GLO-7030: Deep Learning
(Apprentissage par réseaux de neurones profonds)

Practical work 2

The report is for the second practical work of the course

Prepared By

Parham NOORALISHAHI

Teacher

Dr. Pascal Germain

Université Laval
Quebec

Department of Electrical and Computer Engineering
Faculty of Science and Engineering
Université Laval

Winter Semester 2021

1 Question 1 - fine-tuning and normalization (50%)

> In this question, you have to perform fine-grained classification of bird species. To do so, download the images of the dataset (CUB-200). For each class, sort the images in ascending order by file name and use the first 15 images as a test set. Use the other images for training. To do this, you can use the function provided in the file *question1.py*.
>
> Afterwards, do the following training using the *resnet18* network provided in *torchvision*, (a) Using the default random initialization; (b) Using the pre-trained model, but freezing all convolution parameters; (c) Using the pre-trained model, but only freezing the parameters in "layer1"; (d) Using the pre-trained model, but letting all the parameters (including the convolution layers) be adjusted by back-propagation.
>
> Do each training twice. The first time, use the values from the training dataset to normalize the data. The second time, use the following values that were used for the ImageNet training.

In this question, we are going to classify bird species using CUB-200 dataset. For each class, we sort the images in ascending order by filename and use the first 15 images as the test subset and the remaining as the training subset. Later, we perform the following exercises using `RESNET18` provided in `torchvision` module.

RESNET18 A residual network (ResNet) is an artificial neural network that helps to build deeper neural network by utilizing skip connections or shortcuts to jump over some layers. The presented architecture assist to avoid falling into the problem of vanishing gradients [1]. Figure 1 shows the residual block architecture. Additionally, the architecture of `RESNET18` used in this project is demonstrated in Figure 2. It is worth mentioning that in order to prepare the model for the purpose of bird classification using CUB200 dataset, a linear layer with the output dimension of 200 is added to the network instead of the last year of the RESNET18 model.

