Painting Era Classification Using Transfer Learning with EfficientNet

Team Members: Putri Khairani Azzahra

Course Code: COMP6826001

Semester: Odd Semester 2025/2026

Instructor: Wawan Cenggoro, S.Kom. M.TI

Institution: Binus University

Abstract

Art era classification is an emerging application of deep learning in the cultural heritage and art history domain. This project develops a painting classifier capable of identifying three major historical art periods (Medieval art, Renaissance paintings, and Baroque paintings) using transfer learning with EfficientNetB0. A curated subset of the Painting Eras Detection Classification Dataset is preprocessed using resizing, normalization, and data augmentation. The model is trained through a two-phase strategy: (1) training a frozen EfficientNet classification head, and (2) fine-tuning selected upper layers. Evaluation metrics include accuracy and loss on validation and test sets. Results demonstrate strong classification performance, with consistent separation between stylistically distinct eras. The model is exported in TensorFlow SavedModel format and deployed via a Streamlit interface for user interaction. This work highlights the effectiveness of transfer learning for fine-art analysis and provides a foundation for expanding toward broader multi-era painting classification systems.

Introduction

Deep learning has enabled significant advances in visual recognition tasks, including object detection, scene classification, and image-based cultural analytics. In the domain of art history, automated classification of paintings can assist museums, digital archives, and researchers in cataloging artworks more efficiently. However, art classification presents unique challenges: historical styles can be visually subtle, datasets are often limited, and styles from adjacent time periods may overlap.

This project focuses on the classification of paintings into three major Western art eras: Medieval, Renaissance, and Baroque. These eras represent crucial periods in the evolution of European art, and their stylistic differences, ranging from Medieval symbolism to Renaissance humanism and Baroque dramatic realism, provide rich visual features for analysis.

A. Problem Statement

Given an input painting image, predict its historical era with high accuracy.

B. Objectives

1. Build a cleaned and well-structured dataset of three painting eras.
2. Preprocess and augment the dataset to improve generalization.
3. Train a transfer learning model using EfficientNetB0.
4. Evaluate classification performance on validation and test sets.
5. Deploy the classifier using a simple user interface.

C. Significance

This work demonstrates how modern transfer learning architectures can be adapted for cultural heritage analysis, enabling new tools for art scholars and educational platforms.

Related Work

Prior research on artwork classification commonly employs convolutional neural networks (CNNs) due to their strong feature extraction capabilities. Studies have used architectures such as VGG, ResNet, and Inception for tasks including artist identification, style prediction, and art movement classification. Transfer learning has proven effective when datasets are limited, as fine-art collections often are.

EfficientNet, introduced by Tan and Le (2019), scales network depth, width, and resolution efficiently, achieving state-of-the-art performance with fewer parameters. Recent applications of EfficientNet in art classification demonstrate improved accuracy and faster convergence compared to earlier CNN architectures.

Existing datasets such as WikiArt and OmniArt showcase the feasibility of deep learning for artistic style recognition. However, few works focus specifically on historical eras, particularly Medieval-to-Baroque transitions. This project contributes a focused classifier distinguishing these historically adjacent styles.

Methodology

3.1 Dataset

The raw dataset originates from:
Painting Eras Detection Classification Dataset by ArtAncestry
Source: https://share.google/rqdLnLG0PWmTDK0zf

The original dataset contains **44 art-era classes**, but this project focuses on only three:

1. **Medieval Art**
2. **Renaissance Paintings**
3. **Baroque Paintings**

The trained dataset reorganized for this project is publicly available:
https://drive.google.com/drive/folders/1heZ_RkMHTFwX5y_xuMbqNWr0fuvIr1w8?usp=sharing

The final folder structure follows TensorFlow conventions:

```
dataset/
  train/
    Baroque paintings/
    Medieval art/
    Renaissance paintings/
  test/
    Baroque paintings/
    Medieval art/
    Renaissance paintings/
```

A validation set (20%) was automatically generated using tf.keras.utils.image_dataset_from_directory through the validation_split parameter.

3.2 Preprocessing

The following preprocessing steps were applied:

- Resize all images to 224×224
- EfficientNet preprocessing (tf.keras.applications.efficientnet.preprocess_input)
- Normalize pixel values according to ImageNet statistics
- Convert labels to categorical form
- Additionally, the training and validation pipelines were optimized using TensorFlow's cache() and prefetch(buffer_size = tf.data.AUTOTUNE) functions, enabling faster data loading and improved training efficiency.

