Superresolution of Images by different deep models

Prepared by
Shreya Roy
INT939

Introduction

- Existing image SR algorithms can be classified into two categories:
  1. Reconstruction based-
  2. Learning-based algorithms.

- Reconstruction-based algorithms require multiple spatial/spectral/temporal low-resolution images of the same scene.
- Learning-based approaches rely on prior information which can be extracted from the existing dataset consisting of high and corresponding low-resolution images.
- Learning-based SR algorithms can be divided into three categories regression, representation, and deep-learning-based algorithms.

Literature Survey

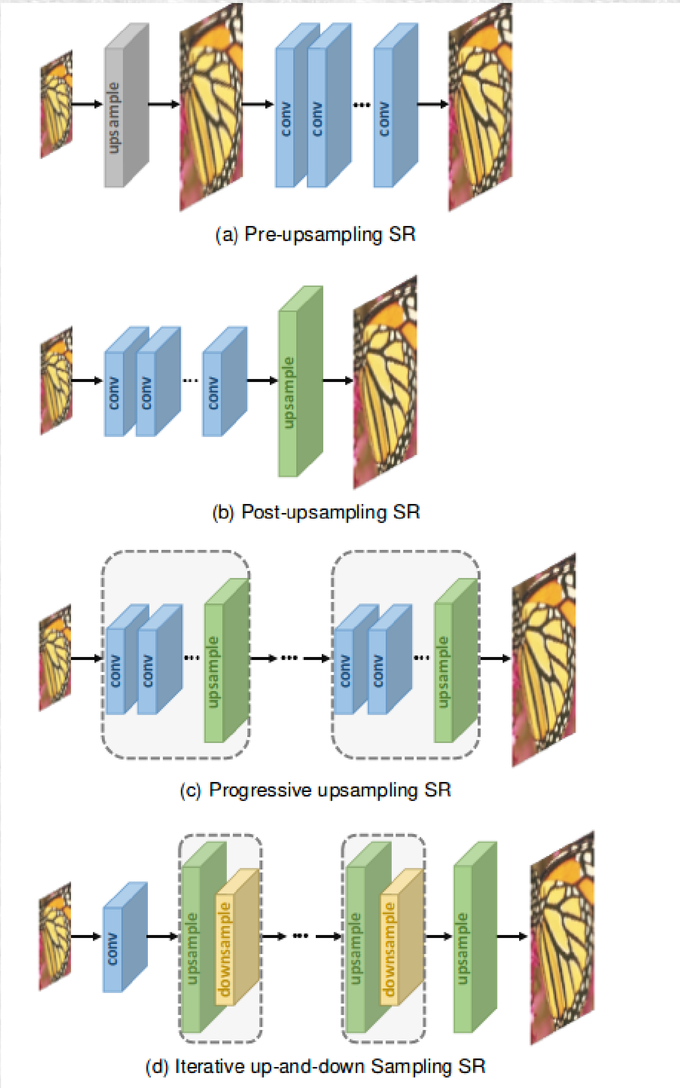
SISR deep algorithms into 4 broad Categories:

A. Pre-upsampling Super-resolution: SRCNN , VDSR , DRRN, IRCNN, DNCNN

B.Post-upsampling Super-resolution: FSRCNN,ESPCN.

C.Progressive Upsampling Super-Resolution

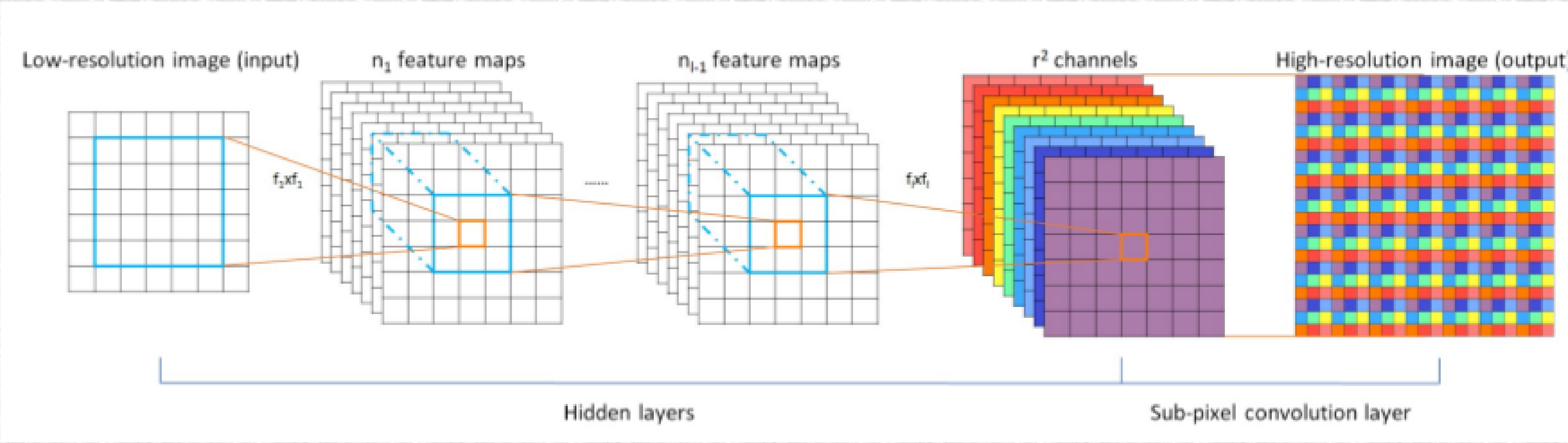
D.Iterative up-and-down sampling:



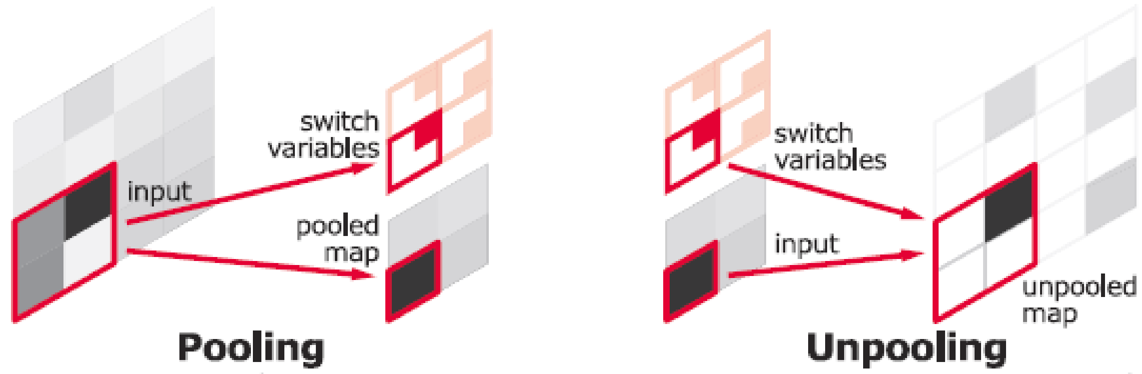
Issues with Pre-upsampling and Post upsampling

- Bicubic interpolation
- L2 loss: L2 loss fails to capture the underlying multi-modal distributions of HR patches.
- Reconstructed HR images are often over-smoothed and inconsistent to human visual perception on natural images.
- The one-step upsampling does not super-resolve the fine structures well, so learning mapping functions for large scaling factors (e.g., 8×) is difficult.
- To address these issues, the deep Laplacian Pyramid Super-Resolution Network (LapSRN) was proposed to progressively reconstruct HR images in a coarse-to-fine fashion.

ESPCN

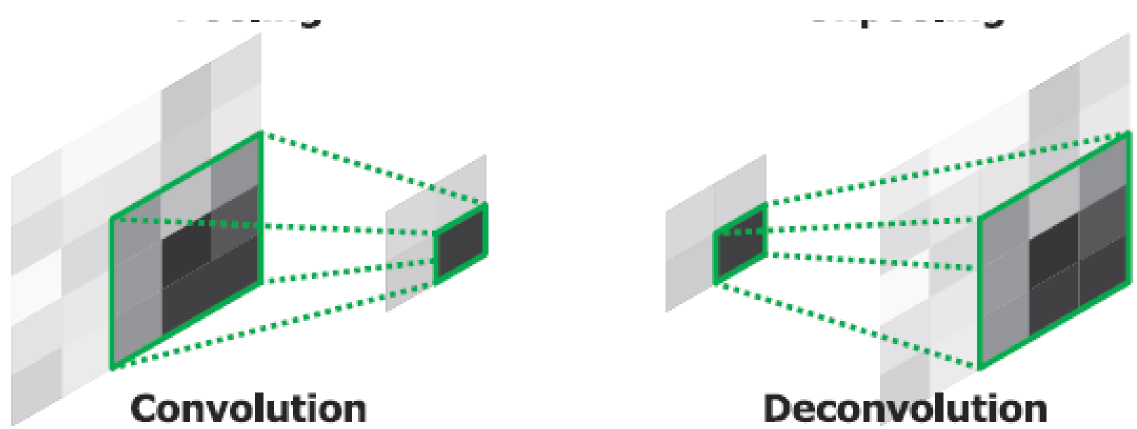


Low-resolution image (input)
n_1 feature maps
n_{l-1} feature maps
r^2 channels
High-resolution image (output)
$f_1 x f_1$
$f_l x f_l$
Hidden layers
Sub-pixel convolution layer



Remember positions when Pooling (Left), Reuse the position information during Unpooling (right)

