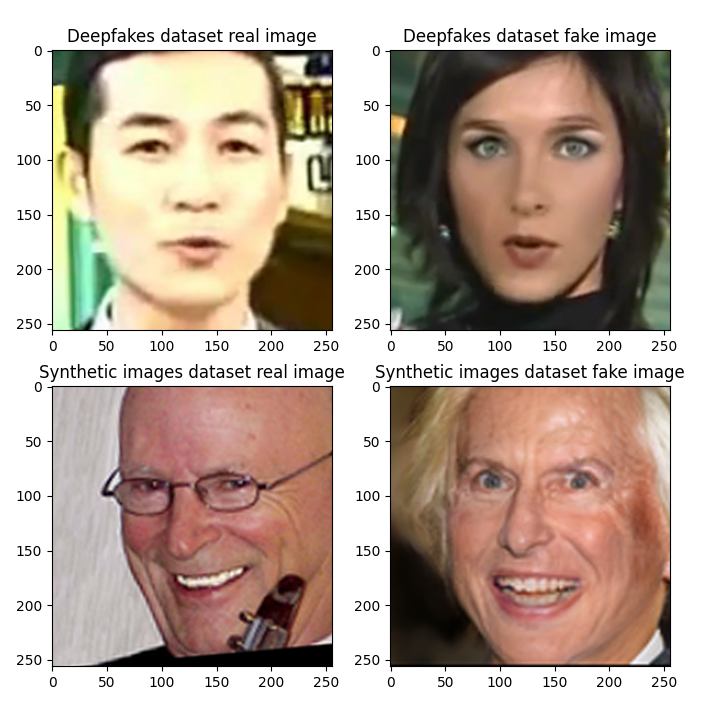
# Computer Vision - Final

Lior Soffer 203135058

Idan Daniel 308088624

# Chapter 2

Q2:Result of **plot\_samples\_of\_faces\_dataset.py**:



# Chapter 3

3.3 Analyze the Deepfake Detection Classifier

Q6 -

Looking only at the test data results, the accuracy increased throughout the epochs to 98.6%, and the loss decreased to 0.04. But, when observing the results on the test and validation data, we received the opposite results – while the train results improved, the test and validation results decreased. This means that the network had done an over-fit training towards to train set, and did not solve the general problem to identify real vs fake images.

Q7 – visualize the data held in the JSON

Chart, line chart

Description automatically generated

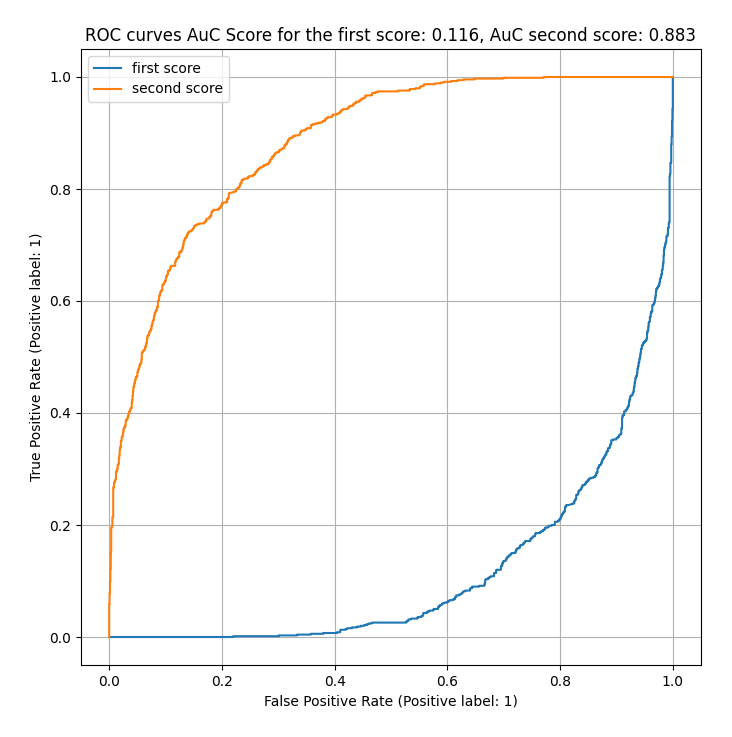
Chart, line chart

Description automatically generated

Q8 – The validation accuracy had decreased between epochs, therefore the highest validation accuracy was received in the first epoch, in which the test accuracy was 78.62%

Q9 – The proportion of fake images to real images in the test set is 1:2 (on every fake image there are 2 real images)

Q10 - Generate ROC and DET graphs for the simple network.



