# EX3 Super Resolution & Image Colorization

#### Asaf Liberman 313230187

#### Nirel Yehoyada 205386675

#### Niv Zatelman 314764812

**Dataset:**

The dataset includes 8050 images of flowers:

50 of them used for test, 7200 for training and 800 for validation.

**1st Try – using RGB:**

After following few articles we decided the use the following model as our base line (input is 32X32):

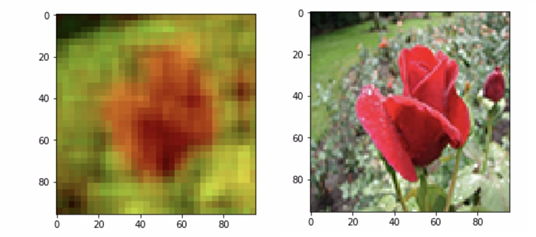
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Layer # | Type | Amount | Shape | Activation | Stride |
| 1 | Conv2D | 64 | (3,3) | Relu |  |
| 2 | Conv2D | 64 | (3,3) | Relu | 2 |
| 3 | Conv2D | 128 | (3,3) | Relu |  |
| 4 | Conv2D | 128 | (3,3) | Relu | 2 |
| 5 | Conv2D | 256 | (3,3) | Relu |  |
| 6 | Conv2D | 256 | (3,3) | Relu | 2 |
| 7 | Conv2D | 512 | (3,3) | Relu |  |
| 8 | Conv2D | 256 | (3,3) | Relu |  |
| 9 | Conv2D | 128 | (3,3) | Relu |  |
| 10 | UpSampling2D | 1 | **(2,2)** |  |  |
| 11 | Conv2D | 64 | (3,3) | Relu |  |
| 12 | UpSampling2D | 1 | **(2,2)** |  |  |
| 13 | Conv2D | 32 | (3,3) | Relu |  |
| 14 | Conv2D | 2 | (3,3) | tanh |  |
| 15 | UpSampling2D | 1 | **(2,2)** |  |  |
| 16 | Conv2D | 3 | (3,3) | Relu |  |

We used adam as our optimizer and mean\_squared\_error as our loss.

Explanation – we apply filters to the image while we downsize it using strides of 2– it cuts the height and the width of the image by 2 and make it sharper.

After that we try to add as the 17th layer, another Upsampling2D of (3,3), to make the image 96x96 .

We trainned the network for few hours, we got this output for an image from out training set:



As we can see the Upsampling in the 17th layer didn’t work out pretty well.. The predict was very blur.

After searching the web for a bit, we understood that except for the colorization of the image, we need to build a super resolution model in order the make the image look **sharper**, and we must use LAB instead of RGB when coloring it.



LAB RGB

When trying using the RGB color scheme we lose all the data we have about the B&W image,  
but when we use the LAB color scheme (**L** stands for lightness, **a** and **b** for the color spectra green–red and blue–yellow) we can save the lightness channel while predicting only the a and b channels, this helped us for keeping the photo sharp and not losing data.

While there are many options out there, we decided to go for the following architecture:

**Resize the image to 96x96 -> Super Resolution -> Colorization**

That means we are going to feed the Colorization network with the output of the Super Resolution model. The colorization and the super resolution models will be trained separately.

**Super Resolution Model**

Input: array of images (each one 32x32)

Output: prediction of the input (each one 96x96)

