Dogan Aykas
6227725

ACML: Assignment 2

General Information

For the assignment, I have chosen tensorflow to design and train the convolutional autoencoders. The jupyter notebook file and its pdf is also available in the zip folder.

Exercise 1.2

Below is the training progress of the suggested model in the assignment. The test error is also indicated at the last row.

```
Epoch 1/10
44444/44444 [==============================] - 59s 1ms/step - loss:
0.0130 - val_loss: 0.0067
Epoch 2/10
44444/44444 [==============================] - 63s 1ms/step - loss:
0.0061 - val_loss: 0.0056
Epoch 3/10
44444/44444 [==============================] - 60s 1ms/step - loss:
0.0054 - val_loss: 0.0051
Epoch 4/10
44444/44444 [==============================] - 61s 1ms/step - loss:
0.0050 - val_loss: 0.0048
Epoch 5/10
44444/44444 [==============================] - 57s 1ms/step - loss:
0.0047 - val_loss: 0.0046
Epoch 6/10
44444/44444 [==============================] - 64s 1ms/step - loss:
0.0045 - val_loss: 0.0044
Epoch 7/10
44444/44444 [==============================] - 60s 1ms/step - loss:
0.0044 - val_loss: 0.0043
Epoch 8/10
44444/44444 [==============================] - 56s 1ms/step - loss:
0.0042 - val_loss: 0.0042
Epoch 9/10
44444/44444 [==============================] - 56s 1ms/step - loss:
0.0041 - val_loss: 0.0041
Epoch 10/10
44444/44444 [==============================] - 59s 1ms/step - loss:
0.0040 - val_loss: 0.0039
```

```
10000/10000 [==============================] - 3s 286us/step

0.00395868329256773
```

Dogan Aykas
6227725

Exercise 2.1

Below are the parameters for the latent space for the default configuration

W = 8

K = 3

P = 1

S = 1

C = 16

s3 = C*((((W-K+2*P)/S) + 1)**2) = 1024

Thus, the size of the latent space representation is **1024.**

Exercise 2.2

The detailed experimentation of different configurations is given in the jupyter notebook file and pdf. Below is a summary of errors after 1 epoch trainings of different configurations together with the size of the latent space of the CAE configuration.

Explanation	W	K	P	S	C	Latent space	Training error
Fewer intermediate layers	16	3	1	1	12	3072	0.0059
Different number of channels	8	3	1	1	128	8192	0.0054
Different filter sizes	8	5	1	1	16	576	0.0118

During my experimentation, I have found that the greater the size of the latent space, the less error the model produced, thus better accuracy. This makes intuitive sense since the size of the latent space is related to the encoding of the information embedded in the inputs. The higher the size of the latent space, less information is lost during encoding which results in less error. However, since the aim of autoencoders is related with representing information with less dimensions to extract some insightful

information, it is not always helpful to keep the latent space as big as possible. For instance, if the size of the latent space is virtually the same as the inputs, the model does not have much to learn, in such cases it can mimic the identity function which is not something useful.

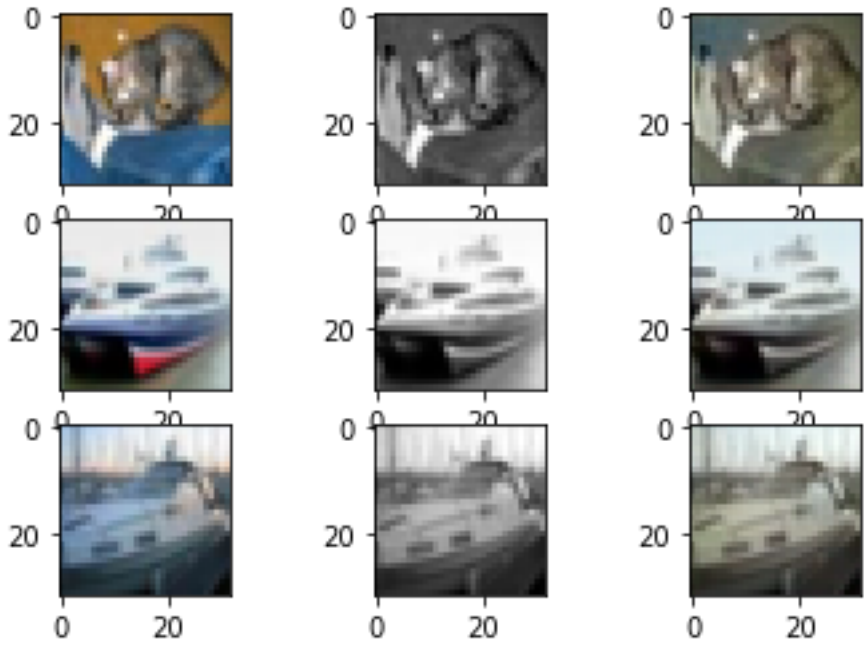
For the colorization part, I have decided to increase the number of channels per convolutional layer since this was the most effective way to increase the latent space without violating the architecture and aim of CAEs.

Exercise 3.1

For the colorization part, I used the YCrCb representation for the model's inputs and outputs. The reason for that is in the YCrCb representation, the Y part is related to the luminance and the remaining two channels are the color differences with respect to red and blue colors. This was a more practical representation than feeding a grayscale image and expecting an RGB representation which includes 3 channels in the output. Thus, using chrominance was useful since using the YCrCb representation, the luminance channel is fed into the model as inputs and only 2 color channels are predicted. In the end, for visualization purposes, the predicted channels and the initial luminance of the objects are concatenated and converted back to RGB representation for representational purposes. Below is the architecture of the model I have used for this colorization part, together with some image results.

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_80 (Conv2D)           (None, 32, 32, 16)        160
_________________________________________________________________
max_pooling2d_32 (MaxPooling (None, 16, 16, 16)        0
_________________________________________________________________
conv2d_81 (Conv2D)           (None, 16, 16, 32)        4640
_________________________________________________________________
max_pooling2d_33 (MaxPooling (None, 8, 8, 32)          0
_________________________________________________________________
conv2d_82 (Conv2D)           (None, 8, 8, 64)          18496
_________________________________________________________________
up_sampling2d_32 (UpSampling (None, 16, 16, 64)        0
_________________________________________________________________
conv2d_83 (Conv2D)           (None, 16, 16, 32)        18464
_________________________________________________________________
up_sampling2d_33 (UpSampling (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_84 (Conv2D)           (None, 32, 32, 2)         578
=================================================================
Total params: 42,338
Trainable params: 42,338
Non-trainable params: 0
```

Dogan Aykas
6227725



Exercise 3.2

For the potential shortcomings, I wanted to mention that I have used max-min normalization which normalized the pixel values between 0 and 1. I have also implemented this normalization for the outputs of the network, which I am not sure if it was necessary or not. Besides the hyperparameters, I could not really check out different stride and padding configurations computationally which may have brought some more accurate results. Furthermore, as it is indicated in the images above, the sharpness of the colors is somehow lost in the model which is also something expected, but could have been improved.