Final Project

ArtStyles Classification Model

Group 24

Kasey Le 301479381

Grace Kim 301321070

Github Repository:

https://github.com/k1gk/IAT481_FinalProject_ArtStylesIdentifier

IAT 481 - Spring 2024

Instructor: Dr. O.Nilay Yalcin

TA: Maryiam Zahoor

Introduction

The main objective of the **ArtStyles Identifier** project is to develop a supervised **classification** model that can accurately determine the art movement to which a painting or artwork belongs to. ArtStyles Identifier covers 6 prominent art movements throughout history including Baroque, Cubism, Expressionism, Renaissance, Realism, and Romanticism. The model aims to serve as a powerful educational tool that helps students and art enthusiasts to better distinguish and recognize the different art styles.This model does not only facilitate an easy classification of artworks for its users but also fosters a sense of appreciation and engagement with art/art history in general. The primary users of the ArtStyles Identifier are educational institutions, art history students, and art enthusiasts. By integrating this model into educational curriculums, teachers can incorporate this model into multimedia presentations or interactive quizzes via desktops and laptops to help students learn and identify art styles effectively.

Dataset Creation

Our final dataset was created by:

- Using a subset of images from the existing dataset WikiaArt Art Movements/Styles on Kaggle: We extracted 5 out of 13 classes from the Kaggle dataset including Baroque, Expressionism, Renaissance, Realism, and Romanticism. Then, we only extracted 1000 files for each of the classes to be included in our final training dataset due to computational training power.
- Scouting online images to create a new added class - Cubism: We each looked for more than 150 new images each totalling an approximate of 300+ images of artworks from the Cubism art movement. These images are mixed up from two online datasets to create a diverse and well-balanced class in addition to the 5 classes we have from the Kaggle dataset. Sources to retrieve the Cubism images: Wiki Art Dataset

Painting Eras Detection

- Data augmentation: Since the number of images we have for our new class is way less than each of the 5 existing classes. We flipped, adjusted brightness, and resized a part of our Cubism images to diversify and increase its size to meet the 1000 files equal to the other classes.
- Final Dataset: 6 classes, 1000 files each, a total of 6000 images.

Please access our final dataset through this link (Password: IAT481spring2024).

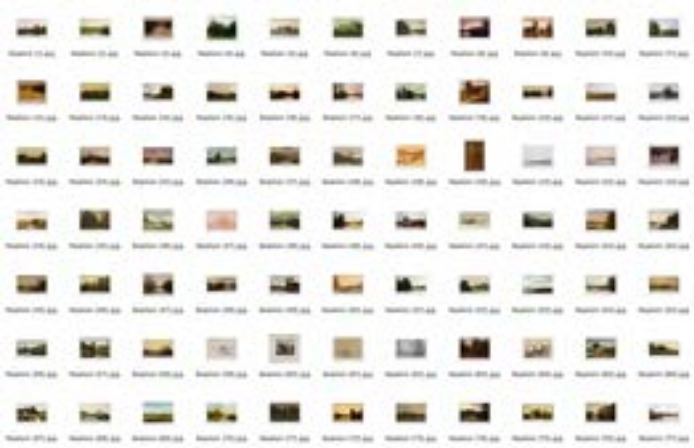
Then, we used code to organize and split our image dataset into a structure that is accepted by the YOLO framework, which is as follows:

Root Folder:

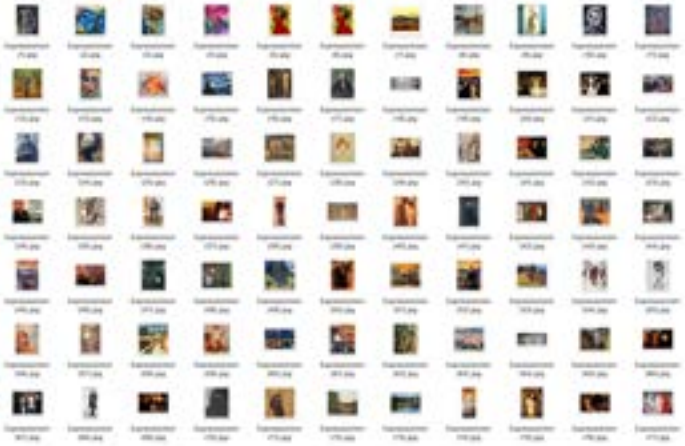
- Train: (80% of dataset - 4800 files)
  + Baroque
  + Cubism
  + Expressionism
  + Realism
  + Romanticism
  + Renaissance
- Val: (20% of dataset - 1200 files)
  + Baroque
  + Cubism
  + Expressionism
  + Realism
  + Romanticism
  + Renaissance

