CENG 483

Introduction to Computer Vision

Spring 2018-2019

Take Home Exam 3

Image Colorization

Student Random ID: 70

Please fill in the sections below only with the requested information. If you have additional things to mention, you can use the last section. Please note that all of the results in this report should be given for the **validation set** by default, unless otherwise specified. Also, when you are expected to comment on the effect of a parameter, please make sure to **fix** other parameters. You may support your comments with visuals (i.e. loss plot).

1 Baseline Architecture (30 pts)

Based on your qualitative results (do not forget to give them),

- Discuss effect of the number of conv layers:

  I have fixed the hyperparameters other than number of convolution layers such that kernel size is **3** except the last convolution layer, number of kernels is **2** and learning rate is **0.1**.

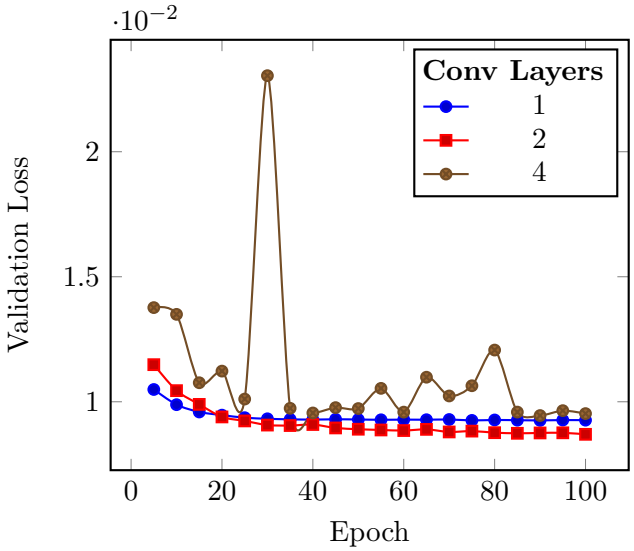

Figure 1: Validation loss of convolutional neural networks with different number of convolutional layers in the network over 100 epoch

Convolutional Layers	Validation Loss
1	0.009255
2	0.008698
4	0.009449

Table 1: The lowest validation losses of convolutional neural networks achieved with different number of convolutional layers in the network

For the effect of the number of convolutional layers, the training process has achieved minimum validation loss with **2 convolutional layers** where kernel size is 3, number of kernels is 2 and learning rate 0.1 when maximum number of epoch is 100.
Minimum validation loss, 0.008698, is achieved in 100^{th} epoch. There is a chance that model would be better if I had trained it more. Architecture with 2 convolutional layers seems better than having 1 or 4 layers for the fixed hyperparameters with the given dataset. I do think that 4 convolutional layers might be better if the model is trained with more epochs. The reason that 4 layers has achieved worse result than 2 layers might be insufficient epochs as it has more weights to optimize.

- Discuss effect of the kernel size(except the last conv layer):

  I have fixed the hyperparameters other than kernel size such that the number of convolutional layer is **2**, the number of kernels is **2** and learning rate is **0.1**.

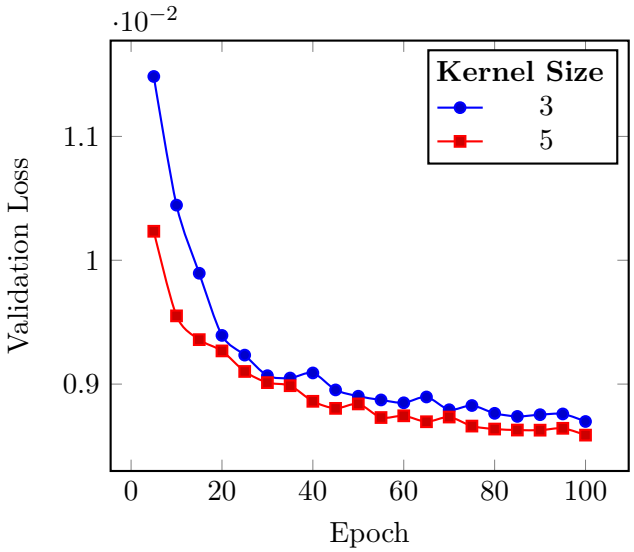

Figure 2: Validation loss of convolutional neural network with different kernel sizes over 100 epoch

