Art Style Classification: Enhancing Accuracy with Shallow Neural Network Adapter

Lu Yingxi*
2023011435
Class 31
lu-yx23@mails.tsinghua.edu.cn

Zhu Fangyu†
2023011410
Class 32
xx@mails.tsinghua.edu.cn

Abstract

This is the abstract.

1. Introduction

In this project, we aim to address the challenge of classifying paintings according to their artistic styles. This task lies at the intersection of computer vision, machine learning, and art history, making it a multifaceted problem with significant implications for both technological and cultural domains[2]. In recent years, the automated classification of art painting styles using deep convolutional neural networks (CNNs) has become essential for analyzing and categorizing vast digitized art collections. These models can learn hierarchical visual features directly from raw image data, providing a powerful tool for art analysis[8][6]. However, accurately recognizing and distinguishing stylistic characteristics across diverse art movements remains a challenge due to high intra-class variability and inter-class similarity[1]. Developing robust classification methods is critical for scalable digital archiving, enhancing curatorial workflows, and enabling large-scale quantitative analysis of artistic trends to support art historical research.

Our project builds upon the approach proposed in[6], which involves a two-stage architecture combining a deep neural network (DNN) and a shallow neural network (SNN) adapter. The DNN acts as a feature extractor, while the SNN adapter is responsible for decision-making. Each input image is divided into five distinct patches: top-left, top-right, bottom-left, bottom-right, and center. The DNN independently classifies each of these five patches, and the SNN adapter aggregates the results to produce a final classification. This architecture allows for a more detailed examination of different regions within an artwork, capturing fine-grained information and preserving important artistic details[6]. Importantly, the SNN operates independently of the DNN, meaning that the introduction of the SNN does not alter the architecture or weights of the DNN. This independence allows the SNN to function as a flexible adapter, enhancing the classification process without imposing any architectural constraints on the DNN. Thus, the integrity and performance of the original DNN are maintained while adding an additional layer of refinement to the classification results.

By comparing the prediction results of the DNN direct output and the SNN adapter output, we observed that while the SNN achieves higher overall accuracy, the DNN direct prediction exhibits superior accuracy in certain specific classes. To integrate the strengths of both networks, we proposed a hierarchical architecture. We trained an SNN adapter using four image patches: left-top, left-bottom, right-top, and right-bottom. The final prediction is then calculated as a weighted sum of the DNN's direct prediction and the SNN adapter's output. This combined approach not only yields higher total accuracy than the original SNN adapter architecture but also results in more stable accuracy across different classes.

Furthermore, we also explored how the selection of patches affects the performance of the model. [To be continued.]

Building on the above methodology, our project makes the following key contributions:

- We implement DNN + SNN sdapter architecture proposed in [6], and proof its effectiveness by ablation study.
- We propose a hierarchical architecture that combines the strengths of both the DNN and SNN adapter, resulting in improved accuracy and stability across different classes.
- We conduct an extensive analysis of the impact of different patch selections on the model's perfor-

mance, providing insights into the optimal configuration for art style classification.
- We provide a conprehensive evaluation of our proposed method on 3 various already-trained DNN models, namely DenseNet-121, VGG-19, and ResNet-50, demonstrating its effectiveness and generalizability across various models.

2. Related Work

Discuss published work that relates to your project. How is your approach similar or different from others?

3. Method

Our method consist of three main parts: basic SNN adapter proposed in [6], hierarchical SNN architecture, and patch selection for SNN.

3.1. Basic SNN Adapter

The basic classification pipeline of base DNN with an SNN adapter is shown in Figure 1.

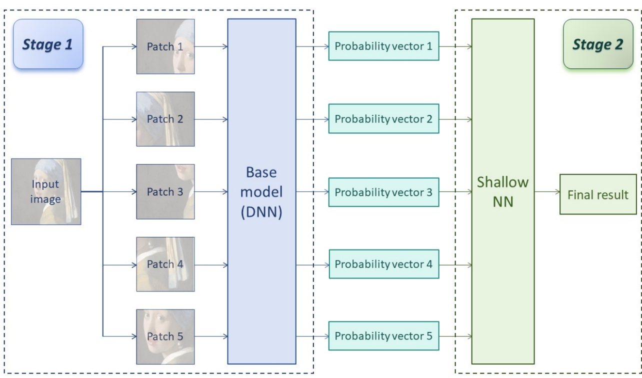

Figure 1. The architecture of the basic SNN adapter

The prediction consist of 2 stages. At the first stage, the image is dividing into 5 patches, as shown in Figure 2. The DNN independently classifies each of these five patches. In the second stage, the SNN functions as a decision-maker: it takes the probability distributions generated by the DNN for the five patches as input and produces the final classification result.

