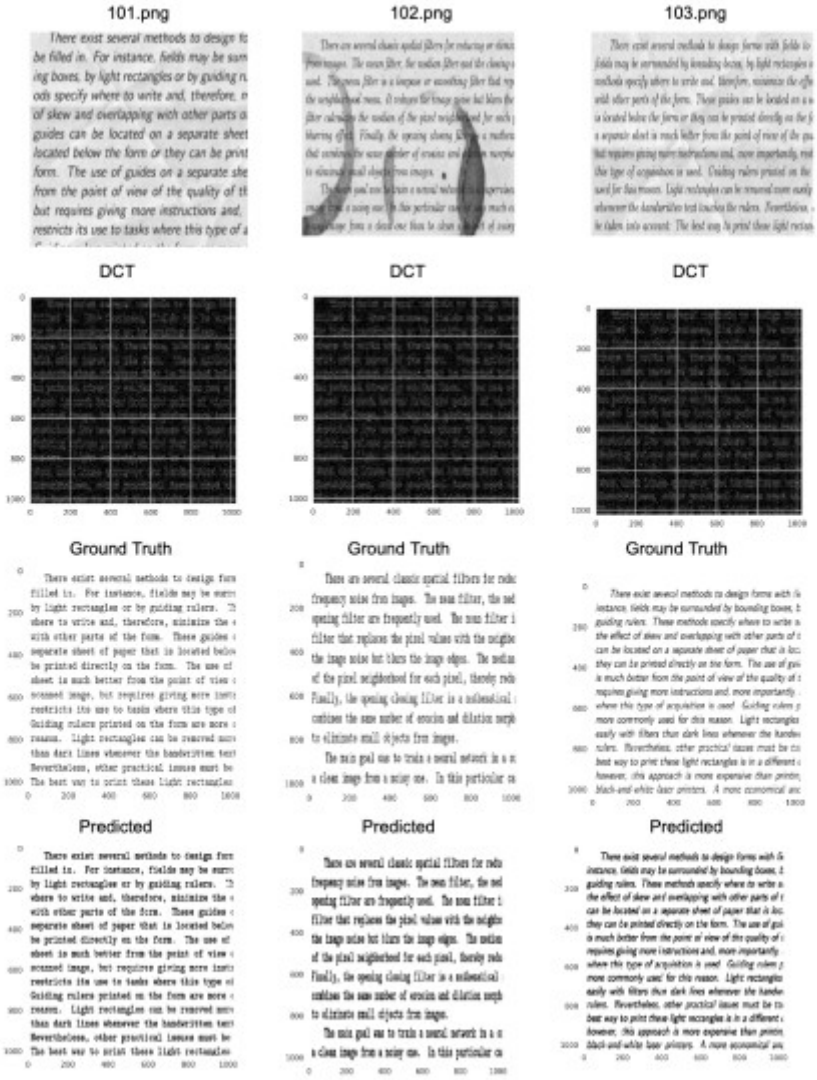



Fig. 8. observations