Bring Van Gogh Back to Life! – Final Project Report for Introduction to Deep Learning – 0571418201

Git-hub:

futuremobilitylabTAU/Van_Gogh_Back_to_Life: Neural Style Transfer: Reimagining Images in the Iconic Brushstrokes of Van Gogh Using Deep Learning

[1]Niv Danieli

M.sc student

ID: 318507217
Email: nivdanieli@mail.tau.ac.il

[2]Gabriel (Gabi) Dadashev

PhD student

ID: 204596456
Email: dadashev@tauex.tau.ac.il

[1]Department of Industrial Engineering, Iby and Aladar Fleischman Faculty of Engineering, Tel-Aviv University, Ramat Aviv 6997801, Israel

[2]The Porter School of the Environment and Earth Sciences, Tel-Aviv University, Ramat Aviv 6997801, Israel

1.0 Introduction

The entry of deep learning algorithms into the mainstream has revolutionized many fields. The ability to create a complex function, built from a long chain of functional assemblies one on top of the other – to the point where the exact “meaning” of the formula becomes indecipherable – and to train it to fit a particular phenomenon with great precision, has provided significant advantages over traditional approaches. These advantages are particularly noticeable in terms of accuracy, but they sometimes come at the cost of dealing with a “black box.”

This report was written with the aim of summarizing the main points of the basic course in deep learning, in which the processes of building neural networks, training them, validating them and applying them to more complex tasks were studied. In this project, we focused on the practical application of two main processes: classification and style transfer. We first used a subset of the WIKIART dataset to train two pre-trained models, VGG-19 and AlexNet, to classify paintings in the post-imperial style and determine whether they were created by the famous painter Vincent van Gogh. The classification process is detailed in the first part of the work. In the second part, we applied the Style Transfer technique to give each image fed into the network the unique artistic touch of Van Gogh, while upgrading the images according to his iconic style.

2.0 Methodology

2.1 VGG-19 and AlexNet

In this work, two advanced models, AlexNet and VGG-19, were defined, which aim to analyze images and perform classification (classification) within the framework of predefined tasks. AlexNet is an 8-layer convolutional neural network (CNN): 5 convolutional layers (with filters of sizes 11x11, 5x5 and 3x3) and 3 fully connected layers, with approximately 60 million parameters. The network is divided into two training runs on two graphics processing units (GPUs). Figure 1 shows the network structure.

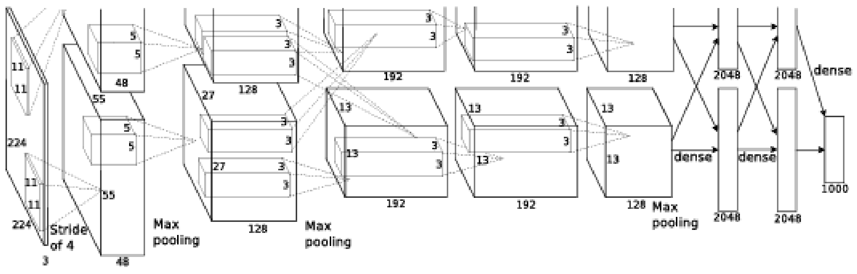


Figure 1: Illustration of the AlexNet architecture in the original training process on two GPUs. Source: Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012).

VGG-19 is a 19-layer CNN: 16 convolutional layers and 3 fully connected layers. It uses small 3x3 filters with a stride and a pad of 1, and 2x2 max pool layers with a stride of 2. With 138 million parameters, it was trained on a portion of the ImageNet database, classifying images into 1000 categories. In ILSVRC 2014, it achieved an error rate of 7.3%, ranking second in classification and first in localization. Compared to AlexNet, it is deeper and uses small filters to reduce the number of parameters. Figure 2 illustrates the network structure.

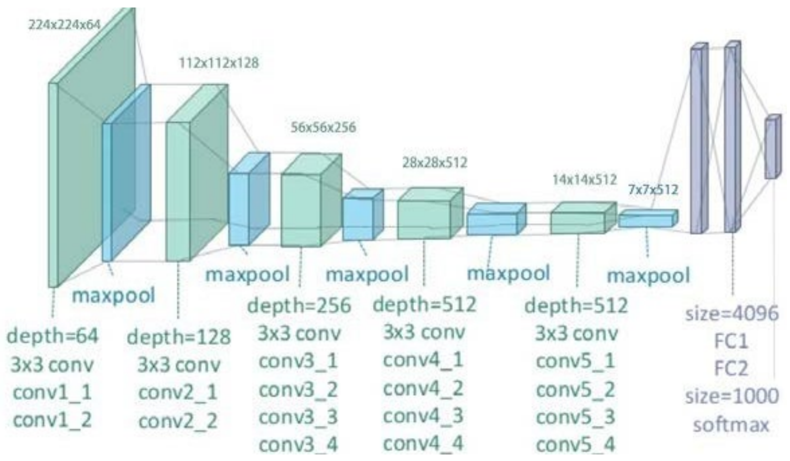


Figure 2: Illustration of the VGG-19 architecture, where "conv" denotes convolution and "FC" denotes fully connected layers. Source: Zheng, Y., Yang, C., & Merkulov, A. (2018).