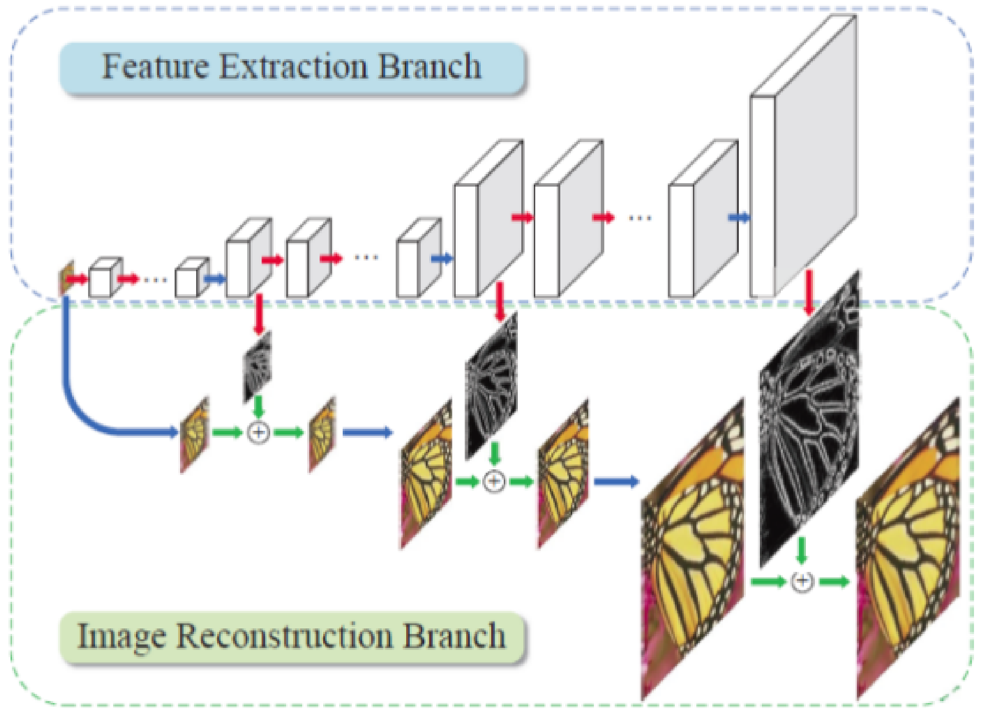
To perform unpooling, we need to remember the position of each maximum activation value when doing max pooling, as shown above. Then, the remembered position is used for unpooling as shown above.



Convolution is to conv the input to smaller size (Left) Deconvolution is to conv the input back to larger size (Right)

Lap SRN

- It consists of two branches Feature Extraction and Feature Reconstruction.



LapSRN / MS-LapSRN Architecture

Robust Loss to optimize Super-resolution deep network

Also, the use of **robust Charbonnier loss** function better handles outliers and improves the SR performance over the L2 loss function.

$y_s = x_s + r_s$

$$\mathcal{L}(\hat{y}, y; \theta) = \frac{1}{N} \sum_{i=1}^{N} \sum_{s=1}^{L} \rho\left(\hat{y}_s^{(i)} - y_s^{(i)}\right)$$

$$= \frac{1}{N} \sum_{i=1}^{N} \sum_{s=1}^{L} \rho\left((\hat{y}_s^{(i)} - x_s^{(i)}) - r_s^{(i)}\right)$$

where $\rho(x) = \sqrt{x^2 + \varepsilon^2}$ is the Charbonnier penalty

Some other works such as **MS-LapSRN and progressive SR(ProSR)** also adopt this framework and achieve relatively high performance.

In contrast to the LapSRN and MS-LapSRN which use the intermediate reconstructed images as the ”base images” for subsequent modules, the ProSR only keeps the main information stream and reconstructs intermediate-resolution images by individual heads.

Difference between LapGAN (introduced in 2016) and LapSRN

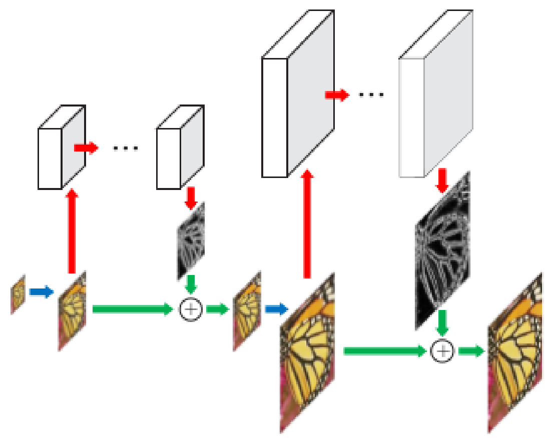
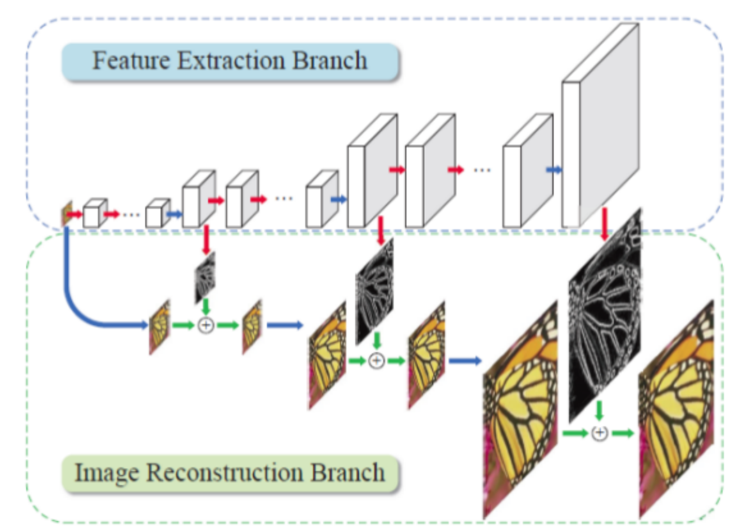


Fig. 2. **Generative network of LAPGAN [34].** The LAPGAN first upsamples the input images before applying convolution for predicting residuals at each pyramid level.



LapSRN / MS-LapSRN Architecture

1.The LAPGAN is a generative model which is designed to synthesize diverse natural images from random noise and sample inputs. On the contrary, the LapSRN is a super-resolution model that predicts a particular HR image based on the given LR image and upsampling scale factor.

2.The LAPGAN upsamples input images before applying convolution at each level, while **LapSRN extracts features directly from the LR space and upscales images at the end of each level** which effectively alleviates the computational cost and increases the size of receptive fields.

LapGAN Architecture Detail

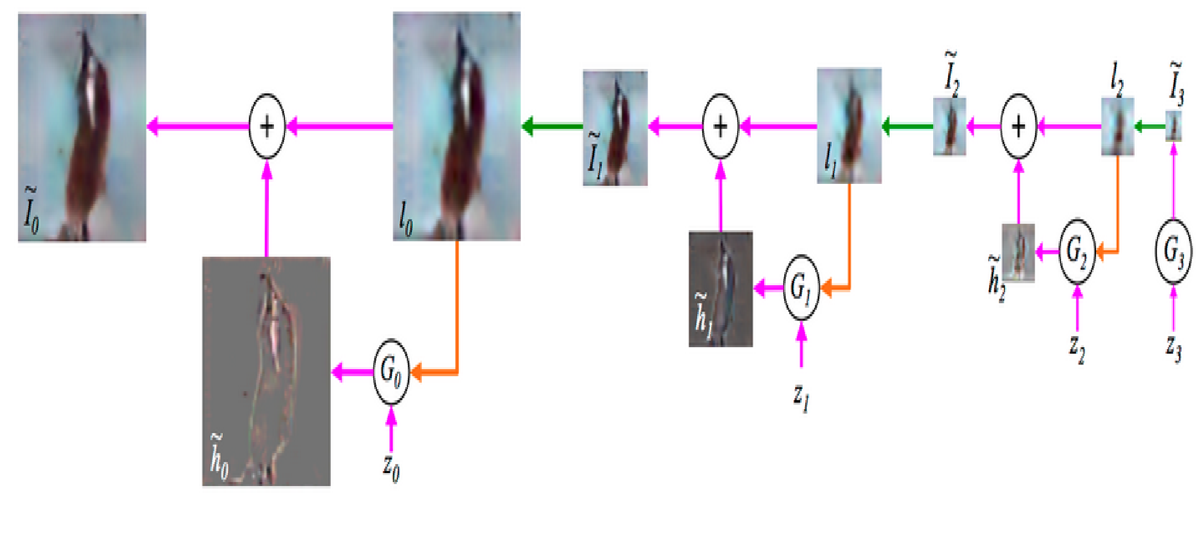


Figure 1: The sampling procedure for our LAPGAN model.

