ArtDecode: Leveraging Convolutional Neural Networks for Multi-Class Art Style Classification

Colinares, Gian Karl L.
College of Computing and Information Technologies
National University
Sampaloc, Manila
colinaresgl@students.national-u.edu.ph

Foryasen, Blix D.
College of Computing and Information Technologies
National University
Sampaloc, Manila
foryasenbd@students.national-u.edu.ph

Lim, Jerico C.
College of Computing and Information Technologies
National University
Sampaloc, Manila
limjc@students.national-u.edu.ph

Mapanao, Jessy Cassandra M.
College of Computing and Information Technologies
National University
Sampaloc, Manila
mapanaojm@students.national-u.edu.ph

***Abstract*—Challenges in art style classification persist due to the subjective nature of manual analysis and the limitations of automated methods in discerning subtle artistic nuances. This study addresses these issues by introducing an automated art style classification approach that leverages advanced computer vision and a convolutional neural network (CNN) model. Inspired by architectures such as AlexNet, VGGNet, and ResNet, the CNN employed in this study was optimized to effectively capture the complexities of diverse artistic features. Findings indicate that the VGG-13 model achieved the highest accuracy at 63.08%. This research holds significant potential for enhancing art education by making art historical knowledge more accessible and engaging for a wider audience.**

***Index Terms*—Art style, Image Classification, VGG-13, CNN**

I. Introduction

Classifying art styles is fundamental to art historical research, curatorial practices, and aesthetic appreciation, offering a crucial framework for understanding artistic evolution and cultural context. Historically, this intricate task has relied heavily on art historians' specialized knowledge and interpretive skills. Concurrently, the past decade has witnessed significant advancements in computer vision and deep learning, leading to transformative breakthroughs in image classification across diverse domains. However, a persistent challenge remains in bridging the inherently objective nature of digital image representation with the complex, often subjective, conceptualizations of art styles [1].

To address this, numerous researchers have explored automated art style classification. Machine learning models, particularly Convolutional Neural Networks (CNNs) and various pre-trained architectures, have emerged as prominent tools for artwork identification. These approaches frequently involve fine-tuning parameters and optimizing network architectures to discern art style elements. Prior studies have effectively utilized features such as distinct color palettes [2] and characteristic brushstroke patterns [3], both crucial elements in differentiating art styles. Despite these efforts, current automated methodologies, alongside traditional manual classification, face notable limitations. Manual art classification requires extensive artistic expertise and is inherently time-consuming, while existing computational models often struggle with the nuanced and subtle variations within art styles, limiting their robustness and scalability across vast and diverse art datasets.

This study addresses these limitations by leveraging computer vision, specifically advanced image classification techniques, for the accurate and automated identification of art styles from digitized artworks. This paper proposes to contribute to existing art style classification frameworks by developing an improved convolutional neural network (CNN) architecture, meticulously optimized to discern the complexities of diverse artistic features. It aims to deliver a highly efficient and accurate preliminary tool that can significantly assist art experts in identifying stylistic inconsistencies, thereby streamlining and enhancing processes for art authentication and attribution. Furthermore, this research holds substantial potential to impact art education, making art historical knowledge more accessible and engaging for broader audiences, including those without a formal art background.

II. Related Work

A. Role of CNNs in Classifying Artistic Styles

Convolutional Neural Networks (CNNs) have become foundational in the field of image classification due to their ability to learn hierarchical visual features. In the context of art, CNNs have been widely adopted to classify artworks by style, genre, artist, and medium. Their layered architecture allows them to capture intricate patterns in brushstrokes, color palettes, and composition, and features that are essential for distinguishing between artistic styles. Studies such as those by Cetinic et al. [4] and DuBois [5] demonstrate the effectiveness

Identify applicable funding agency here. If none, delete this.

of CNNs in art classification, achieving accuracies above 80 percent when trained on curated datasets. CNNs have also been extended with attention mechanisms [6], patch-based learning [7], and multi-scale architectures [8], [9] to further enhance their performance in this domain.

B. Key Factors and Challenges Influencing Model Accuracy

Convolutional Neural Networks (CNNs) face key challenges in art classification, notably due to dataset imbalance and scarcity, which can bias learning and hinder generalization for underrepresented styles [5], [10]. Visual similarities between certain styles, such as Impressionism and Post-Impressionism, further complicate classification by blurring stylistic boundaries [10], [11]. Real-world noise like compression artifacts also degrades performance, though some studies, such as Xu and Xu [12], have proposed architectural optimizations to improve robustness. Additionally, while high-performing models like ResNet-50-NTS [6] and BiT [13] offer strong accuracy, their computational demands and the limited effectiveness of training from scratch without transfer learning [14] restrict practical deployment.

C. Comparative Analysis of the Accuracy of Classification Models

Several studies reported classification accuracies below 70 percent, often due to methodological constraints. Iliadis et al. [14] trained Vision Transformers and MLP Mixers from scratch on 21 art style classes, achieving only 39 percent accuracy due to the absence of pre- training and the inherent complexity of the task. Falomir et al. [11] employed qualitative color descriptors with traditional classifiers like k-NN and SVM, reaching about 65 percent accuracy, though their models lacked the representational power of deep learning approaches. Zhao and Zhang [15] explored few- shot learning using the Difference Component model, which improved performance but still struggled with generalization due to limited data. In contrast, transfer learning approaches by Cetinic et al. [A], Sandoval et al. [7], and Zhao et al. [13] achieved significantly higher accuracies (75–95 percent) by leveraging pre-trained models on large datasets like ImageNet. Notably, Li [16] attained 93 percent accuracy with a custom CNN trained from scratch, though this required meticulous architectural design and extensive hyperparameter tuning.

D. Review of Gaps and Implications for Experimental Design

Transfer learning, while effective, often introduces a semantic mismatch between source domains like object recognition and target domains such as art style classification. Additionally, many top-performing models are computationally intensive, making them unsuitable for deployment in resource-limited settings. To address these issues, our study proposes a one-stage training approach using a CNN trained from scratch on the WikiArt dataset, designed to capture domain-specific features without external pre-training. This streamlined architecture aims to balance simplicity, training efficiency, and classification accuracy, contributing to the development of scalable and interpretable models for art classification.

