Image Compression using Encoders

Dobre Dragos

Dobre Radu-Cristian

1231EA

In this project we tried to see different types of compression for images and compare between them to see the best result, time to train, compression rate etc. We are using Simple Encoding, Denoising and Deep CNN.

We are using the Mnist dataset ( <https://en.wikipedia.org/wiki/MNIST_database> ) so that the comparison will be easier. Because they are pretty small in size, we can’t see a big difference in size ,but we can see difference of displaying the image.

We use **Conv2D** which creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs.

**Maxpooling2D** which down samples the input along its spatial dimensions (height and width) by taking the maximum value over an input window for each channel of the input.

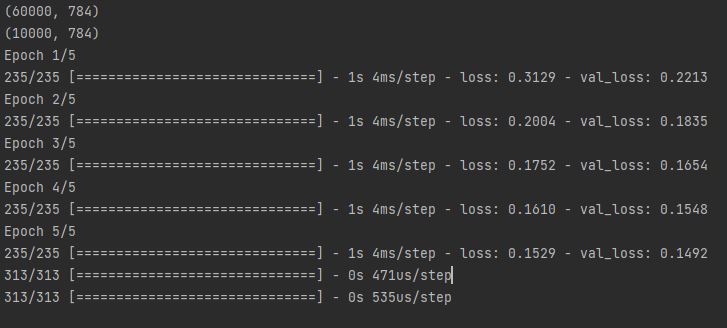
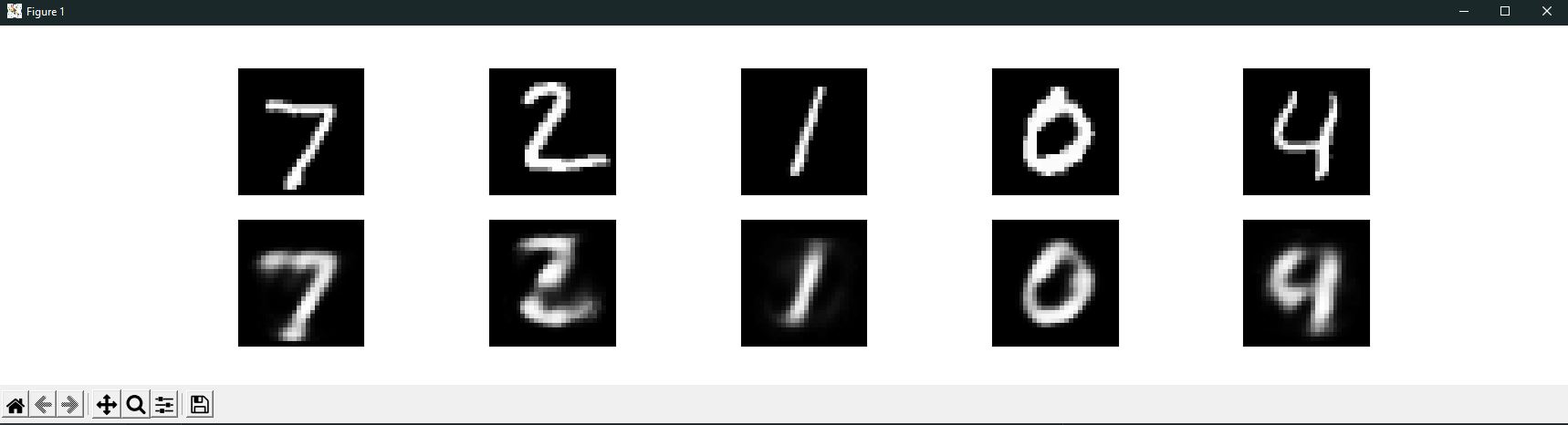
**UpSampling2D** which up samples the layers for 2D inputs.