Kernel Size	Validation Loss
3	0.008698
5	0.008588

Table 2: The lowest validation losses of convolutional neural networks achieved with different kernel sizes

For the effect of the number of the kernel size, the training process has achieved minimum validation loss with **kernel size 5** where number of conv layers is 2, number of kernels is 2 and learning rate 0.1 when maximum number of epoch is 100.
Minimum validation loss, 0.008588, is achieved in 100^{th} epoch. Again, there is a chance that model would achieve lesser validation loss if I had trained it more. Having 5x5 kernels seems better than having 3x3 kernels for the fixed hyperparameters with the given dataset. Lesser validation loss might be achieved with the combination of different kernel sizes.

- Discuss effect of the number of kernels(except the last conv layer):

  I have fixed the hyperparameters other than the number of kernels (except the last convolutional layer) such that the number of convolutional layer is **2**, kernel size is **5** and learning rate is **0.1**.

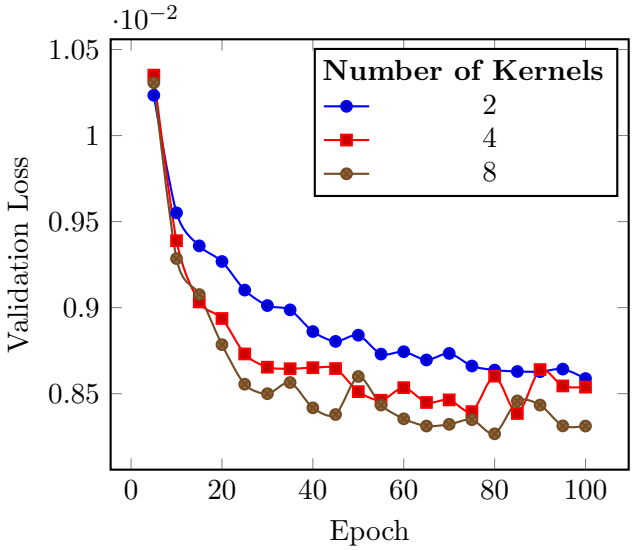

Figure 3: Validation loss of convolutional neural networks with different number of kernels over 100 epoch

The Number of Kernels	Validation Loss
2	0.008588
4	0.008385
8	0.008267

Table 3: The lowest validation losses of convolutional neural networks achieved with different number of kernels

For the effect of the number of kernels, the training process has achieved minimum validation loss with **8 channel kernels** where number of conv layers is 2, kernel size is 5 and learning rate 0.1 when maximum number of epoch is 100.
Minimum validation loss, 0.008267, is achieved in 80^{th} epoch. There is a slight chance that model would achieve lesser validation loss if I had trained it more. Having more channel makes the model better for this task because the model extends its capabilities in colorization task.

- Discuss effect of the learning rate by choosing three values: a very large one, a very small one and a value of your choice:

  I have fixed the hyperparameters other than the learning rate such that the number of convolutional layer is **2**, kernel size is **5** and the number of kernels is **8**.

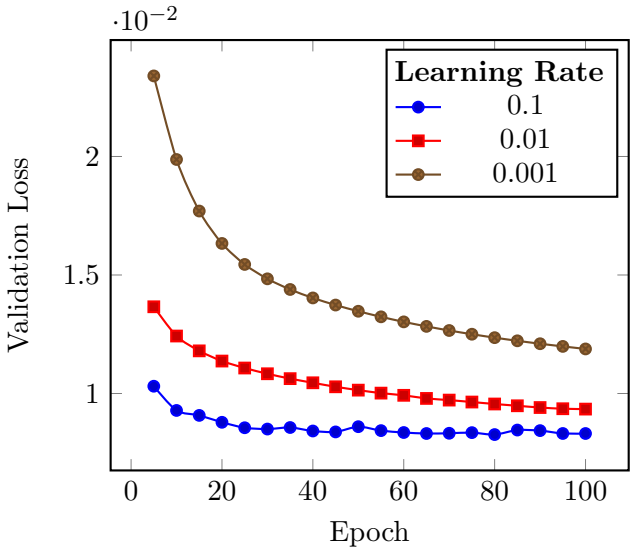

Figure 4: Validation loss of convolutional neural networks with different learning rates over 100 epoch

