Image De-raining using Deconvolutional Neural Networks

Kushal P
2014B4A70624G
BITS Pilani, Goa Campus

Adwait Bauskar
2014B4A70595G
BITS Pilani, Goa Campus

Niranjan Jayaraman
2014B4A70614G
BITS Pilani, Goa Campus

***Abstract*—This project aims to filter out a clear image from a rainy image. The method implemented is a deconvolutional neural network. Simulations show an accuracy of around 80%.**

***Keywords*—De-raining, deconvolutional nerual networks, neural networks, image processing.**

I. Introduction

Rainy weather affects image quality of photos taken in these conditions, and make their processing and sharing difficult. Therefore it is important to solve this problem of image deraining. In this paper, we use the concept of deconvolutional neural networks to filter out the rain from an image. We compare our resultant image with the original image using the original accuracy metric defined by Keras. Simulations show that our resultant image after training acheives an accuracy of about 80

II. Background

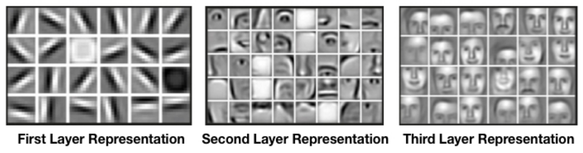
Here are the basic concepts required to understand the rest of the paper. We mainly describe two components. The first one is the convolutional neural networks. In this project, these are not used directly, but the idea behind these networks is significant in characterising the images.

A. Convolutional Neural Networks

Image recognition involves not only the current pixels, but also the relative position of individual pixels. CNNs architecture is built on capturing different spacial features that are crucial in learning. Single pixel is merely a color and independently mean nothing, but more of them together form patterns and whole images. That is because spatial relationship of pixels is very important, especially in case of pictures. Single pixel is merely a color and independently mean nothing, but more of them together form patterns and whole images. That is because spatial relationship of pixels is very important, especially in case of pictures.

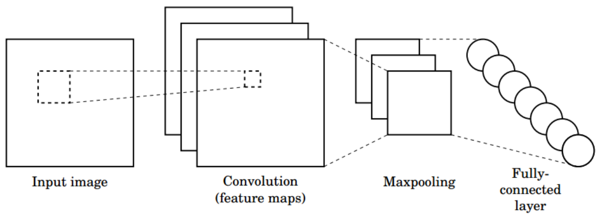
Lets say we want to detect human face from image. Simple neural network would assign every pixel to one neuron in input layer. But what does that mean? It mean that we do not keep spatial information of pixels. We split the image into individual neurons and then feed the network with them.

But in case of face recognition you have parts like eyes. Eyes are complex objects composed of several parts. You have pupil, iris, sclera and even eyelids. Every eye has them. Would you be able to detect the eye only by one pixel or only by one part? Probably not. Only the whole in specific order make sense.If you train neural network on image of eye it will only works if the eye will be on the same exact position on the image every time. When you move, scale or rotate the eye, the network will inevitably fail to predict the correct output.We need some way to look for specific patters instead of individual pixels. And that is what convolutional neural networks do.


First Layer Representation
Second Layer Representation
Third Layer Representation

In case of image detection, the input is an image transformed into matrix of pixel values. If you have grayscale image (like MNIST) each pixels has only one value, whereas colored image has three values per pixel (RGB). During learning, convolutional network will choose appropriate kernels and all you need to do is define their amount (more filters can detect more features, but increases required time for convolution) and size (usually 3x3 or 5x5). This step is called convolution.

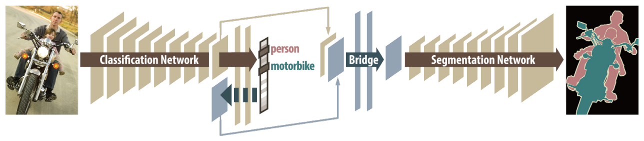
This is then passed on to an activation function which adds non linearity to the network. Then, the image resolution is reduced by using polling layers. And finally multiple of such groups of layers are given to a fully connected network.



B. Deconvolutional Neural Networks

Deconvolutional Networks, a framework that permits the unsupervised construction of hierarchical image representations. These representations can be used for both low-level tasks such as denoising, as well as providing features for object recognition. Each level of the hierarchy groups information from the level beneath to form more complex features that exist over a larger scale in the image. The grouping mechanism is sparsity: by encouraging parsimonious representations at each level of the hierarchy, features naturally assemble into more complex structures. However, sparsity itself is not enough it must be deployed within the correct architecture to have the desired effect.