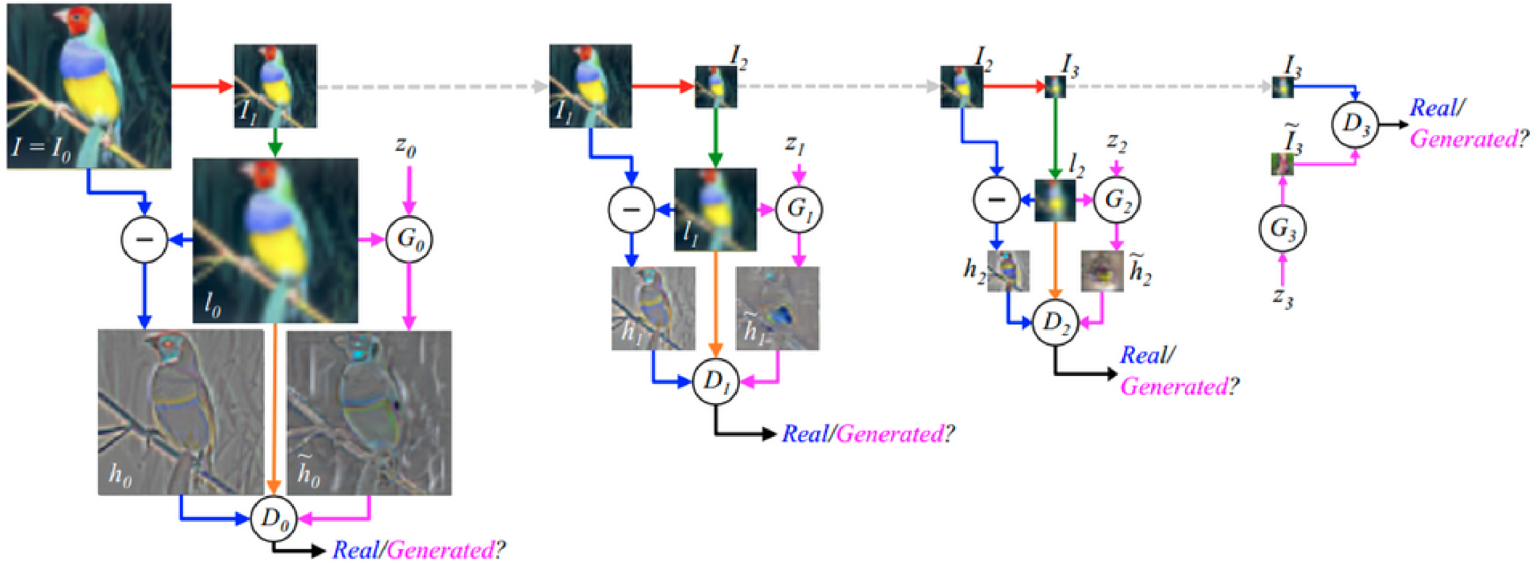


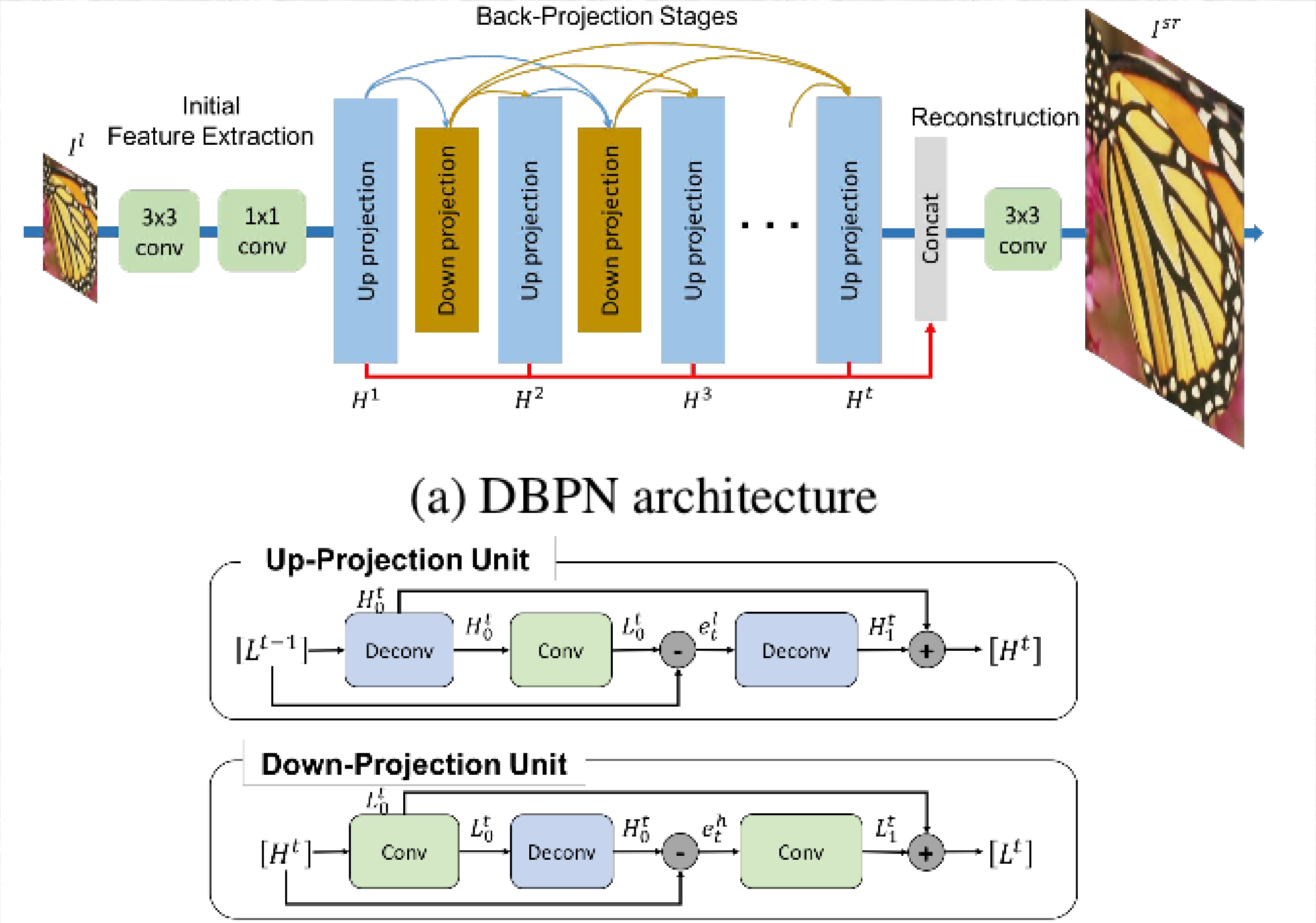
Figure 2: The training procedure for our LAPGAN model.

Iterative up-and-down sampling

DBPN architecture:

1. Initial feature extraction
2. Back-projection stages
3. Reconstruction

- Coupled with other techniques (e.g., dense connections), the DBPN wins the championship on the classical track of NTIRE 2018.



(a) DBPN architecture

(b) the up- and down-projection units in DBPN

Figure 2. Toyota-TI's DBPN network structure.

DBPN

The up-projection unit is defined as follows:

$$\text{scale up:} \quad H_0^t = (L^{t-1} * p_t) \uparrow_s, \tag{1}$$

$$\text{scale down:} \quad L_0^t = (H_0^t * g_t) \downarrow_s, \tag{2}$$

$$\text{residual:} \quad e_t^l = L_0^t - L^{t-1}, \tag{3}$$

$$\text{scale residual up:} \quad H_1^t = (e_t^l * q_t) \uparrow_s, \tag{4}$$

$$\text{output feature map:} \quad H^t = H_0^t + H_1^t \tag{5}$$

The down-projection unit is defined very similarly, but now its job is to map its input HR map H^t to the LR map

$$\text{scale down:} \quad L_0^t = (H^t * g_t') \downarrow_s, \tag{6}$$

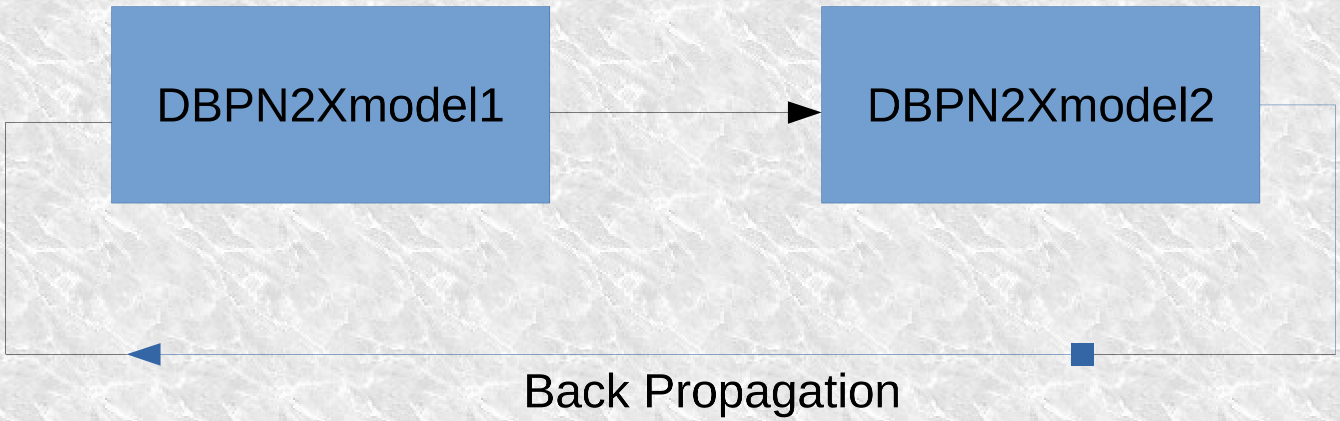
$$\text{scale up:} \quad H_0^t = (L_0^t * p_t') \uparrow_s, \tag{7}$$

$$\text{residual:} \quad e_t^h = H_0^t - H^t, \tag{8}$$

$$\text{scale residual down:} \quad L_1^t = (e_t^h * g_t') \downarrow_s, \tag{9}$$

$$\text{output feature map:} \quad L^t = L_0^t + L_1^t \tag{10}$$

Cascaded DBPN

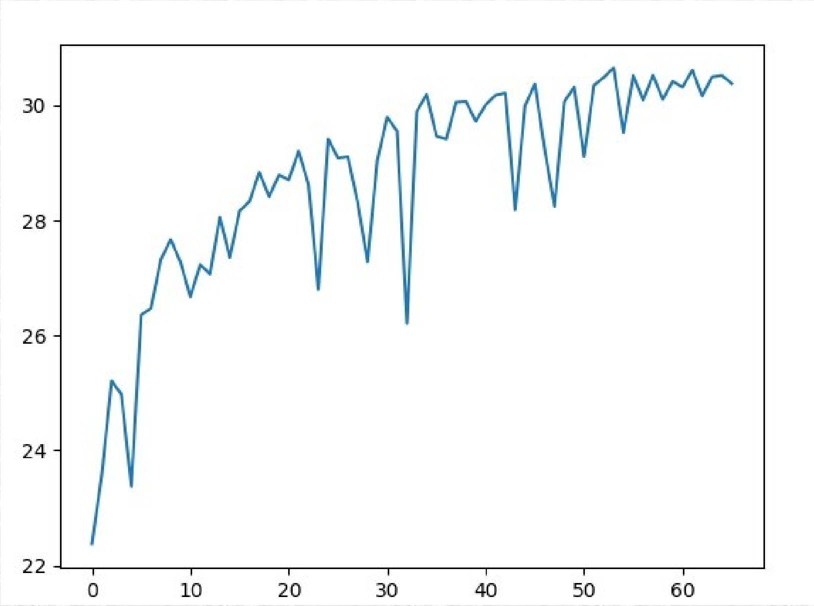


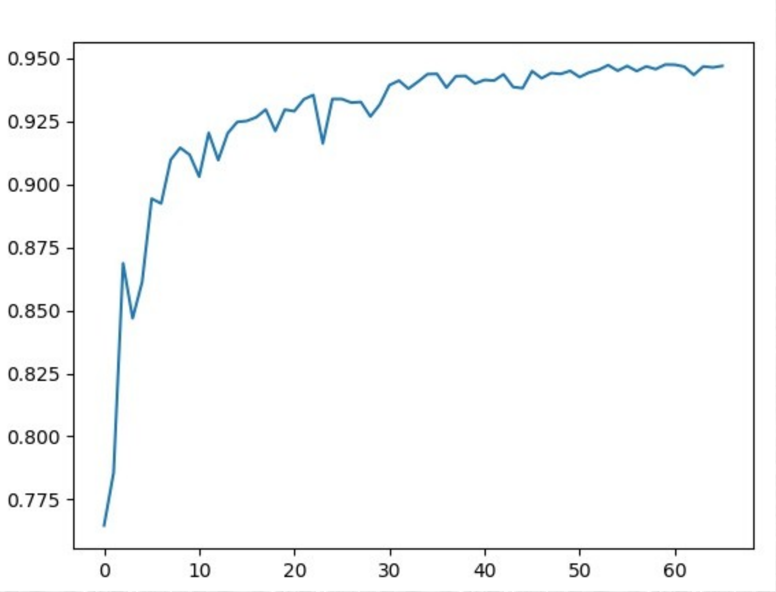
A slight modification of already existing DBPN4X model that introduce the step by step upsampling