Learning Rate	Validation Loss
0.1	0.008267
0.01	0.009342
0.001	0.011880

Table 4: The lowest validation losses of convolutional neural networks achieved with different learning rates

For the effect of the learning rate, the training process has achieved minimum validation loss with **learning rate 0.1** where number of conv layers is 2, kernel size is 5 and number of kernels is 8 when maximum number of epoch is 100.
Minimum validation loss, 0.008267, is achieved in 80^{th} epoch. Again, there is a slight chance that model would achieve lesser validation loss if I had trained it more. Learning rate is related with maximum number of epochs because we can achieve similar results if a model with smaller learning rate is trained with bigger maximum number of epochs. However, bigger learning rate is not always good; for example, if a model with learning rate 0.9 or 1 is trained, it doesn't learn anything as its learning loss and validation loss is **NaN**.
From the validation losses above, it can be concluded that learning rate is crucial for training duration. Small learning rate extends the training duration as it takes much more time to optimize the weights to their desired values. Big learning rate makes the model untrainable as it prevents the model from optimization.

2 Further Experiments (20 pts)

I will do further experiments on the best models I have trained so far:

I. Conv Layers: **2**, Kernel Size: **5**, Kernels: **8**, Learning Rate: **0.1** $\Rightarrow$ Validation Loss: **0.008267**

II. Conv Layers: **2**, Kernel Size: **3**, Kernels: **8**, Learning Rate: **0.1** $\Rightarrow$ Validation Loss: **0.008289**

III. Conv Layers: **4**, Kernel Size: **5**, Kernels: **8**, Learning Rate: **0.1** $\Rightarrow$ Validation Loss: **0.008308**

IV. Conv Layers: **2**, Kernel Size: **5**, Kernels: **4**, Learning Rate: **0.1** $\Rightarrow$ Validation Loss: **0.008385**

- Try adding a batch-norm layer (torch.nn.BatchNorm2d) into each convolutional layer. How does it affect the results, and, why? Keep it if it is beneficial.

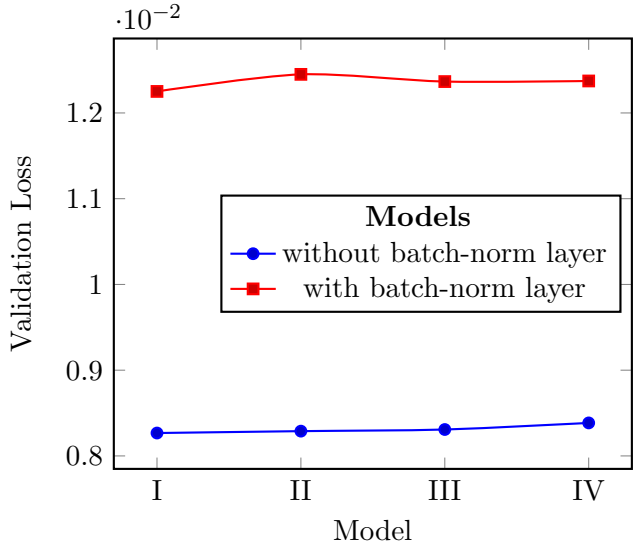

Figure 5: Comparison of the best models with and without batch normalization layer after every convolutional layer in terms of validation loss

I have added batch-norm layer after ReLU activation function. I've also tested adding batch-norm layer before ReLU activation function. Applying batch-norm layer after ReLU has achieved validation loss 0.012251 whereas applying it before ReLU has achieved validation loss 0.012328.
In terms of validation loss, adding batch-normalization layer seems to be unnecessary as it increases the validation loss. Our dataset is already in range $[-1, 1]$. Therefore, normalization might be unnecessary for the input layer as the input is already in normalized shape; however, batch-normalization layer might be useful for the following layers. The current models tend to output broken or sharp pixels due to lack of activation layer in the last layer. Batch-normalization layer can reduce the number of those pixels. Yet, it might not be sufficient for that problem.

- Try adding a tanh activation function after the very last convolutional layer. How does it affect the results, and, why? Keep it if it is beneficial.