3.3 Data Augmentation

Applied only to training images:

- Random horizontal flip
- Random rotation (0.05)

- Random zoom (0.08)

These augmentations improve generalization and reduce overfitting.

3.4 Model Architecture

This project uses EfficientNetB0 as the base model. It was chosen due to its computational efficiency and strong performance for image classification tasks.
The architecture consists of:

1. Input Layer (224×224×3)
2. Data Augmentation Block
3. EfficientNetB0 (frozen initially)
4. Global Average Pooling Layer
5. Dropout (0.3)
6. Dense Softmax Layer (3 units)

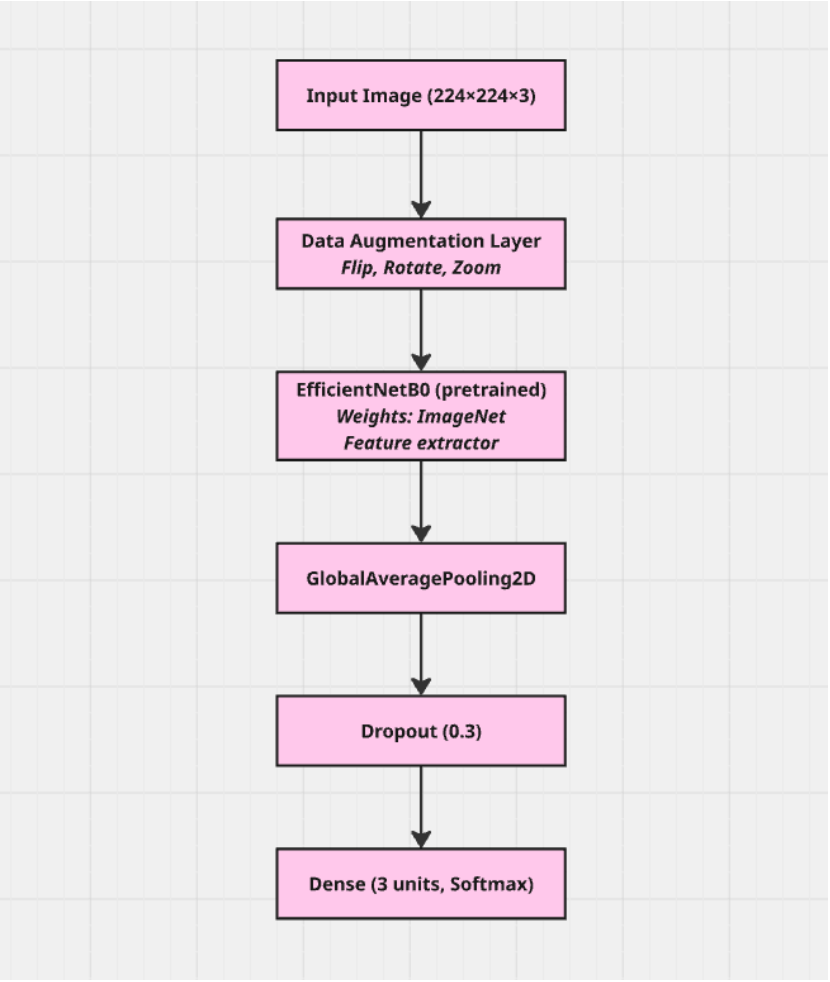

Image 1.1 Architecture Diagram

3.5 Training Setup

Phase 1: Train the Classification Head

- Freeze EfficientNet layers
- Train only the top classification layers
- Optimizer: Adam
- Loss: SparseCategoricalCrossentropy

Training employed an EarlyStopping callback with patience = 6, monitoring validation loss and automatically restoring the best-performing weights to prevent overfitting.

Phase 2: Fine-Tuning

- Unfreeze upper EfficientNet blocks
- Lower learning rate
- Improve model accuracy and generalization

This two-phase training approach prevents catastrophic forgetting and stabilizes training.

3.6 Evaluation Metrics

- Accuracy
- Loss
- Confusion Matrix
- Classification Report

Implementation & Result

4.1 System Implementation

Implemented in Python using:

- TensorFlow / Keras
- NumPy
- Matplotlib
- Streamlit

Data loading used: tf.keras.utils.image_dataset_from_directory(...)
Preprocessing and augmentation were applied using tf.data transformations for efficiency.