2.2 Fine tuning

Fine Tuning is a process in deep learning where you take an existing model, such as AlexNet or VGG-19, that has been pre-trained on a large dataset and adapt it to a new specific task. Instead of training the network from scratch, we use the knowledge you've already learned, such as general features of contours, shapes, or textures, and make slight changes to its parameters. For example, working with AlexNet or VGG-19, you can use them as a basis for classifying medical images, such as detecting breast cancer, while tuning the last layers to identify categories such as "healthy" or "suspicious." In our case, the models are adapted to detect whether an image is a painting by Van Gogh or not, using the existing knowledge for a binary classification: "Van Gogh" or "not Van Gogh." This saves time and resources compared to full training and improves accuracy when data is limited.

Early layers, which detect basic features such as edges and textures and are considered more general, are often left locked (i.e., their weights do not change) to preserve the basic knowledge already acquired. In contrast, later layers, which are closer to the classification stage and focus on more specific features, are released for training to adapt them to the new task. The decision depends on the amount of data available and the similarity between the original and new tasks.

The models were loaded with pre-trained weights from the ImageNet database, which contained millions of images and 1000 categories, allowing for the exploitation of general features such as contours, shapes, and textures as a basis for the task. The final layer of each model, which originally classified 1000 categories, was replaced with a new layer with only one output, incorporating a Sigmoid function, to classify images binary – whether it was a Van Gogh painting (1) or not (0). Next, in AlexNet, most of the feature recognition layers remained fixed, and only the last 6 parts (which include late convolutional layers and additional processes such as pooling) were released for updating; in VGG-19, most of the feature recognition layers also remained fixed, and the last 6 parts (a combination of convolutions and additional processes) were released for updating. In addition, the last layer of the fully connected layers, which performs the final classification, was released for full updating in both models.

2.3 Data Set Adjustments

Visual adjustments (transforms) are designed to increase the variety of data, improve the generalizability of the model, prevent overfitting, and adapt the images to the technical requirements of the models, such as size and standard values, to enable more accurate identification of features such as Van Gogh's style under varying conditions. As part of the work, the dataset underwent precise adjustments for the task of identifying Van Gogh paintings: a comprehensive data file with image details was loaded, filtered to 6307 images from the Post-Impressionism genre only, and a binary label was added – 1 for Van Gogh's images and 0 for others. A default image was set as a backup in case of deficiencies (this was how 20 deficiencies were handled at the beginning of the runs due to incorrectly recognized French letters), and the images underwent visual modifications during training: random cuts to 224x224 pixels, horizontal flips, rotations up to 15 degrees, and changes in brightness, contrast, and colors. Finally, the images were normalized to standard values to match AlexNet and VGG-19 requirements, creating a dataset ready for efficient training.

2.4 Hyper parameters

The work processes The hyperparameters were optimized in order to improve the performance of the AlexNet and VGG-19 models in the Van Gogh painting classification task using the Optuna optimization tool, which was used to search intelligently and efficiently for the optimal hyperparameter combinations. The optimization included the following parameters: learning rate, which was chosen in a logarithmic range between 0.00001 and 0.001, weight decay, which was chosen in a logarithmic range between 0.000001 and 0.0001, and the training set size, which was chosen from the values 4, 8, 12, and 16. In addition, in each experiment, one of the two models was chosen - AlexNet or VGG-19. The training process was performed using the K-Fold Cross Validation technique with 5 splits to ensure a reliable and balanced performance assessment, with the model being trained for 4 training cycles in each experiment with an Early Stopping mechanism that stopped training in the event that it did not An improvement in the validation set loss was recorded for 7 consecutive cycles during training. The Weights & Biases platform was used to document and track the performance of the models, which provided detailed graphs of the training and validation losses along with a report on the hyperparameters selected in each experiment. At the end of the process, it was found that the optimal combination of hyperparameters for AlexNet included a learning rate of 0.000010369, weight decay of 0.000082922, and a training set size of 4, while for VGG-19 it was found that the optimal combination included a learning rate of 0.000022012, weight decay of 0.0000015497, and a training set size of 4. This approach allowed us to maximize the performance of the models and reduce the loss on the validation set efficiently and accurately.

2.5 Training

For training, the Binary Cross Entropy Loss (BCELoss) loss function was used, which is particularly suitable for binary classification problems and allows for accurate calculation of model errors based on the probabilities predicted by the outputs. In addition, an Adam optimizer was used, which was chosen for its efficiency in the learning process. The training process was implemented in a K-Fold Cross Validation structure with five folds to ensure better generalization of the model and reduce the chance of overfitting. An Early Stopping mechanism with a patience=2 parameter was used to stop training when there is no improvement in the loss during two consecutive training cycles, which helps prevent overtraining and reduce waste of computational resources. In addition, the model was evaluated throughout all training cycles using various metrics such as Accuracy, F1 index, and AUC, which allowed for ongoing monitoring of the model's performance.

