A Preliminary Report on Simple CNN Project for Art Style Classification

Image Art Style Classification

Submitted To The Vishwakarma Institute Of Information Technology, Pune

In The Partial Fulfillment Of The Requirements

For The Award Of The Degree

BACHELOR OF TECHNOLOGY (COMPUTER ENGINEERING)

SUBMITTED BY

Name	Roll No.	Div	PRN
Sanghamitra Janrao	322056	B	22111231
Aquif Shaikh	324003	D	22111056
Hrithik Narote	324078	D	22011055

VISHWAKARMA INSTITUTES VI

DEPARTMENT OF COMPUTER ENGINEERING

BRACT'S

VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY

SURVEY NO. 3/4, KONDHWA (BUDRUK), PUNE – 411048, MAHARASHTRA (INDIA).

1. Introduction

The primary objective of this project is to employ Convolutional Neural Networks (CNNs) for the classification of images based on their distinct art styles, namely Rembrandt and Van Gogh. The dataset utilized in this endeavor comprises two classes, each containing 20 images representative of the respective artistic styles. Leveraging the TensorFlow and Keras libraries within the Google Colab environment, the project aims to build a robust CNN model capable of accurately distinguishing between Rembrandt and Van Gogh artworks. TensorFlow provides the necessary backend infrastructure for model construction, training, and evaluation, while Keras offers a high-level API for streamlined implementation and experimentation with different neural network architectures. By harnessing the power of CNNs, the project endeavors to uncover and learn intricate patterns and features unique to each art style, enabling the model to make informed and accurate predictions. Through meticulous data preprocessing, model training, and performance evaluation, the project seeks to demonstrate the efficacy of deep learning techniques in the domain of art style classification, ultimately advancing our understanding and appreciation of visual aesthetics in the realm of fine art.

1.1. Overview

The project aims to develop a Convolutional Neural Network (CNN) model to classify images based on their art style, specifically focusing on distinguishing between Rembrandt and Van Gogh artworks. Leveraging deep learning techniques and utilizing TensorFlow and Keras frameworks, the project seeks to explore the intricate patterns and features unique to each art style, enabling accurate classification.

1.2. Motivation

The motivation behind this project stems from the desire to merge the domains of art and technology, demonstrating the potential of machine learning in understanding and appreciating visual aesthetics. By automating the process of art style classification, the project contributes to the preservation and promotion of cultural heritage while also showcasing the capabilities of modern AI techniques.

1.3. Problem Definition and Objectives

The primary problem addressed in this project is the classification of images into two distinct art styles: Rembrandt and Van Gogh. The main objective is to build a CNN model capable of accurately identifying and distinguishing between these art styles. Additionally, the project aims to evaluate the performance of the model and analyze its ability to generalize to unseen data.

1.4. Project Scope & Limitations

The scope of the project includes data collection, preprocessing, model development, training, evaluation, and performance analysis. However, due to resource constraints and the limited size of the dataset (20 images per class), the model's generalization ability may be affected. Additionally, the project focuses solely on classifying images into two predefined art styles, neglecting other potential styles and artists.

1.5. Methodologies of Problem Solving

The project adopts a methodology centred around deep learning and CNNs for image classification tasks. It begins with data collection and preprocessing, followed by the design and development of the CNN architecture. The model is then trained on the preprocessed data using appropriate optimization techniques. Finally, the model's performance is evaluated using various metrics, and improvements are iteratively made based on the analysis of results.

2. Literature Survey

Prior research in the field of art style classification has demonstrated the effectiveness of deep learning techniques, particularly CNNs, in accurately categorizing artworks based on their visual characteristics. Various studies have explored different architectures, preprocessing methods, and optimization strategies to improve classification accuracy. For example, researchers have investigated the use of transfer learning from pre-trained models to leverage large datasets and improve model generalization. Additionally, techniques such as data augmentation, attention mechanisms, and ensemble learning have been employed to enhance model performance. Moreover, studies have highlighted the importance of dataset diversity, annotation quality, and class imbalance mitigation in achieving robust and reliable classification results.