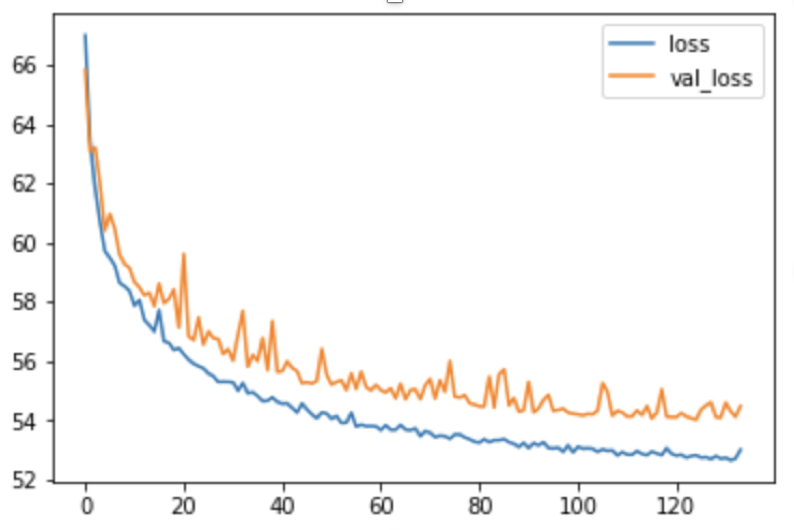
Optimizer: adam with 0.001 LR

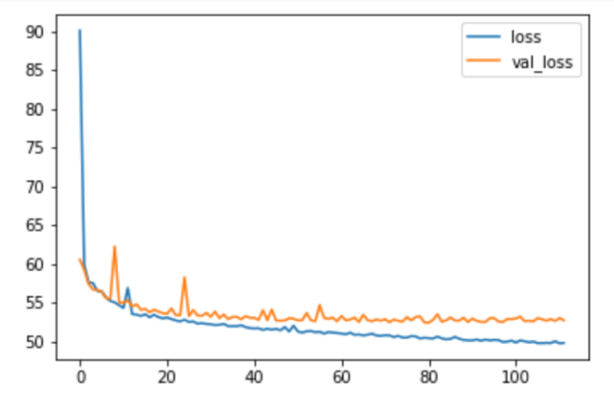
batch\_size = 50

Loss function: mean\_squared\_error

At first we started with the following model:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer # | Type | Amount | Shape | Activation |
| 1 | UpSampling2D | 1 | (3,3) |  |
| 2 | Conv2D | 64 | (6,6) | Relu |
| 3 | Conv2D | 128 | (3,3) | Relu |
| 4 | Conv2D | 256 | (6,6) | Relu |
| 5 | Conv2D | 1 | (4,4) | Relu |

The results were:

Then we applied different filter sizes and shape and got a slightly better result, so we went for this model.

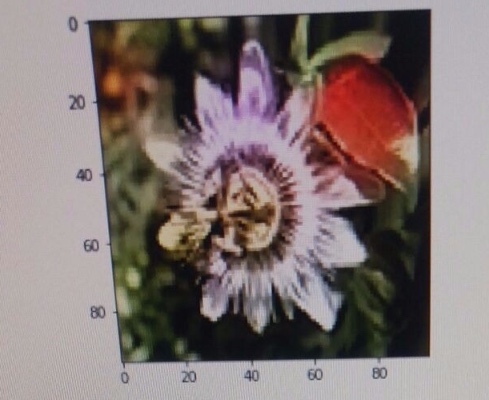
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer # | Type | Amount | Shape | Activation |
| 1 | UpSampling2D | 1 | (3,3) |  |
| 2 | Conv2D | 128 | (3,3) | Relu |
| 3 | Conv2D | 64 | (3,3) | Relu |
| 4 | Conv2D | 32 | (3,3) | Relu |
| 5 | Conv2D | 16 | (3,3) | Relu |
| 6 | Conv2D | 1 |  | linear |

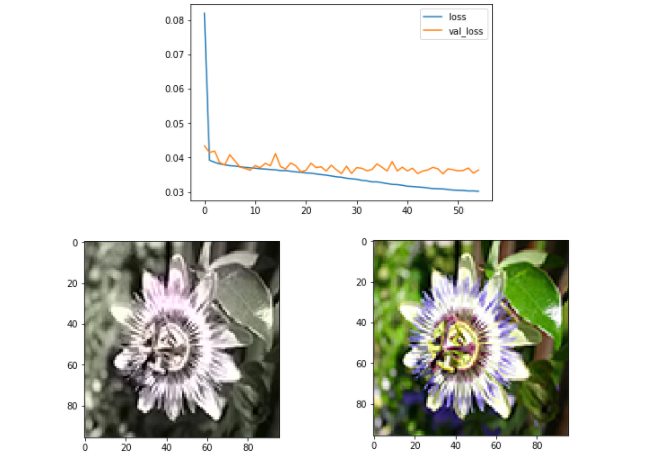
Results example (from test set):

****

**Left – original image, Middle – super resolution, Right – naïve resize**

**Colorization Model**

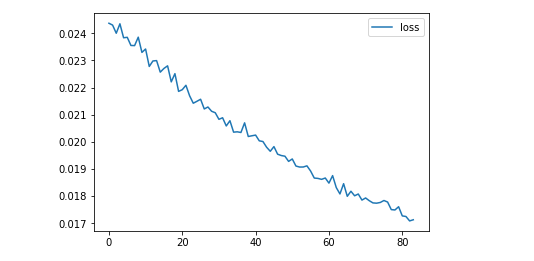
Now we decided to train our colorization model with 96x96 input. We noticed a problem of overfitting and not giving the correct colors for images that are not in our training set. Example for that can be seen here:

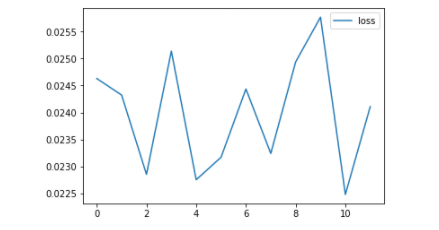
So after we encountered some overfitting problems so we tried to add some dropout layers. The results are below:

As we can see, dropout layers didn’t really help with fixing the overfitting, and our image lacks a lot of colors... We had to find another solution!

After reading few articles, we understood we can use an existing feature extraction models (such as InceptionV3) and use them in addition to our network in order to get better results, and it worked!

We also noticed that using adam as optimizer is more stable than rmsprop- both used very small LR – 0.0005

**Adam:**

**Rmsprop:**

**At the end we choose adam as out optimizer with 0.00005 LR.**

**Mean squared error as our loss**

**Batch size of 100**

**The final model is defined as that:**

****

And finally, an example for the whole process can be seen here - left and right photos are the originals.

****