* First one is the **Simple Encoding**. It is the fastest out of the three, but the result is not that promising. Even at 100 epochs it did a great job in terms of time, but the compression has some weak points. We tried with four configurations Epoch:*5,15,30,100.*

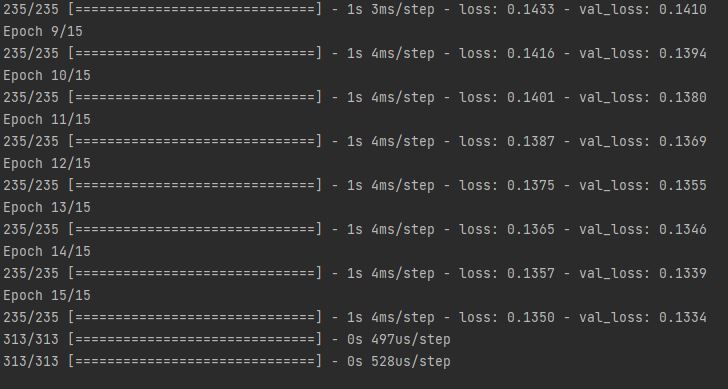
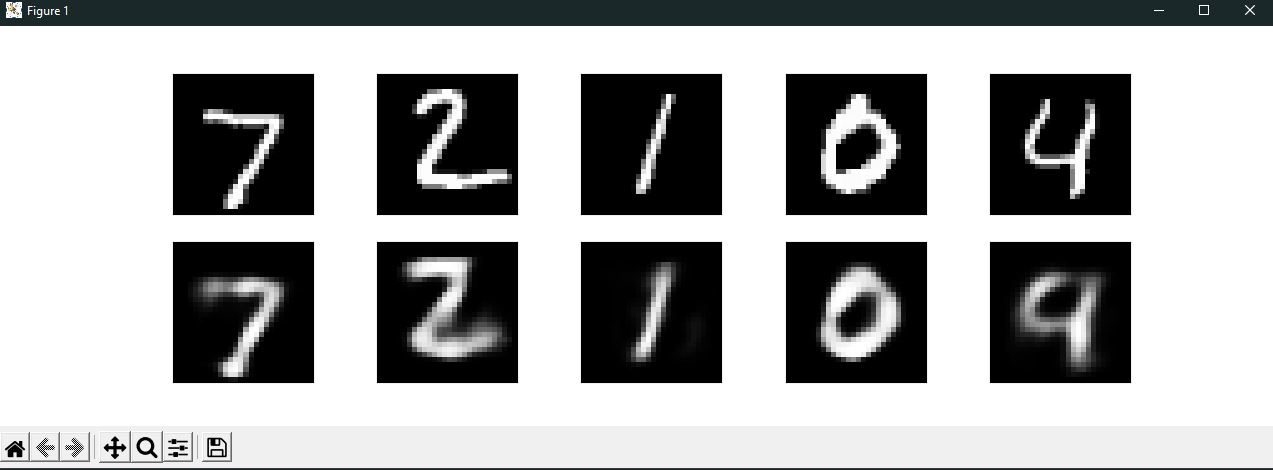
From the images below it can be seen that the value loss for this is *0.1492,0.1334,0.1307,0.1289* for each epoch. The improvement of loss can be seen is logarithmic because it is getting less and less affect over each epoch increase. The time for each step is 4ms and average for each epoch is 1 second.