Images from the Kaggle dataset of the 5 classes:

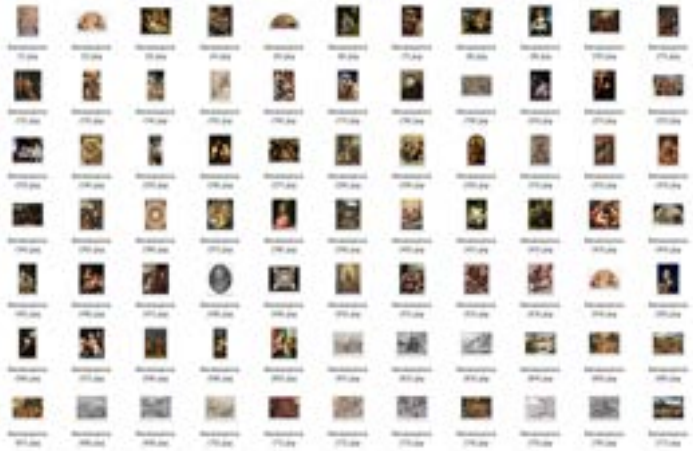
Romanticism Class



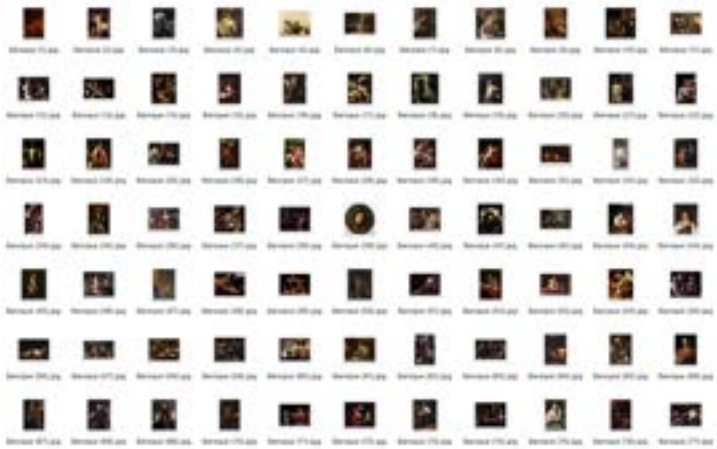
Realism Class



Expressionism Class

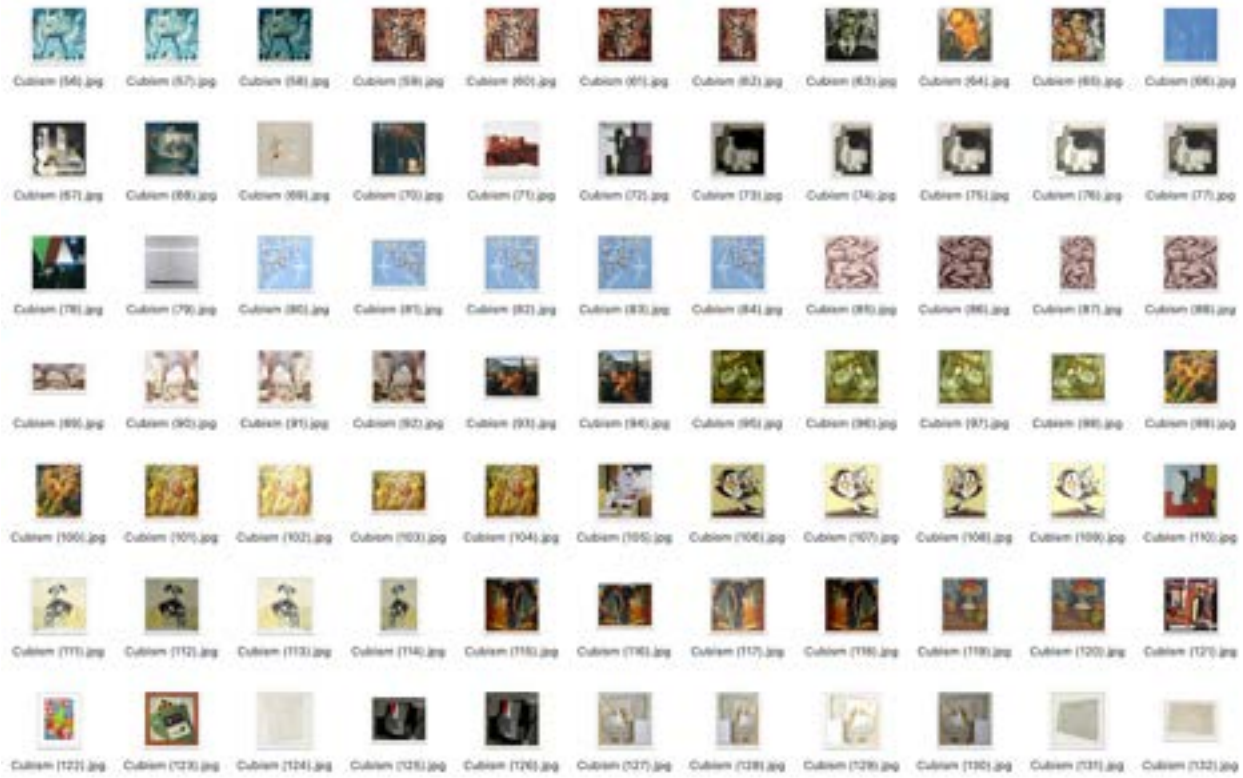


Renaissance Class



Baroque Class

Overview of the Cubism class we added:



Close-up of some of the data augmentation we did (Flipped, resized, adjust brightness)

The first code in the image below was used to create new directories that match the structure above:

```
directories = [
    '../Downloads/RefinedArtStyles/train',
    '../Downloads/RefinedArtStyles/val',

    '../Downloads/RefinedArtStyles/train/Baroque',
    '../Downloads/RefinedArtStyles/train/Cubism',
    '../Downloads/RefinedArtStyles/train/Expressionism',
    '../Downloads/RefinedArtStyles/train/Realism',
    '../Downloads/RefinedArtStyles/train/Renaissance',
    '../Downloads/RefinedArtStyles/train/Romanticism',

    '../Downloads/RefinedArtStyles/val/Baroque',
    '../Downloads/RefinedArtStyles/val/Cubism',
    '../Downloads/RefinedArtStyles/val/Expressionism',
    '../Downloads/RefinedArtStyles/val/Realism',
    '../Downloads/RefinedArtStyles/val/Renaissance',
    '../Downloads/RefinedArtStyles/val/Romanticism'