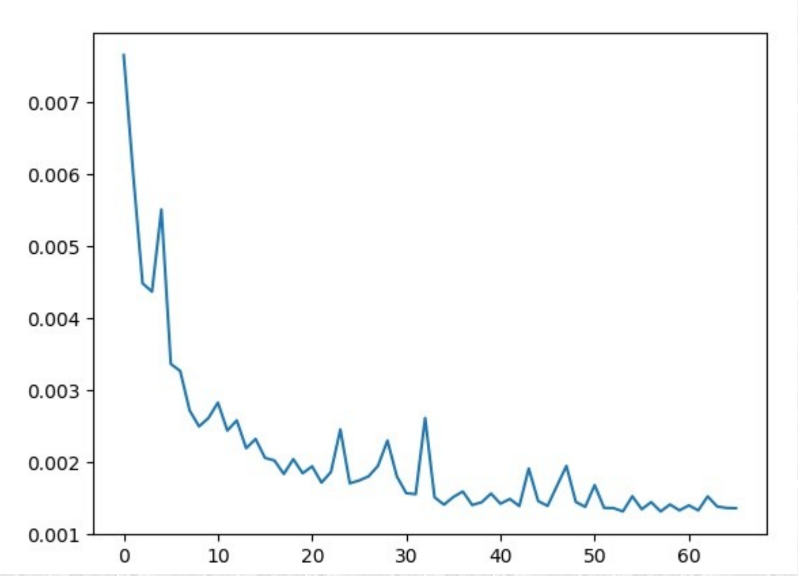
Dataset

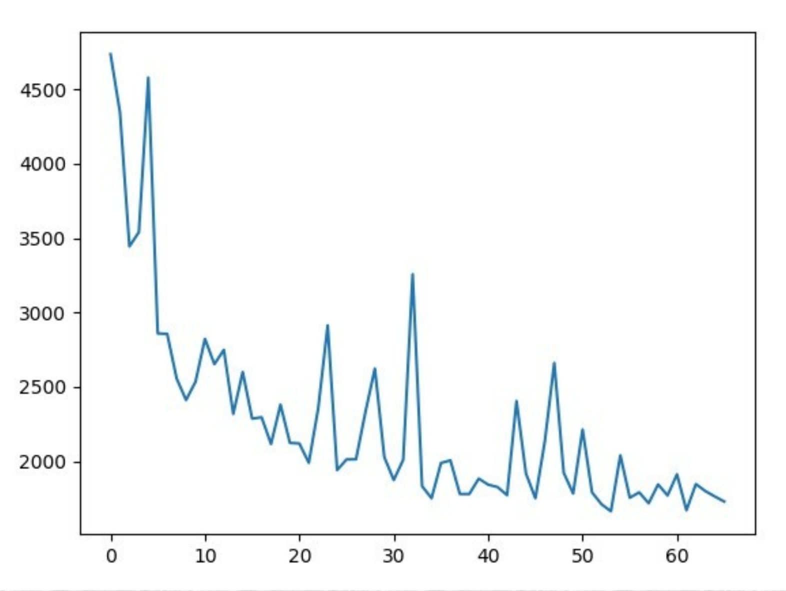
- I have performed the experiment on ICDAR 2015 data. The dataset had varying size of images and hence for training in batch I had to do padding