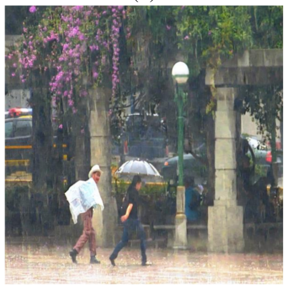
Like a convolutional neural network, the idea here is just the reverse. In unpooling, we place activations to pooled locations and preserve structure of activations. In deconvolutional neural networks, we try to extract the activations as they were convoluted.



III. Methodology

In this paper, we try to use this architecture to remove rain from images. The convolutional neural network with ReLu makes sure that the image values doesn't go negative and also to introduce non linearity. We also use Average Pooling layers to make sure that out network is invariant to rain and snow, and also to the direction that the rain is falling. An example is given below.

(c)


(d)

IV. Simulation

The architecture we simulate in this project contains 4 convolutional layers and 4 deconvolutional layers. The depths of convolutional layers are as follows.

The format is (no. of filters, filter size, stride, activation)

Conv1	: (64, 2, 1, ReLu)
Conv2	: (128, 2, 1, ReLu)
AveragePool1	: PoolSize = (2, 2)
Conv3	: (256, 2, 1, ReLu)
AveragePool2	: PoolSize = (2, 2)
Conv4	: (512, 2, 1, ReLu)
AveragePool3	: PoolSize = (2, 2)

The deconvolutional layers are

Unpool1	: PoolSize = (2, 2)
Deconv1	: (512, 2, 1, ReLu)
Unpool2	: PoolSize = (2, 2)
Deconv2	: (256, 2, 1, ReLu)
Unpool3	: PoolSize = (2, 2)
Deconv3	: (128, 2, 1, ReLu)
Deconv4	: (3, 2, 1, ReLu)

The following architecture was built using Keras which is a python library based on TensorFlow. The training accuracy obtained was close to 80%, the best model was saved and the evaluation accuracy was 76.92%, loss was 655.07, PSNR score of 20.5.

Here are two of the simulation results. The first one is the original image, the second is the rainy one and the third is rainy image.

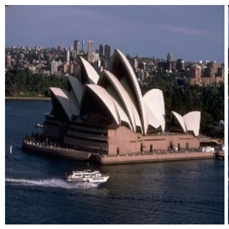


Fig. 1. Ground Truth

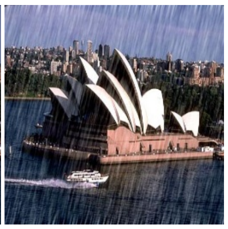


Fig. 2. Rainy

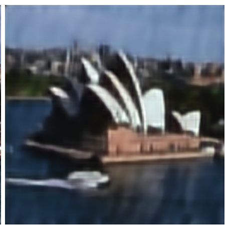


Fig. 3. De-rained

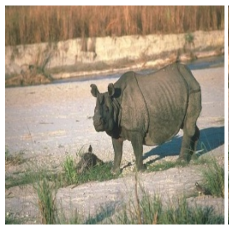


Fig. 4. Ground Truth

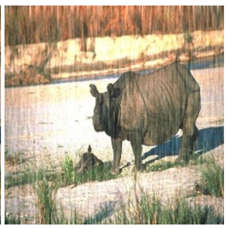


Fig. 5. Rainy

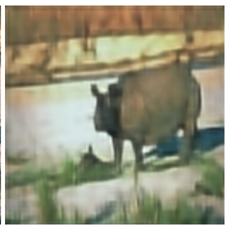


Fig. 6. De-rained

V. Conclusion

To sum up, our simulations result in an accuracy of about 80% and the rain is significantly filtered out in the resultant images while preserving the features of the original images. The new features that are generated are consistent with the image and no extra noise is observed.

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[2] Vojtech Pavlovsky, *Introduction To Convolutional Neural Networks*, 2017.

[3] Matthew D. Zeiler, Dilip Krishnan, Graham W. Taylor and Rob Fergus, *Deconvolutional Networks*, 2010.

[4] Seunghoon Hong, Hyeonwoo Noh, Bohyung Han, *Decoupled Deep Neural Network for Semi-supervised Semantic Segmentation*, NIPS, 2015.