Image De-raining using Deconvolution Network

*

Malhotra Gyanesh
Department of Computer Science
BITS Pilani K K Birla Goa Campus
f20140495@goa.bits-pilani.ac.in

Arora Bharat
Department of Computer Science
BITS Pilani K K Birla Goa Campus
f20140629@goa.bits-pilani.ac.in

***Abstract*—Degradation in the image quality due adverse weather conditions like Rain, Snow and others can make it difficult to identify useful information for not only for computers but also for the humans.So, it has become increasingly important to remove the noises due to weather conditions before the application of any image algorithm to get reliable results. This paper aims to provide a solution for de-raining images by using deconvolution neural network. We used supervised learning techniques to train the Deconv network and used truncated normal for weight initialization of the filters. Images generated using our network are visually clear and natural because of high PSNR and high accuracy.**

***Index Terms*—PSNR, GAN, Relu, Truncated Normal, Pooling**

I. Introduction

Field of computer vision and Machine learning has become increasingly popular these days and their application in day to day life have brought unprecedented benefits but the benefits come with cost of being able to work on dataset generated in uncontrolled conditions. As dataset may come for different weather conditions and as well it is not possible to collect data in controlled conditions.But various computer vision algorithm fail to work on such data. So, for any such algorithm, prepossessing to remove noises due to weather conditions must be done. Convolution neural network provide us with the power of doing such a complex task easily and reliably. There are several model for image de-raining in the literature which have been sued for image de raining problem deconvolution neural network GAN, IDGAN. Generative adversarial give more accurate result are very tough to rain and have high chances of exploding. So, applied deconvolution neural network solve the problem at hand.

II. Literature Review

A. Deconvolution Neural Network and GAN

General adversarial networks is based on min-max optimization problem which is a game theoretic approach to solve a particular unsupervised problem. In the context of deraining problem He Zhang et al, incorporated perceptual loss to make images less grainier.But the problem with GANs is you need to have the right value of the hyperparameters for the generator as well as discriminator.This is to ensure that one doesnt overpowers the other during training and nash equilibrium is obtained. This makes it harder to train GANs effectively, keeping aside the cost related to good enough GPUs. On the other hand, deconvolution neural network are easy to train and also give comparable accuracy as well.

III. Training methodology

A. Image Pre-processing

Images are first split into half to separate the input and output images. The sizes are of both the images are adjusted to get make them acceptable for model to without any cropping and undesirable operations.All the images are resized to a desired size of 256X256 for training and testing purpose.If image length and breadth is greater than 256 then images are shrinked to desired size. If either of the length or breadth is greater than desired then images are centered by shrinking only along that particular axis and zero padding along other axis. Rest all images are zero padded to get desired size. Rational behind using the zero padding was to make it distinguishable from the rain and sky color which are usually in higher range of rgb values.

B. Architecture

- In brief, our model contains 4 layers of convolution,1 layer of dropout and 4 deconvolution layers. Initialisation of the biases and kernel is done through **truncated normal**.
- We used batch normalisation after each layer of the convolution and deconvolution to improve the learning rate and avoid internal covariant shifts. Activation function used in the neural network is relu which do not have any vanishing gradient problems.
- Max pool operation is used after each layer to reduce the parameters. For doing the max pool operation we choose a filter size of 2X2 and stride size of 2X2 as well.
- Dropout is used before the last fully connected convolution layer to reduce the number of parameters in the system.
- After dropout layer we used deconvolution layers with upsampling. For choosing the hyperparameters of the model, 10 percent of the total training dataset provided is kept for validation purpose and rest 90 percent is used to train.

- We used early stopping , to avoid unnecessary iterations after a point when change in loss function become stagnant. We used patience limit as 15 which means the program will stop after 15 iterations if there is no significant change in loss function.
- As we need to generate the output image from final layer. So, size of the final deconvolutional layer is kept 256x256x3.

1) Loss functions: Mean squared error function is used to calculate the loss.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

2) Optimizer: We tried on adadelta and adam optimiser while training but the result from adam are better than adadelta. So, during final training adam optimiser is used.