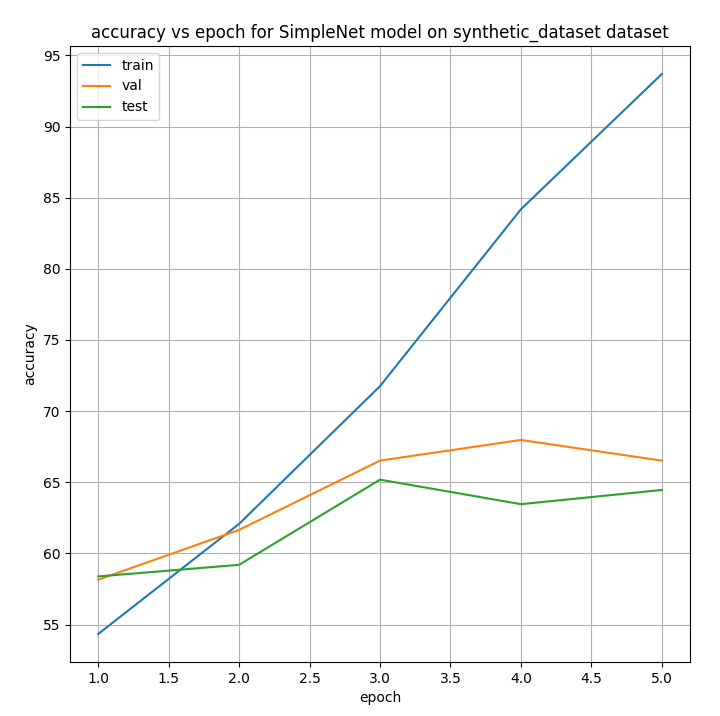
Chart, line chart

Description automatically generated

Q11 –

The first score (A – in blue) tells how real the image is, we can see from the ROC curve that the classifier we have created for this classification problem does not have a good result, whereas the second score (B – in orange) approximate to a perfect classifier. This can be shown also by the AuC score of the two scores. The results show that the simple network we have created is not capable of detecting correctly if the image is real or not.

3.5 Analyze the Synthetic Image Detection Classifier  
Q13 - Generate ROC and DET graphs for the simple network



Chart, line chart

Description automatically generated

Q14 – The highest validation accuracy was received in the fourth epoch, in which the test accuracy was 63.46%

Q15 – The proportion of synthetic images to real images in the test set is 1:1 (on every synthetic image there is a real image)

Q16 – For the synthetic classifier we received a classifier that is not overfitting the train data and can be used to classify unseen data. Saying that, the accuracy of the test data is ~65%, therefore, it is not a good classifier, and needs to be improved to classify correctly.

Q17 – When looking at the sample presented in Q2, it can be seen that the synthetic images are much closer to real images than the fake images. Therefore, even though we got an overfitted classifier for the Fake dataset, the accuracy of the test data was higher than the accuracy of the test data for the synthetic classifier, which was closer to a random classifier.

# Chapter 4

## Q18 and Q20: What is Xception pre-trained on?

retrained version of the network trained on **more than a million images from the ImageNet database** . The pretrained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals.

The researches conduct their comparison on two image classification tasks: one is the well-known 1000-class single-label classification task on the ImageNet dataset , and the other is a 17,000-class multi-label classification task on the large-scale JFT dataset.

## Q19: What are the basic building blocks of Xception?

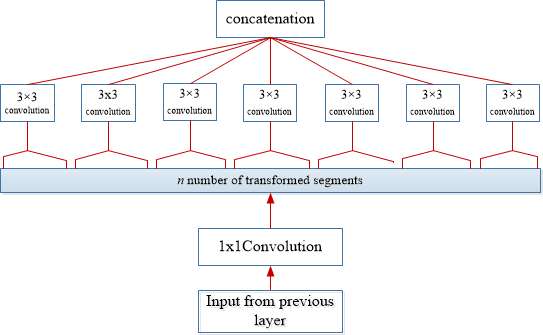
Xception basic building block is **depthwise Separable Convolutions**.