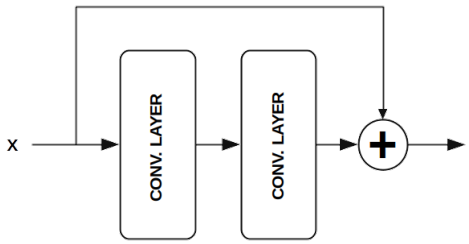


Figure 1: Residual block

[1] `https://www.pluralsight.com/guides/introduction-to-resnet`

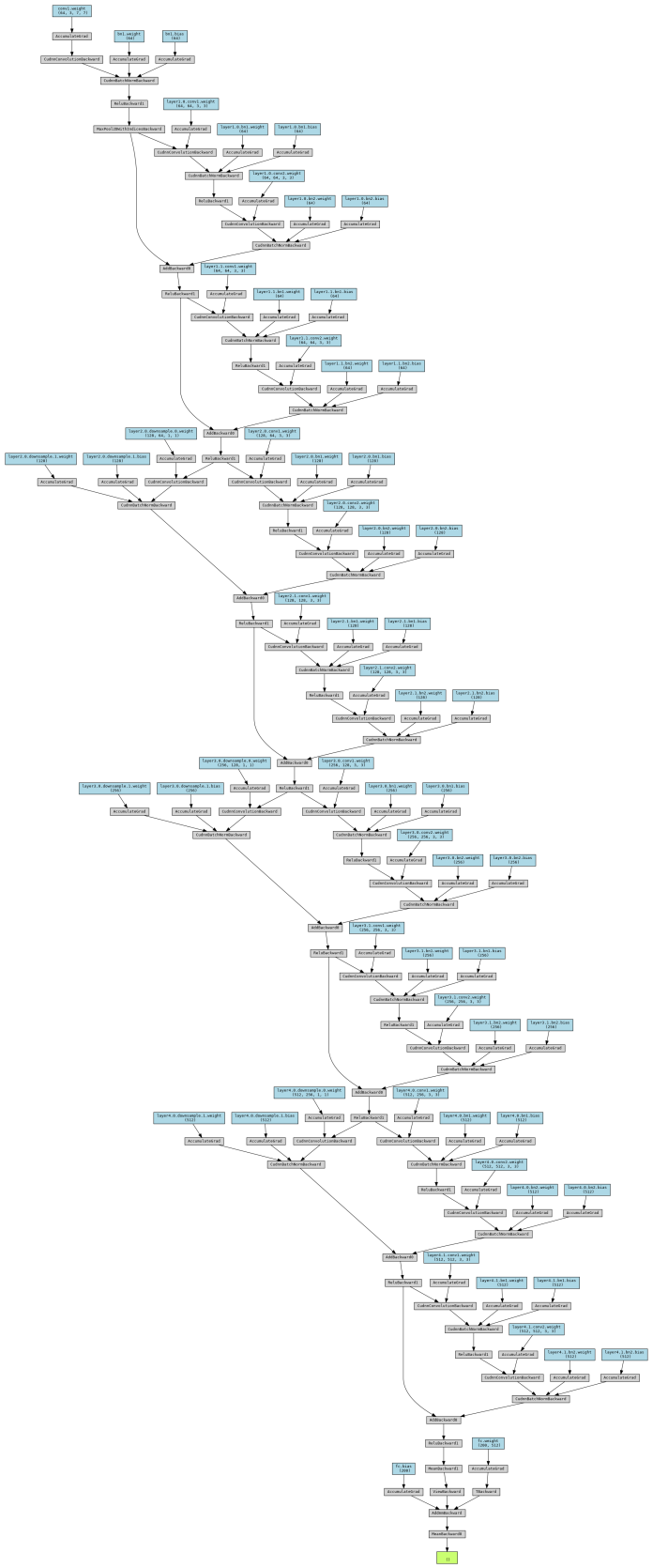


Figure 2: RESNET18 architecture is presented here. The figure is generated by `torchviz`. In addition, the figure is available in the project root folder.

In this question, four scenarios are defined to train the `RESNET18` network for bird classification. In each section, we train the network: (a) using the default random initialization, (b) using the pre-trained model, but freezing all `convolution` parameters, (c) using the pre-trained model, but freezing only the parameters in `layer1`, and (d) using the pre-trained model, but leaving all parameters (including layers convolution) be adjusted by back-propagation. For each of the mentioned scenarios, the network trained using two sets of different data normalization values as shown in Table 1: (a) data normalization's weights calculated using the CUB200 dataset, (b) data normalization's weight used in ImageNet training. Additionally, the values of hyper-parameters selected for the following experiments are explained in Table 2. Furthermore, as mentioned earlier, the dataset is divided into two training and testing subsets. During the training process, the testing subset is passed as validation subset to be able to track the progress of model's accuracy on testing subset.

Table 1: Employed data normalization's weights. * values are calculated based on the data from CUB200 dataset.

		R	**G**	**B**
ImageNet	**MEAN**	0.485	0.456	0.406
	STD	0.229	0.224	0.225
CUB200 *	**MEAN**	0.4750	0.4917	0.4242
	STD	0.2287	0.2244	0.2656

Table 2: The hyper-parameters which are employed for all the experiments in the first question.

Parameters	**Values**
Epoch Number	31
Batch Size	68
Learning Rate	0.1
Momentum	0.9

Beside the results and discussions presented in this article, the supplementary data is also available. The result of Question 1 is placed in `results/q1_colab_30_epoch`. The result of each scenario is placed in a directory named based on the conducted experiment. The supplementary files are placed as explained in Table 3.

Table 3: A description of resulted files.

File	**Description**
log.tsv	A epoch summary of training process, including the accuracy and loss of the model at the end of each epoch.
*.phm	The serialized model saved after the training process is finished.
result.gif	An animated figure showing the loss and accuracy of validation and training processes.
result.png	The trend of showing the loss and accuracy during the validation and training processes.
test_log.tsv	The result of testing process.

1.1 Using default random initialization

In this section, the model with random initialization is trained using the data without any presumption or condition. As shown in Figure 3, both of the data normalization strategies got similar results in testing and training phases. Although, the initial loss of ImageNet is less than the other method, but the final loss of both methods are similar at the end.