PSNR,SSIM,MSE,RobustLoss vs number of iterations With Cascaded DBPN

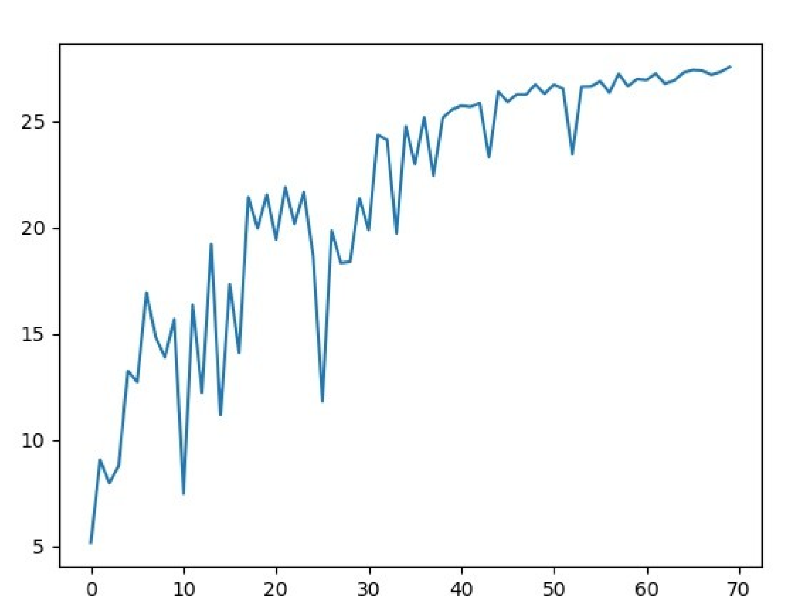


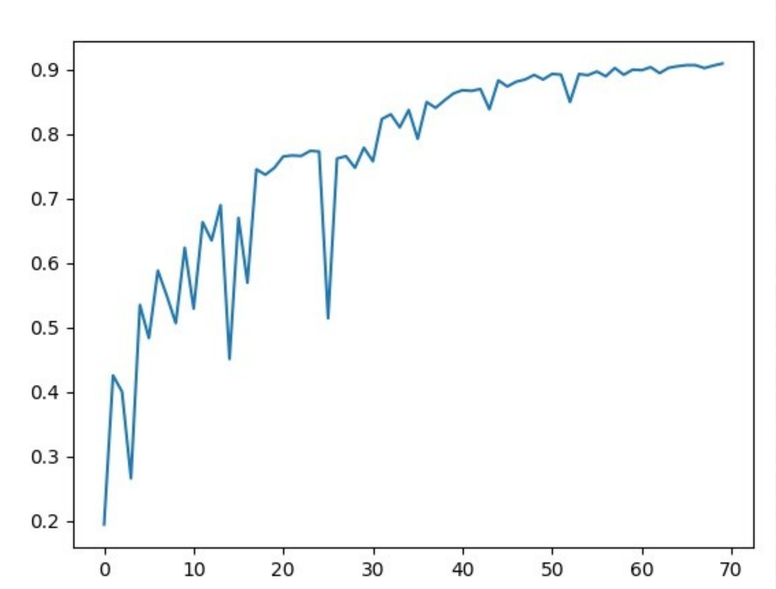


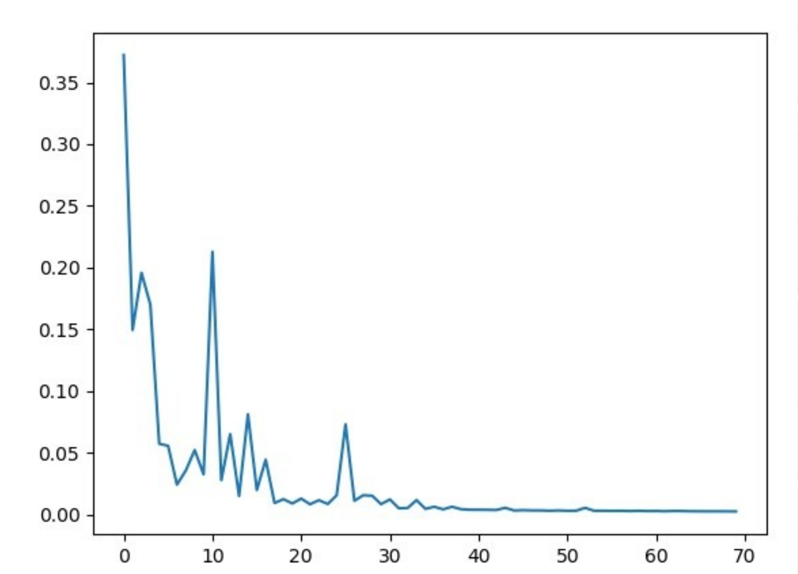


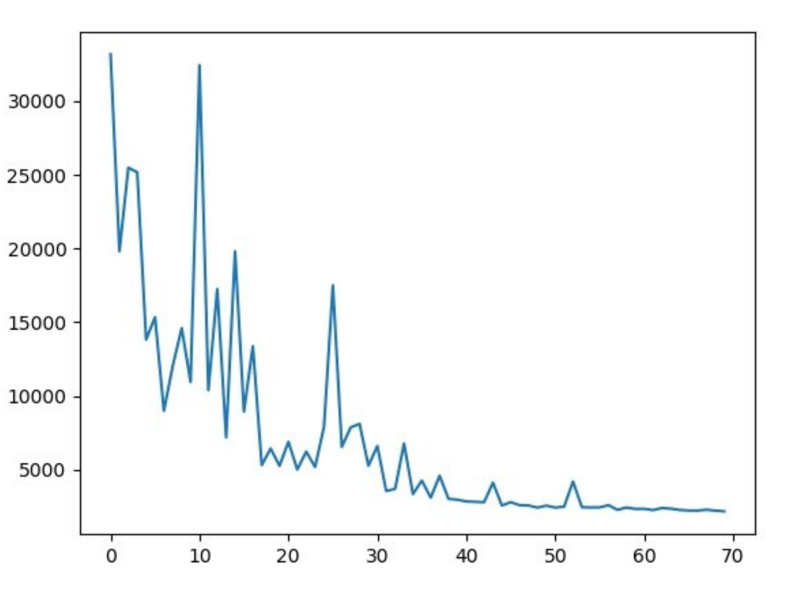


PSNR,SSIM,MSE,RobustLoss vs number of iterations With LapSRN

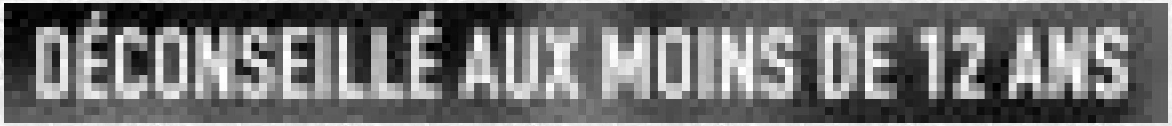








Result on an image from validation set with Cascaded DBPN



Low Resolution

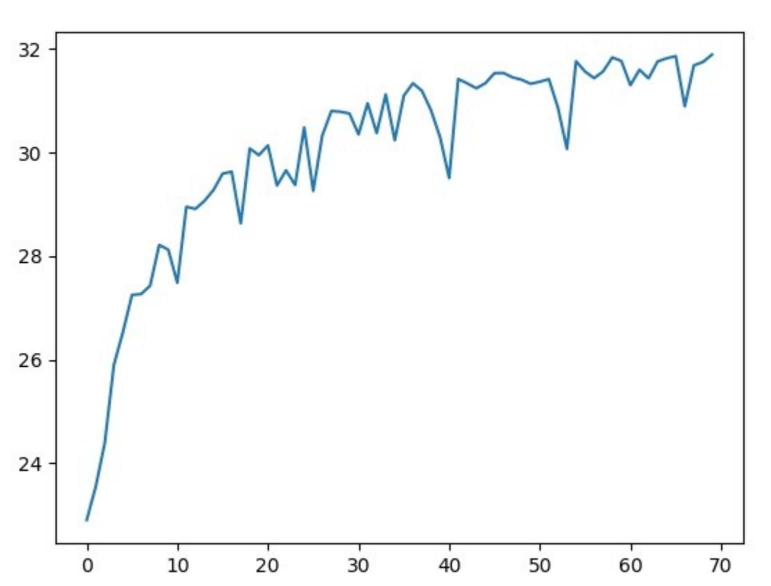
BiCubic

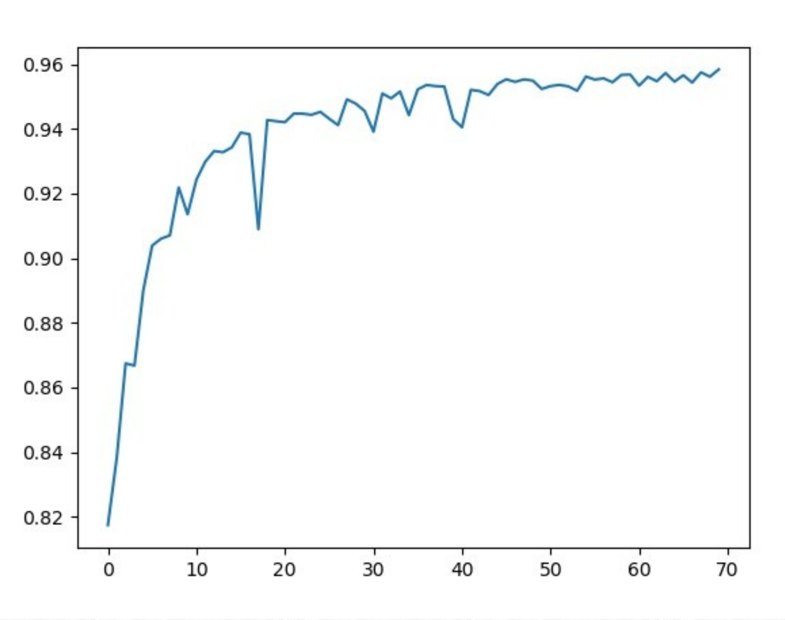
Original high resolution

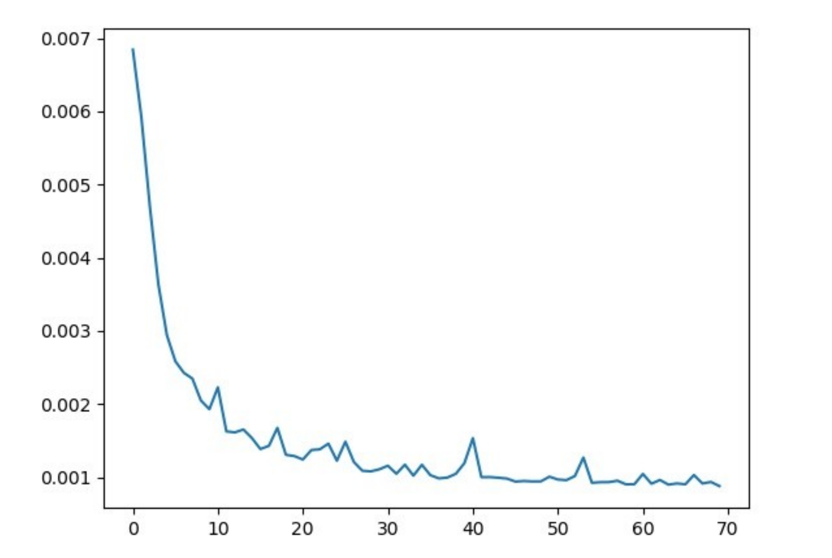
Predicted High Resolution

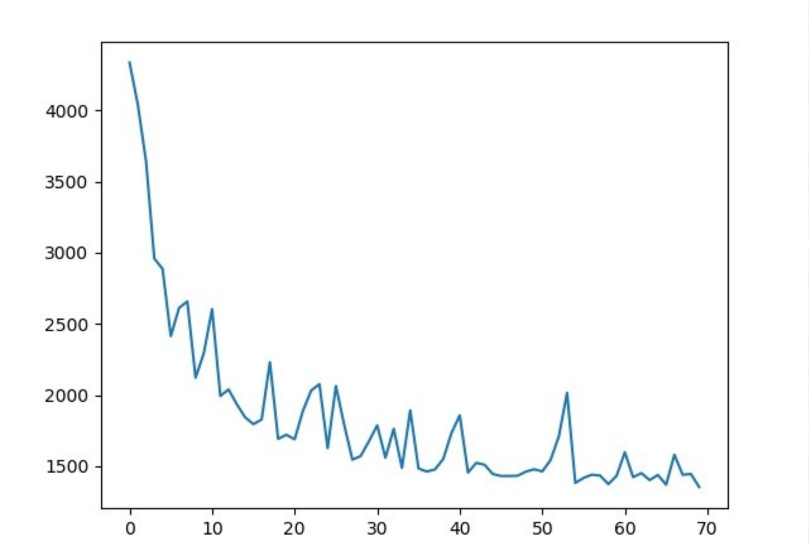
Residual: |Predicted - Original High|

PSNR,SSIM,MSE,RobustLoss vs number of iterations With DRLN

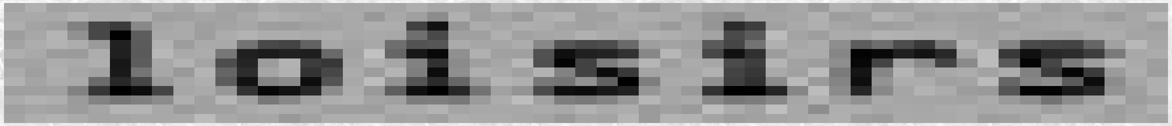




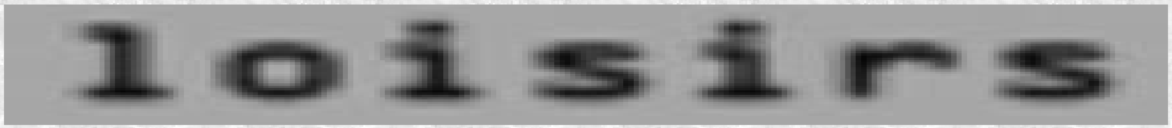




Result on an image from validation set with Cascaded DBPN

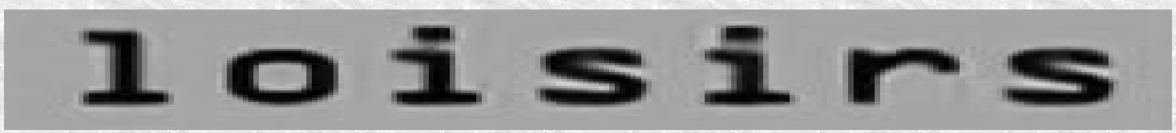


Low Resolution

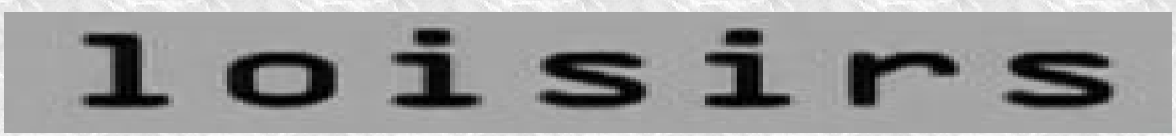


BiCubic

Original high resolution



Predicted High Resolution With Cascaded DBPN