1.     **Depthwise convolution** is the **channel-wise n×n spatial convolution**. Suppose in the figure above, we have 3 channels, then we will have 3 n×n spatial convolution.

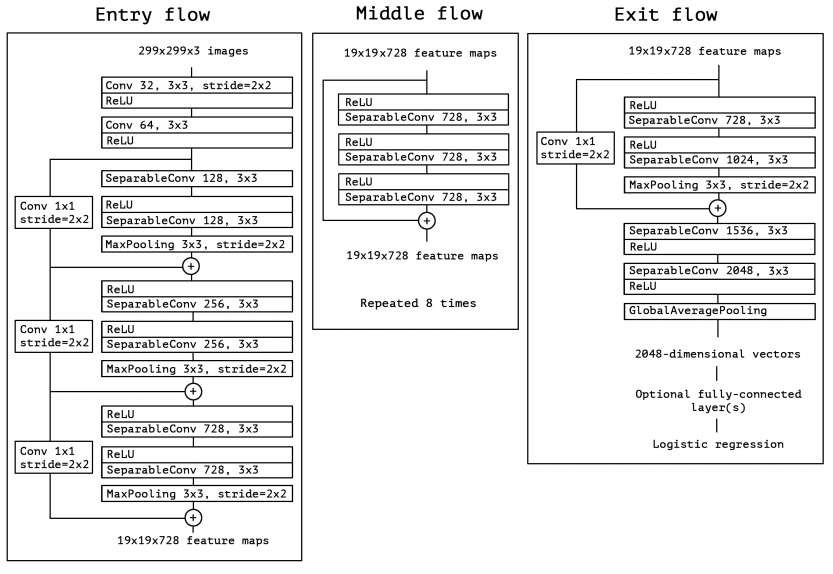
2.    **Pointwise convolution** actually is the **1×1 convolution** to change the dimension.

Compared with conventional convolution, we do not need to perform convolution across all channels. That means **the number of connections are fewer and the model is lighter.**

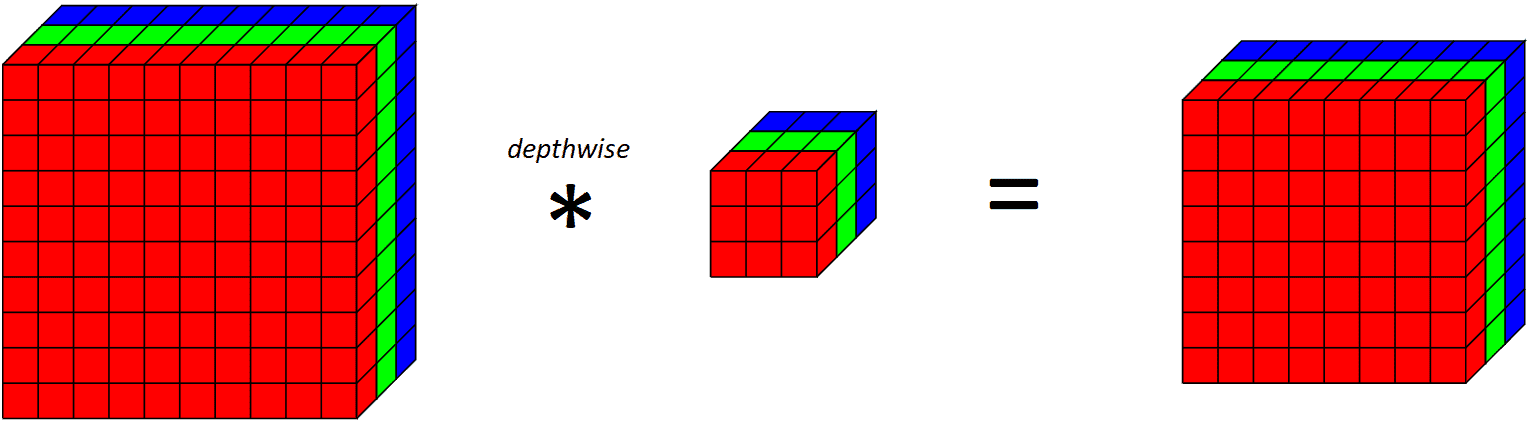
The modified depthwise separable convolution is the **pointwise convolution followed by a depthwise convolution**. **In Xception**.