C. Training methodology

From the graph of loss function give in "Fig. 2" we can observe that the curve is monotonic in nature. It keeps on decreasing until 160 epochs around which it starts to plateau. This signifies that our model capacity has been fully exploited for the given parameter setting to obtain the best model possible.Similar nature is observed for the validation loss curve shown in "Fig. 3" also. It is not completely monotonic but overall there is a clear downward trend observed. Few peaks in the validation loss in early epochs correspond to our model still learning about the dataset and trying to generalise it. Moreover, from graph of training accuracy shown in "Fig. 1", we can observe that it clearly shows an upward trend, with few peaks and lows .This is also due to unseen data and our model trying to generalise better.Also, we can observe that after 130 epochs our models accuracy doesnt seem to go up. But our model is still learning about the data since the loss value keeps on decreasing even after 130 epochs. This pattern is validated by the behaviour of validation loss and validation accuracy, in "Fig. 4".

a) Accuracy: Accuracy of the model increases significantly for few initial epochs during training, later it become stagnant. "Fig. 1", shows the snapshot of stats collected by tensor-board during training.

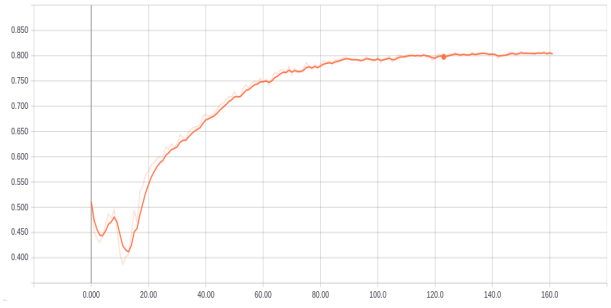

Fig. 1. Y axis = 'accuracy' & X axis = 'number of epochs'

b) Loss: Loss of the model decreases inversely for few initial epochs during training, later it become stagnant. "Fig. 2", shows the snapshot of stats collected by tensor-board during training.

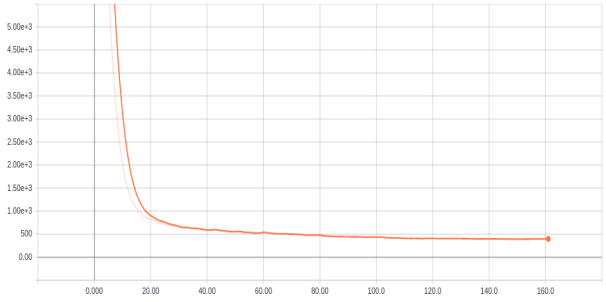

Fig. 2. Y axis = 'loss' & X axis = 'number of epochs'

c) Validation Loss: Validation Loss of the model increase but with huge variability for few initial epochs during training, later it become stagnant. "Fig. 3", shows the snapshot of stats collected by tensor-board during training.

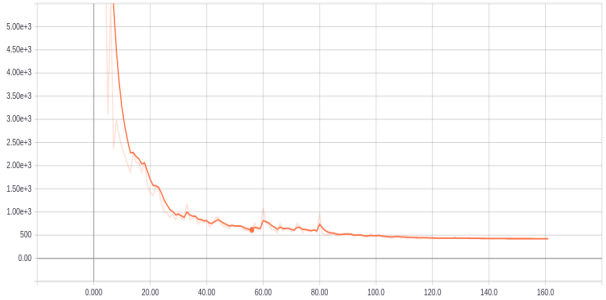

Fig. 3. Y axis = 'validation loss' & X axis = 'number of epochs'

d) Validation Accuracy: Validation accuracy of the model decreases with some variability for few initial epochs during training, later it become stagnant."Fig. 4", shows the snapshot of stats collected by tensor-board during training.