Predicted High Resolution With DRLN

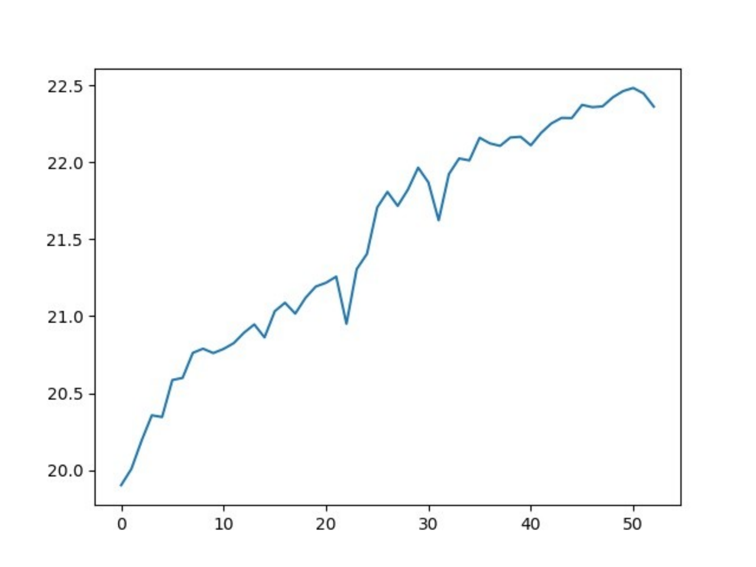
Discussion

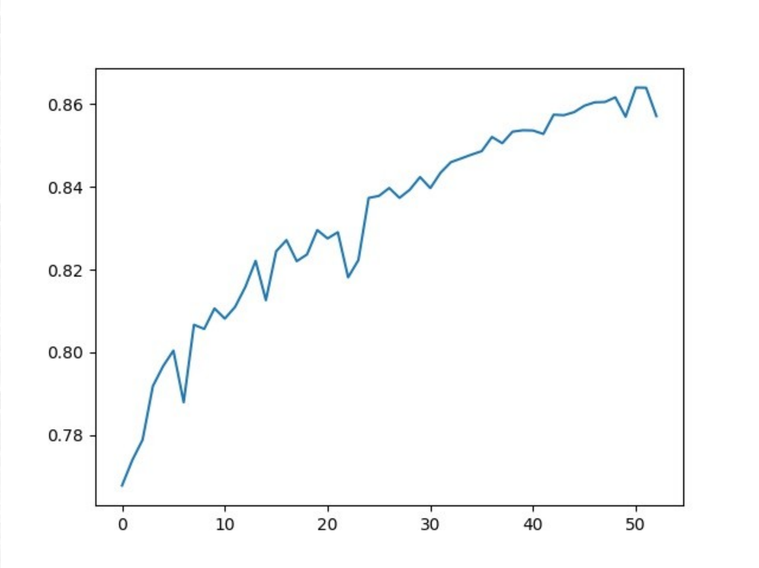
- So as of now DRLN gives the best result on ICDAR15 dataset. (PSNR reaches 32 while it reaches 30 in cascaded DBPN)
- In the above example we see that the predicted example with DRLN is more sharp than the one predicted with cascaded DBPN

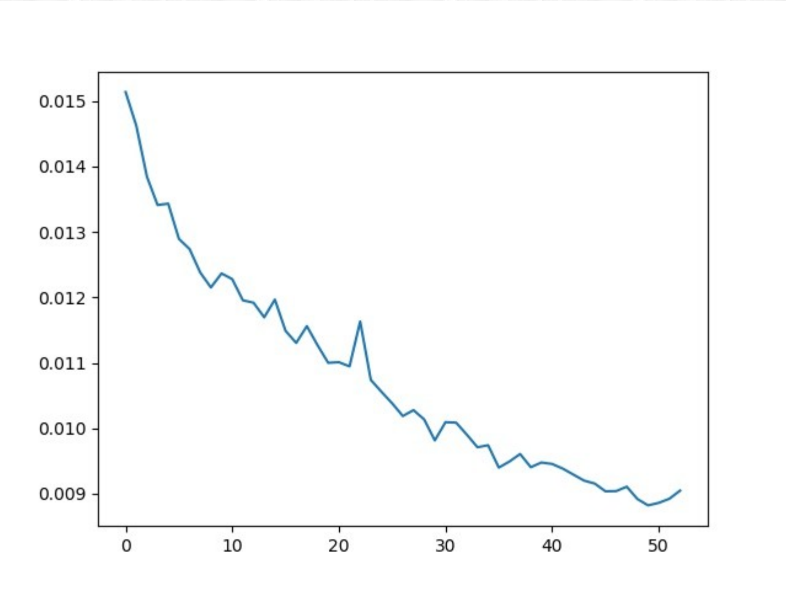
Walmart Data

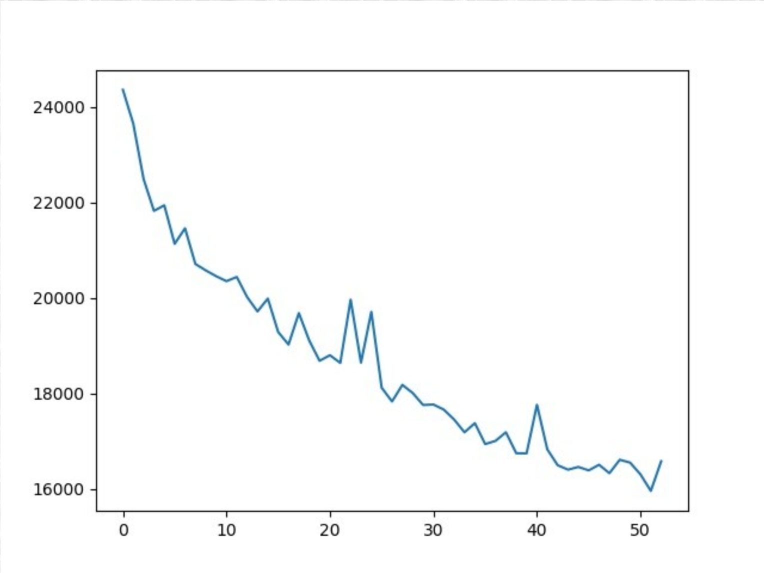
- I have augmented the dataset with different rotation, jitter, flip etc as the number of original images in the training set was less and images were more diverse so augmentation is a must to avoid overfitting.
- Even with the light weight network (DPSR) for high resolution images more than 2000 X 2000 size is giving CUDA out of memory error. I have tried with batch size=1. So, I will try training excluding the ones with large sizes, but again that will reduce the number of images to 50%.
- I am already working on creating a new dataset of product package images with reasonable size.