III. Methodology

A. Dataset Description

The dataset utilized for this study is the WikiArt dataset, acquired from Kaggle and originally compiled by Kaggle User Stefano Morelli from WikiArt. This is a comprehensive collection of digitized artworks, representing 80,020 artworks from 1,119 distinct artists across 27 different art styles.

However, there exists a massive class imbalance between art styles where artworks have as little as 98 (Action Painting) or as large as 13,100 (Impressionism). Such a gap comes from the varying number of artists primarily practicing the art style throughout history. Moreover, artworks vary in quality, attributed to the deterioration of materials used in paintings over the passage of time. Artists also contribute to multiple art styles such as Pablo Picasso's artworks existing in Cubism, Fauvism, and Impressionism. Both instances contribute to noise and may affect generalization capability of the model.

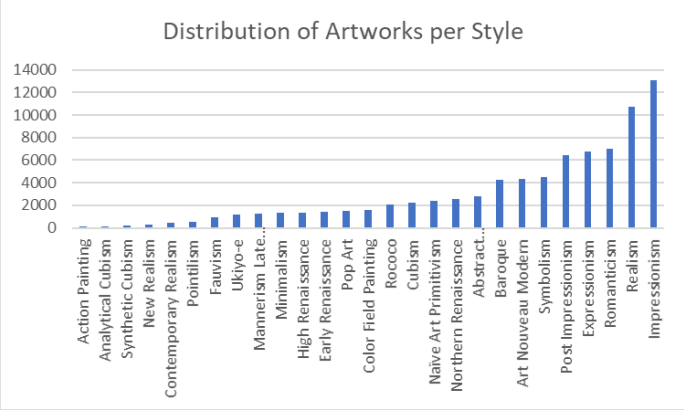

Fig. 1. Initial Distribution of Artworks per Style

Various preprocessing treatments were applied to minimize the effects of class imbalance and improve the quality of the data which shall be discussed further on in this section. A stratified train-test-validation split of 70-15-15 was implemented in the model architecture, ensuring an adequate training data and validation split across all art styles to minimize overfitting and potential biases.

B. Experimental Setup

Two computers were utilized to train models of varying complexity locally through Visual Studio Code. An RTX5070Ti with 16 GB of VRAM served bigger, more complex model architectures, while an RTX3060 with 12GB of VRAM trained simpler, smaller models. Both setups used a TensorFlow 2.10 and Python 3.10 Conda environment that supports GPU model training on a Windows Native Operating System.

C. Data Preprocessing

With the presence of class imbalance and noise within the original dataset, several preprocessing steps are employed

to reach the optimal model performance and limit possible classification bias.

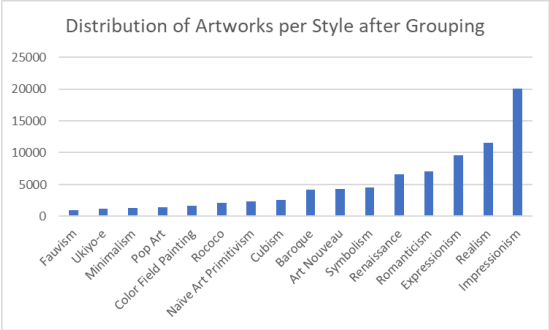

Fig. 2. Number of Art Styles after Grouping

Grouping of art styles

Some art styles in the dataset are either progressions over time of a single art style (e.g. Early and High Renaissance) or substyles of a larger artstyle (e.g. Analytical and Synthetic Cubism). Both instances are combined into a single group to reduce the number of classes to be predicted, focusing on the general features of a specific art movement instead of its progression over time. The following are grouped together:

- Impressionism (Impressionism, Post-Impressionism, Pointillism)
- Expressionism (Expressionism, Abstract Expressionism, Action Painting)
- Cubism (Cubism, Synthetic Cubism, Analytical Cubism
- Realism (Realism, New Realism, Contemporary Realism)
- Renaissance (Early, High, Northern, and Late Renaissance)

Grouping of art styles reduced the number of classes from 27 to 16. These styles include Art Nouveau, Baroque, Color Field Painting, Cubism, Expressionism, Fauvism, Impressionism, Minimalism, Naive Art Primitivism, Pop Art, Realism, Renaissance, Rococo, Romanticism, Symbolism, and Ukiyo-e. Each art style is organized into its respective subdirectory within a base directory.

Reduction of Artworks per Style

Even after grouping art styles, there exists a massive class imbalance between art styles with Fauvism only having 934 samples compared to Impressionism with 20010. To have equal representation of classes, samples are reduced down to 900 samples to match the number contained within Fauvism. Artworks from pioneering and defining artists for each art movement are selected and added up to help ensure distinction between art styles. For example, Pablo Picasso is assigned to Cubism as a pioneering artist, making all of his artworks from other art styles not considered in sampling.

Data Cleaning

In combination with reducing the sample size, manual data cleaning was conducted to select high-quality paintings and artworks. Non-paintings (e.g. sculptures and architecture), sketches or Drawings, Paintings with large frames or walls, and black and white paintings are omitted from each selected artist's artwork. It is worth noting that due to the varying art production and conservation across the centuries, some paintings of certain styles are in greater quality than others; thus, the classes are not balanced in terms of quality of painting. After cleaning, each art style consists of around 900 samples, further broken down into around 630 training samples and 135 samples each for validation and testing.

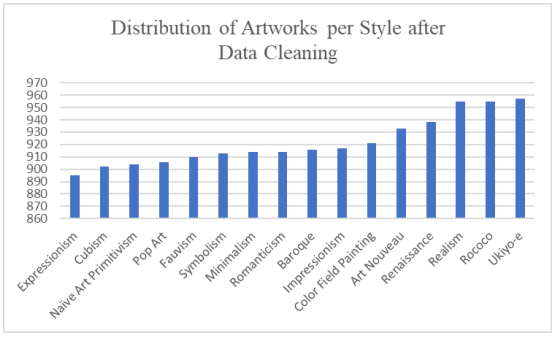

Fig. 3. Distribution of Artworks per Style after Data Cleaning

Style	**# of Training Samples**	**# of Test & Validation Samples**
Art Nouveau	653	140
Baroque	641	137
Color Field Painting	645	138
Cubism	631	136
Expressionism	626	135
Fauvism	637	137
Impressionism	642	137
Minimalism	640	137
Naive Art Primitivism	633	136
Pop Art	634	136
Realism	668	143
Renaissance	657	137
Rococo	669	137
Romanticism	640	137
Symbolism	639	137
Ukiyo-e	670	143

TABLE I
DISTRIBUTION OF STRATIFIED TRAIN, TEST, AND VALIDATION SAMPLING PER STYLE

Image Normalization

All images loaded from the base directory were loaded and decoded into 3-channel (RGB) tensor representations. Each decoded image is uniformly resized to a fixed spatial resolution of 224x224 pixels to be normalized into the [0,1] range by dividing by 255.0. Furthermore, the training dataset is shuffled with a buffer size of 1024, randomizing the order of training samples in each epoch for improving model generalization. All datasets are subsequently batched into fixed-size chunks of 32 samples.