4.2 Experiments

The model was trained for multiple epochs under the two-phase strategy. Augmentation improved generalization, and EfficientNetB0 allowed training with minimal computational cost.

4.3 Results

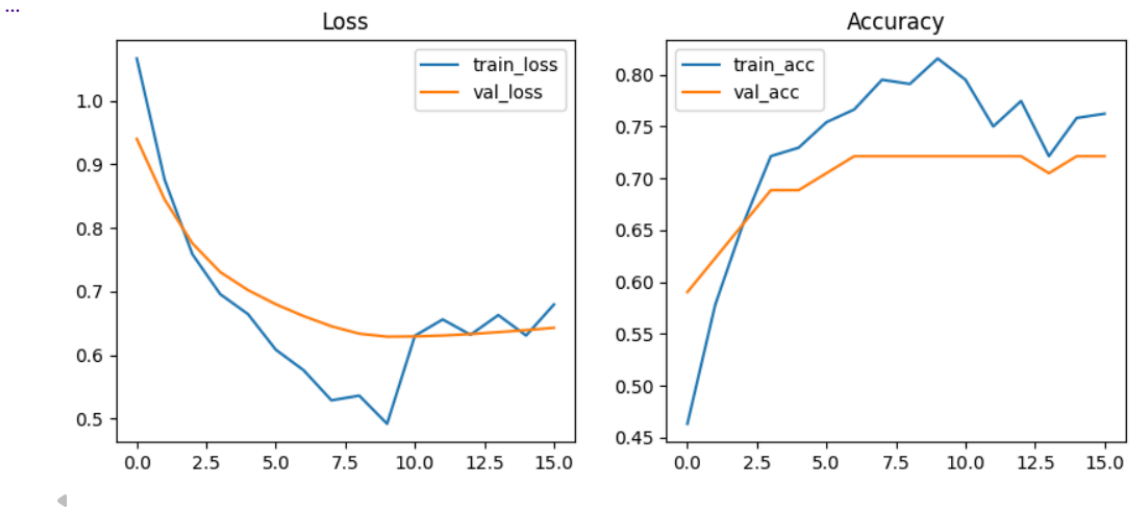

Image 1.2 Train Accuracy & Loss

```
[17]  test_loss, test_acc = model.evaluate(test_ds)
      print(f"Test loss: {test_loss:.4f}   Test accuracy: {test_acc:.4f}")
```

```
5/5 [==============================] - 6s 726ms/step - loss: 0.5912 - accuracy: 0.7415
Test loss: 0.5912   Test accuracy: 0.7415
```

Image 1.3 Test Accuracy & Loss

```
Classification Report:
                          precision    recall  f1-score   support

      Baroque paintings       0.83      0.74      0.78        58
         Medieval art         0.89      0.79      0.84        52
Renaissance paintings         0.51      0.68      0.58        37

               accuracy                           0.74       147
              macro avg       0.74      0.74      0.73       147
           weighted avg       0.77      0.74      0.75       147
```