DRLN on Walmart Data Result







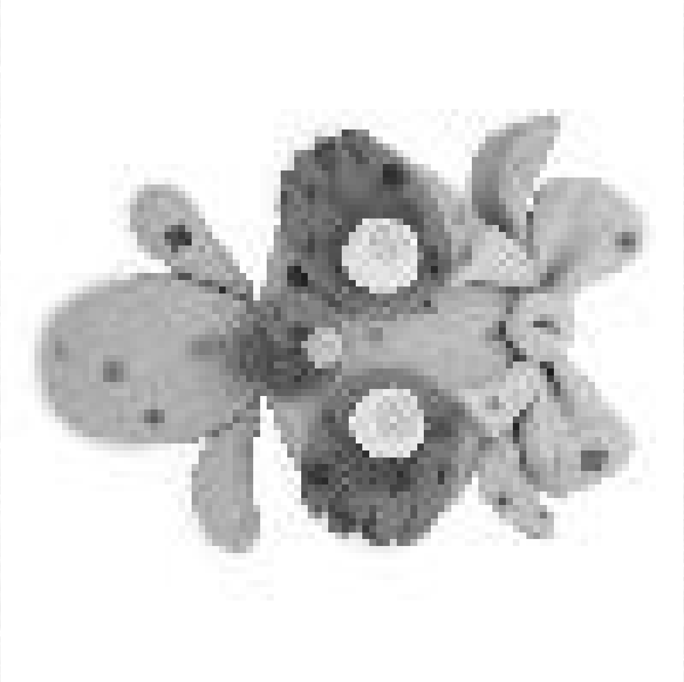


Low Res image

BiCubic interpi

Original image

Predicte d image



Low Res image

BiCubic interpi

Original image

Predicte d image

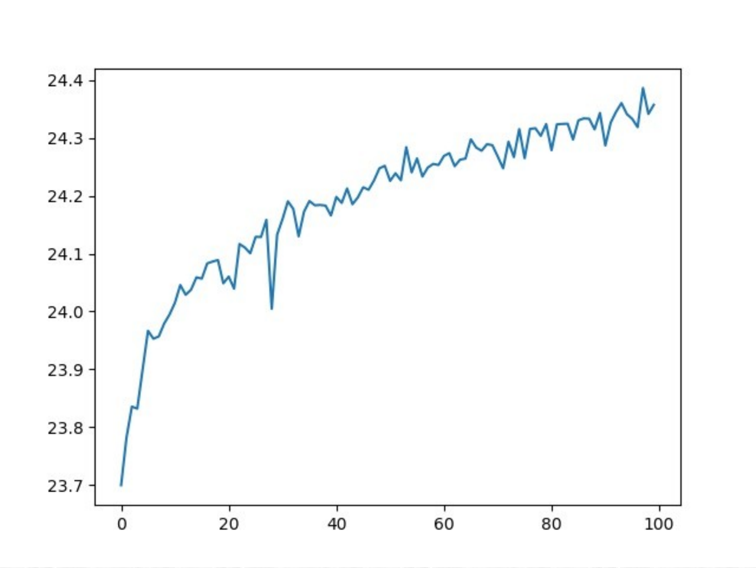
Low Res image

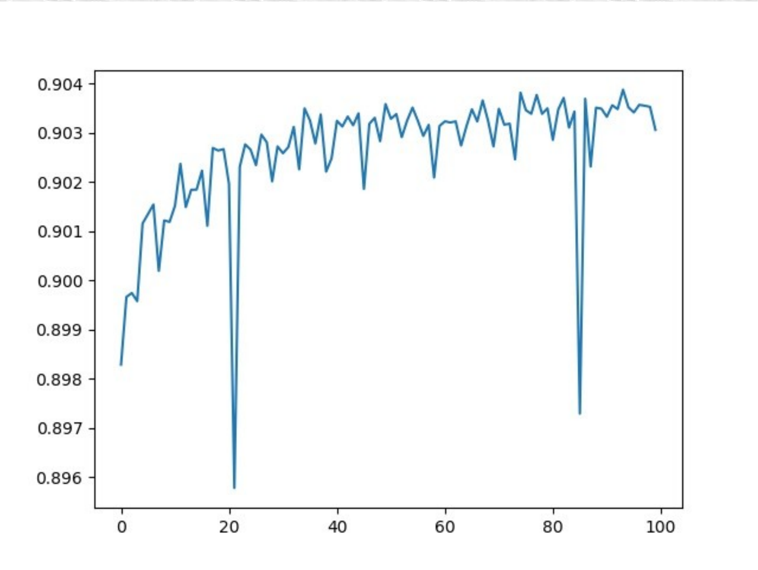
BiCubic interpi

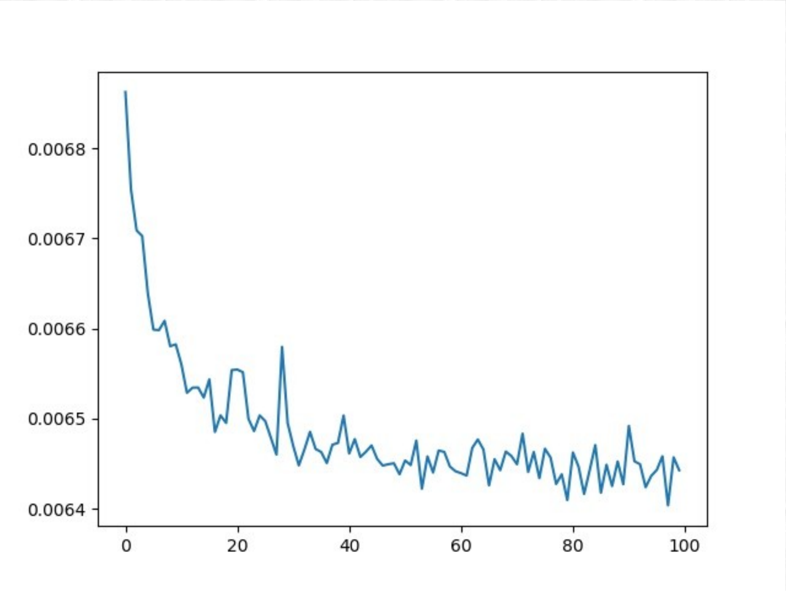
Original image

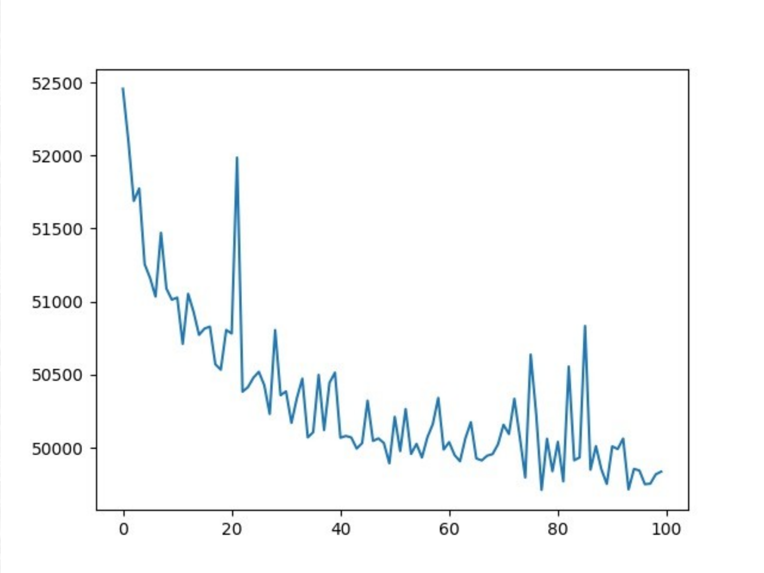
Predicte d image

DRLN next 100 iterations on Walmart Data Result



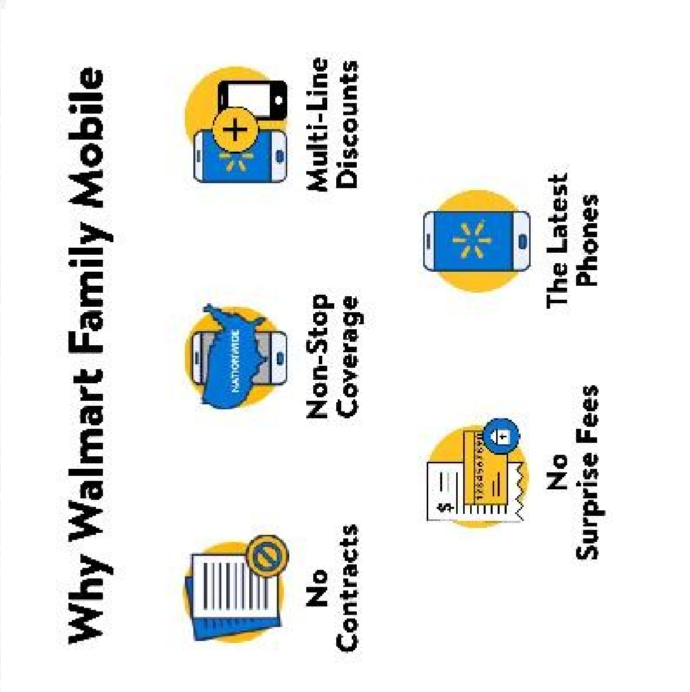






Low Res image

BiCubic interpi



Original image

Predicte d image

Low Res image

BiCubic interpi

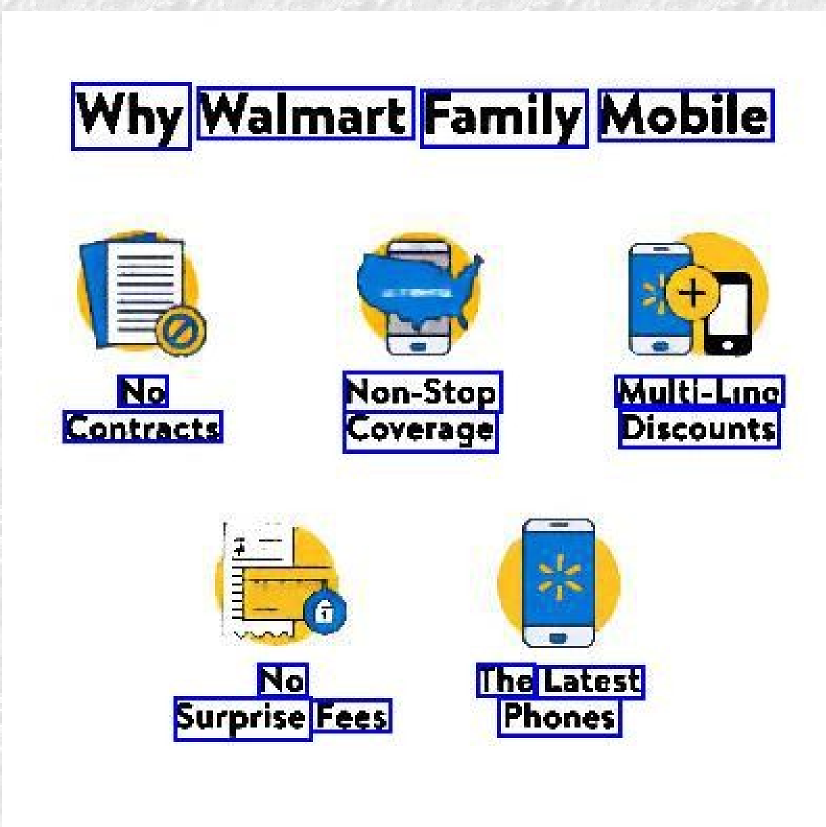
Original image

Predicte d image

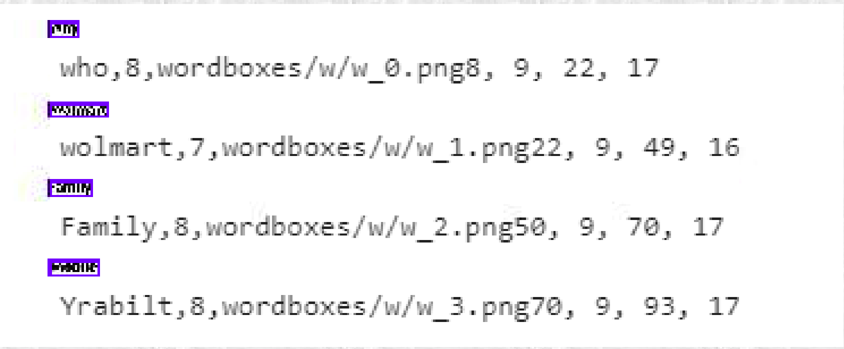

EWOK
EWOK

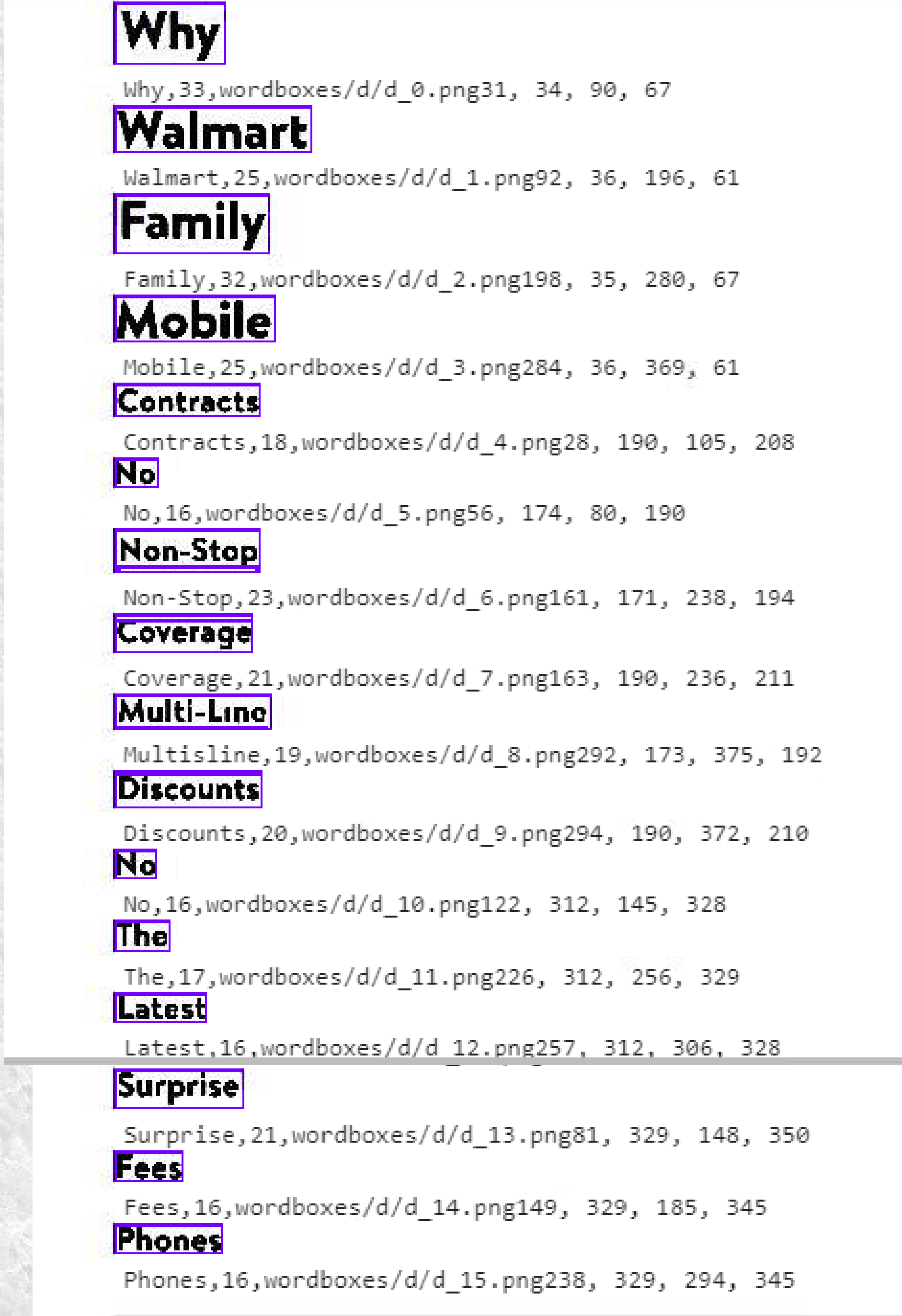

EWOK
EWOK

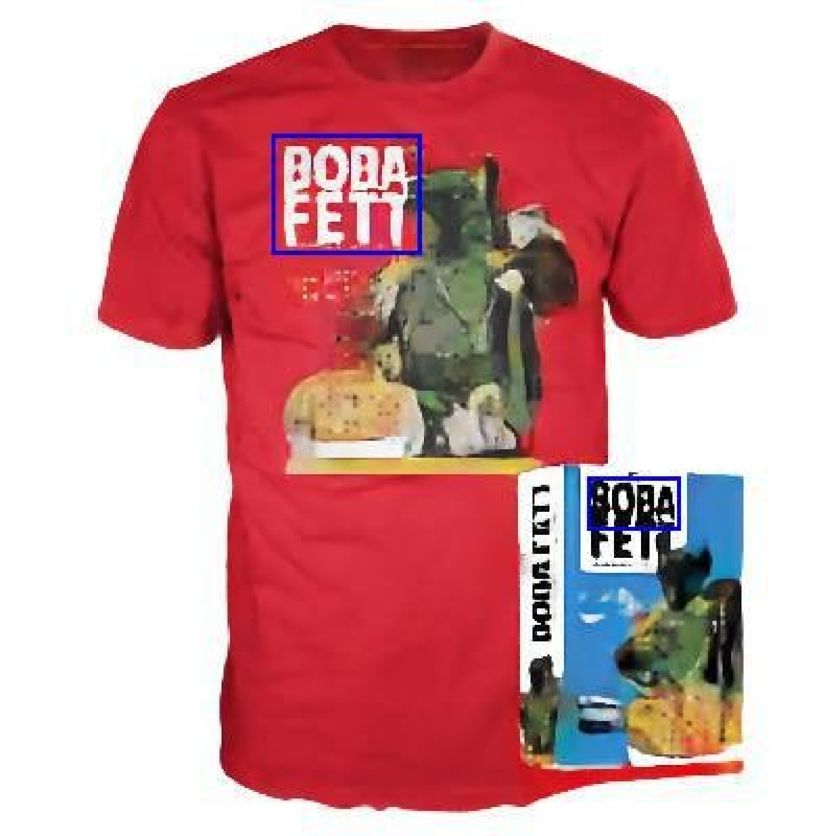

Mobile


Why Walmart Family Mobile
No Contracts
Non-Stop Coverage
Multi-Line Discounts
No Surprise Fees
The Latest Phones

Text recognized from rotated image





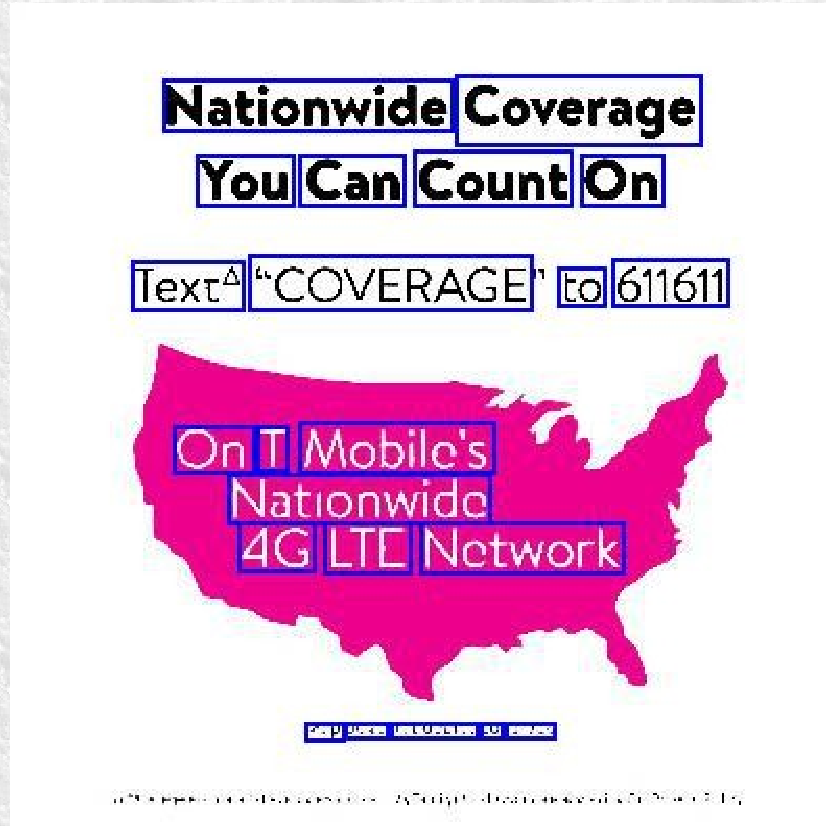


BOBA
FETT

Text recognized

Heft,16,wordboxes/w/w_0.png30, 16, 53, 32

PORA,57,wordboxes/d/d_0.png130, 64, 201, 121

BOBA,25,wordboxes/d/d_1.png281, 228, 325, 253


Nationwide Coverage
You Can Count On
Text∆ "COVERAGE" to 611611
On T Mobile's
Nationwide
4G LTE Network

Text Recognized

```
Natiamakic,8,wordboxes/w/w_0.png17, 9, 53, 17
You,7,wordboxes/w/w_1.png22, 18, 34, 25
