DEEP BOOSTING FOR IMAGE DENOISING

TEAM – SMAI_TEAM

- BHAVANAM SAI VARUN REDDY (2021201026)
- G SAI TEJA (2021201040)
- NALLURU VSS MANEESH GUPTA (2021201041)

GOAL OF THE PAPER

- Integrating several CNNs in a feed-forward fashion where each network serves as a boosting unit.
- Construction of a Deep Boosting Framework (DBF).
- Dilated Dense Fusion Networks (DDFN) are used as boosting units to fully exploit the potential of DBF.

INTRODUCTION

With the increase of GPU based parallel computing, learning-based models have become popular, sidelining traditional frameworks based on boosting frameworks

The DBF in this paper focuses on denoising, but it can be used for other denoising tasks as well

Theoretically, boosting unit in DBF can be of any type.

In practice, not all structures are suitable, since the depth of DBF is substantially increased and it causes problems during training convergence.

Steps to convert plainly connected network into DDFN -

- To overcome vanishing gradients during training, we use dense connections which will help in reusage of features which will in turn guarantee efficiency. (Since in dense net, all modules are interconnected, which means each module will have access to every other module without traversing through layers)
- To obtain better performance on densely connected networks, we use dilated convolution for widening the receptive field without introducing additional parameters
- Path widening fusion scheme is combined with dilation to make DDFN more efficient

MODELS

Four kinds of Models were explored in this paper which are as follows:

1. Plain Network
2. Dense Network
3. Dilated Dense Network
4. Dilated Dense Fusion Network

DESCRIPTION OF MODEL

DBF – deep boosting framework

- DBF integrates several CNN in feed-forward fashion
- Depth of the framework is substantially increased which becomes as a difficulty during training
- Dense connections are used which overcome the problem of vanishing gradient during TRAINING

DDFN – dilated dense fusion network – a single boosting unit

- Incorporates path widening fusion scheme incorporated with dilated convolution
- Standard Convolution vs dilated convolution

TOOLS AND LIBRARIES

- numpy
- matplotlib
- Scikit-learn
- tensorflow
- keras
- Google Colab notebooks

TRAIN DATASET DESCRIPTION

We have used the train-400 as our initial dataset, as there were only 400 images in the dataset all are grayscale, so it was difficult to train the model using only 400 images. So, we have taken each image and divided each image into multiple smaller patches of the images (size 50x50) and have used it as the dataset.

Then we have added the gaussian noise (using standard deviation of 50) for each and every patch of all the images and gave that data to the model for training.

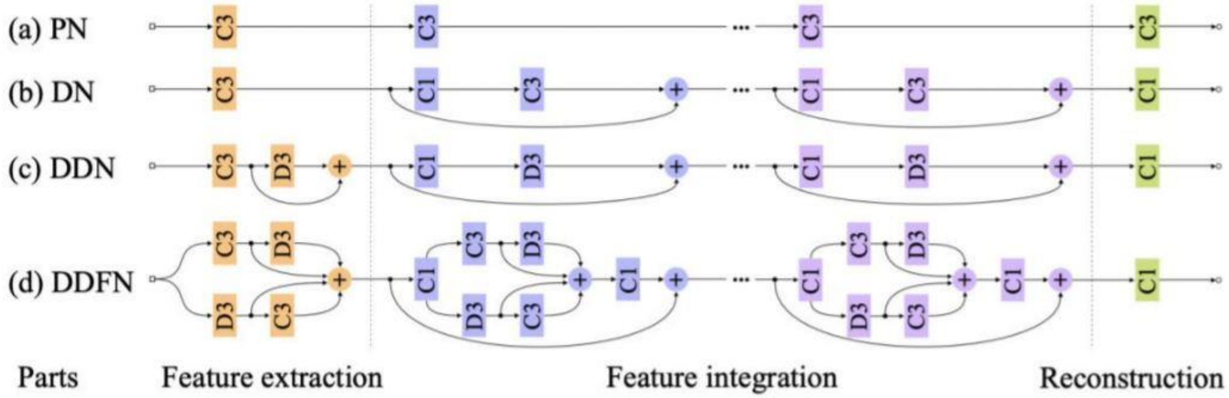
TEST DATASET DESCRIPTION

We have used the set-12 as our test dataset, as there are only 12 images, we have taken each image and divided each image into multiple smaller patches of the images (size 50x50) and have used it as the dataset.

Then we have added the gaussian noise (using standard deviation of 50) for each and every patch of all the images and used for testing our model.