]

for directory in directories:
    os.makedirs(directory, exist_ok=True)
```

```
baroque_source = '../Downloads/RefinedArtStyles/Baroque'
baroque_train = '../Downloads/RefinedArtStyles/train/Baroque'
baroque_val = '../Downloads/RefinedArtStyles/val/Baroque'

expression_source = '../Downloads/RefinedArtStyles/Expressionism'
expression_train = '../Downloads/RefinedArtStyles/train/Expressionism'
expression_val = '../Downloads/RefinedArtStyles/val/Expressionism'

realism_source = '../Downloads/RefinedArtStyles/Realism'
realism_train = '../Downloads/RefinedArtStyles/train/Realism'
realism_val = '../Downloads/RefinedArtStyles/val/Realism'

renaissance_source = '../Downloads/RefinedArtStyles/Renaissance'
renaissance_train = '../Downloads/RefinedArtStyles/train/Renaissance'
renaissance_val = '../Downloads/RefinedArtStyles/val/Renaissance'

romanticism_source = '../Downloads/RefinedArtStyles/Romanticism'
romanticism_train = '../Downloads/RefinedArtStyles/train/Romanticism'
romanticism_val = '../Downloads/RefinedArtStyles/val/Romanticism'

cubism_source = '../Downloads/RefinedArtStyles/Cubism'
cubism_train = '../Downloads/RefinedArtStyles/train/Cubism'
cubism_val = '../Downloads/RefinedArtStyles/val/Cubism'
```