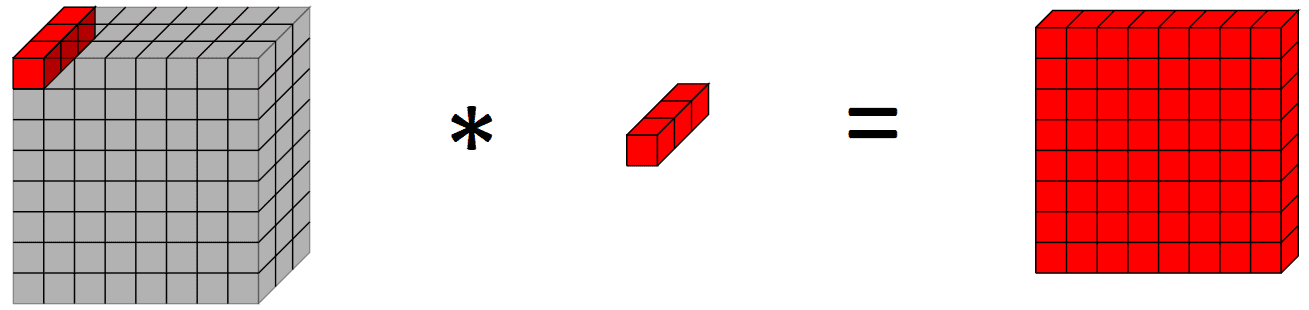
depthwise Separable Convolutions diagram



Xception model diagram



Applying a depthwise convolutional filter on 10x10x3 input volume outputs 8x8x3 volume



Applying a pointwise convolution on a 10x10x3 input volume outputs a 10x10x1 output volume

## Q21: What is the input feature dimension to the final classification block “fc”?

2048.

## Q22: What is the number of parameters the Xception network holds by de-fault? That is, without architectural change and default parameters.

In paper: Xception has 22.8 million parameters.