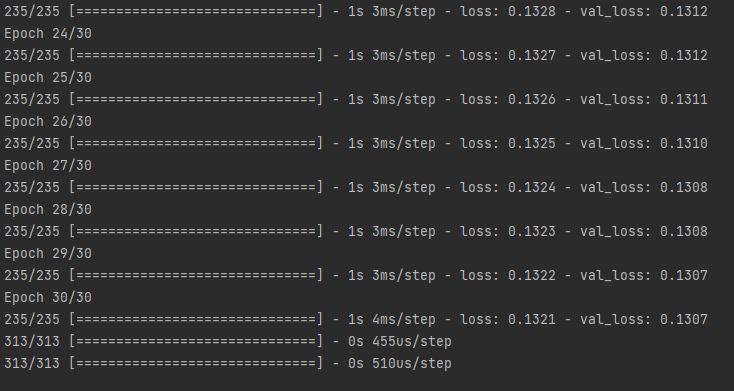
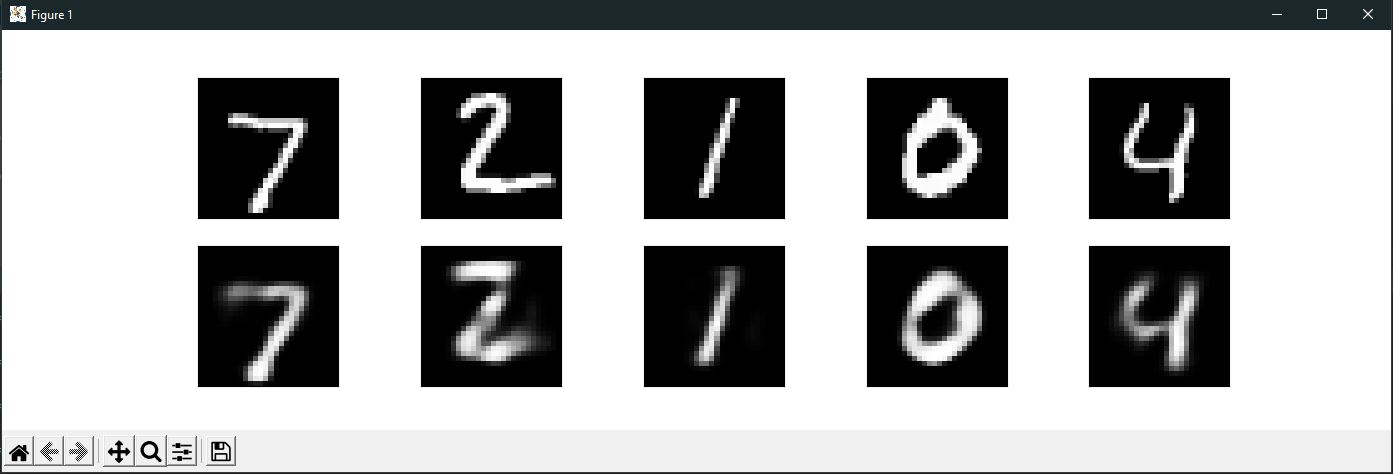
This is the fastest as we will see compered to the other ones. The quality for the compression as it can be seen is pretty bad, but it is something. In the first image when the epoch is 5 we can even mistake some numbers because of the little time it got to train. At 100 we can see an improvement as we can differentiate the numbers between them.