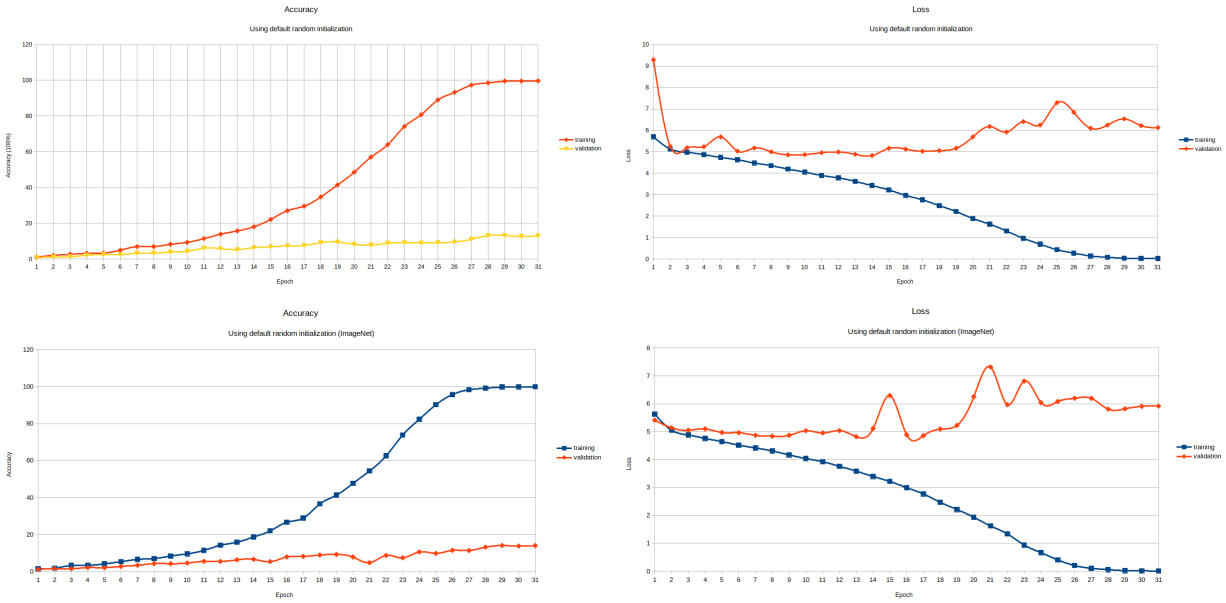


Figure 3: The trends of two training process for 31 epochs when using default random initialization.

1.2 Using the pre-trained model, but freezing all convolution parameters

In this section, the pretrained `RESNET18` model is trained while all convolution parameters are freezed during the training process. As shown in Figure 4, the model used ImageNet data normalization values reached the high accuracy in training process sooner than when CUB200 values used for data normalization. Moreover, although the initial loss of the model used ImageNet values is very higher than the other method, it provides higher accuracy in the testing phase.

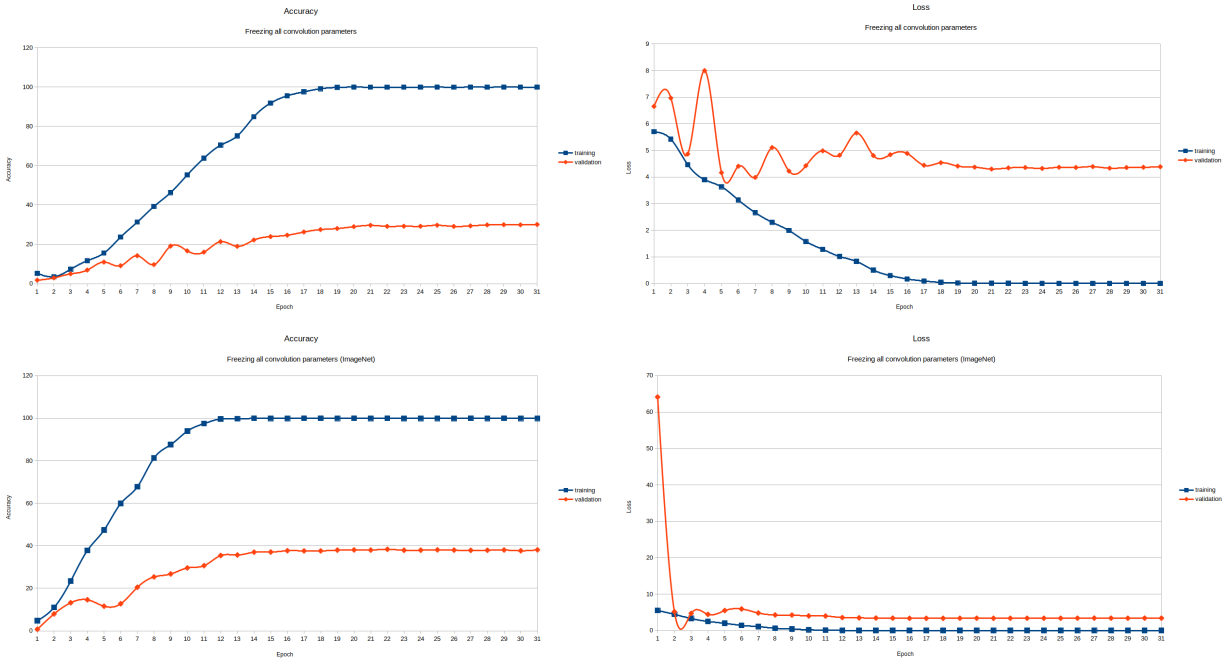


Figure 4: The trend of two training process for 31 epochs when using freezing all convolution parameters.