We then defined a function to take images from a source directory where our dataset is located and move them to the newly created validation and train folders according to their category. We also split this data using train_test_split with a ratio of 80-20. We then defined the source directories for each category as well as directories for

where they should be organized as shown in the screenshot below:

Dataset Organization

```
from sklearn.model_selection import train_test_split
def organize_file(source_dir, train_dir, val_dir):
    files = []
    for f in os.listdir(source_dir):
        files.append(f)

    if len(files) == 0:
        print("No files found in the source directory:", source_dir)
        return

    train_files, val_files = train_test_split(files, test_size=0.2, random_state=42)

    for file in train_files:
        file_path = os.path.join(source_dir, file)
        image = Image.open(file_path).convert("RGB")
        image.save(os.path.join(train_dir, file), "JPEG")

    for file in val_files:
        file_path = os.path.join(source_dir, file)
        image = Image.open(file_path).convert("RGB")
        image.save(os.path.join(val_dir, file), "JPEG")
```

```
organize_file(baroque_source, baroque_train, baroque_val)
```

```
organize_file(expression_source, expression_train, expression_val)
```

```
organize_file(realism_source, realism_train, realism_val)
```

```
organize_file(renaissance_source, renaissance_train, renaissance_val)
```

```
organize_file(romanticism_source, romanticism_train, romanticism_val)
```

```
organize_file(cubism_source, cubism_train, cubism_val)
```

Bias in the Dataset:

- Class imbalance: The Cubism class had significantly fewer images compared to other classes. Although data augmentation techniques such as flipping and brightness adjustment were used to increase the number of Cubism images, these methods might introduce their own biases. Augmented images could cause the model to learn less about the variability and nuance of original Cubism artwork and more about the modified versions, potentially affecting the model's ability to generalize to new, unmodified images of Cubism.
- Historically representation bias: The dataset might not fully represent the diversity within each art movement. Art movements often encompass a wide range of styles

and techniques, influenced by different artists, geographical regions, and cultural contexts. By limiting each class to 1000 images and sourcing from primarily Western-centric databases, there's a risk of underrepresenting the global and historical diversity of these movements.

- Sourcing primarily online: The choice of sources, primarily Kaggle and other online datasets may also introduce a bias towards the types of paintings that are more commonly digitized and available online, which are often those held in major Western museums.

If we had the computational resources and skill sets, we could have mitigated these biases by expanding the dataset to include a wider array of images from diverse sources and employing more sophisticated augmentation techniques could be beneficial. Advanced techniques such as style transfer and GAN-based augmentations might help the model learn more robust features without overfitting to the augmented styles.

Note on privacy concerns: Even though privacy and copyright considerations are minimal with historical artworks, typically in the public domain, the digital reproductions that we are using might not be. Some of them might be copyrighted by the photographer or the entity that digitized the artwork. Therefore, we need to be mindful of these potential copyrights by keeping the modifications private to us and the teaching team and ensure it is only used for academic purposes.

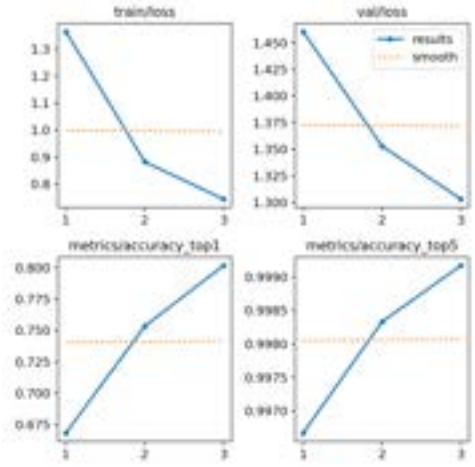
Training Report

For hypertuning, our team modified the number of times the dataset is passed through (epoch), as well as the batch size. Initially, we configured four models with the same YOLO architecture, switching between 3 or 5 epochs and a batch size of 16 and 20. Specifically, the models are:

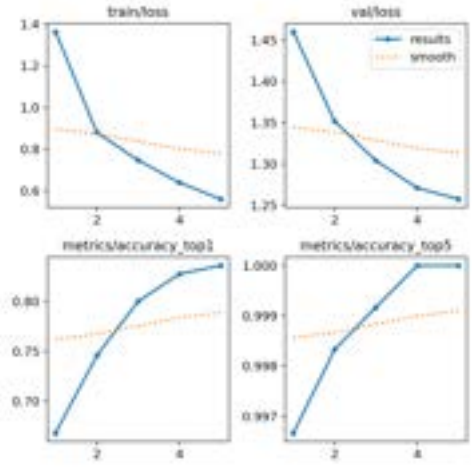
- Model 1: 3 epochs with batch size of 16
- Model 2: 5 epochs with batch size of 16
- Model 3: 3 epochs with batch size of 20

- Model 4: 5 epochs with batch size of 20

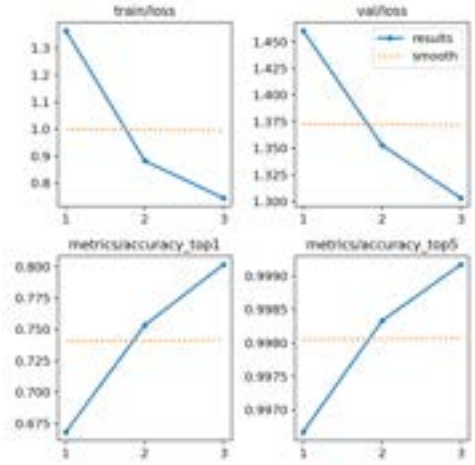
1. <u>Overall Comparison</u>



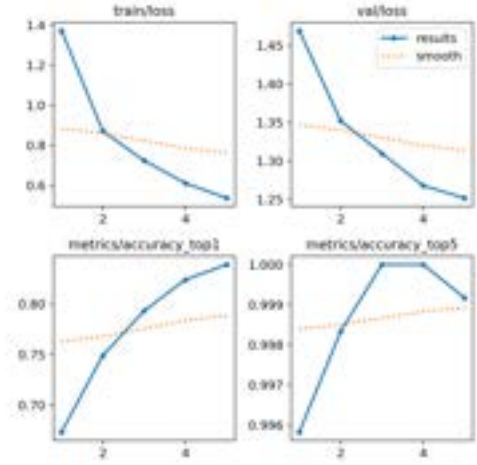
Model 1: 3 epochs, batch-size 16 (left)



Model 2: 5 epochs, batch-size 16 (right)



Model 3: 3 epochs, batch-size 20 (left)



Model 4: 5 epochs batch-size 20 (right)

With the first glance at the overall results, we noticed that the pattern of the models do not change much in terms of the increase in batch size, with both models with 3 epochs but different batch sizes reaching the same 80% of the accuracy rate and both models with 5 epochs but different batch sizes reaching the same 83% of the accuracy rate. However, there is a clear indication that models that were trained with higher epochs

resulted in higher accuracy rate. We then looked deeper into the results to better evaluate the models through the specifics of the evaluation metrics.

2. Accuracy

We use two evaluation metrics for our model, which are Top 1 Accuracy and Top 5 Accuracy. Top 1 measures the percentage of times the class with the highest probability score predicted by the model is exactly the same as the true class label of the input data. Meanwhile, Top 5 Accuracy extends this concept by checking whether the true class label is among the model's top five predictions, regardless of their order. However, in our case, there are only 6 classes so the relevance and impact of Top 5 Accuracy are somewhat diminished. Since there are only 6 classes, achieving high Top 5 Accuracy is significantly less challenging compared to scenarios with many classes.

	train/loss	metrics/accuracy_top1	metrics/accuracy_top5	val/loss	lr/pg0	lr/pg1	lr/pg2
1	1.3641	0.66833	0.99667	1.4604	0.00023721	0.00023721	0.00023721
2	0.88239	0.75333	0.99833	1.3529	0.00031839	0.00031839	0.00031839
3	0.74434	0.80167	0.99917	1.3031	0.00024249	0.00024249	0.00024249

Model 1 with 3 epochs batch-size 16

	train/loss	metrics/accuracy_top1	metrics/accuracy_top5	val/loss	lr/pg0	lr/pg1	lr/pg2
1	1.3641	0.66833	0.99667	1.4604	0.00023721	0.00023721	0.00023721
2	0.8802	0.74583	0.99833	1.3517	0.00038112	0.00038112	0.00038112
3	0.74688	0.8	0.99917	1.304	0.00043078	0.00043078	0.00043078
4	0.6402	0.8275	1	1.2709	0.00028988	0.00028988	0.00028988
5	0.5607	0.83583	1	1.2574	0.00014851	0.00014851	0.00014851