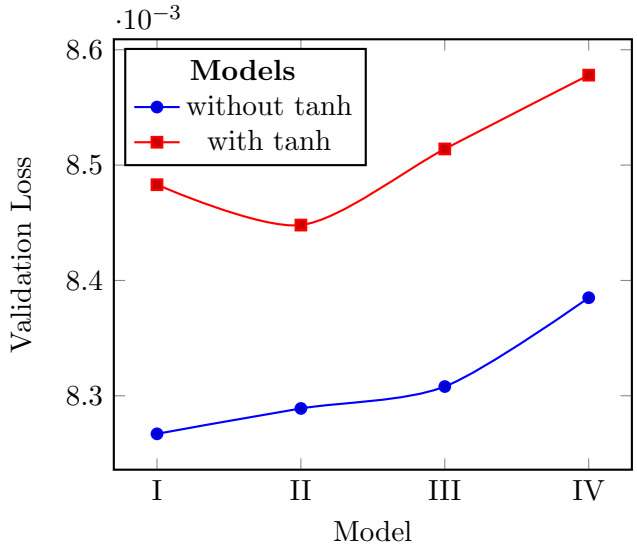

Figure 6: Comparison of the best models with and without tanh activation function after the very last convolutional layer in terms of validation loss

It doesn't affect the models in a good way as the models' validation losses are not decreased. However, it improves the outputs of models even though it doesn't reduce the validation losses. There is nothing after the last layer, so our model technically doesn't map $[-1, 1]^2$ to $[-1, 1]^2$ because the last layer has no activation function. Therefore, current models do $[-1, 1]^2 \to [-\infty, \infty]^2$. Due to lack of activation function after the last layer, the models' outputs have broken or sharp pixels. tanh activation function prevents this from happening.
I used mean square error (MSE) as loss function when training the models. It might be the reason for broken/sharp pixels in the outputs. If I used 12-margin error as loss function, broken/sharp pixel problem might be fixed.

- Try setting the number of channels parameter to 16. How does it affect the results, and, why? Keep it if it is beneficial.

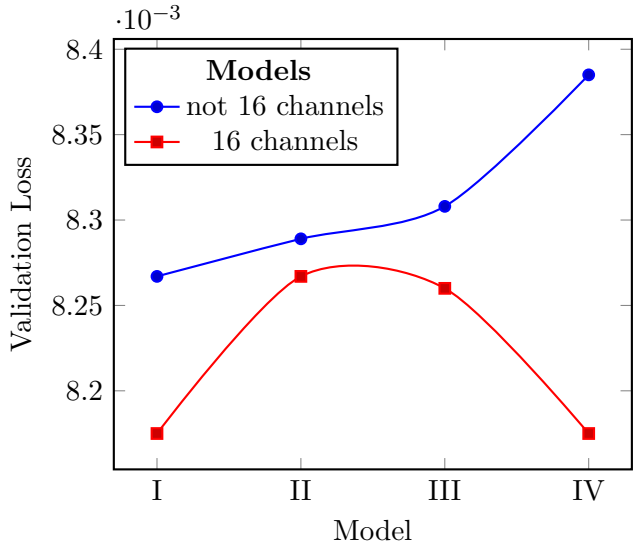

Figure 7: Comparison of the best models and their 16 channel versions in terms of validation loss

16 channel counterpart of model I and IV is same because rest of the hyperparameters are identical. In terms of validation loss, having 16 channel kernels rather than 2, 4 or 8 channels is better because having more channels means new capabilities, more feature extraction/learning. Therefore, models with 16 channels have had lesser validation loss than their counterparts.

- Broken/Sharp pixel problem in outputs and above experiments:

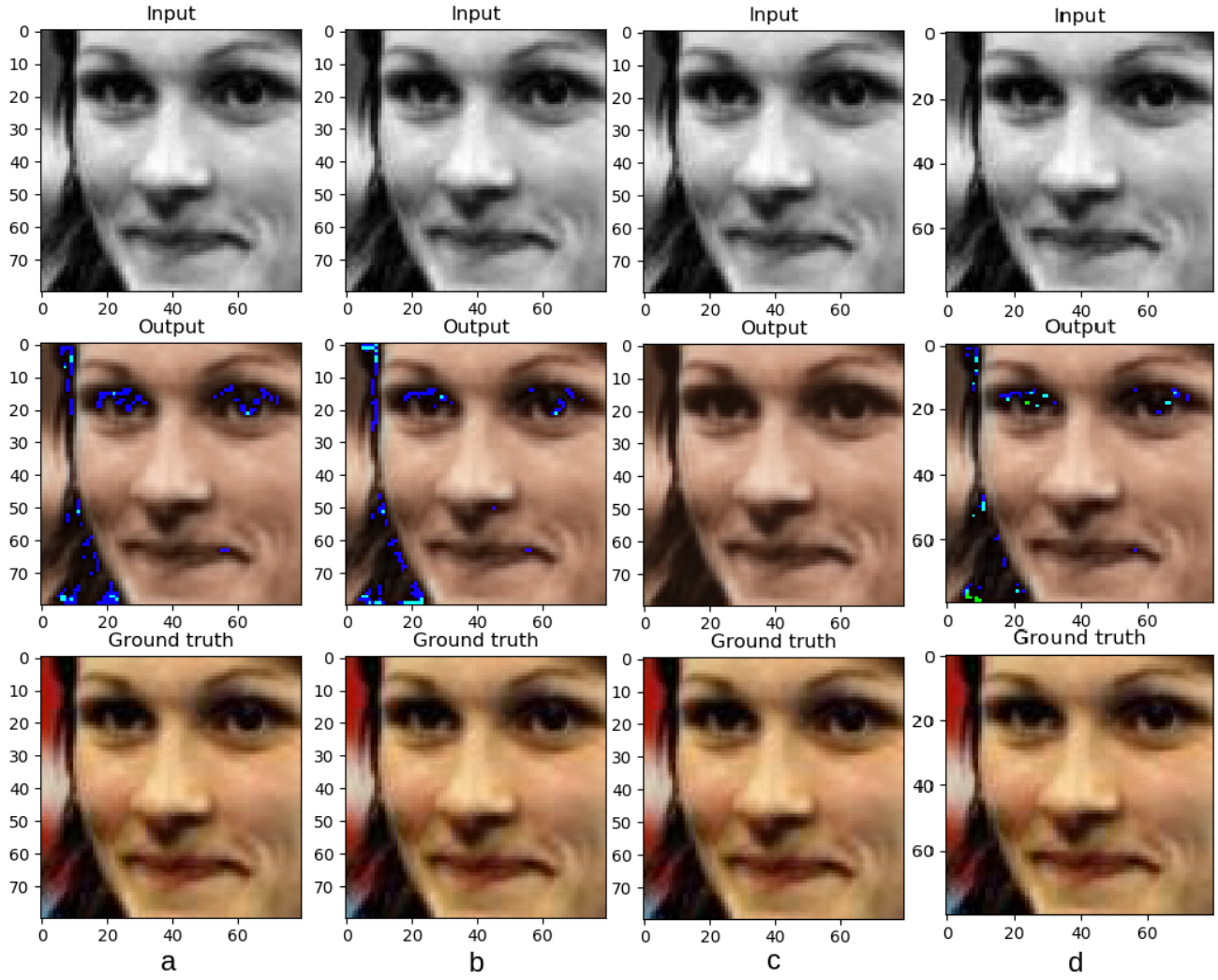

Figure 8: Outputs from different models with same input: **(a)** Model I, **(b)** Model I with Batch-Normalization, **(c)** Model I with tanh activation fuction, **(d)** Model I with 16 channels instead of 8 channels

3 Your Best Configuration (20 pts)

My best configuration is **2 convolutional layers**, **kernel size 5**, **16 channels in kernels**, **learning rate 0.1** with **tanh** activation function after the last convolutional layer.

- The automatically chosen number of epochs(what was your strategy?):

  I have ended the training process if the latest validation loss is greater than the three times of the minimum validation loss. I have also saved the model with minimum validation losses. Therefore, I will have the best trained model saved even though the training process is finished due to extreme increase in validation loss. In this way, training phase is terminated when the model is overfitting. However, I haven't came upon termination of the training. Maybe, thats reason is that I do get the validation loss at every 5 epoch in order to fasten the training process.