The proposed two-stage architecture combining a DNN and an SNN offers several key advantages:
- It enables a more detailed examination of different regions within an artwork. By capturing fine-grained information and preserving important artistic details, this architecture enhances classification accuracy[6].
- Using probability vectors instead of images as inputs to the SNN reduces computational costs and avoids potential errors during image processing.

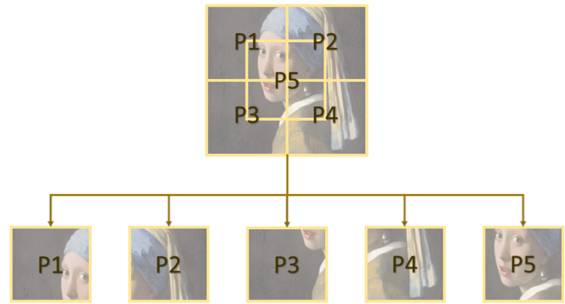

Figure 2. Patch dividing of each image

- The SNN functions as an independent decision-making adapter, making the model more flexible and generalizable. Since the DNN and SNN are trained independently, we can adjust their weights and architectures according to our needs. Each SNN adapter can fit different kinds of DNN models.

3.2. Hierarchy SNN Architecture

Despite the strong performance of the basic SNN adapter, it still faces certain limitations. Upon comparing the comprehensive prediction results of the DNN direct output and the SNN adapter output, we observed that while the SNN achieves higher overall accuracy, the DNN direct prediction exhibits superior accuracy in certain specific classes.

To integrate the strengths of both networks, we proposed a two-layer hierarchical classification architecture. This architecture aims to extract both global and local features to generate the final prediction. The prediction pipeline is illustrated in Figure 3.

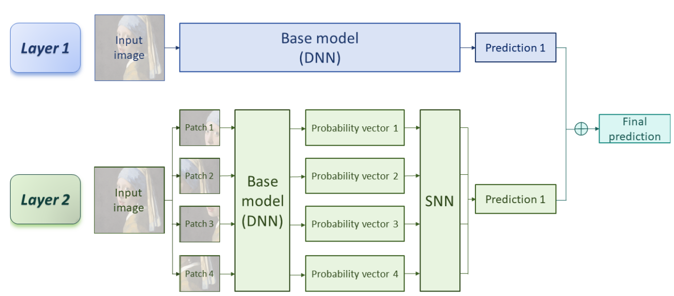

Figure 3. The architecture of the hierarchical SNN adapter

In this architecture, we employ a two-layer prediction scheme. In the first layer, the input image is fed directly into the base model (DNN), yielding the layer 1 prediction p_{layer1} in the form of a probability vector. In the second layer, the input image is divided into four patches: left-top, left-bottom, right-top, and right-bottom. The DNN independently classifies each of these patches, generating probability vectors p_1, p_2, p_3, p_4. These probability vectors are then fed into the SNN adapter, which produces the layer 2 prediction p_{layer2}, also in the form of a probability vector.

The final prediction is a weighted sum of the predictions from the two layers:

$$p_{\text{final}} = w_1 \cdot p_{\text{layer1}} + w_2 \cdot p_{\text{layer2}}, \tag{1}$$

where the weights w_1 and w_2 are proportional to the "reliability" of the respective layers. As suggested in [5], the "reliability" can be evaluated based on the entropy of the probability vectors. Specificly, the entropy of the layer 1 prediction is calculated as:

$$H^1 = -\sum_{i=1}^{C} p_{\text{layer1}}(i) \cdot \log(p_{\text{layer1}}(i)), \tag{2}$$

and the corresponding weight w_1 is given by:

$$w_1 = 1 + 1/\exp(H^1), \tag{3}$$

For the second layer, the average probability vector $\overline{p}_{\text{layer2}}$ is fist computed as:

$$\overline{p}_{\text{layer2}} = \frac{1}{4}(p_1 + p_2 + p_3 + p_4). \tag{4}$$

The entropy of the average probability vector is then calculated as:

$$H^2 = -\sum_{i=1}^{C} \overline{p}_{\text{layer2}}(i) \cdot \log(\overline{p}_{\text{layer2}}(i)), \tag{5}$$

and the corresponding weight w_2 is given by:

$$w_2 = 1 + 1/\exp(H^2). \tag{6}$$

3.3. Patch Selection for SNN

4. Experiment

In this section, we present an comprehensive evaluation of our proposed method.