D. Model Architecture

The study focused on training and comparing CNN models loosely based on model architectures of state-of-the-art CNNs, specifically AlexNet, VGGNet, and ResNet.

AlexNet

Model: "AlexNet"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 56, 56, 64)	23,296
max_pooling2d (MaxPooling2D)	(None, 27, 27, 64)	0
conv2d_1 (Conv2D)	(None, 27, 27, 192)	307,392
max_pooling2d_1 (MaxPooling2D)	(None, 13, 13, 192)	0
conv2d_2 (Conv2D)	(None, 13, 13, 384)	663,936
conv2d_3 (Conv2D)	(None, 13, 13, 256)	884,992
conv2d_4 (Conv2D)	(None, 13, 13, 256)	590,080
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 256)	0
global_average_pooling2d (GlobalAveragePooling2D)	(None, 256)	0
dense (Dense)	(None, 512)	131,584
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 512)	262,656
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 16)	8,208

Total params: 2,872,144 (10.96 MB)
Trainable params: 2,872,144 (10.96 MB)
Non-trainable params: 0 (0.00 B)

Fig. 4. Modified AlexNet Architecture

AlexNet consists of five convolutional layers and three fully connected layers. The initial two convolutional layers utilize 64 and 192 filters, respectively, followed by max-pooling for spatial downsampling. Subsequent convolutional layers progressively increase in depth, using 384, 256, and 256 filters, distributed with max-pooling operations. Instead of a large flattened layer, the modified architecture uses a GlobalAveragePooling2D layer after the final convolutional block, effectively reducing the feature maps to a 256-unit vector, leading to the classification head. The classification head comprises two 4096-unit Dense layers, each followed by Dropout (0.5) layers for regularization, culminating in a 16-unit Dense Classification output layer.

VGGNet (VGG-11 and VGG-13)

Layer (type)	Output Shape	Param #
conv2d_5 (Conv2D)	(None, 224, 224, 64)	1,792
batch_normalization (BatchNormalization)	(None, 224, 224, 64)	256
activation (Activation)	(None, 224, 224, 64)	0
max_pooling2d_3 (MaxPooling2D)	(None, 112, 112, 64)	0
conv2d_6 (Conv2D)	(None, 112, 112, 128)	73,856
batch_normalization_1 (BatchNormalization)	(None, 112, 112, 128)	512
activation_1 (Activation)	(None, 112, 112, 128)	0
max_pooling2d_4 (MaxPooling2D)	(None, 56, 56, 128)	0
conv2d_7 (Conv2D)	(None, 56, 56, 256)	295,168
batch_normalization_2 (BatchNormalization)	(None, 56, 56, 256)	1,024
activation_2 (Activation)	(None, 56, 56, 256)	0
conv2d_8 (Conv2D)	(None, 56, 56, 256)	590,080
batch_normalization_3 (BatchNormalization)	(None, 56, 56, 256)	1,024
activation_3 (Activation)	(None, 56, 56, 256)	0
max_pooling2d_5 (MaxPooling2D)	(None, 28, 28, 256)	0
conv2d_9 (Conv2D)	(None, 28, 28, 512)	1,180,160
batch_normalization_4 (BatchNormalization)	(None, 28, 28, 512)	2,048
activation_4 (Activation)	(None, 28, 28, 512)	0
conv2d_10 (Conv2D)	(None, 28, 28, 512)	2,359,808
batch_normalization_5 (BatchNormalization)	(None, 28, 28, 512)	2,048
activation_5 (Activation)	(None, 28, 28, 512)	0
max_pooling2d_6 (MaxPooling2D)	(None, 14, 14, 512)	0
conv2d_11 (Conv2D)	(None, 14, 14, 512)	2,359,808
batch_normalization_6 (BatchNormalization)	(None, 14, 14, 512)	2,048
activation_6 (Activation)	(None, 14, 14, 512)	0
conv2d_12 (Conv2D)	(None, 14, 14, 512)	2,359,808
batch_normalization_7 (BatchNormalization)	(None, 14, 14, 512)	2,048
activation_7 (Activation)	(None, 14, 14, 512)	0
max_pooling2d_7 (MaxPooling2D)	(None, 7, 7, 512)	0
global_average_pooling2d_1 (GlobalAveragePooling2D)	(None, 512)	0
batch_normalization_8 (BatchNormalization)	(None, 512)	2,048
dense_3 (Dense)	(None, 512)	262,656
batch_normalization_9 (BatchNormalization)	(None, 512)	2,048
dropout_2 (Dropout)	(None, 512)	0
dense_4 (Dense)	(None, 512)	262,656
batch_normalization_10 (BatchNormalization)	(None, 512)	2,048
dropout_3 (Dropout)	(None, 512)	0
dense_5 (Dense)	(None, 16)	8,208

Total params: 9,771,152 (37.27 MB)
Trainable params: 9,762,576 (37.24 MB)
Non-trainable params: 8,576 (33.50 KB)

Fig. 5. Modified VGG-11 Architecture

VGG Models utilize stacked 3x3 convolutional filters and max-pooling. The modified VGG-11 in this study incorporates BatchNormalization and ReLU after each convolutional layer, enhancing training stability. A key difference is using GlobalAveragePooling2D in the classification head, replacing traditional large flattened layers to reduce parameters significantly. The head includes two 512-unit Dense layers with batch normalization and Dropout(0.3) for regularization, ending in a 16-unit softmax output. To increase depth, the modified VGG-13 architecture extends VGG-11 by adding an extra convolutional layer within blocks 1 and 2. L2 kernel regularization (0.001) exists throughout all layers. The modified VGG-11 and VGG-13 consist of 9,771,152 and 9,956,432 parameters, respectively. This architectural refinement not only improves generalization by reducing overfitting through regularization techniques but also enhances computational efficiency with fewer parameters and streamlined operations. The deeper VGG-13 variant allows for more complex feature extraction, potentially boosting performance on intricate classification tasks.