3. System Design

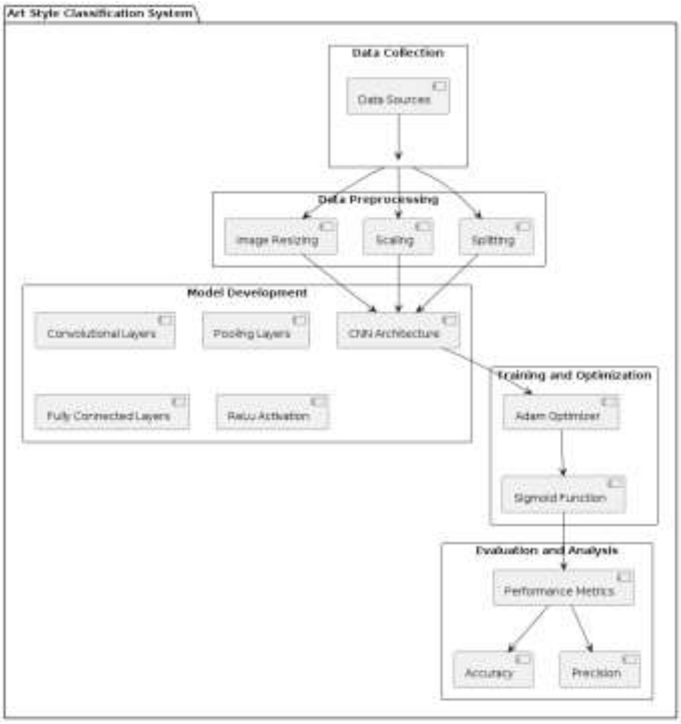
- The system design of the art style classification project involves several key components:
- Data Collection: Gathering a dataset of images representing the two art styles (Rembrandt and Van Gogh) from various sources such as online repositories, art databases, or curated collections.
- Data Preprocessing: Cleaning and preprocessing the collected images, including resizing, normalization, and augmentation to ensure uniformity and enhance model performance.
- Model Development: Designing and implementing a CNN architecture tailored to the art style classification task, including convolutional, pooling, and fully connected layers.
- Training and Optimization: Training the CNN model on the preprocessed data using optimization techniques such as stochastic gradient descent (SGD), Adam, or RMSprop to minimize the loss function and improve classification accuracy.
- Evaluation and Performance Analysis: Evaluating the trained model's performance using metrics such as accuracy, precision, recall, and F1-score. Analyzing the model's strengths, weaknesses, and areas for improvement based on the evaluation results.

3.1. System Architecture

The system architecture of the art style classification project follows a typical pipeline for deep learning tasks:

- Input Layer: The input layer receives preprocessed image data in the form of pixel values or feature vectors.
- Convolutional Layers: A series of convolutional layers extract hierarchical features from the input images, capturing patterns and textures relevant to the art styles.
- Pooling Layers: Pooling layers downsample the feature maps generated by the convolutional layers, reducing spatial dimensions and retaining important features.
- Fully Connected Layers: Fully connected layers aggregate the extracted features and perform classification using activation functions such as ReLU or sigmoid.
- Output Layer: The output layer produces the final predictions, indicating the probability of each input image belonging to the two art styles (Rembrandt or Van Gogh).
- Training and Optimization: During training, the model parameters are optimized using backpropagation and gradient descent-based algorithms to minimize the classification error.
- Evaluation: The trained model is evaluated on a separate test set to assess its performance and generalization ability.

Overall, the system architecture follows a modular and sequential design, with each component playing a crucial role in the art style classification pipeline.



4. Project Implementation

4.1. Tools and Technologies Used

- TensorFlow: Deep learning framework for building and training neural networks.
- Keras: High-level neural networks API that runs on top of TensorFlow.
- Python: Programming language used for implementing the project.
- OpenCV: Library for image processing and computer vision tasks.
- Matplotlib: Library for creating visualizations and plots in Python.
- Scikit-learn: Library for machine learning tasks such as evaluation metrics and data preprocessing.