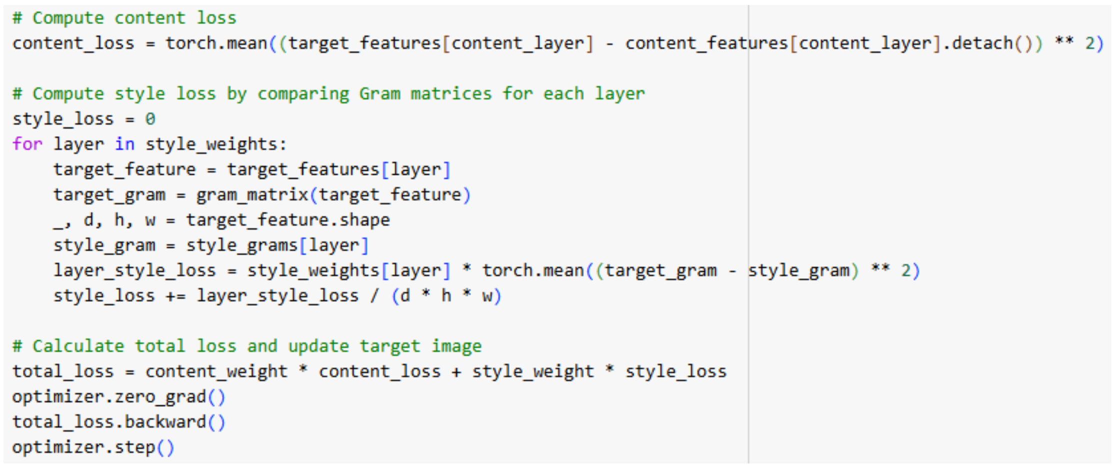
2.6 Style Transfer

Style transfer is a technique that combines the content of one image with the artistic style of another. This process relies on neural networks for image recognition to extract and recombine representations of different features. The method involves extracting content features from a given "content" image using the deeper layers of the network and extracting style features from the "style" image using the initial layers. The output image is then iteratively transformed to minimize a loss function, which consists of content loss, which ensures that the produced image preserves the main details of the source, and style loss, which encourages the transfer of artistic patterns.

Content loss is calculated by comparing the feature map of the produced image to the feature map of the content image in a selected layer, ensuring that the overall structure remains the same. In contrast, Style loss is calculated using a Gram matrix of feature activations, which “captures” the relationships between different textures and colors. By minimizing the difference between these matrices, the resulting image adopts the stylistic characteristics of the artwork.

The total loss function is a weighted sum of the content loss and the style loss, with adjusting the weights allowing control over how much the result is biased toward preserving content or emphasizing style. Through iterative optimization, the model gradually refines the image to achieve an aesthetic fusion of the two elements.

Figure 3. Core implementation of style transfer: computation of content loss, style loss based on Gram matrices, and iterative optimization of the target image.



```
# Compute content loss
content_loss = torch.mean((target_features[content_layer] - content_features[content_layer].detach()) ** 2)

# Compute style loss by comparing Gram matrices for each layer
style_loss = 0
for layer in style_weights:
    target_feature = target_features[layer]
    target_gram = gram_matrix(target_feature)
    _, d, h, w = target_feature.shape
    style_gram = style_grams[layer]
    layer_style_loss = style_weights[layer] * torch.mean((target_gram - style_gram) ** 2)
    style_loss += layer_style_loss / (d * h * w)

# Calculate total loss and update target image
total_loss = content_weight * content_loss + style_weight * style_loss
optimizer.zero_grad()
total_loss.backward()
optimizer.step()
```

Figure 3. Style Transfer core code

2.7 Normalizing Importance

The importance of normalizing Style Loss layers is critical in the style transfer process. Each layer in the model contributes differently to the style representation of the image: lower layers detect fine details, such as edges and textures, while deeper layers "capture" abstract patterns and larger scal es. Without normalization, layers with a larger number of parameters or operations can disproportionately affect the optimization process, leading to unbalanced style transfer.

To address this issue, we normalize each style loss term by dividing it by the square of the number of elements in the feature map. This normalization ensures that all selected layers contribute equally and proportionally to the style representation and prevents dominant layers from overly influencing the result. This is essential for achieving a natural and consistent transfer of artistic features across the entire image.

To evaluate the effectiveness of style transfer, we tested different scenarios using pre-trained VGG19 and AlexNet models. The goal was to test how well each model was able to preserve and implement the artistic style features of a Van Gogh painting.