ResNet-18

```python
def conv_block(x, filters, kernel_size=3, strides=1, use_projection=False, weight_decay=1e-3, name=None):
    l2 = regularizers.l2(weight_decay)

    shortcut = x
    if use_projection:
        shortcut = layers.Conv2D(filters, 1, strides=strides, padding='same',
                                 kernel_regularizer=l2, name=f'{name}_proj')(x)
        shortcut = layers.BatchNormalization(name=f'{name}_proj_bn')(shortcut)

    x = layers.Conv2D(filters, kernel_size, strides=strides, padding='same',
                      kernel_regularizer=l2, name=f'{name}_conv1')(x)
    x = layers.BatchNormalization(name=f'{name}_bn1')(x)
    x = layers.ReLU()(x)

    x = layers.Conv2D(filters, kernel_size, strides=1, padding='same',
                      kernel_regularizer=l2, name=f'{name}_conv2')(x)
    x = layers.BatchNormalization(name=f'{name}_bn2')(x)

    x = layers.Add()([shortcut, x])
    x = layers.ReLU()(x)
    return x

def build_resnet18_gradcam(input_shape=(224, 224, 3), num_classes=1000, dropout_rate=0.5, weight_decay=1e-3):
    inputs = tf.keras.Input(shape=input_shape)
    l2 = regularizers.l2(weight_decay)

    # Initial Conv + MaxPool
    x = layers.Conv2D(64, 7, strides=2, padding='same', kernel_regularizer=l2, name='conv1')(inputs)
    x = layers.BatchNormalization(name='bn_conv1')(x)
    x = layers.ReLU()(x)
    x = layers.MaxPooling2D(3, strides=2, padding='same')(x)

    # Conv2_x
    x = conv_block(x, 64, strides=1, use_projection=False, weight_decay=weight_decay, name='conv2_1')
    x = conv_block(x, 64, strides=1, use_projection=False, weight_decay=weight_decay, name='conv2_2')

    # Conv3_x
    x = conv_block(x, 128, strides=2, use_projection=True, weight_decay=weight_decay, name='conv3_1')
    x = conv_block(x, 128, strides=1, use_projection=False, weight_decay=weight_decay, name='conv3_2')

    # Conv4_x
    x = conv_block(x, 256, strides=2, use_projection=True, weight_decay=weight_decay, name='conv4_1')
    x = conv_block(x, 256, strides=1, use_projection=False, weight_decay=weight_decay, name='conv4_2')

    # Conv5_x
    x = conv_block(x, 512, strides=2, use_projection=True, weight_decay=weight_decay, name='conv5_1')
    x = conv_block(x, 512, strides=1, use_projection=False, weight_decay=weight_decay, name='conv5_2')
    last_conv_output = x  # For Grad-CAM

    # Classification head
    x = layers.GlobalAveragePooling2D()(x)
    x = layers.Dropout(dropout_rate)(x)
    x = layers.Dense(16, activation='softmax', kernel_regularizer=l2, name='fc')(x)

    model = models.Model(inputs=inputs, outputs=x, name='ResNet18')
    return model
```

Fig. 6. Model Definition of Modified ResNet-18 Architecture

ResNet (Residual Network) introduced ”residual connections” or ”skip connections,” which enable gradients to flow directly through the neural network, mitigating the vanishing/exploding gradient problem commonly seen in deep models. The Modified ResNet-18 architecture utilizes an initial 7x7 convolutional layer with a stride of 2, followed by BatchNormalization, ReLU, and 3x3 max-pooling. The network’s core consists of four stages, each containing two residual blocks. These blocks employ two 3x3 convolutional layers with BatchNormalization and ReLU activations. A key modification from standard identity shortcuts is using 1x1 convolutional projections for shortcuts when dimensions change (strides=2) or filter depth increases, ensuring dimension matching for residual addition. Filter counts progressively increase from 64 to 512 across the stages. The classification head employs GlobalAveragePooling2D directly after the last convolutional block, followed by a Dropout layer (0.5 rate) and a final 16-unit Dense layer with softmax activation. L2 regularization (1e-3) is consistently applied to all convolutional and dense layer kernels, totalling 11,199,120 parameters.

E. Model Training

Considering compute capability, AlexNet and VGG-11 were trained in the RTX3060 environment, while the RTX5070Ti environment handled both VGG-13 and ResNet18. All models follow a consistent methodology to ensure comparison, optimizing for robust performance and generalization. Each model used Adam as optimizer, initialized with a base learning rate of 1×104. Categorical Cross-Entropy (CCE), coupled with a 0.1 Label smoothing, served as the loss function employed in each training job. Label Smoothing introduces a small amount of uniform noise to the one-hot encoded labels, which helps to regularize the model and prevent it from becoming overconfident in its predictions.

Accuracy served as the primary metric for monitoring model performance. Training persisted until 1000 epochs, with batch sizes consistently set to 32. EarlyStopping and ReduceLROnPlateau aided in preventing overfitting and optimizing training efficiency. EarlyStopping monitored validation loss with a patience of 15 epochs and a minimum change of 1e-5, restoring the best weights if validation loss did not improve. ReduceLROnPlateau also monitored val_loss with a patience of 10 epochs and a minimum change of 1e-5, reducing the learning rate by 0.5 when the validation loss plateaued, with a minimum learning rate set to 1e-7. The training process used the train_ds for training data and val_ds for validation data.

F. Model Evaluation

Model evaluation was conducted through a multifaceted approach to comprehensively assess performance and generalization and identify specific class-wise strengths and weaknesses. Training progress was visually monitored by plotting the training and validation accuracy and loss curves across epochs, providing insights into convergence and the presence of overfitting or underfitting.