Image 1.4 Classification Report

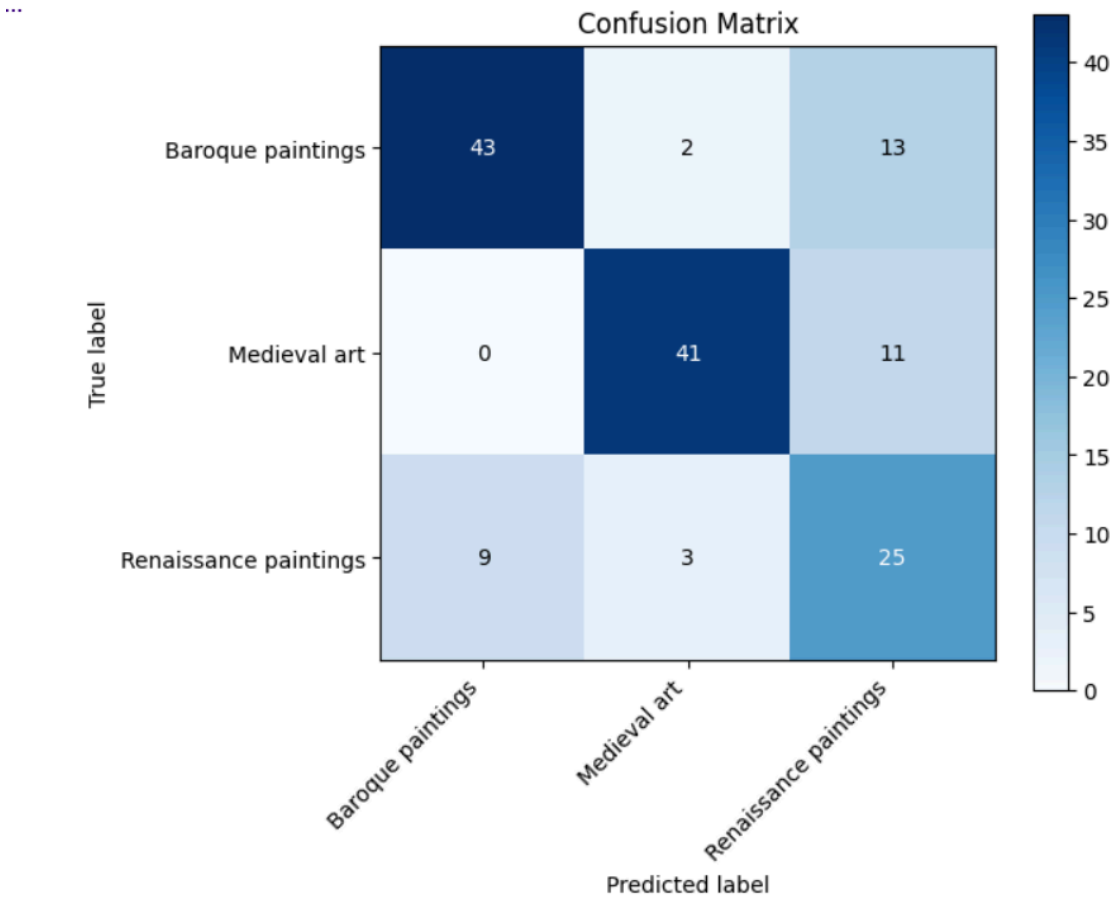

Image 1.5 Confusion Matrix

The model shows strong classification performance, especially between stylistically distinct classes, though Renaissance and Baroque exhibit some overlap due to shared techniques such as realism and chiaroscuro.

4.4 System Deployment

A simple deployment pipeline was created to allow users to upload an image and receive a predicted painting era.

The final model was exported in TensorFlow SavedModel format:
tf.saved_model.save(model, "painting_era_model")

Saved elements include:

- assets/
- variables/
- saved_model.pb

Class names were stored in a separate JSON file:
["Baroque paintings", "Medieval art", "Renaissance paintings"]

4.5 User Interface

Implemented using Streamlit, the deployment UI allows users to:

1. Upload an image
2. Run preprocessing
3. Predict art era
4. Display prediction probabilities

This makes the model accessible for museums, educators, and art enthusiasts.

GitHub Repository: https://github.com/babytokki/painting-era-classifier

Discussion & Limitations

A. Strengths

- Transfer learning allowed high accuracy with limited training data.
- EfficientNetB0 achieved strong feature extraction at low computational cost.
- Augmentation and preprocessing improved robustness.

B. Limitations

- Only three classes were used; performance on the full 44-class dataset remains untested.
- Some misclassifications occurred between Renaissance and Baroque paintings, likely due to stylistic similarity.
- Dataset size and diversity could be expanded for better generalization.

C. Trade-Offs

- Freezing the EfficientNet base reduces training time but limits adaptability.
- Fine-tuning improves accuracy but increases risk of overfitting.
- SavedModel format simplifies deployment but increases file size.

Conclusion & Future Work

This project successfully developed a painting classifier using EfficientNet transfer learning, achieving strong performance across three historically significant art periods. The methodology demonstrated the viability of deep learning for cultural heritage applications and provided a deployable interface through Streamlit.

Future Enhancements

- Expand classification to all 44 art-era classes
- Integrate artist/style detection
- Apply Grad-CAM for interpretability
- Deploy as a web API or mobile app
- Train larger architectures (EfficientNetB3-B7)

References

ArtAncestry. (n.d.). *Painting Eras Detection Classification Dataset*. https://share.google/rqdLnLG0PWmTDK0zf

Azzahra, Putri. (2025). *Painting-era-classifier* [Source code]. GitHub. https://github.com/babytokki/painting-era-classifier

sagecadence. (2025). *My attempt at combining probability theory with art* [Instagram Reel]. Instagram. https://www.instagram.com/reel/DRR-V0Eip_V/