- The plot of the training mean-squared error loss over epochs:

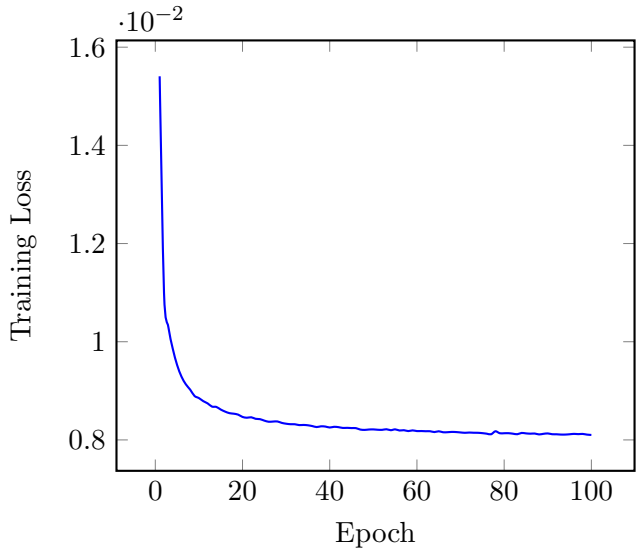

Figure 9: Mean-Squared Error Loss of Training Data over 100 Epochs

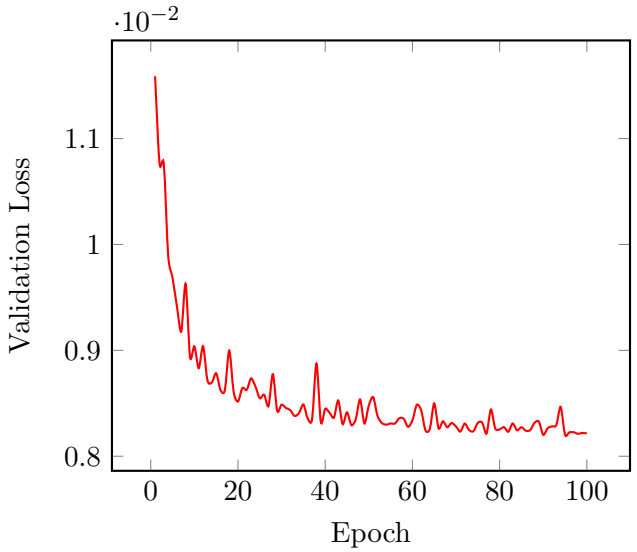

Figure 10: Mean-Squared Error Loss of Validation Data over 100 Epochs

- The plot of the validation 12-margin error over epochs:

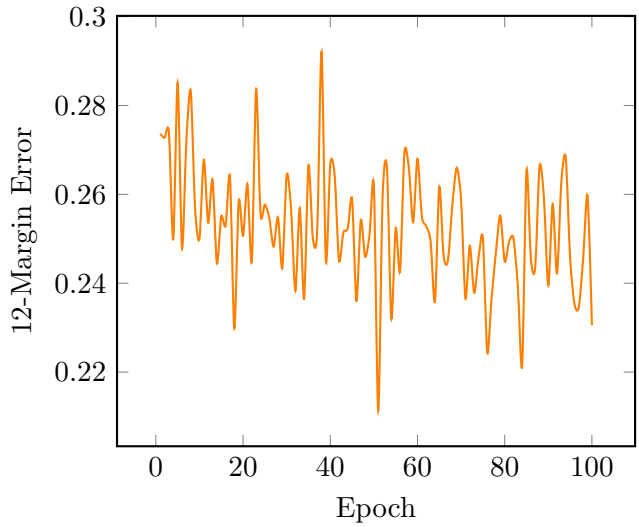

Figure 11: 12-Margin Error Loss of Validation Data over 100 Epochs

- At least 5 qualitative results on the validation set, showing the prediction and the target colored image:

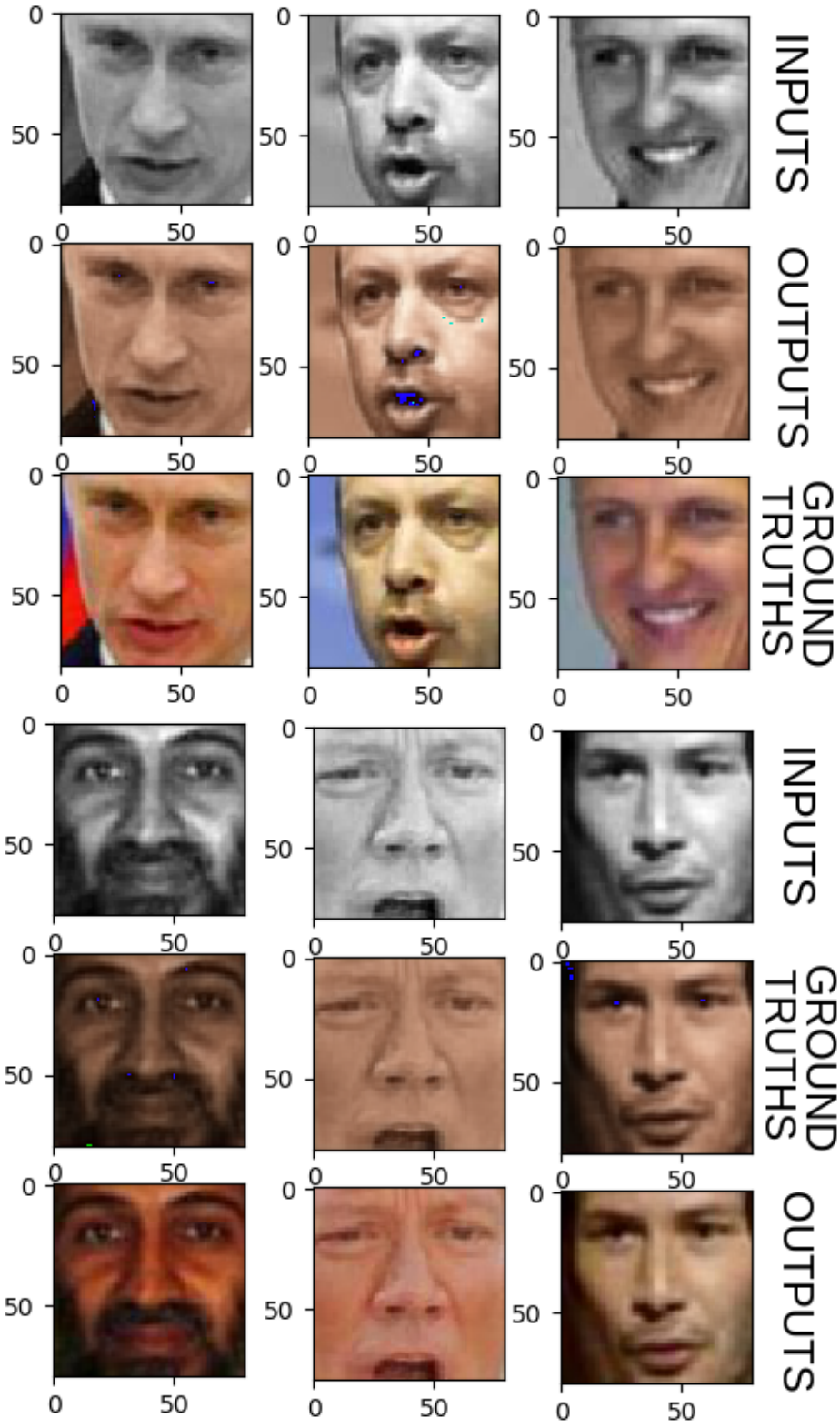

Figure 12: 6 qualitative results from the validation set

- Discuss the advantages and disadvantages of the model, based on your qualitative results, and, briefly discuss potential ways to improve the model:

  The model currently achieves **0.008201 validation loss** and **0.211376 12-margin error** in the validation set. The model has still sharp pixels as it can be seen from the outputs in ***Figure 12***. Those are caused from the output of last convolutional layer whose value is greater than 1 or smaller than -1. It can be fixed with simple value cropping or tanh activation function after the last convolutional layer. However, the latter solution isn't preferable as it decreases the loss and 12-margin error values.
  The model can be improved even more with increasing the number of kernel channels, better optimizers such as *adam* or optimizing the parameters of SGD optimizer. Introducing new layers might also improve the model if it is used with combinations of different sized kernels. However, adding new layers hasn't improved the model in this homework.
  The model can also be trained with more epochs and suitable learning rate with proper decay rate.

4 Your Results on the Test Set(30 pts)

This part will be obtained by us using the estimations you will provide. Please tell us how should we run your code in case of a problem:

5 Additional Comments and References

(if there any)