1.3 Using the pre-trained model, but freezing only the parameters in layer1

In this section, the pretrained `RESNET18` model is trained while the layer named `layer1` is freezed during the training process. As shown in Figure 5, unlike the previous section, the model used CUB200 normalization values reached the highest accuracy range sooner than the model used ImageNet's normalization values. Also, it provides higher accuracy during testing phase compared to the other method. It is worth mentioning that similar to previous section, the model used ImageNet normalization values has higher initial loss.

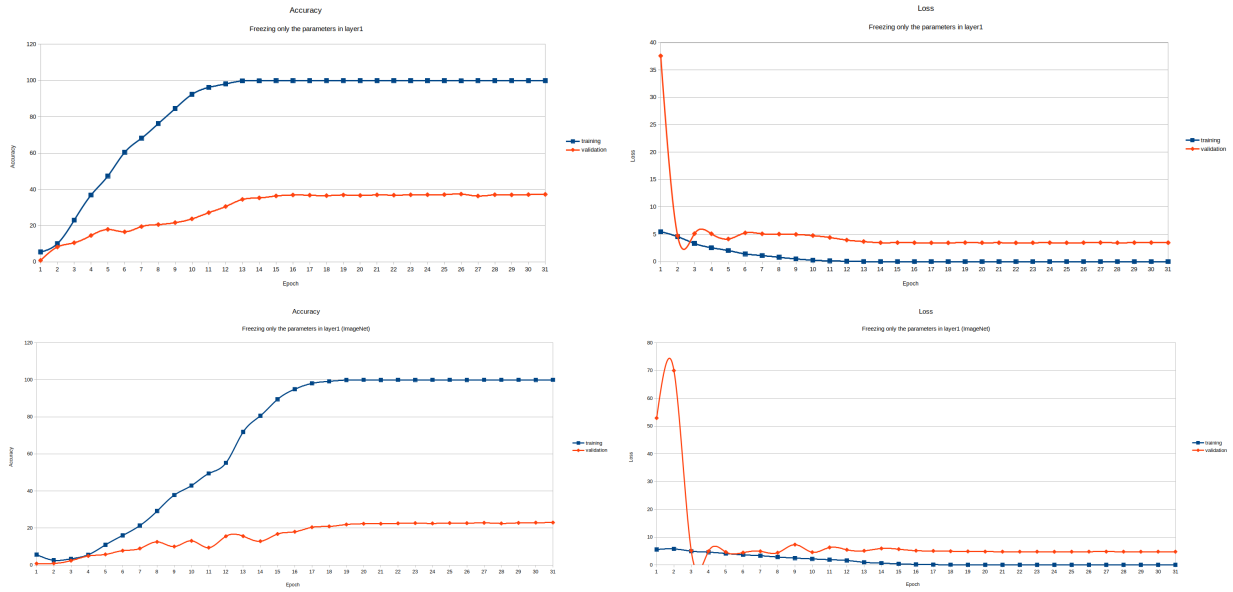


Figure 5: The trend of two training process for 31 epochs when using freezing only the parameters in layer1.

1.4 Using the pre-trained model, but leaving all parameters (including layers convolution) be adjusted by backprop

In this section, the model is trained while all parameters are involved in the training process. As shown in Figure 6, in this experiment, the model shows very slow progress when CUB-200 normalization's weight are employed and shows a very poor accuracy, however the use of ImageNet's normalization weight shows much better results and faster progress.