4.1. Experiment Setup

4.1.1. Dataset

We utilize the Painter by Numbers dataset [7] as the primary source of paintings for our study. This dataset is a vast collection of approximately 23,000 unique paintings, each labeled with the painter's name, style (or movement), genre, and additional information[3]. For each experiment, we select a subset of 4,000 to 5,000 paintings from the training data. Additionally, we use 500 paintings from the test set for evaluation purposes.

4.1.2. Evaluation

In order to get a comprehensive understanding of the model's performance, we evaluate the model using four metrics: accuracy, precision, recall and F1 score.

- Accuracy: The ratio of correctly predicted instances to the total instances in the dataset.
- Precision: The ratio of true positive predictions to the total predicted positives. It indicates the quality of the positive predictions.
- Recall: The ratio of true positive predictions to the total actual positives. It measures the model's ability to identify all relevant instances.
- F1 Score: The harmonic mean of precision and recall. It provides a balance between precision and recall, especially useful when dealing with imbalanced datasets.

Moreover, we not only evaluate the overall model performance but also analyze each class's performance. This provides deeper insights into the model's strengths and weaknesses across categories, helping identify specific areas needing improvement and ensuring consistent performance across all classes.

4.1.3. Baseline

We evaluate out SNN adapter on 3 various already-trained DNN models, DenseNet-121, VGG-19, and ResNet-50.

- DenseNet-121: We adopt ArtNet model[3] trained with DenseNet-121 architecture as our first baseline A key feature of ArtNet is its preprocessing strategy, where each input image is first padded to a uniform size and then divided into five patches. Both the full image and its patches are included in the training set, enabling the model to learn from global composition as well as local texture details, making it a perfect choice for our baseline model.
- VGG-19: We utilize the model trained by [9] as our baseline for VGG-19. This model is trained on the same dataset as our method, with a test accuracy lower than ArtNet.
- ResNet-50: We also employ the pretrained ResNet-50 model from [4]. Our results show that even if the base model's performance is subpar, the SNN adapter can still significantly enhance the overall performance.

5. Conclusion

References

[1] Moritz Alkofer. Using convolutional neural networks to compare paintings by style. Intersect: The Stanford Journal of Science, Technology, and Society, 15(1), 2021. 1

[2] Eva Cetinic, Tomislav Lipic, and Sonja Grgic. Fine-tuning convolutional neural networks for fine art classification. Expert Systems with Applications, 114:107–118, 2018. 1

[3] Rendi Chevi. Artnet - painting style classification and feature visualization. https://github.com/rendchevi/artnet-app. Accessed: 2025-04-24. 3

[4] Kayra Coşkun. Art style classification with transfer learning (vgg16 and resnet50v2). https://github.com/kayracoskun/Art_Style_Classification/tree/main, 2023.

[5] Kai-Lung Hua, Trang-Thi Ho, Kevin-Alfianto Jangtjik, Yu-Jen Chen, and Mei-Chen Yeh. Artist-based painting classification using markov random fields with convolution neural network. Multimedia Tools and Applications, 79:12635–12658, 2020. 3

[6] Saqib Imran, Rizwan Ali Naqvi, Muhammad Sajid, Tauqeer Safdar Malik, Saif Ullah, Syed Atif Moqurrab, and Dong Keon Yon. Artistic style recognition: combining deep and shallow neural networks for painting classification. Mathematics, 11(22):4564, 2023. 1, 2

[7] Kaggle. Painter by numbers. https://www.kaggle.com/c/painter-by-numbers/data. Accessed: 2025-04-24. 3

[8] Weiwei Li. Enhanced automated art curation using supervised modified cnn for art style classification. Scientific Reports, 15(1):7319, 2025. 1

[9] Qchaldemer. painting-classification: Classifier of paintings depending on their style (impressionism, barocco, ...). https://github.com/qchaldemer/painting-classification/tree/master, 2023. 3