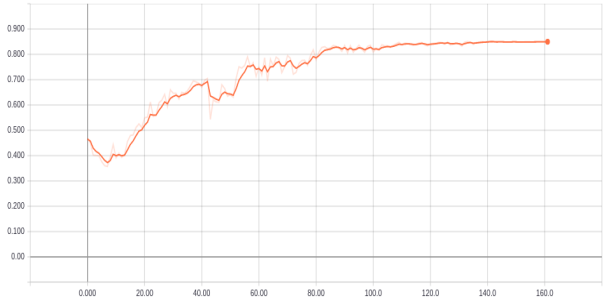

Fig. 4. Y axis = 'validation accuracy' & X axis = 'number of epochs'

D. Model Summary

We tried with different depths for the convolution and deconvolution layers for a small number of epoch.Among

8conv-8deconv , 5conv-5deconv and 4conv-4deconv, 4conv-4deconv gave us the best results and so we have chosen following model as our architecture for the image deraining problem.

TABLE I
MODEL SUMMARY

S.NO	Layer Description		
	layer(type)	***Output Shape***	***Parama***
1	conv2d_1	None, 254, 254, 64	1792
2	batch_normalization_1	None, 254, 254, 64	256
3	activation_1	None, 254, 254, 64	0
4	max_pooling2d_1	None, 127, 127, 64	0
5	conv2d_2	None, 126, 126, 128	32896
6	batch_normalization_2	None, 126, 126, 128	512
7	activation_2	None, 126, 126, 128	0
8	max_pooling2d_2	None, 63, 63, 128	0
9	conv2d_3	None, 62, 62, 256	131328
10	batch_normalization_3	None, 62, 62, 256	1024
11	activation_3	None, 62, 62, 256	0
12	max_pooling2d_3	None, 31, 31, 256	0
13	conv2d_4	None, 30, 30, 512	524800
14	batch_normalization_4	None, 30, 30, 512	2048
15	activation_4	None, 30, 30, 512	0
16	dropout_1 Dropout	None, 30, 30, 512	0
17	conv2d_transpose_1	None, 31, 31, 512	1049088
18	batch_normalization_5	None, 31, 31, 512	2048
19	activation_5	None, 31, 31, 512	0
20	up_sampling2d_1	None, 62, 62, 512	0
21	conv2d_transpose_2	None, 63, 63, 256	524544
22	batch_normalization_6	None, 63, 63, 256	1024
23	activation_6	None, 63, 63, 256	0
24	up_sampling2d_2	None, 126, 126, 256	0
25	conv2d_transpose_3	None, 127, 127, 128	131200
	batch_normalization_7	None, 127, 127, 128	512
53	activation_7	None, 127, 127, 128	0
55	up_sampling2d_3	None, 254, 254, 128	0
57	conv2d_transpose_4	None, 256, 256, 3	3459

E. Conclusion

We tested it for 97 images on the following evaluation meterics and got the following results.

TABLE II
RESULTS

Metric	Value
loos	560.21
accuracy	21.38
psnr	84.64%

Following are few samples of images generated by our neural network.

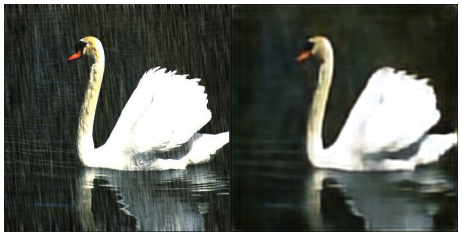

Fig. 5. Left Side: Input Image, Right Side:Output Image

Fig. 6. Left Side: Input Image, Right Side:Output Image

REFERENCES

[1] He Zhang, Student Member, IEEE, Vishwanath Sindagi, Student Member, IEEE Vishal M. Patel, Senior Member, IEEE (2017) Image Deraining Using a Conditional Generative Adversarial Network, Department of Electrical and Computer Engineering at Rutgers University, Piscataway, NJ USA.

[2] an J. Goodfellow , Jean Pouget-Abadie , Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio (2012) Generative Adversarial Nets, Department of Monteral , University of Monteral .

[3] Li Xu, Lenovo Research & Technology, Jimmy SJ. Ren,Lenovo Research Technology,Ce Liu,Microsoft Research (2015) Deep Convolutional Neural Network for Image Deconvolution, Research Grants Council of the Hong Kong Special Administrative Region. Valley, CA: University Science, 1989.