Test loss and accuracy on the unseen test dataset define the quantitative evaluation of the final model’s performance. Predictions allow for a better analysis of each art style’s classification performance. Classification reports provided class-wise metrics, including precision, recall, and F1-score. By definition,

$$\text{Precision} = \frac{TP}{TP + FP}$$

Precision quantifies the accuracy of correct predictions for a specific class, formally defined as the ratio of true positive (TP) predictions to the total number of positive predictions made by the model for that class (TP + False Positives (FP)). High precision indicates a low rate of false negatives for that specific class.

$$\text{Recall} = \frac{TP}{TP + FN}$$

Recall, also known as sensitivity, quantifies a model’s ability to identify all relevant instances of a particular class. Sensitivity represents the ratio of true positive (TP) predictions to the total number of actual positive instances of that class (TP + False Negatives (FN). High recall suggests that the model is effectively capturing most of the true cases of a class.

$$\text{F1-Score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

F1-score represents the harmonic mean of precision and recall, providing a single metric that balances both, especially in cases where there might be an uneven class distribution. A

higher value indicates better performance and is less susceptible to the impact of class imbalance than accuracy.

Finally, a Confusion Matrix was computed and visualized as a heatmap to graphically represent the counts of correct and incorrect predictions for each class of the best model. This matrix identified which classes were frequently confused with one another, offering valuable insights into the model's discriminative capabilities across different art styles.

IV. Results and Discussion

The following chapter discusses the findings from training to model predictions of CNN model architectures employed in the methodology, in order of complexity. Loss and Accuracy Graphs portray the model's performance over each training epoch, while classification reports analyze accuracy, top-predicted classes, and models' potential weak points or biases. A confusion matrix visually summarizes the best model's performance across all classes. All models have been trained with a batch size of 32, 224x224 image size, and optimized through Adam with a starting learning rate of 1e-4.

A. AlexNet

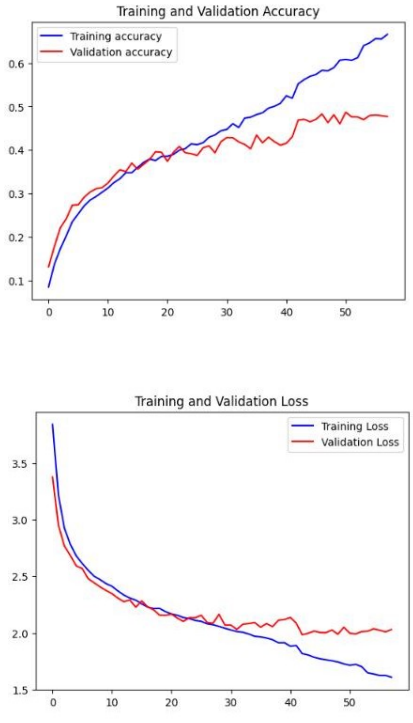

Fig. 7. Accuracy and Loss Graphs of AlexNet

AlexNet's training accuracy improved, reaching over 60% as training progressed, while the training loss steadily decreased. Validation accuracy plateaued around 45-50% and began to diverge significantly from the training accuracy, indicating a struggle to generalize to unseen data. Correspondingly, validation loss leveled and showed upward fluctuations, failing to keep up with the continuous decrease of training loss, pointing to overfitting.

	precision	recall	f1-score	support
Art Nouveau	0.47	0.41	0.44	140
Baroque	0.50	0.42	0.46	138
Color Field Painting	0.75	0.61	0.67	138
Cubism	0.56	0.39	0.46	135
Expressionism	0.29	0.30	0.29	134
Fauvism	0.44	0.32	0.37	136
Impressionism	0.47	0.86	0.61	138
Minimalism	0.62	0.54	0.58	137
Naive Art Primitivism	0.36	0.24	0.28	135
Pop Art	0.45	0.43	0.44	136
Realism	0.32	0.20	0.25	144
Renaissance	0.45	0.62	0.52	141
Rococo	0.41	0.60	0.49	143
Romanticism	0.42	0.45	0.44	137
Symbolism	0.20	0.16	0.18	137
Ukiyo_e	0.68	0.84	0.75	144
accuracy			0.46	2213
macro avg	0.46	0.46	0.45	2213
weighted avg	0.46	0.46	0.45	2213

Fig. 8. Classification Report for AlexNet

With an overall accuracy of 46% at epoch 43, AlexNet shows significant performance variability across different art styles. Color Field Painting and Ukiyo-e demonstrated comparatively stronger performance than other styles, with an F1-score of 0.67 and 0.75, respectively. Conversely, Symbolism, Realism, Naive Art Primitivism, and Expressionism exhibited weaker performance with an F1-score less than 0.30, indicating challenges in effectively distinguishing these styles. Significant imbalances exist between precision and recall for Impressionism, achieving a high recall (0.86) but a lower precision (0.47).

B. VGG-11

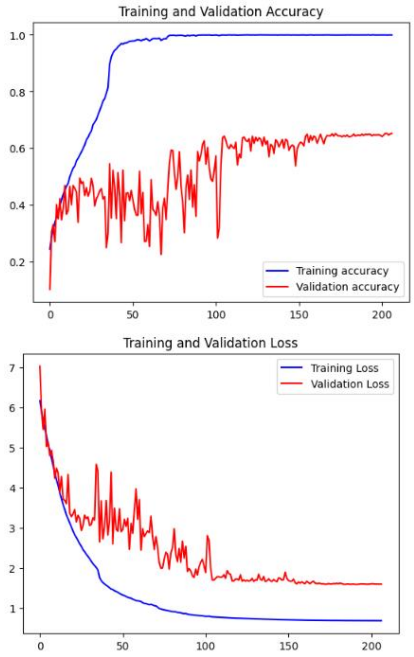

Fig. 9. Accuracy and Loss Graphs of VGG-11

The training and validation accuracy and loss curves for the VGG-11 model confirm a clear case of overfitting, with effective memorization of the training dataset, but plateaus and exhibits significant fluctuations around 60-65% validation

accuracy for most of the training epochs. Despite an initial decrease, the validation loss stabilizes at a value considerably higher than the training loss, proving the model's struggle to generalize to unseen data.