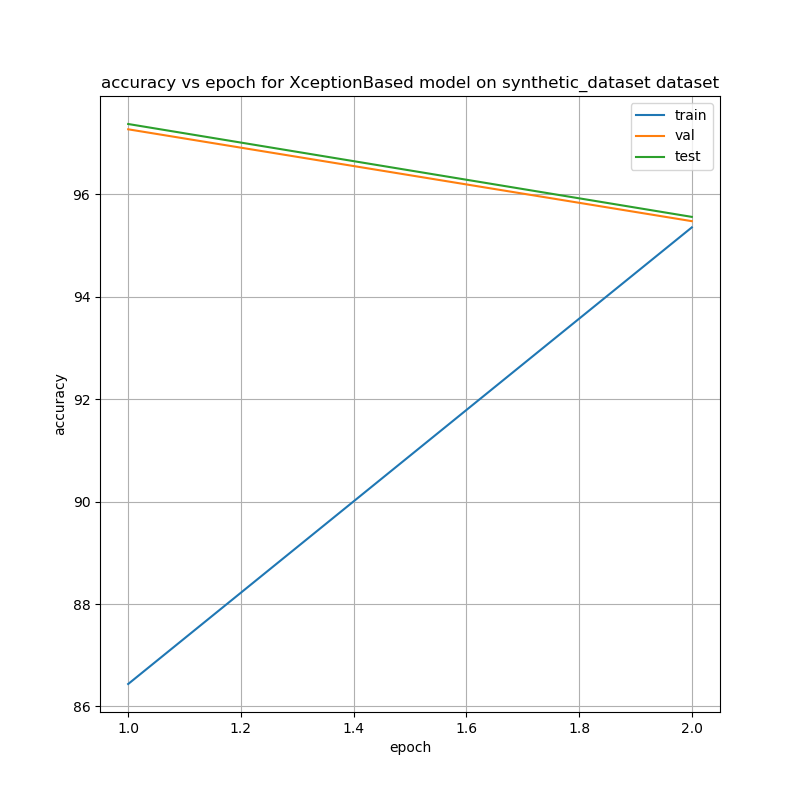
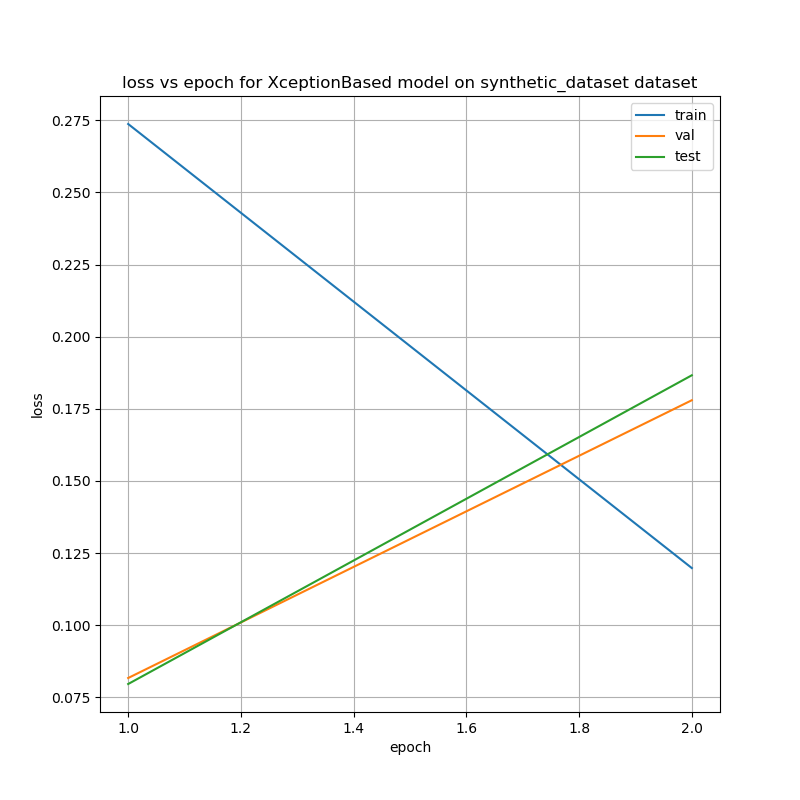
By method: 22,855,952 parameters.

## Q24: How many parameters did we add with the MLP on top of the original Xception’s parameters count?

We added 272,834 parameters.