MODEL EQUATIONS AND DIAGRAM

Parts	PN	DN	DDN	DDFN
1	$C(3\times3\times24)$	$C(3\times3\times32)$	$\begin{bmatrix}C(3\times3\times16)\\D(3\times3\times16)\end{bmatrix}$	$\begin{bmatrix}C(3\times3\times8/6)\\D(3\times3\times8/6)\end{bmatrix}$
2	$C(3\times3\times24)\times8$	$\begin{bmatrix}C(1\times1\times32)\\C(3\times3\times8)\end{bmatrix}\times8$	$\begin{bmatrix}C(1\times1\times32)\\D(3\times3\times8)\end{bmatrix}\times8$	$\left\{\begin{matrix}C(1\times1\times24)\\\begin{bmatrix}C(3\times3\times6/3)\\D(3\times3\times6/3)\end{bmatrix}\\C(1\times1\times8)\end{matrix}\right\}\times8$
3	$C(3\times3\times1)$	$C(1\times1\times1)$	$C(1\times1\times1)$	$C(1\times1\times1)$



PARTS OF MODEL

- FEATURE EXTRACTION
- FEATURE INTEGRATION
- RECONSTRUCTION

EVALUATION METRICS

We used 2 metrics for evaluation

1. MSE
2. PSNR

We have used the MSE loss as our evaluation metrics for model performance, as the number of epochs were increasing the training loss is decreasing accordingly.

PSNR was also increasing correspondingly when we increased the depth.

Also, for non-wide model, PSNR is lower than PSNR for wide model. Suggesting that the wide variant is better than non-wide variant.

MEAN SQUARED ERROR(MSE)

- MSE refers to the average loss on an observed data set.
- An MSE of zero, meaning that the estimator predicts observations of the parameter with perfect accuracy, is ideal (but typically not possible).

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y_i})^2.$$

PEAK SIGNAL TO NOISE RATIO(PSNR)

- Peak signal-to-noise ratio (PSNR) is the ratio between the maximum possible power of an image and the power of corrupting noise that affects the quality of its representation. To estimate the PSNR of an image, it is necessary to compare that image to an ideal clean image with the maximum possible power.

$$PSNR = 20\log_{10}\left(\frac{MAX_f}{\sqrt{MSE}}\right)$$

- Here, MAX_f is the number of maximum possible intensity levels that is equal to 1.

WEIGHT INITIALIZATION

- We used **He initialization** according to this paper

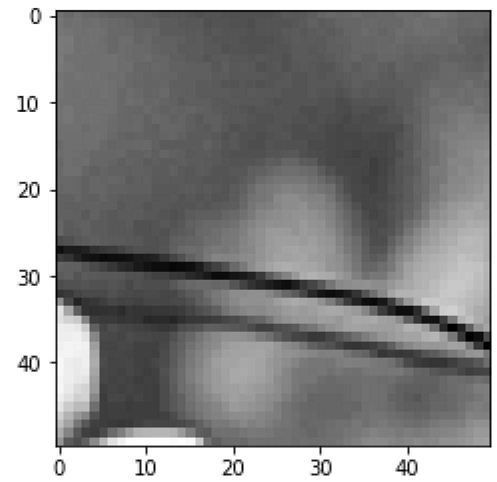
https://www.cv-foundation.org/openaccess/content_iccv_2015/papers/He_Delving_Deep_into_ICCV_2015_paper.pdf

- This initialization ensures that the convergence is faster than usual random initialization of weights.
- For the last layer, we initialized the weights using normal distribution
- Biases in each and every layer is 0 by default

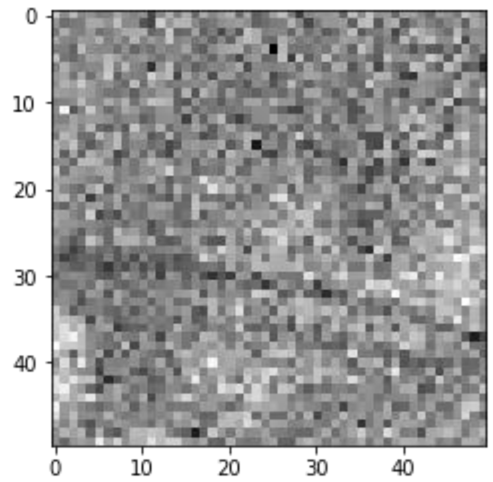
HYPER-PARAMETERS CONSIDERATION

- As mentioned in the paper, we used Adam solver for optimization with
  - Coefficient of weight decay as 0.0001
  - Learning rate as 0.001
  - Number of epochs as 50
  - Batch size as 64

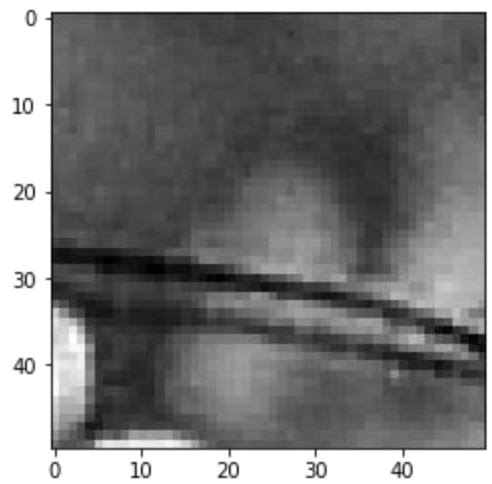
OUTPUT OF THE MODEL



Original



Noise



Reconstructed

The above reconstructed image is the output of DBF network with 8 layers of Feature Integration and 25 epochs