	precision	recall	f1-score	support
Art Nouveau	0.68	0.60	0.64	140
Baroque	0.58	0.51	0.55	138
Color Field Painting	0.76	0.77	0.76	138
Cubism	0.74	0.81	0.77	135
Expressionism	0.48	0.41	0.44	134
Fauvism	0.51	0.56	0.54	136
Impressionism	0.69	0.89	0.78	138
Minimalism	0.71	0.65	0.68	137
Naive Art Primitivism	0.53	0.60	0.56	135
Pop Art	0.58	0.64	0.61	136
Realism	0.55	0.45	0.49	144
Renaissance	0.61	0.70	0.65	141
Rococo	0.72	0.63	0.67	143
Romanticism	0.51	0.52	0.51	137
Symbolism	0.51	0.47	0.49	137
Ukiyo_e	0.91	0.88	0.89	144
accuracy			0.63	2213
macro avg	0.63	0.63	0.63	2213
weighted avg	0.63	0.63	0.63	2213

Fig. 10. Classification Report for VGG-11

Despite overfitting, the VGG-11 model demonstrated strong performance on Ukiyo-e, Impressionism, Color Field Painting, and Cubism. With its best iteration at epoch 192, VGG-11 learned highly discriminative features for the abovementioned styles. However, performance was comparatively weaker for Expressionism, Symbolism, and Realism, suggesting persistent challenges in distinguishing these more ambiguous styles or those with subtle visual characteristics.

C. VGG-13

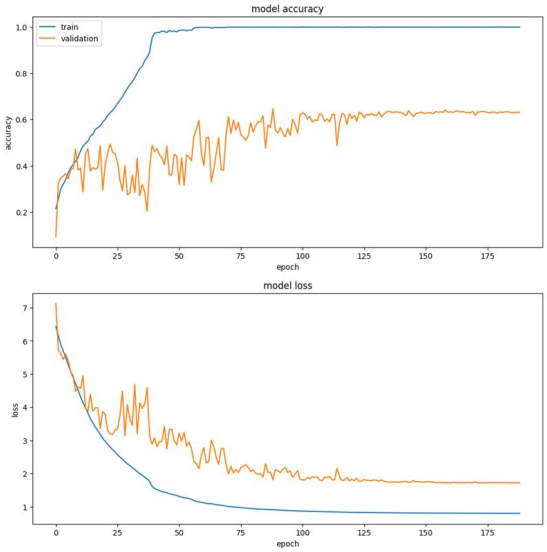

Fig. 11. Accuracy and Loss Graphs of VGG-13

The VGG-13 model shows clear signs of overfitting. While the training accuracy rapidly approaches 1.0 (100%), validation accuracy plateaus and fluctuates significantly around 60-65%. Loss graphs exhibit the exact comparisons between training and validation performance, where training loss steadily decreases as validation loss plateaus.

Classification Report:

	precision	recall	f1-score	support
Art Nouveau	0.58	0.59	0.58	140
Baroque	0.57	0.50	0.53	138
Color Field Painting	0.76	0.78	0.77	138
Cubism	0.76	0.82	0.79	135
Expressionism	0.55	0.45	0.49	134
Fauvism	0.51	0.56	0.53	136
Impressionism	0.83	0.86	0.84	138
Minimalism	0.69	0.66	0.68	137
Naive Art Primitivism	0.60	0.57	0.59	135
Pop Art	0.60	0.59	0.59	136
Realism	0.54	0.44	0.49	144
Renaissance	0.64	0.68	0.66	141
Rococo	0.64	0.60	0.62	143
Romanticism	0.46	0.52	0.49	137
Symbolism	0.39	0.51	0.44	137
Ukiyo_e	0.95	0.85	0.89	144
accuracy			0.62	2213
macro avg	0.63	0.62	0.62	2213
weighted avg	0.63	0.62	0.62	2213

Fig. 12. Classification Report for VGG-13

At epoch 174, VGG-13 achieved its best accuracy at 62%. Analysis of the classification report highlights that Ukiyo-e, Impressionism, and Cubism return the best results with exceptionally high accuracy and balanced performance across all evaluation metrics. Conversely, symbolism, expressionism, and realism remain classes more challenging to classify, exhibiting lower F1-scores.

D. ResNet-18

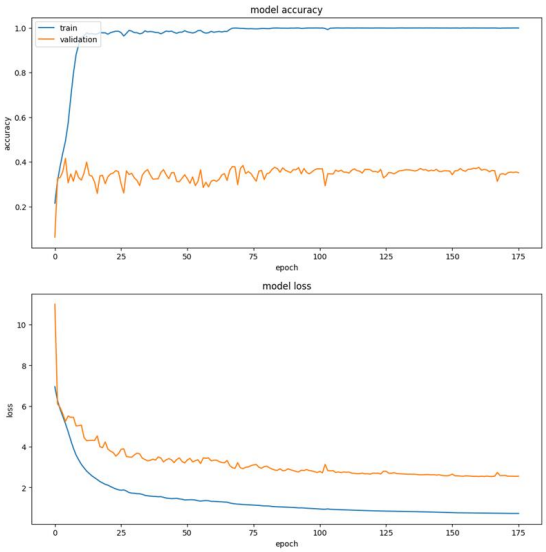

Fig. 13. Accuracy and Loss Graphs of ResNet-18

Visual observation of both loss and accuracy graphs shows a distinct divergence when comparing the training performance to the validation performance of the ResNet-18 model. After an initial increase, the validation accuracy plateaus and fluctuates around 35-40% for most of the training epochs, failing

to keep pace with the increasing training accuracy. Similarly, while initially decreasing, the validation loss stabilizes at a value significantly higher than the training loss, indicative of overfitting.