2.8 Selecting layers and their role in conveying style

Neural networks "capture" different levels of abstraction depending on the depth of their layers. Shallow layers primarily detect simple patterns, such as edges and shapes, while deeper layers capture more complex elements, such as textures and object-related information. This hierarchical feature extraction plays a critical role in conveying style, as it determines how content and style information are preserved and combined.

For VGG19, we chose conv4_2 as the content layer. This is because deeper layers preserve high-level structural details while filtering out low-level textural information, ensuring that the result preserves the overall structure of the original content image. For style extraction, we used conv1_1, conv2_1, conv3_1, conv4_1. By choosing layers at different depths, we capture a wide range of stylistic elements: low-level textures from earlier layers and more abstract artistic patterns from later layers. This balance ensures that the conveyed style reflects both precise textures and general artistic characteristics.

AlexNet has a shallower architecture compared to VGG19, so its layers do not capture a deep hierarchy of features. Because of this, we chose conv4 as the content layer. This choice ensures that the content structure is preserved while maintaining some flexibility for changes. For style, we used conv1, conv2, conv3, and avoided conv4, since in a shallower network the later layers lose less textural information compared to a deep network like VGG19. By restricting the style layers to earlier stages of the convolution, we match the depth of AlexNet's feature extraction while still ensuring that stylistic elements are emphasized without compromising the content structure.

2.9 Choosing a Classification Model

To assess the quality of style transfer, we used VGG19 and AlexNet as classification models. Using two models instead of one provided a more comprehensive analysis of the effectiveness of style transfer. Each model has unique architecture and processes images in a different way, allowing us to examine how the transferred style aligns with Van Gogh's artistic characteristics at different levels of abstraction.

VGG19, as a deeper network, captures complex patterns and detailed textures, making it more sensitive to subtle stylistic elements. In contrast, AlexNet, as a shallower network, relies more on low-level features, such as edges and colors, which may make it more sensitive to general stylistic patterns than to precise details.

Another important factor is that one of these models was used directly to transfer style. This creates a built-in bias: images transferred using VGG19 may be more easily identified as Van Gogh paintings by VGG19, and vice versa with AlexNet. By using two classification models, we reduce this bias and ensure that the assessment is not overly biased in favor of the network that performed the transfer. If a single model were used, especially the one that performed the transfer, the results could be misleading, as the network might favor features that it itself created. Including a second classifier provides a necessary balance and highlights differences in how each model interprets the transfer results. Furthermore, comparing the classifications of the two models provides insights into the nature of the transfer. If both models agree that an image resembles a work by Van Gogh, this indicates a strong and consistent implementation of the style. On the other hand, if one model recognizes the conveyed style and the other does not, this indicates gaps in feature extraction and the level of abstraction at which stylistic patterns are perceived.

3.0 Results

3.1 Classification results

During the training process of the AlexNet model, particularly strong results were obtained in Fold 1, which recorded the highest Val AUC value in Epoch 9. However, Fold 5 presented the best combination of metrics, which included exceptionally high AUC and F1 values along with a low Val Loss value. Given the balance between the different metrics and consistent performance, Fold 5 was selected as the best model representative for AlexNet as Fig 4 shows.