**Epoch 5**:

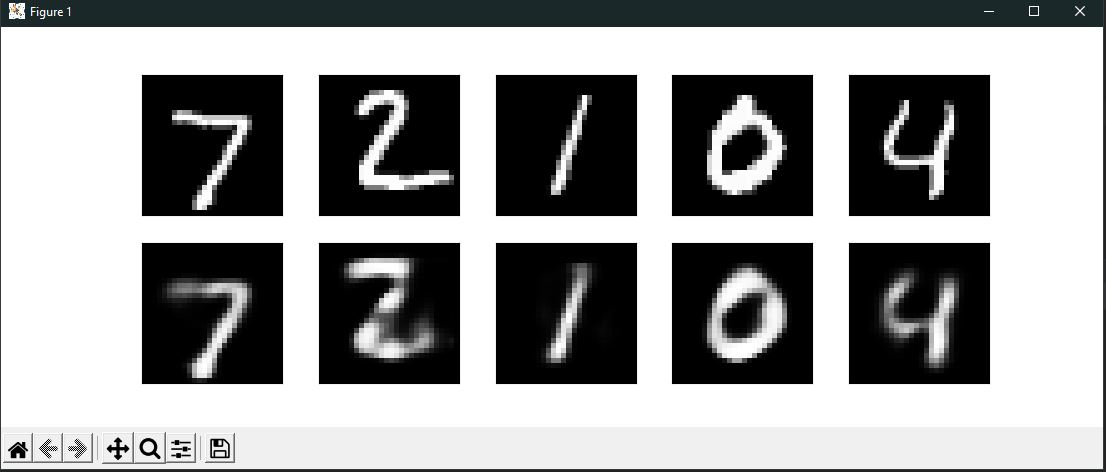


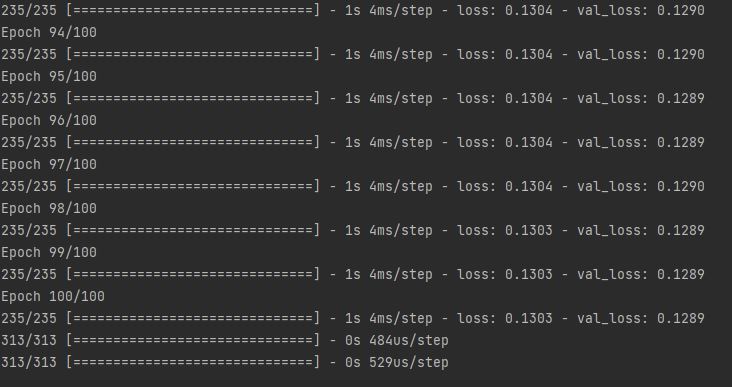
**Epoch 15**:



**Epoch30**:

**Epoch 100**:



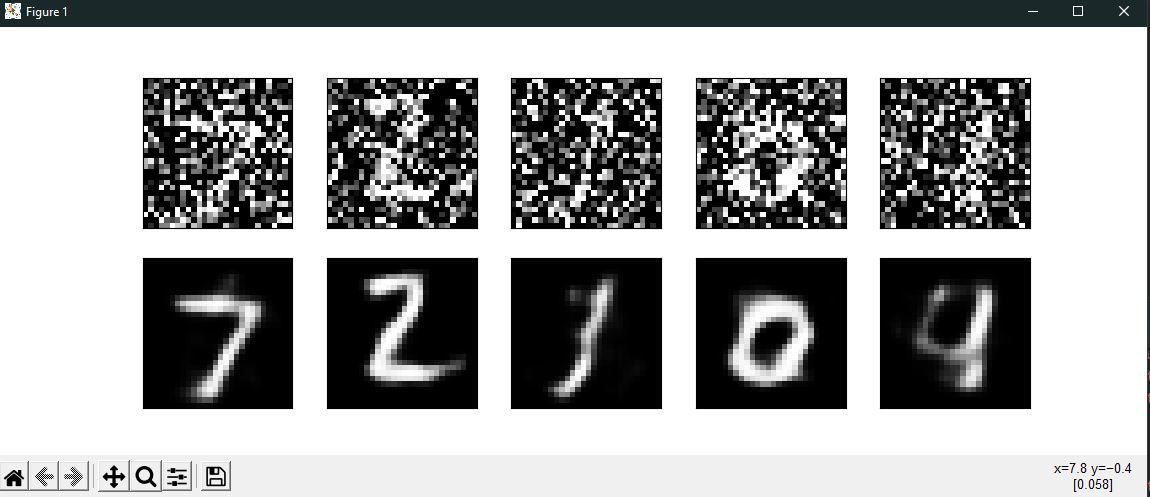


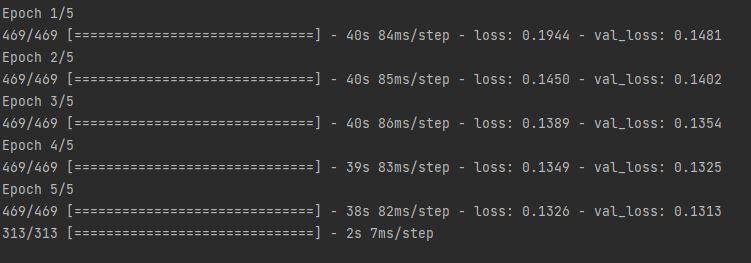
* Second is the **Denoising Encoder**. Is a better encoding then Simple one but it takes much longer. It took a lot even for the 15 epochs. I did it for 5 and 15 epochs. First it makes a noise image of the actual image. After that we train it to decode the encoded image.