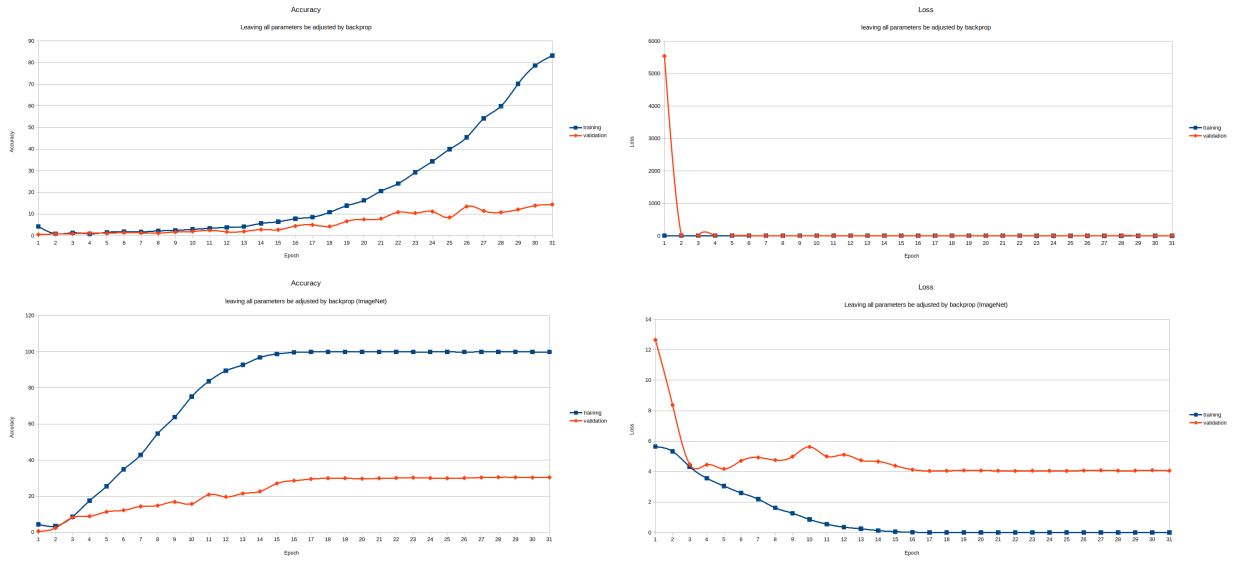


Figure 6: The trend of two training process for 31 epochs when using leaving all parameters (including layers convolution) be adjusted by backprop.

1.5 Discussion

In Table 4, the results of training and testing processes are explained. Based on the results, the model *used ImageNet data normalization values and trained while all convolutional layers are freezed*, demonstrated the best results compared to other models. Also, it is worth mentioning that the maximum testing accuracy achieved in this question is 38.06%. Generally speaking, the results show that the use of ImageNet's normalization weights can improve the model's performance and it can assist the model to converge sooner.

Table 4: The table demonstrates the training and testing results conducted by different experiments. * ERM: Epoch that reached maximum

	TRAINING			TESTING	
	ACC	**LOSS**	**ERM**	**ACC**	**LOSS**
Using default random initialization	99.670	0.0238	31	12.93	6.1283
Using default random initialization (ImageNet)	99.934	0.0079	31	14.03	5.916
Freezing all conv. parameters	99.934	0.0024	25	30.06	4.297
Freezing all conv. parameters (ImageNet)	99.934	0.0018	17	38.066	3.398
Freezing layer1 parameters	99.934	0.0018	15	37.2	3.48
Freezing layer1 parameters (ImageNet)	99.934	0.00234	20	22.93	4.732
Leaving all parameters	83.25	0.556	31	14.366	5.975
Leaving all parameters (ImageNet)	99.934	0.002	20	30.49	4.054

2 Question 2 - Neural Style Transfer (45%)

This question is a pretext to introduce you to style transfer using neural networks. First read the following tutorial,

`https://pytorch.org/tutorials/advanced/neural_style_tutorial.html`

Next, modify the code to use a content image of your own (for example, it could be your self-portrait or your favorite travel photo). Repeat the experiment with the style image provided with the tutorial, corresponding to a Picasso painting, and then with two other style images of your choice (be original !). For each of the chosen images, present the results for the hyper

parameters as set in the tutorial, then : (a) With at least two other weight combinations for style loss (style weight) and content loss (content weight); (b) With at least one other selection of layers to calculate the loss functions.

For each of the experiments, present the obtained results in your report. Explain the hyper-parameter choices that led to the results and discuss the results. Here, the discussion will be qualitative. Interpret as best you can how each choice influenced the visual appearance of the resulting images.

For this question, one content image and three style images are selected.