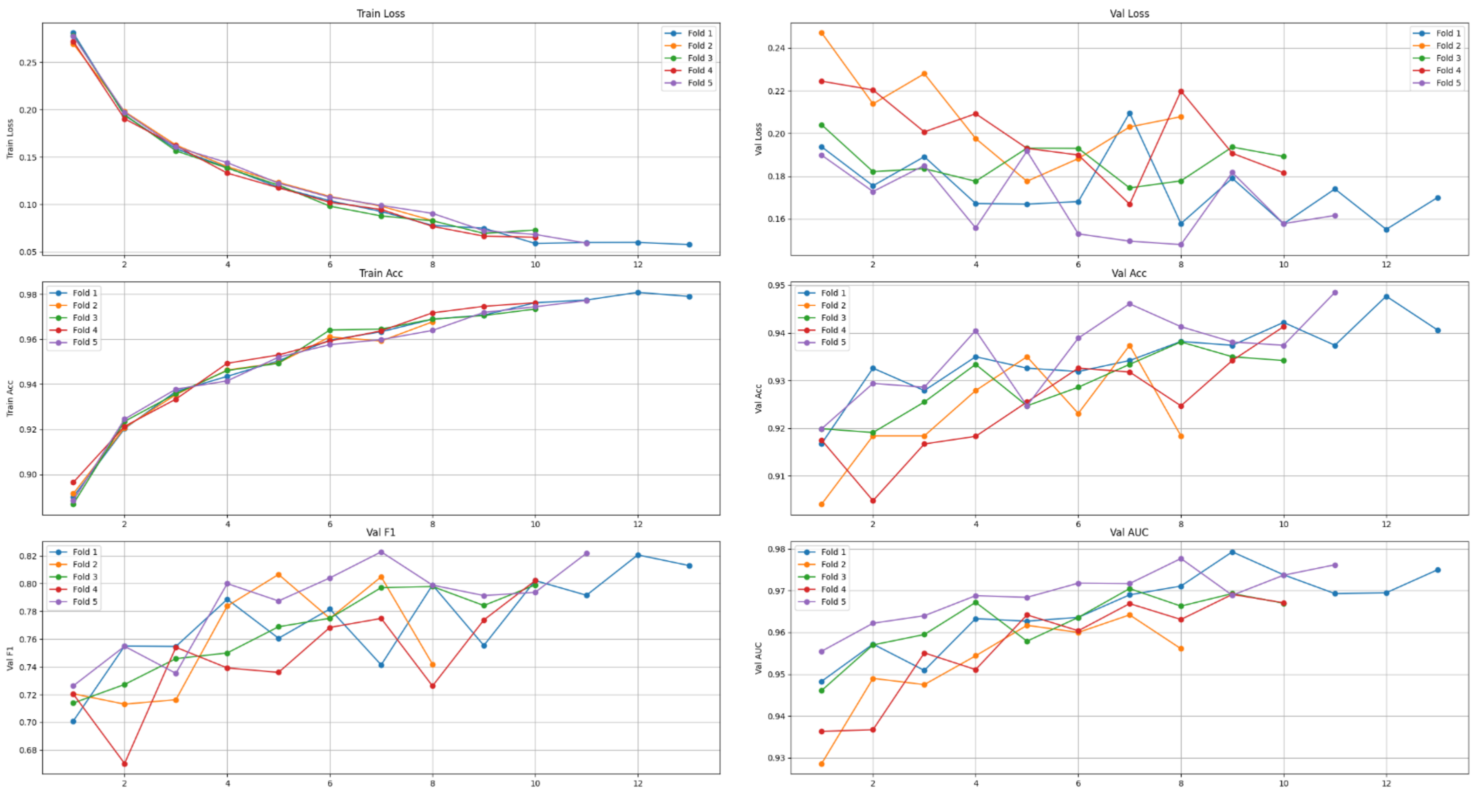


Figure 4 AlexNet training.

In training the VGG model, Fold 1 stood out with the highest Val AUC value achieved in Epoch 6, while also performing well in other metrics such as Val F1, Val Acc, and Val Loss. Fold 3 achieved a very similar performance, with almost the same AUC value and an even higher F1, while Fold 5 showed the highest F1 and Accuracy values of all folds. Nevertheless, due to its clear superiority in AUC and overall stability, Fold 1 was selected as the best model representative for VGG as Fig 5 shows.