Classification Report:

	precision	recall	f1-score	support
Art Nouveau	0.34	0.21	0.26	140
Baroque	0.37	0.34	0.36	138
Color Field Painting	0.55	0.63	0.59	138
Cubism	0.30	0.33	0.31	135
Expressionism	0.21	0.13	0.16	134
Fauvism	0.31	0.29	0.30	136
Impressionism	0.39	0.64	0.48	138
Minimalism	0.48	0.47	0.48	137
Naive Art Primitivism	0.26	0.14	0.18	135
Pop Art	0.30	0.30	0.30	136
Realism	0.21	0.15	0.18	144
Renaissance	0.38	0.31	0.34	141
Rococo	0.36	0.34	0.35	143
Romanticism	0.30	0.38	0.34	137
Symbolism	0.14	0.21	0.17	137
Ukiyo_e	0.50	0.62	0.55	144
accuracy			0.34	2213
macro avg	0.34	0.34	0.33	2213
weighted avg	0.34	0.34	0.33	2213

Fig. 14. Classification Report for ResNet-18

Despite training for 161 epochs, ResNet-18 struggled across many art styles except Color Field Painting (F1-score: 0.59) and Ukiyo-e (F1-score: 0.55). The model frequently misclassified instances of most styles and failed to identify many actual instances correctly, as proved by low accuracy f-1 scores across all classes. Despite the architectural advantages of residual connections, ResNet-18's current configuration or training parameters were insufficient for robust discrimination across all 16 art styles, especially for those with subtle differences or smaller representation.

E. Best and Worst Predicted Art Styles

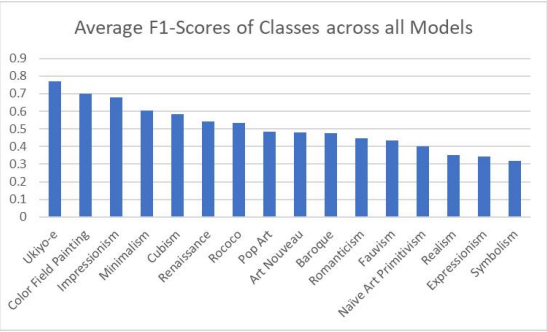

Fig. 15. Average F-1 Scores of Styles across all Models

Across all models, Ukiyo-e and Color Field Painting were generally well-classified. In contrast, Symbolism, Expressionism, Naive Art Primitivism, and Realism consistently proved challenging, indicating inherent visual ambiguities among these styles. While deep learning models show promise in art style classification, the trade-off between model capacity and effective regularization remains critical for improved generalization.

F. Comparison Models

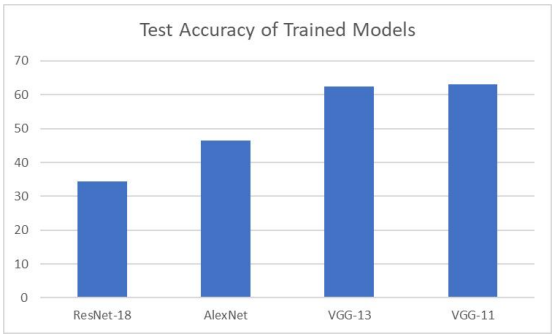

Fig. 16. Model Accuracy across all CNN Architectures

Analysis of AlexNet, VGG-11, VGG-13, and ResNet-18 models revealed a consistent trend of overfitting across all architectures when applied to a multi-class art style classification task. In terms of overall generalization, the VGG family demonstrated superior performance. VGG-11 achieved the highest accuracy at 63%, closely followed by VGG-13 at 62%. VGG architectures, with their deeper, uniform convolutional layers and the inclusion of BatchNormalization (in modified versions), were more effective in extracting and generalizing robust features for art style recognition.

Conversely, AlexNet yielded a moderate overall accuracy of 46%, while ResNet-18 exhibited the lowest accuracy at 34%. These results indicate that despite their architectural strengths, AlexNet and ResNet-18 struggled more significantly with the fine-grained discrimination required for this task in their current configurations compared to the VGG variants.

G. Predictions of the Selected Best Model

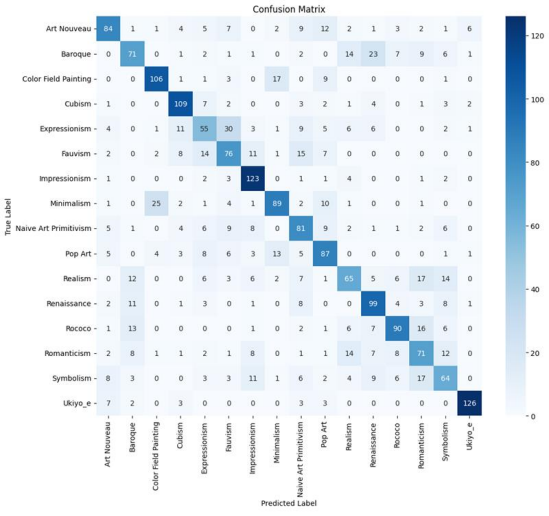

Fig. 17. Confusion Matrix of Best Model

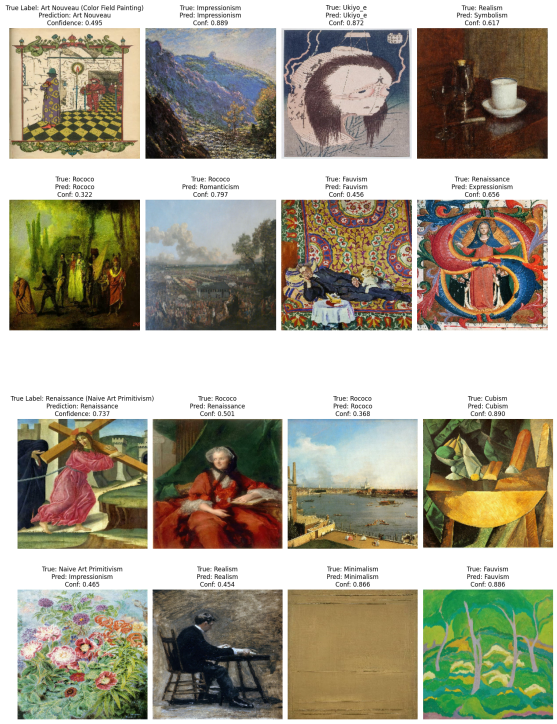

Fig. 18. Sample Predictions and Confidence Score in Test Set

The confusion matrix and sample predictions for VGG-11 offer a detailed visual representation of the model's classification performance on the test set, complementing the insights from its classification report. Ukiyo_e boasts 126 correct predictions out of 144 instances, which aligns with its high F1 Score of 0.89 and precision of 0.91. Impressionism and Color Field Painting also show strong classification, supplementing their high F1 Scores of 0.78 and 0.76, respectively. VGG-11 effectively learned highly discriminative features for these visually distinct art styles.