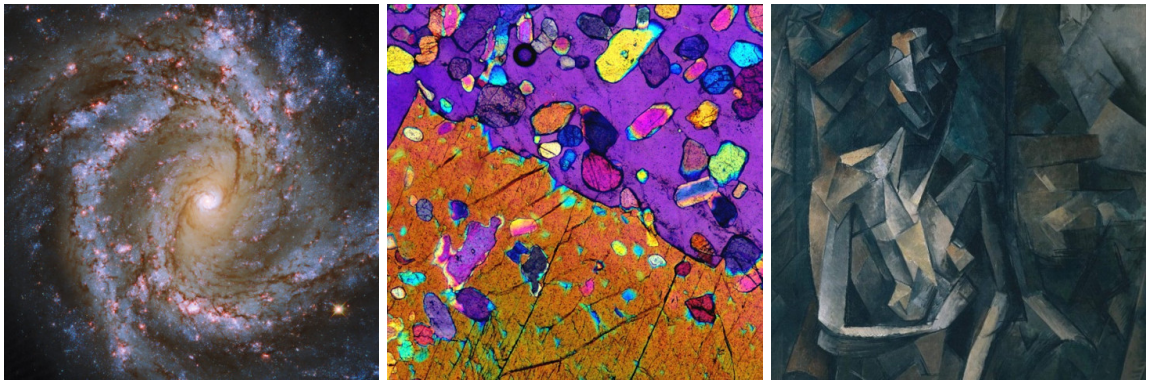
2.1 Content Image

This image is a thermal image of my own face captured by **DALSA Teledyne GXM640** as shown in Figure 7. The areas with higher intensity present higher temperatures and the areas with lower intensity demonstrate lower temperature.

Figure 7: Employed content image

2.2 Style Images

For this question, we have selected three different images with unique patterns to be used in the defined experiment.



(a) Luminous heart of the galaxy (b) Magmatic Rock Olivine (c) Seated Nude (1909)

Figure 8: The selected style images for the second question.

2.2.1 Luminous heart of the galaxy

The first image is the luminous heart of the galaxy [2] M61 dominates this image as shown in Figure 8a, framed by its winding spiral arms threaded with dark tendrils of dust.

2.2.2 Magmatic Rock Olivine

The second image is a magmatic rock olivine [3] as shown in Figure 8b (a mineral which is in a pure form is the germ peridot) inclusions, rumbles in at just 5x magnification. Bernardo Cesare of the University of Padova's Geoscience Department in Italy captured them with a polarized transmitted-light microscope. This image won ninth place in the 20 Microscopic Photo Competition Prizewinners.

2.2.3 Seated Nude (1909)

The last image is a Picasso painting named Seated Nude (1909) [4] as shown in Figure 8c. Seated Nude is part of a series from late 1909 to spring 1910, and a summation of earlier Cubist three-dimensional experimental work on still life and portraits. This time of experiment and research gives this period the title of Analytical Cubism, with its manipulation and fragmentation of space and multiple angles of vision Picasso's whole preoccupation with the notion of vision, explored in the earlier blind man images, now finds its thematic challenge in Cubis.

2.3 Experiment

To analyze the algorithm and understand the influence of defined hyper-parameters on the behavior of the algorithm, seven experiments are defined with different sets of configurations as shown in Table 5. Each configuration is applied to the selected style images and the content image. In the following, the results and configurations for each experiment, are explained in separated sections and the conclusion and discussion regarding the experiments are in the last section. It is worth mentioning that for each experiment, it is tried to change only one of the parameters to investigate their effects on the behavior of the algorithm.

Table 5: Configuration of the hyper-parameters for the experiments.

Experiment	**A**	**B**	**C**	**D**	**E**	**F**	**G**
Content Layers	conv_4	conv_2, conv_4, conv_5	conv_4	conv_4	conv_4	conv_1	conv_1
Style Layers	conv_1, conv_2, conv_3, conv_4, conv_5	conv_3, conv_4, conv_5	conv_1, conv_2, conv_3, conv_4, conv_5	conv_1, conv_2, conv_3, conv_4, conv_5	conv_1, conv_2, conv_3, conv_4, conv_5	conv_1	conv_5
Style Weight	1000000	1000000	8000000	1000000	80	80	80
Content Weight	1	1	1	200	1	1	1

Inside the source code directory, the code of question 2 is placed in `question_2.py` and the Jupyter notebook named `question_2.ipynb`. The experiments are implemented in the order explained in Table 6.

Table 6: The lookup table for the mapping of code sections.

	Style 1	**Style 2**	**Style 3**
Experiment A	1-a	2-a	3-a
Experiment B	1-b	2-b	3-b
Experiment C	1-c	2-c	3-c
Experiment D	1-d	2-d	3-d
Experiment E	1-e	2-e	3-e
Experiment F	1-f	2-f	3-f
Experiment G	1-g	2-g	3-g

[2] `https://www.nasa.gov/mission_pages/hubble/multimedia/index.html`
[3] `https://www.scientificamerican.com/slideshow/small-world-microscope-photography/`
[4] `https://www.pablopicasso.org/seated-nude.jsp`

2.3.1 Experiment A

For this experiment, the algorithm is executed with the preliminary values of hyper-parameters. The algorithm uses the first, second, third, fourth, and fifth convolutional layers to calculate style loss. Additionally, the algorithm uses the fourth convolutional layer to calculate the content loss. The style weight set to 1000000 and the content weight set to one! The result is presented in Figure 9 where each output is associated with one of the selected style images. The pattern of style image is integrated into the content image, however, the contours and general shape of the components inside the content image are visible and distinguishable.