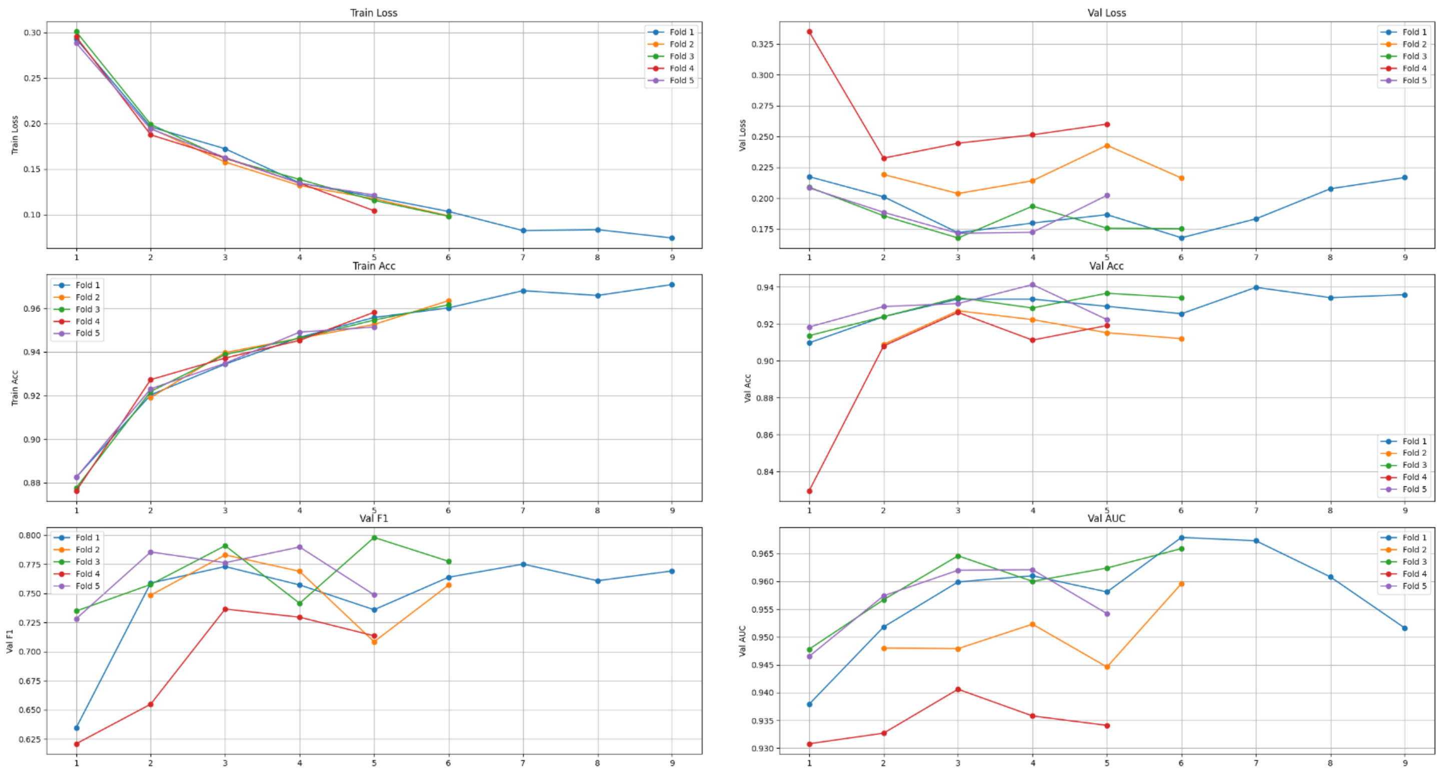
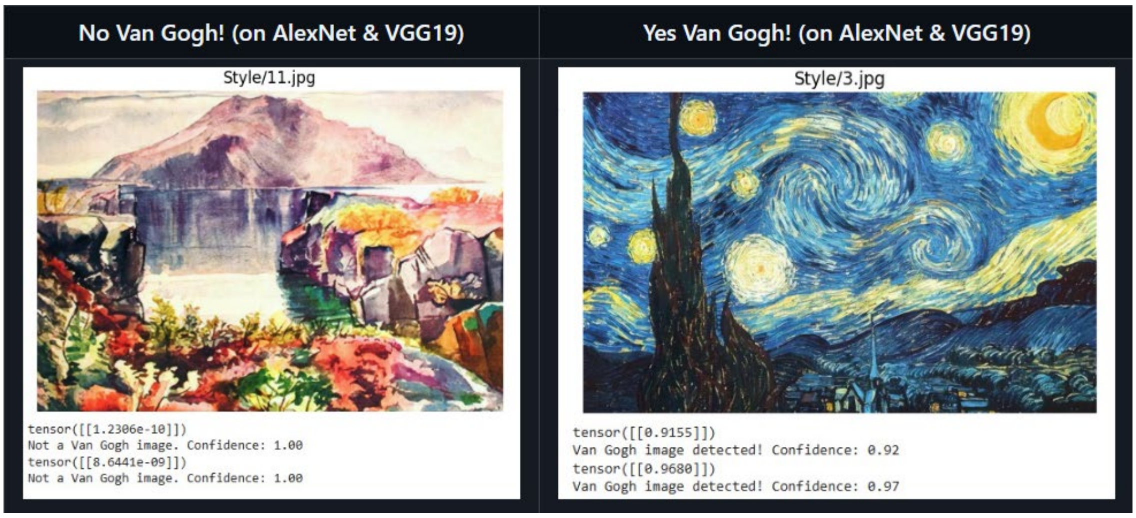


Figure 5 VGG training

These results are also presented in a slightly different format in the Jupiter notebooks included in this work and serve to further visualize the training process and model selection. Fig 6 shows the results of the classification model for the two models tested. It shows an example in which both models successfully identify a Van Gogh painting, alongside another example in which both fail to identify the painting correctly.



No Van Gogh! (on AlexNet & VGG19)

Style/11.jpg

```
tensor([[1.2306e-10]])
Not a Van Gogh image. Confidence: 1.00
tensor([[8.6441e-09]])
Not a Van Gogh image. Confidence: 1.00
```

Yes Van Gogh! (on AlexNet & VGG19)

Style/3.jpg

```
tensor([[0.9155]])
Van Gogh image detected! Confidence: 0.92
tensor([[0.9680]])
Van Gogh image detected! Confidence: 0.97
```

Figure 6 classification prosses.

3.2 Style Transfer results

In cases where both models classified the output as Van Gogh, the style transfer was convincing and successfully captured brushstrokes and textures characteristic of Van Gogh's style. This finding suggests that both networks identified key stylistic patterns consistent with those found in Van Gogh's artwork. For example, Figure 7 was classified as Van Gogh by both models.'