want,7,wordboxes/w/w_2.png30, 30, 64, 37
Concountion,8,wordboxes/w/w_3.png34, 17, 78, 25
Comcrage,8,wordboxes/w/w_4.png54, 9, 84, 17
wener,7,wordboxes/w/w_5.png65, 30, 86, 37
```

```
Nationwide,25,wordboxes/d/d_0.png73, 36, 212, 61
Coverage,34,wordboxes/d/d_1.png213, 34, 331, 68
You,25,wordboxes/d/d_2.png89, 72, 136, 97
Can,25,wordboxes/d/d_3.png138, 72, 189, 97
Count,27,wordboxes/d/d_4.png193, 70, 269, 97
On,25,wordboxes/d/d_5.png273, 72, 313, 97
Texta,24,wordboxes/d/d_6.png58, 123, 112, 147
#COVERAGE,27,wordboxes/d/d_7.png114, 120, 250, 147
to,19,wordboxes/d/d_8.png262, 126, 285, 145
611611,23,wordboxes/d/d_9.png288, 122, 344, 145
On,23,wordboxes/d/d_10.png78, 202, 117, 225
```

```
T,22,wordboxes/d/d_11.png118, 202, 133, 224
Mobilels,25,wordboxes/d/d_12.png138, 200, 232, 225
Nationwide,24,wordboxes/d/d_13.png104, 225, 230, 249
4G,24,wordboxes/d/d_14.png109, 248, 146, 272
LTO,25,wordboxes/d/d_15.png150, 248, 192, 273
Network,25,wordboxes/d/d_16.png196, 248, 294, 273
LEP,9,wordboxes/d/d_17.png141, 344, 160, 353
out,8,wordboxes/d/d_18.png160, 344, 181, 352
INTOREITE,8,wordboxes/d/d_19.png182, 344, 224, 352
co,8,wordboxes/d/d_20.png225, 344, 236, 352
the,8,wordboxes/d/d_21.png237, 344, 261, 352
```

Future Work

- I am creating a new dataset which will contain to the product images only, the trained model on the data will give better result on test images as the ultimate goal is to extract small font from images like product package image and so on.

Thank you