(a) Style: Seated Nude (1909) (b) Style: Galaxy (c) Style: Magmatic Rock Olivine

Figure 9: The results of Experiment A. Content Image : `content_1.jpg` and Style Images: `style_1.jpg, style_2.jpg,style_3.jpg`

2.3.2 Experiment B

For this experiment, the algorithm is configured with different sets of layers for the calculation of content and style losses. The algorithm uses the third, fourth, and fifth convolutional layers to calculate style loss. Additionally, the algorithm uses the second, fourth, and fifth convolutional layers to calculate the content loss. The style and content weights are set the same as Experiment A! The result is presented in Figure 10 where each output is associated with one of the selected style images. The changes in the layers involved in content and style loss create different integration patterns between style and content images. However, still the same as Experiment A, the content image's contours are visible and distinguishable.

(a) Style: Seated Nude (1909) (b) Style: Galaxy (c) Style: Magmatic Rock Olivine

Figure 10: The results of Experiment B. Content Image : `content_1.jpg` and Style Images: `style_1.jpg, style_2.jpg,style_3.jpg`

2.3.3 Experiment C

For this experiment, the algorithm is configured with a very high value for style weight. The rest of the parameters are set as in Experiment A. The result is presented in Figure 11 where each output is associated with one of the selected style images. The results show that the increase in style weight causes the style patterns to prevail over the content patterns which means the resulted image has fewer details related to the content image.

(a) Style: Seated Nude (1909) (b) Style: Galaxy (c) Style: Magmatic Rock Olivine

Figure 11: The results of Experiment C. Content Image : `content_1.jpg` and Style Images: `style_1.jpg, style_2.jpg,style_3.jpg`

2.3.4 Experiment D

For this experiment, the algorithm is configured with a higher value for content weight. The rest of the parameters are set the same as Experiment A. The result is presented in Figure 12 where each output is associated with one of the selected style images. The results show that the increase in content weight makes the result contains more enhanced details of the content image.

(a) Style: Seated Nude (1909) (b) Style: Galaxy (c) Style: Magmatic Rock Olivine

Figure 12: The results of Experiment D. Content Image : `content_1.jpg` and Style Images: `style_1.jpg, style_2.jpg,style_3.jpg`

2.3.5 Experiment E

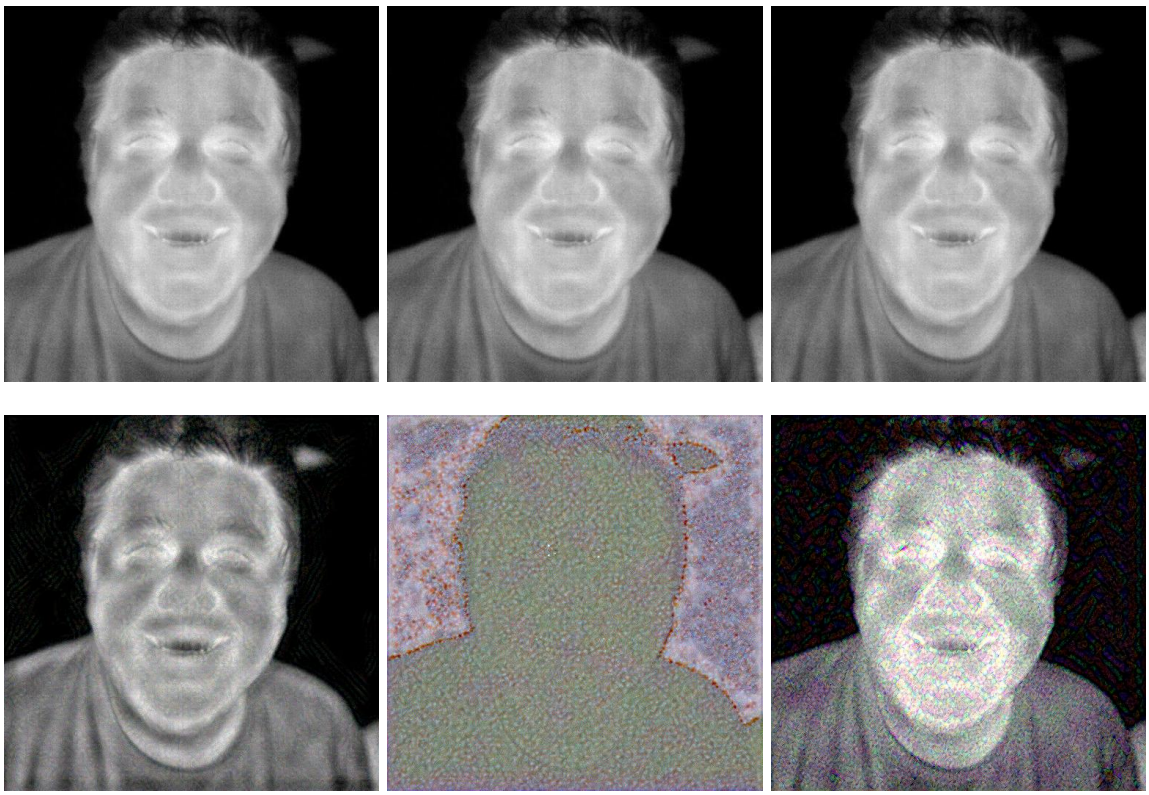
For this experiment, the algorithm is configured with a very low value for style weight. The rest of the parameters are set the same as Experiment A. The result is presented in Figure 13 where each output is associated with one of the selected style images. The results show that the low value in style weight makes the effect of the style image disappear from the resulted image.