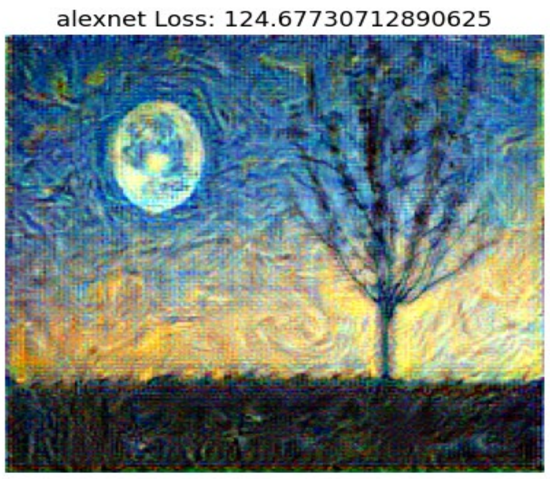


Figure 7: Successful Van Gogh style transfer

However, discrepancies arose when one model recognized the style while the other did not. When VGG19 recognized the result as Van Gogh while AlexNet did not, this indicated that VGG19 captured finer artistic details that AlexNet ignored, perhaps due to its lower sensitivity to these patterns. For example, the following image, created by style transfer with VGG19, was recognized as Van Gogh by the VGG19 model, while AlexNet classified it as "not Van Gogh". In contrast, when AlexNet identified the result as a Van Gogh but VGG19 did not, this suggested that Alene's classification was likely influenced by lower-level textural elements. For example, the following image, created by style transfer with AlexNet, was classified as "not a Van Gogh" by the VGG19 model, while AlexNet classified it as a Van Gogh, despite its strong resemblance to the previous image to the human observer. Figure 8 depicts two cat images that illustrate these examples: The first image, created using style transfer with VGG19, was identified as Van Gogh by the VGG19 model, while AlexNet classified it as “not Van Gogh.” The second image, created using style transfer with AlexNet, was classified as “not Van Gogh” by the VGG19 model, while AlexNet classified it as Van Gogh.

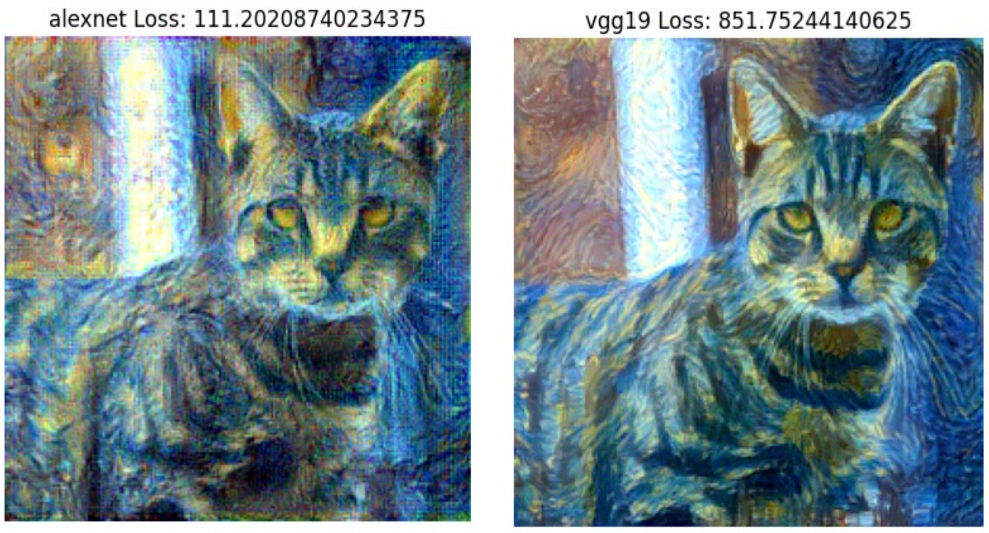


Figure 8: (Left) VGG19 styled and identified by VGG19 alone; (Right) AlexNet styled and identified by AlexNet alone

Fig 8 results suggest that each model tends to favor the stylistic patterns it was trained to recognize. In the first case, since VGG19 was used to transfer the style, its tuned classification may have been better suited to the specific textures and high-level features it applied, making it more likely to classify the result as a Van Gogh. In contrast, AlexNet, with its different feature recognition ability, may not have recognized the same stylistic elements.

Similarly, in the second case, style transfer using AlexNet produced an image that AlexNet itself recognized as a Van Gogh, but VGG19 did not. This suggests that AlexNet's transfer emphasized lower-level textures and color patterns that its model associated with Van Gogh, while VGG19, which captures more complex artistic elements, did not.

Overall, this discrepancy highlights the bias that arises from using the same model for both style transfer and classification. Each model is more likely to validate the images it creates, reinforcing the idea that different architectures capture and interpret artistic styles in different ways

In cases where neither model classified the output as Van Gogh, the style transfer may have failed to effectively capture the artistic elements of the style. Possible reasons may include the complexity of the style itself, where certain nuances of Van Gogh's technique were too subtle or complex to be captured by the models. It may also have been due to a mismatch between the content image and the style image, where the content characteristics made it difficult to convincingly apply the style. For example, Figure 9 were classified as "not Van Gogh" by both models, regardless of the model used to transfer the style.