We can see that the average time for each epoch is around 40 seconds and for each step 84ms/step. So it took around 3 minutes for the 5 epochs to finish and around 10 minutes for the other one. The value loss is *0.1313,0.1214* respectively.

It is an improvement over the simple one but it takes much longer. We can see the difference in quality between the two which is great, but both of them look worst then the 100 epochs simple one. If we would have trained it more, maybe we got a better result.

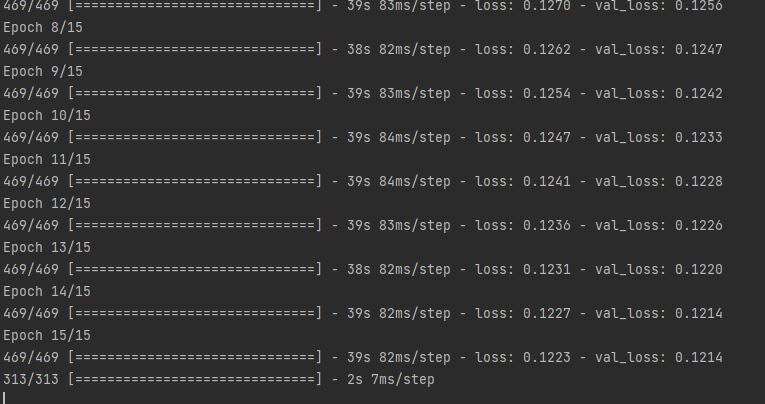
**Epoch 5**:





**Epoch 15**:



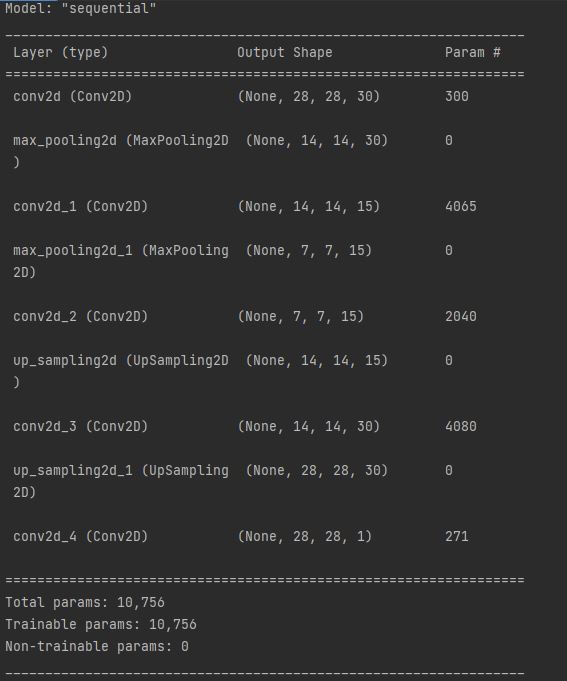


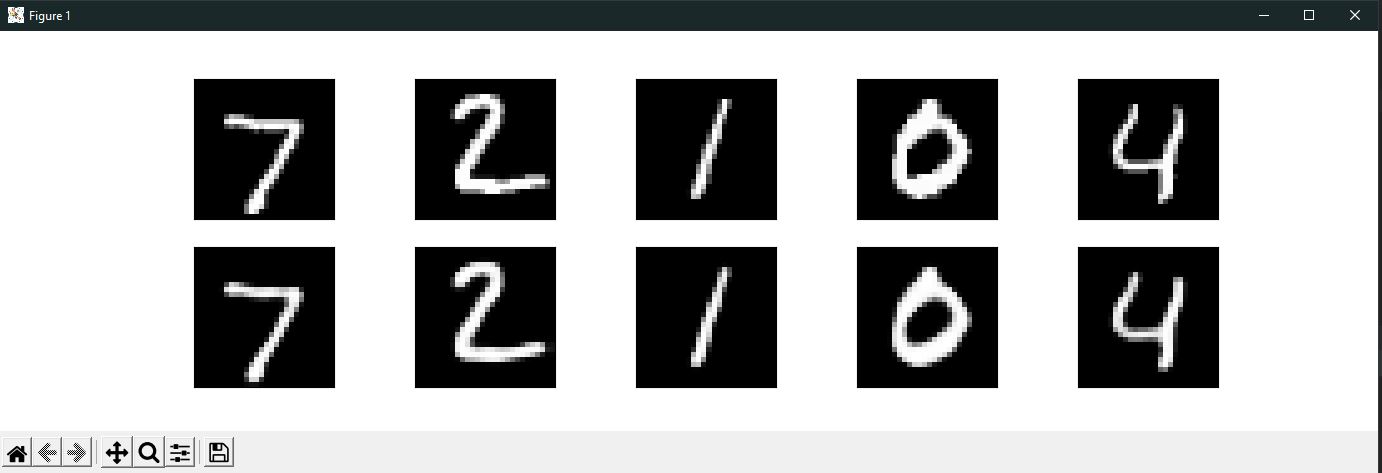
* Third is the **Deep CNN**(Convolutional Neural Network) Encoding. In [deep learning](https://en.wikipedia.org/wiki/Deep_learning), a convolutional neural network (CNN, or ConvNet) is a class of [artificial neural network](https://en.wikipedia.org/wiki/Artificial_neural_network) (ANN), most commonly applied to analyze visual imagery. A **Convolutional Neural Network (ConvNet/CNN)** is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.

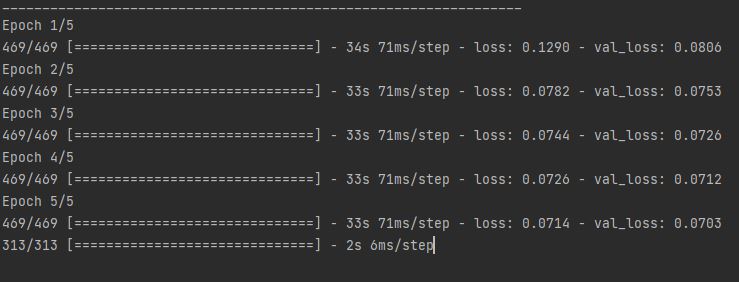
It is the best out of the three. It takes as shorter than Denoising but gives a greater quality. It can be seen from the images below that the recreation is very close. It takes around 33 seconds for an epoch.

Even at 5 epochs the quality is great. At 5 value loss is *0.0703* and for the 15 is *0.0674*. Those numbers are great and we can even see the result. Even at 5 epochs it gives us almost half of the other two bests. For the encoder we are using two Conv2D using RELU and two MaxPooling2D and for the decoder three Conv2D: two RELU and one sigmoid and two UpSampling2D. The model is Sequential.

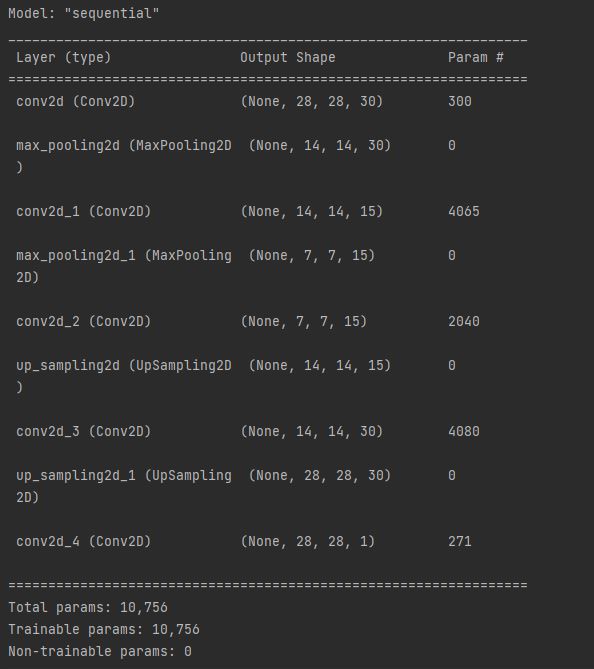
**Epoch 5**:

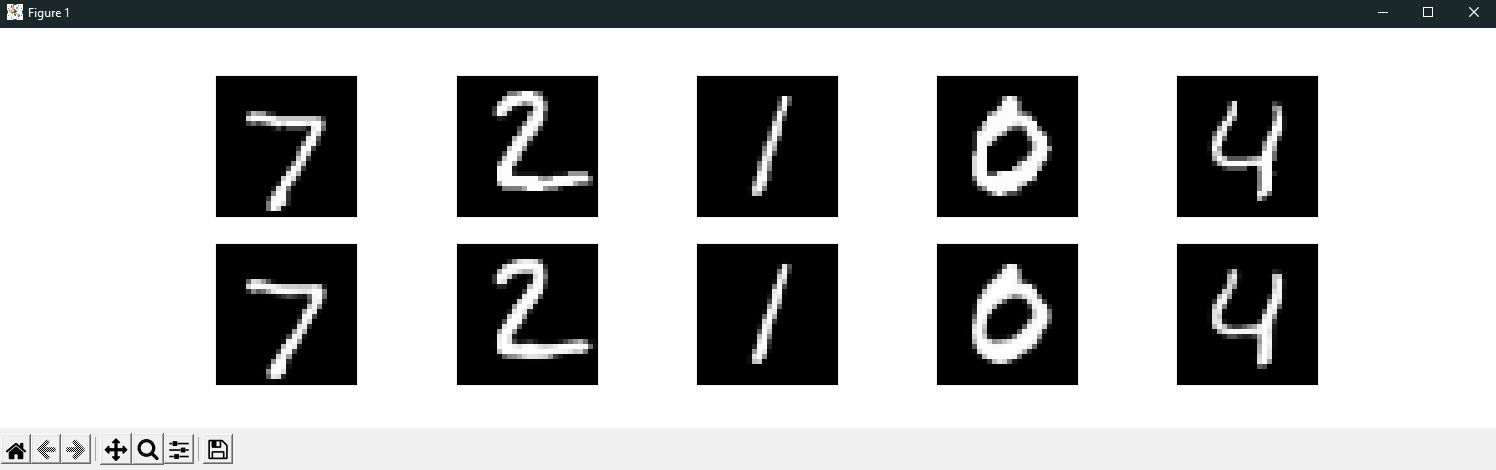


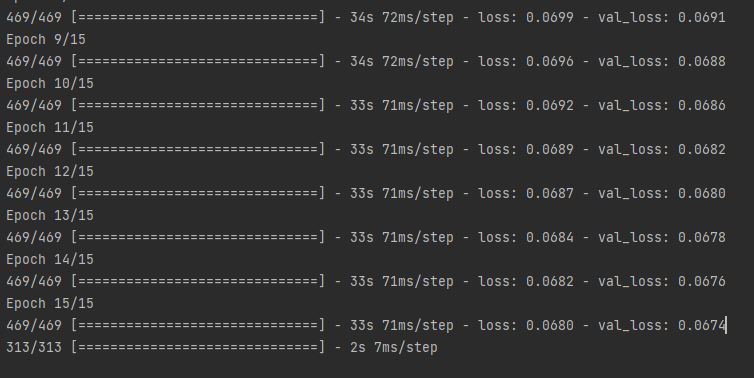




**Epoch 15**:







As a conclusion, after a lot of training and comparisons between them, the third comes first. Deep CNN is the best for this type of tasks as it classifies the clusters. It is slower then simple but faster than Denoising. It gives the best quality and value loss as it can be seen from the images above.

In this period of time, we can see this type of compression applied even for 3D assets using Nanite, a technology developed by Epic Games. It is great as it compresses 8K assets but it keeps the resolution. Even when displaying it is using a cluster type to group pixels so they can be easily displayed.

**REFERENCE:**

<https://www.unrealengine.com/en-US/blog/understanding-nanite---unreal-engine-5-s-new-virtualized-geometry-system?sessionInvalidated=true>

<https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>