## Q26: Run the plot accuracy and loss.py script

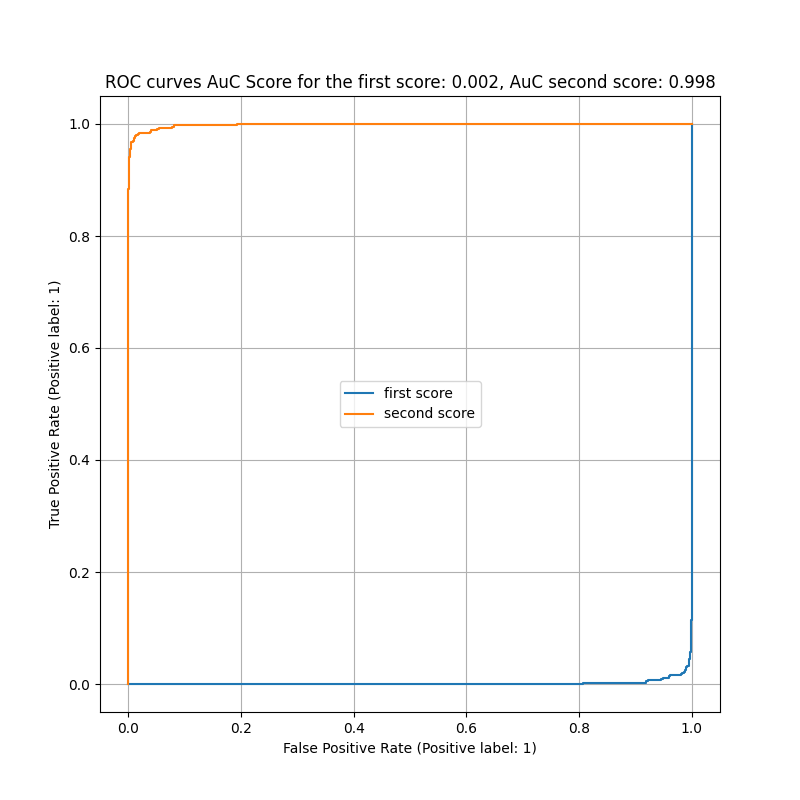
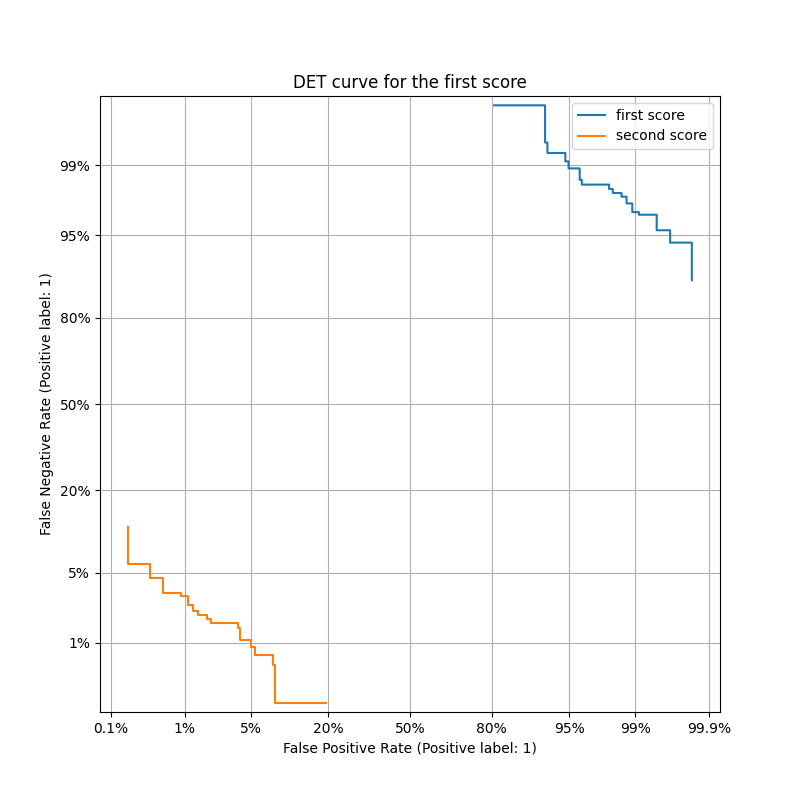
visualize the data held in the json. figures created:



## Q27: What is the test accuracy corresponding to the highest validation accuaracy you received?

97.2

## Q28: Run the numerical analysis.py script for the Xception-Based network trained on the Synthetic Faces dataset.



# Chapter 5

5.1 Intro

## Q29: Image-Specific Class Saliency Visualization

Image-Specific Class Saliency Visualization is a way to figure out which parts of an image are most important in determining what the image is. It does this by making a map that shows which parts of the image had the biggest impact on the image's classification. The biggest impact means highest gradients. This map is specific to each individual image and the class it's been classified as. This technique can also be used to find where an object is located in an image based on what class the object is. In simpler terms, it helps you to figure out what parts of an image are most important in figuring out what it is.

## Q30: Grad-CAMs

Grad-CAM is a way to see which parts of an image a deep learning model is focusing on when it makes a prediction. It does this by making a heatmap that shows which parts of the image had the biggest impact on the prediction. This is done by using the gradients of the last convolutional layer with respect to the final class output and then backpropagating it to the input image. This way we can understand which regions of the image the model was focusing on to make its final prediction. It's a useful tool to understand and interpret the predictions made by a deep learning model, especially in image classification tasks.

5.2 Saliency Maps

Q32

5.3 Grad-CAM