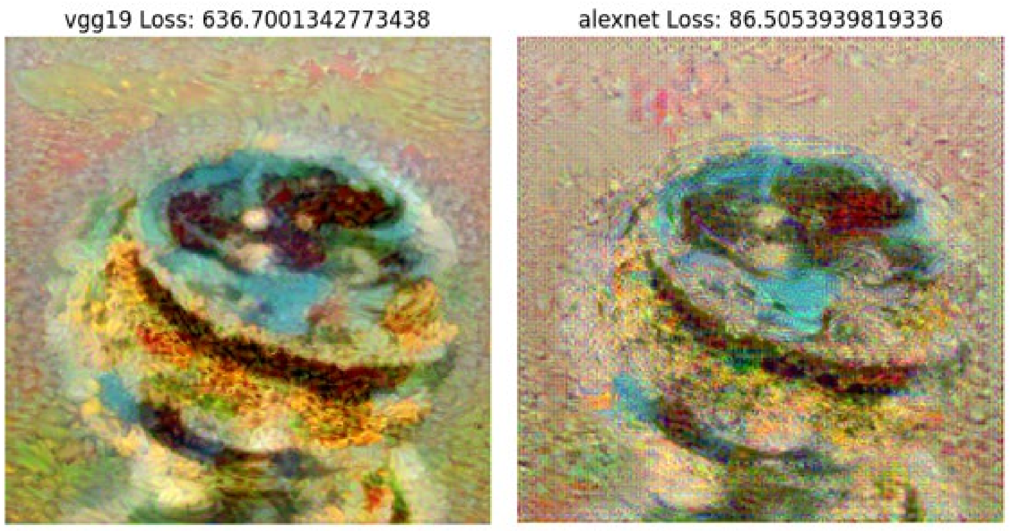


Figure 9: (Left) VGG19-style image, not identified as Van Gogh; (Right) AlexNet-style image, not identified as Van Gogh

When comparing the overall loss function during style transfer, AlexNet exhibited a lower final loss compared to VGG19. While a lower loss value generally indicates more successful optimization, it does not necessarily reflect better style transfer. In this case, the lower loss may indicate that AlexNet's style transfer process converged faster, but this may have come at the expense of fully capturing the complexity of Van Gogh's style.

Our classification models confirmed the observed differences in results. Images styled by AlexNet were more frequently classified as Van Gogh paintings, suggesting that AlexNet's transfer process applied stylistic elements in a way that was more consistent with its own architecture. In contrast, images whose style was transferred by VGG19 exhibited more diverse classifications, indicating a better balance between style and content, even if they were not always clearly identified as Van Gogh paintings. Table 1 shows this difference.

Table 1: Classification Results, VGG19 and AlexNet.

Condition	**VGG19 transfer**	**AlexNet transfer**
VGG19 identified as Van Gogh	25%	40%
AlexNet identified as Van Gogh	35%	**75%**
Both models agreed	10%	35%

The results of the classification models revealed key differences between the VGG19 and AlexNet approaches. These findings indicate that AlexNet's style transfer produced images that were more frequently classified as Van Gogh paintings, likely due to its stronger emphasis on low-level textures. In contrast, VGG19's style transfer applied a more refined combination of style and content, leading to images that were less unambiguously identified as Van Gogh paintings.

4.0 Conclusion

The current work dealt with a practical examination of two central tasks in the field of deep learning: image classification and style transfer. Two distinct models were used – AlexNet and VGG19 – which were fine-tuned for a binary classification task aimed at distinguishing between paintings by Van Gogh and paintings by other artists from the Post-Impressionist movement. By using K-Fold Cross Validation, hyperparameter optimization, and ongoing monitoring of performance metrics, well-tuned models were obtained that performed the classification task with high accuracy.

During the classification phase, it was found that in AlexNet, Fold 5 presented the best combination of AUC, F1, and Val Loss metrics, and was therefore selected as the representative model. In contrast, in VGG19, it was found that Fold 1 demonstrated the highest AUC value and the most stability in additional metrics and was therefore selected as the representative model. The selection process was also presented visually in the graphs that were attached to the work, as well as in the Jupyter Notebooks platform that accompanied the work stages (see GitHub).

In the second part of the project, the effects of style transfer using both architectures were analyzed. It was found that AlexNet, being shallower, applied a prominent and clear style transfer, which was detected more frequently by the classifiers. However, this transfer tended to be based on relatively superficial textual features. In contrast, the VGG19 model performed a more balanced and refined style transfer, which was characterized by better preservation of the source content, although it was not always detected as "Van Gogh" by the models. Comparing the results of the classifiers revealed that there is a direct relationship between the architecture that performed the transfer and the likelihood that the classifier using the same architecture would identify the image as a product of Van Gogh.

The full training process for the models took about 12 hours, performed on an ASUS TUF 15 gaming laptop equipped with an NVIDIA RTX 3060 graphics card 32 GB, RAM memory. This process included hundreds of iterations of training, early stopping, and parameter adjustment, which required high performance and efficient use of computer resources. The hyperparameters run the same scales of time.