In contrast, Expressionism was frequently confused with Romanticism and Pop Art. Realism artworks are misclassified as Romanticism and Rococo, proving their low recall of 0.45 and F1- score of 0.49. Moreover, Symbolism was notably confused with Romanticism and Expressionism, leading to its lower F1-score of 0.49. Such patterns of confusion often suggest visual similarities between the misclassified art styles, or nuances that the model, despite its depth, found challenging to differentiate fully.

V. Conclusion

This paper explores classifying art styles using a deep learning methodology. The primary objective was to develop a convolutional neural network (CNN) to distinguish between 16 distinct art styles compiled from WikiArt. The implemented approach involved a custom-built (ResNet, VGG, and AlexNet) CNN architecture, chosen for its proven effectiveness in image classification tasks. Preprocessed data through steps such as image resizing to 224x224 pixels and normalization, and then efficiently managed using TensorFlow's tf.data pipelines for optimizing training, validation, and testing.

Despite these rigorous efforts, the experimental results highlighted significant challenges during model development and training. The model exhibited a notably low test accuracy, ranging from 34% to 63%. Given the complexity of classifying among 16 distinct art styles, this performance closely approximates random chance, indicating the profound difficulty in enabling the model to discern the unique and often subtle characteristics of each art style effectively.

This outcome suggests that while the methodology incorporated standard practices for deep learning in image classification, the complexity or nuance inherent in distinguishing between these specific art styles presented a considerable challenge for the current model architecture and training regimen. Future work should explore more advanced CNN architectures, potentially leveraging transfer learning from models pre-trained on larger, diverse image datasets. Models could navigate smoothly towards global minima for improved convergence and performance by considering various optimization, preprocessing, and data annotation techniques. Additionally, a deeper analysis of the distinctiveness of the art style categories within the dataset and further hyperparameter tuning for the current model could offer valuable insights and pathways for improving classification accuracy. The successful conversion of the model to TFLite format highlights the potential for deployment in resource-constrained environments, but the model's low accuracy currently hinders this practical application.

References

[1] C. Sandoval, E. Pirogova, and M. Lech, "Adversarial learning approach to unsupervised labeling of fine art paintings," IEEE Access, vol. 9, pp. 81969–81985, Jan. 2021, doi: 10.1109/access.2021.3086476.

[2] Z. Falomir, L. Museros, I. Sanz, and L. Gonzalez-Abril, "Categorizing paintings in art styles based on qualitative color descriptors, quantitative global features and machine learning (QArt-Learn)," Expert Systems With Applications, vol. 97, pp. 83–94, Dec. 2017, doi: 10.1016/j.eswa.2017.11.056.

[3] K. Georgoulaki, "Classification of impressionist and pointillist paintings based on their brushstrokes characteristics," Journal on Computing and Cultural Heritage, vol. 17, no. 3, pp. 1–9, May 2024, doi: 10.1145/3665501.

[4] E. Cetinic, T. Lipic, and S. Grgic, "Fine-tuning Convolutional Neural Networks for fine art classification," Expert Systems With Applications, vol. 114, pp. 107–118, Jul. 2018, doi: 10.1016/j.eswa.2018.07.026.

[5] J. DuBois, "Using convolutional neural networks to classify art genre," Carroll Collected, 2020. [Online]. Available: https://collected.jcu.edu/honorspapers/148/

[6] X. Wang, Q. Ye, L. Liu, H. Niu, and B. Du, "ResNet-50-NTS digital painting image style classification based on Three-Branch convolutional attention," Egyptian Informatics Journal, vol. 29, p. 100614, Feb. 2025, doi: 10.1016/j.eij.2025.100614.

[7] C. Sandoval, E. Pirogova, and M. Lech, "Two-Stage deep learning approach to the classification of Fine-Art paintings," IEEE Access, vol. 7, pp. 41770–41781, Jan. 2019, doi: 10.1109/access.2019.2907986.

[8] Q. Yu and C. Shi, "An image classification approach for painting using improved convolutional neural algorithm," Soft Computing, vol. 28, no. 1, pp. 847–873, Nov. 2023, doi: 10.1007/s00500-023-09420-1.

[9] J. Xiang, Y. Yang, and J. Bai, "Adaptive classification of artistic images using multi-scale convolutional neural networks," PeerJ Computer Science, vol. 10, p. e2336, Oct. 2024, doi: 10.7717/peerj-cs.2336.

[10] J. M. Fortuna-Cervantes, C. Soubervielle-Montalvo, C. A. Puente-Montejano, O. E. Pérez-Cham, and R. Peña-Gallardo, "Evaluation of CNN Models with Transfer Learning in Art Media Classification in Terms of Accuracy and Class Relationship," Computación Y Sistemas, vol. 28, no. 1, Mar. 2024, doi: 10.13053/cys-28-1-4895.

[11] Z. Falomir, L. Museros, I. Sanz, and L. Gonzalez-Abril, "Categorizing paintings in art styles based on qualitative color descriptors, quantitative global features and machine learning (QArt-Learn)," Expert Systems With Applications, vol. 97, pp. 83–94, Dec. 2017, doi: 10.1016/j.eswa.2017.11.056.

[12] X. Xu and S. Xu, "Optimization of the convolutional neural network classification model under the background of innovative art teaching models," Scientific Reports, vol. 14, no. 1, Sep. 2024, doi: 10.1038/s41598-024-71536-7.

[13] W. Zhao, W. Jiang, and X. Qiu, "Big transfer learning for fine art classification," Computational Intelligence and Neuroscience, vol. 2022, pp. 1–19, May 2022, doi: 10.1155/2022/1764606.

[14] L. Iliadis, S. Nikolaidis, P. Sarigiannidis, S. Wan, and S. Goudos, "Artwork style recognition using vision transformers and MLP mixer," Technologies, vol. 10, no. 1, p. 2, Dec. 2021, doi: 10.3390/technologies10010002.

[15] Q. Zhao and R. Zhang, "Classification of painting styles based on the difference component," Expert Systems With Applications, vol. 259, p. 125287, Sep. 2024, doi: 10.1016/j.eswa.2024.125287.

[16] W. Li, "Enhanced automated art curation using supervised modified CNN for art style classification," Scientific Reports, vol. 15, no. 1, Mar. 2025, doi: 10.1038/s41598-025-91671-z.