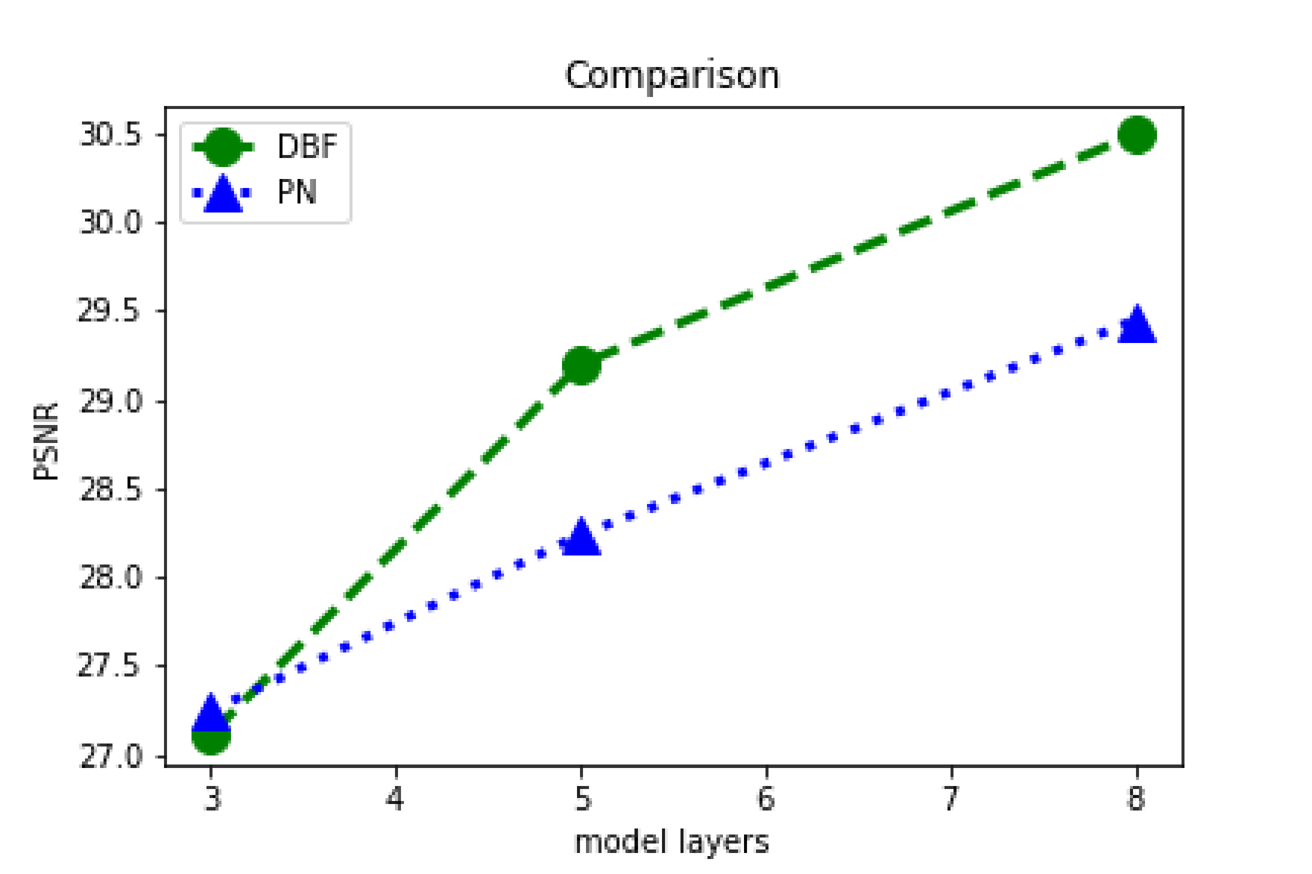
EXPERIMENTED MODELS

We have experimented with different architectures for both the models-

1. DDN - 3 CNN layers
2. DDN - 5 CNN layers
3. DDN - 8 CNN layers
4. DDFN - 3 CNN layers
5. DDFN - 5 CNN layers
6. DDFN - 8 CNN layers

OBSERVATIONS

- We have calculated the PSNR values for both the models along with different variations in the layers of the model.



WHY DDFN IS BETTER THAN DDN?

- Path Widening Fusion (PWF) – this makes the boosting algorithm more efficient.
- This expands the number of forward paths to derive DDFN from DDN.
- The order between the dilated convolutions (Dconv) and convolutions (conv) is exchanged in different branches.
- Conv-ReLU-Dconv and Dconv-ReLU-Conv branches learn different features and PWF exploits their potention at the same time.

WORK DISTRIBUTION

Team Member	Contribution (not limited to the mentioned)
Bhavanam Sai Varun Reddy (2021201026)	Data pre-processing, hyper parameters tuning
G Sai Teja (2021201040)	Implemented DDN model, Implemented DDFN model,
Nalluru VSS Maneesh Gupta (2021201041)	Dataset preparation, PSNR metric evaluation, code debugging and integration.