**Assignment 4 - Course Project**

Sebastian Scholl 11827056

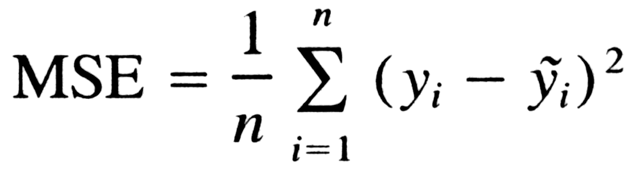
**TODO:** insert other name and matricle numbers

**1) Introduction**

Our group chose topic 1, Correcting Images with Autoencoders. The task is to train an autoencoder that is capable of correcting and re-generating clean images from their distorted versions. As data set we chose Kuzushiji-MNIST. To build a perturbed data set we created our own functions, which modify the data with gaussian noise, rotate or flip the images, add black patches and change the brightness of the original images.

**2) Methods**

The network we build is an autoencoder, therefore we use mean squared error as loss function:

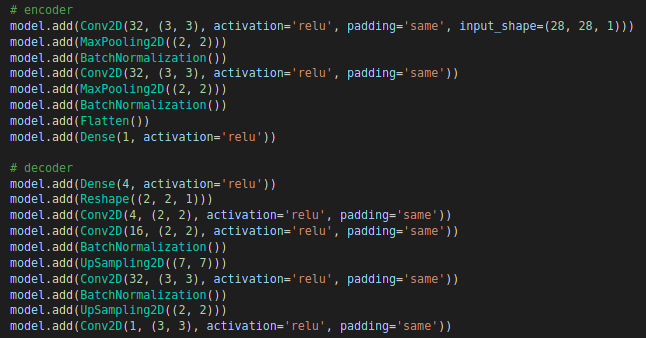


Autoencoder are built as a encoder-decoder architecture, the encoder receives an image, reduces the dimensions and compresses the input data. The decoder then reconstructs the original image from this compressed data.

**3) Results**

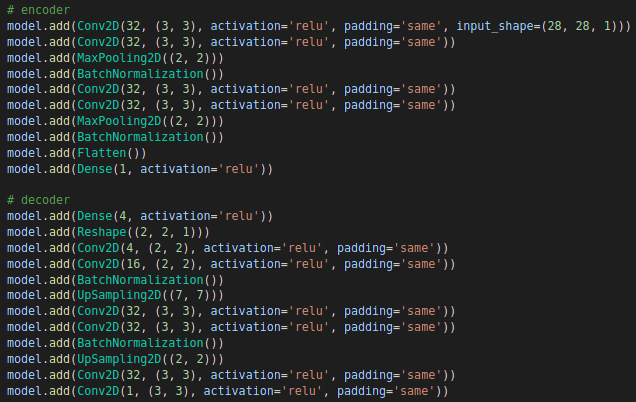
Model 0:

The initial model is a not very deep standard encoder decoder architecture with two convolutions in the encoder and batch normalizations.



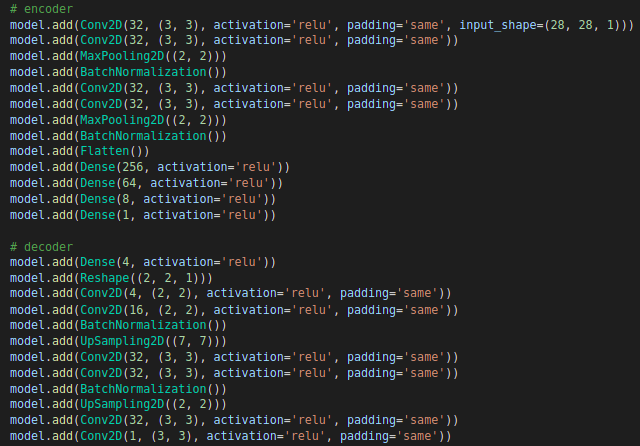
Model 1:

Model 0 with more convolutional layers.



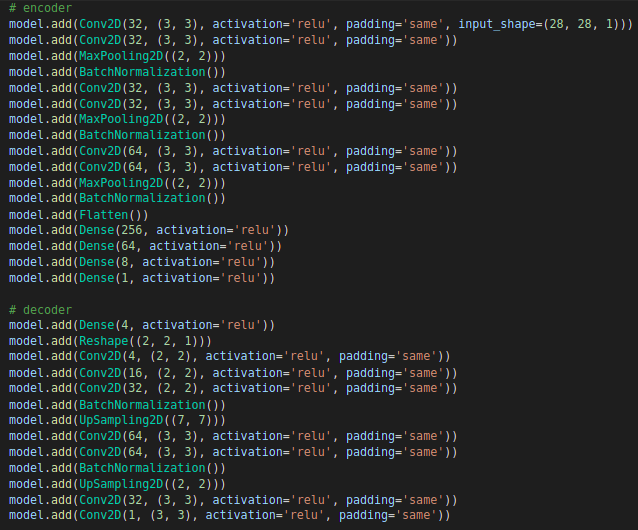
Model 2:

Added more dense layers to model 1.



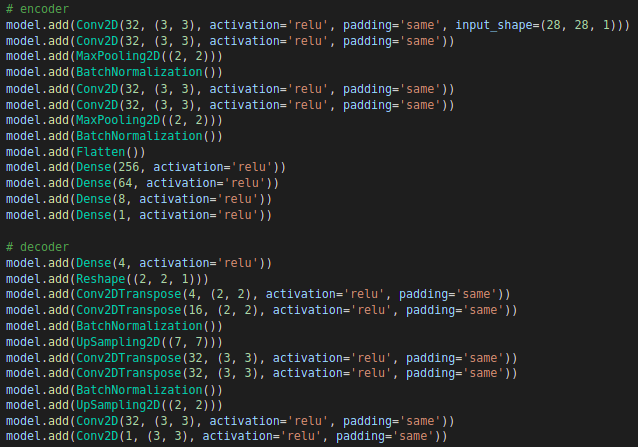
Model 3:

Added another convolutional layer with batch normalization to model 2.

****

Model 4:

Model 2 with Conv2DTranspose instead of Conv2D layers on the decoder.



**TODO:** describe other models

Table of all models:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | 0 | 1 | 2 | 3 | 4 | 5 |
| Training loss | 0.1 | 0.09 | 0.09 | 0.109 | 0.111 |  |
| Validation loss | 0.1 | 0.09 | 0.091 | 0.109 | 0.112 |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | 6 | 7 | 8 | 9 | 10 | 11 |
| Training loss |  |  |  |  |  |  |
| Validation loss |  |  |  |  |  |  |

**TODO:** insert values in table

**3) Discussion**

**TODO:** briefly describing your main observations. Include a clear description of your final deep learning architecture (e.g., regularization approaches, convolutional layer specifications, activations, latent dimensions, etc.). Report your model training details (e.g., loss function and optimization), and the amount of parameters in your networks. Provide tables that depicts your hyper-parameter search, as well as plots of training, validation and test set losses across training iterations.