(a) Style: Seated Nude (1909) (b) Style: Galaxy (c) Style: Magmatic Rock Olivine

Figure 13: The results of Experiment E. Content Image : `content_1.jpg` and Style Images: `style_1.jpg`, `style_2.jpg`,`style_3.jpg`

2.3.6 Experiment F & G

Finally, for these two experiments, the algorithm is configured with two different configurations on style layers. The goal of these experiments is to investigate the effect of different style levels on the resulted photo as shown in Figure 14. In Experiment F, the style layer is set to only the first convolutional layer. The results of this experiment demonstrate that the first style layer does not contain a significant pattern that influences the fusion of two images. On the other hand, in Experiment G, the style layer is set to the fifth convolutional layer. The result of this experiment shows that the higher layers contain more details which can be presented in the fusion of the two images.



(a) Style: Seated Nude (1909) (b) Style: Galaxy (c) Style: Magmatic Rock

Figure 14: The results of Experiments F & G. Content Image : `content_1.jpg` and Style Images: `style_1.jpg,style_2.jpg,style_3.jpg`

2.3.7 Discussion

In the conducted experiments, the influence of determined hyper-parameters is investigated! Content weight is one of the parameters, determining how much the content image influences the generated image. The results of conducted experiments show that the higher values of Content weight make the content image's patterns more appearing. Style weight is also another parameter specifying how much the style image influences the generated image. We tested the algorithm with increasing and decreasing values of style weight and the results demonstrate that the higher values of style weight reduce the details of the content image and increase the appearance of the style patterns. On the other hand, the lower value of style weight, reduce the visibility of style patterns in the resulted image.

Style and content layers are the other parameters presenting the involved convolutional layers for calculating content and style losses. The determined list of layers determines that whether the low-level features or higher-level features are targeted to be involved in the calculation.

3 Question 3 - An open question ! (5%)

> In 2018, an "art piece" generated by a GAN sold at auction for over $400,000 US. Read the provided link in the TP2. Then, express an opinion informed by what you have learned in the course inspired by the question : *Can an artificial neural network demonstrate creativity?* Explain your point of view in just a few sentences. There are no right or wrong answers, only interesting thoughts.

Creativity is the act of perceiving the world from a new perspective, bringing ideas to reality, finding new patterns, making connections between even unrelated phenomena, or even generating novel solutions. In my opinion, creativity is a relative subject! In the context of science, business, or even industry, the objective is to find novel solutions, improving the performance of a process pipeline, etc. Thus, they can be strictly mathematical and the involved factors can be limited! which means they can be modeled mathematically (at least approximately). Generally, I believe, any natural phenomena can be modeled if all the influential factors are considered and modeled properly, even for example the path that a leaf would take while falling from a tree can be modeled and estimated if all the involved factors are known and taken into the consideration! So creativity in this context is completely possible using artificial intelligence. However, creativity in art and literature would be a little bit different. Because, a work of art is created based on a huge number of factors like the artist's life, their experiences, their view about beauty, life, and millions and millions of other factors which would be seen in the art itself; therefore the number of influential factors make the modeling process almost impossible (of course to my knowledge and current computing resources!). So I think, putting all these factors aside, the presented method in the article, simply learns what would be similar to human art and mimic something that would trigger the aesthetic part of the viewer's mind! So the results would have the elements of an art but lacks the inference, essence and originality.

In conclusion, I think the current state of art can generate art in the sense of beauty or similarity to human art. However, at least for now, they are not able to send a message through art.