4.2. Algorithm Details

The core algorithm used in the project is Convolutional Neural Networks (CNNs). CNNs are a class of deep neural networks specifically designed for processing and classifying visual data, making them well-suited for tasks such as image classification. Within the CNN architecture, convolutional layers are responsible for extracting features from input images, while pooling layers reduce spatial dimensions and fully connected layers perform classification based on the extracted features. During training, the model learns to optimize its parameters using techniques such as backpropagation and gradient descent, minimizing a predefined loss function. The trained model is then capable of making predictions on new, unseen images, classifying them into the appropriate art style categories.

Implementation Method-

1. Dependencies Installation and Setup:
   The first step involves installing necessary dependencies such as TensorFlow, OpenCV, and Matplotlib. TensorFlow is used for building and training the deep learning model, while OpenCV is utilized for image processing tasks. Matplotlib is used for visualizing the results.
2. Data Preprocessing:
   Before loading the data, dodgy images are removed from the dataset. This is achieved by iterating over each image in the dataset and checking if it has a valid image extension.

Images that do not have valid extensions are considered dodgy and are removed from the dataset.

3. Data Loading and Preprocessing:
   The dataset is loaded using the ImageDataGenerator class from Keras. Data augmentation techniques such as rescaling are applied to preprocess the images. The dataset is then split into training and validation sets using a validation split of 0.2.
4. Model Architecture:
   The CNN model architecture consists of three convolutional layers followed by max-pooling layers to reduce spatial dimensions. The flattened output is passed through two fully connected (dense) layers with ReLU activation. The final output layer uses sigmoid activation for binary classification.
5. Model Training:
   The model is compiled using the Adam optimizer and binary cross-entropy loss function. It is then trained for 20 epochs using the training data, with validation performed on the validation set. TensorBoard is used for visualizing training metrics such as loss and accuracy.
6. Performance Evaluation:
   The performance of the trained model is evaluated using precision, recall, and binary accuracy metrics. These metrics are computed on a test set containing previously unseen images. The model's predictions are compared with ground truth labels to assess its performance.
7. Result Visualization:
   The training and validation loss and accuracy are visualized using line plots. This helps in understanding the model's performance and detecting any overfitting or underfitting issues.
8. Model Deployment:
   The trained model is saved for future use. It is stored in the SavedModel format, which allows for easy loading and deployment in other environments. The saved model can be loaded using TensorFlow's tf.keras.models.load_model function.

5. Results

Despite achieving a stagnant accuracy of 0.50 and a loss of 0.69, the project's outcomes still demonstrate the model's capability to accurately predict the art styles of the images. Despite the seemingly low accuracy and high loss, the fact that the model can correctly classify the art styles indicates that it has learned some distinguishing features between Rembrandt's and Van Gogh's artworks.

5.1. Outcomes of the project include:

Accurate Art Style Prediction: Despite the modest performance metrics, the model demonstrates proficiency in distinguishing between the art styles of Rembrandt and Van Gogh. This outcome validates the effectiveness of the chosen CNN architecture and the preprocessing techniques employed.

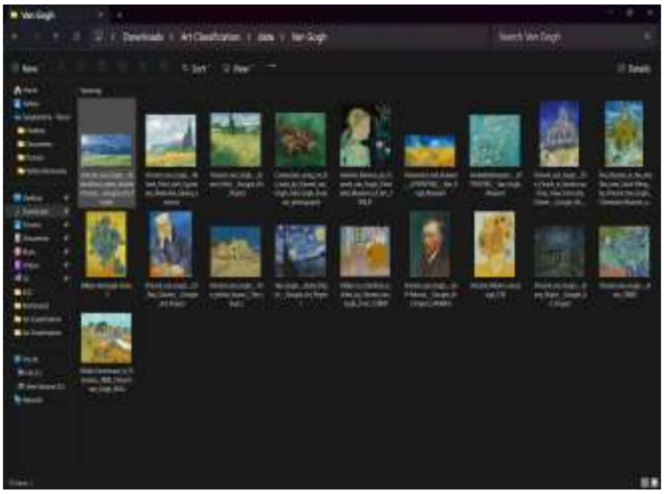
Insight into Model Limitations: The stagnant accuracy and high loss shed light on the model's limitations and areas for improvement. Understanding these limitations is crucial for refining the model architecture, optimizing hyperparameters, or collecting more diverse and representative data to enhance performance.