Streamlit. (n.d). *Streamlit documentation*. https://docs.streamlit.io/

Tan, M., & Le, Q. V. (2019). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.* arXiv:1905.11946. https://arxiv.org/abs/1905.11946

TensorFlow. (2023). *Image classification.* Tensorflow documentation. https://www.tensorflow.org/tutorials/images/classification

Appendix

A. Team Contribution

Name	NIM	Contribution
Putri Khairani Azzahra	2702346931	Data, Preprocessing, Modeling, UI/app, Report

B. Documentation

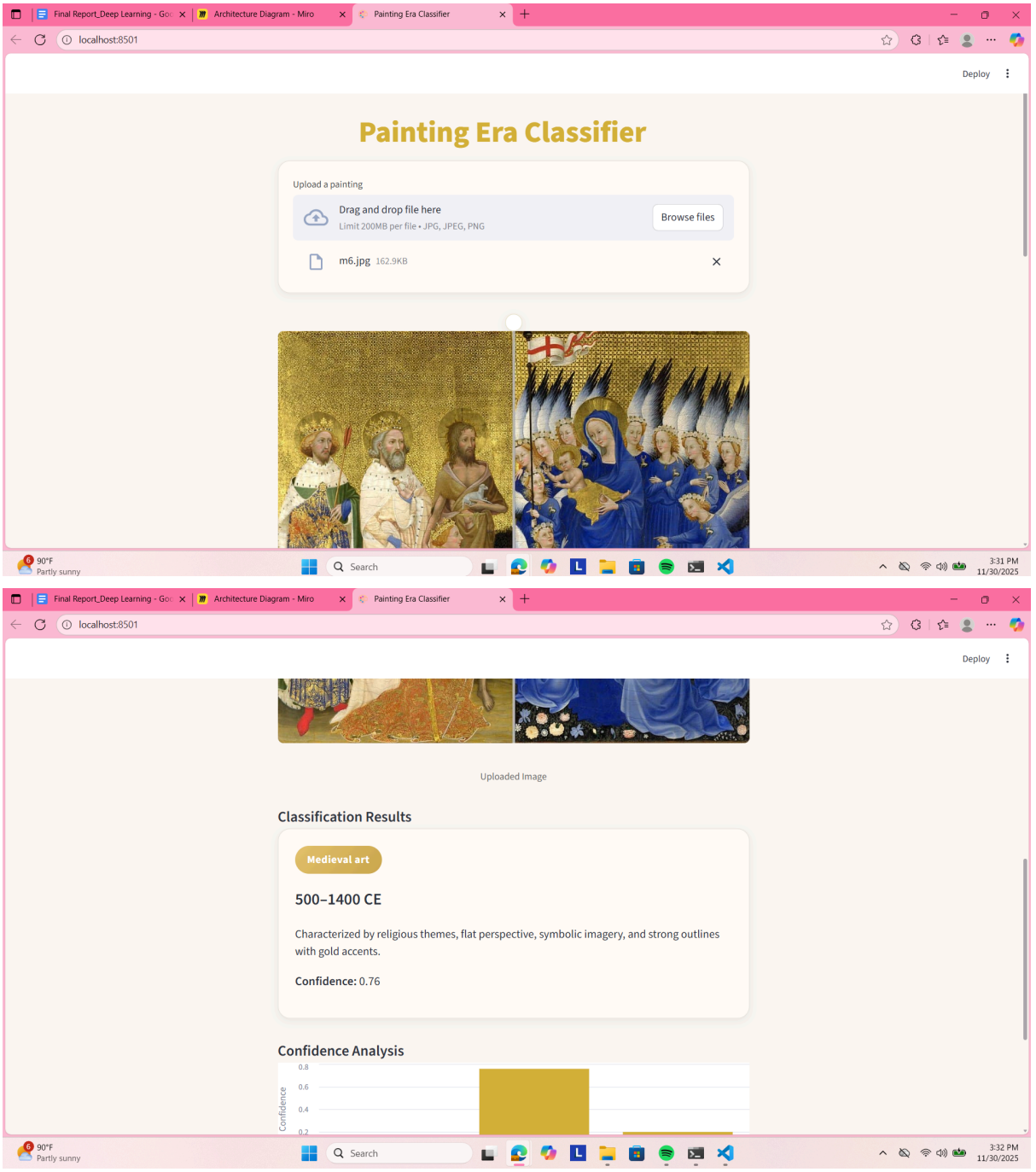

Image 2.1 - 2.2 User Interface