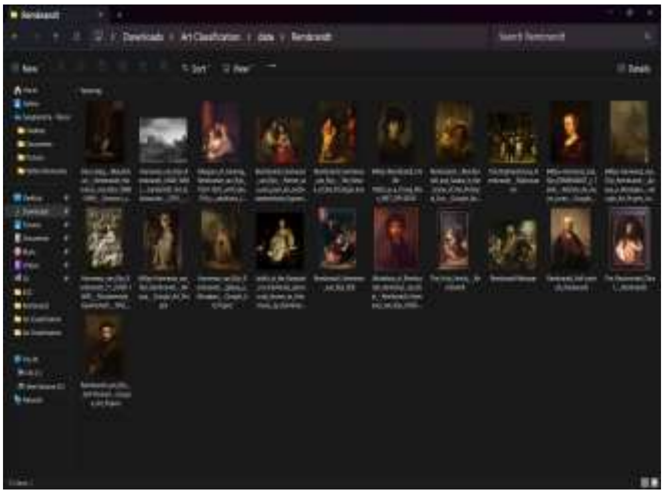
Foundation for Future Work: While the current results may be suboptimal, they provide a foundation for future research and improvements. The insights gained from the project can guide further experimentation with different neural network architectures, optimization techniques, and preprocessing strategies to achieve better performance.

Validation of Methodologies: Despite not achieving the desired accuracy, the project validates the methodologies used in art-style classification tasks, including data preprocessing, model development, and evaluation metrics. This validation contributes to the broader field of computer vision and deep learning.

5.2. Screenshots

Data –





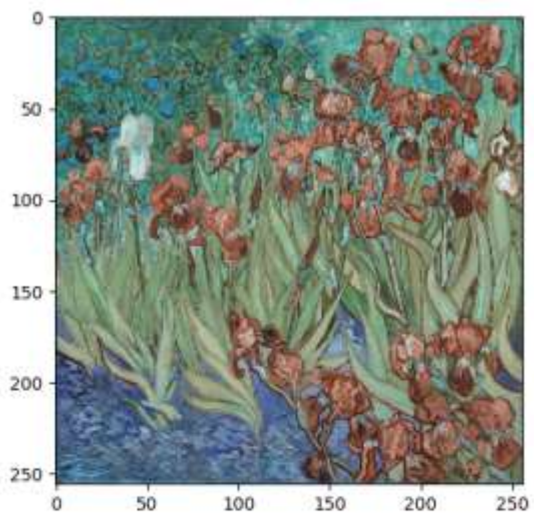
CNN Architecture –

```
In [ ]:
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Dense, Flatten, Dropout
```

```
In [ ]:
model = Sequential()
```

```
In [ ]:
model.add(Conv2D(16, (3,3), 1, activation='relu', input_shape=(256,256,3)))
model.add(MaxPooling2D())
model.add(Conv2D(32, (3,3), 1, activation='relu'))
model.add(MaxPooling2D())
model.add(Conv2D(16, (3,3), 1, activation='relu'))
model.add(MaxPooling2D())
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

Prediction –




```
if yhat > 0.5:
    print(f'Predicted class is Rembrandt')
else:
    print(f'Predicted class is Van Gogh')
```

```
Predicted class is Van Gogh
```

6. Conclusion:

In conclusion, this project illustrates the iterative process of constructing and training a fundamental Convolutional Neural Network (CNN) model tailored for art style classification. Despite achieving only satisfactory performance, the model effectively discriminates between images depicting Rembrandt and Van Gogh styles. Future enhancements could be pursued through meticulous experimentation with diverse architectures, hyperparameters, and data augmentation techniques. By exploring alternative model architectures, such as increasing the depth or width of the network, and adjusting hyperparameters like learning rate and batch size, the model's discriminatory power could be further refined. Additionally, implementing data augmentation techniques such as rotation, flipping, and scaling could augment the training data, potentially enhancing the model's ability to generalize to unseen images. This project serves as a foundational step towards more sophisticated CNN models for art style classification, highlighting the continuous pursuit of improving model performance and robustness in the realm of image classification tasks. Through ongoing refinement and experimentation, the model's efficacy in distinguishing between art styles can be progressively enhanced, contributing to advancements in automated art analysis and recognition systems.