Model 2 with 5 epochs batch-size 16

	train/loss	metrics/accuracy_top1	metrics/accuracy_top5	val/loss	lr/pg0	lr/pg1	lr/pg2
1	1.3714	0.67333	0.99583	1.4695	0.00023701	0.00023701	0.00023701
2	0.87419	0.7525	0.99833	1.3531	0.00031826	0.00031826	0.00031826
3	0.72541	0.8	1	1.3083	0.00024242	0.00024242	0.00024242

Model 3 with 3 epochs batch-size 20

	train/loss	metrics/accuracy_top1	metrics/accuracy_top5	val/loss	lr/pg0	lr/pg1	lr/pg2
1	1.3714	0.67333	0.99583	1.4695	0.00023701	0.00023701	0.00023701
2	0.87116	0.74917	0.99833	1.3523	0.00038096	0.00038096	0.00038096
3	0.72423	0.79333	1	1.3099	0.00043066	0.00043066	0.00043066
4	0.61149	0.82417	1	1.2679	0.00028988	0.00028988	0.00028988
5	0.53747	0.83917	0.99917	1.2517	0.00014851	0.00014851	0.00014851

Model 4 with 5 epochs batch-size 20

Model 1 achieves a top-1 accuracy of 80.2% at the end of epoch 3, which started from 66.8% at the first epoch. showcasing a strong capability to accurately classify images. The top-5 accuracy marginally increased over time, from 99.7% to 99.9%.

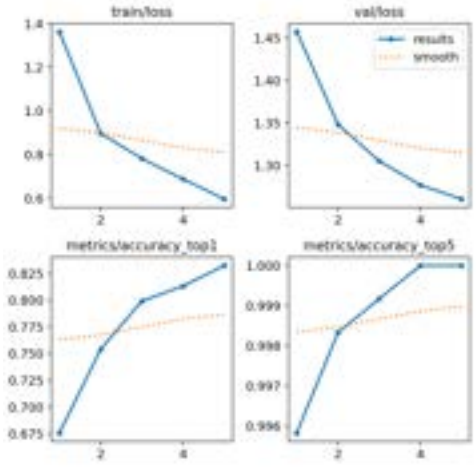
Model 3 has the same epochs but increased batch size which initially started with 67.3% of the top-1 accuracy, which increased to 80% at epoch 3. It had 99.6% of the top-5 accuracy at first, and it increased to 100%.

Comparing these two models, we noticed that although model 1 has an overall higher top 1 and top 5 accuracy rate, it has the larger final loss value in comparison to model 3 with the same batch size, where the gap is slightly larger than the gap between the accuracy rates. Although model 3 started with a slightly higher loss value than model 1, it seems to be training better to result in the final lower loss value. Moreover, model 3 is also increasing its accuracy rates better than model 1 between the epochs. This suggests that Model 3, despite its lower top 1 accuracy, can be a better model.

Model 2, with a longer training duration of 5 epochs, relatively refined its classification performance, achieving a top-1 accuracy of 83.6% by the end of its 5 epochs. The top-5 accuracy, which started from 99.7%, increased to 100% at epoch 4 and maintained the accuracy rate at epoch 5. Compared to **Model 1** with the same batch size of 16, **Model 2** has a better performance in terms of both accuracies.

Model 4 initially achieved 67.3% for top-1 accuracy, which increased to 83.9% at epoch 5. It also achieved a top-5 accuracy of 99.6%, which increased to 100% for epochs 3 and 4, however, it slightly decreased to 99.9% at epoch 5. Compared to **Model 3** with the same batch size of 20, **Model 4** has a better performance in terms of both accuracies most of the time.

Through observations, the team can see that the model with higher epochs and higher batch size seems to be performing better. Model 4 with the higher epochs and higher batch size is the model with the highest accuracy right now. However, our team initially expected the model with lower batch size to be performing better because of the more frequent update and the results above conflicted with our hypothesis. Therefore, we decided to hypertune one last time with a lower batch size to see if there is a difference. We trained Model 5 on 5 epochs with the batch size of 12.



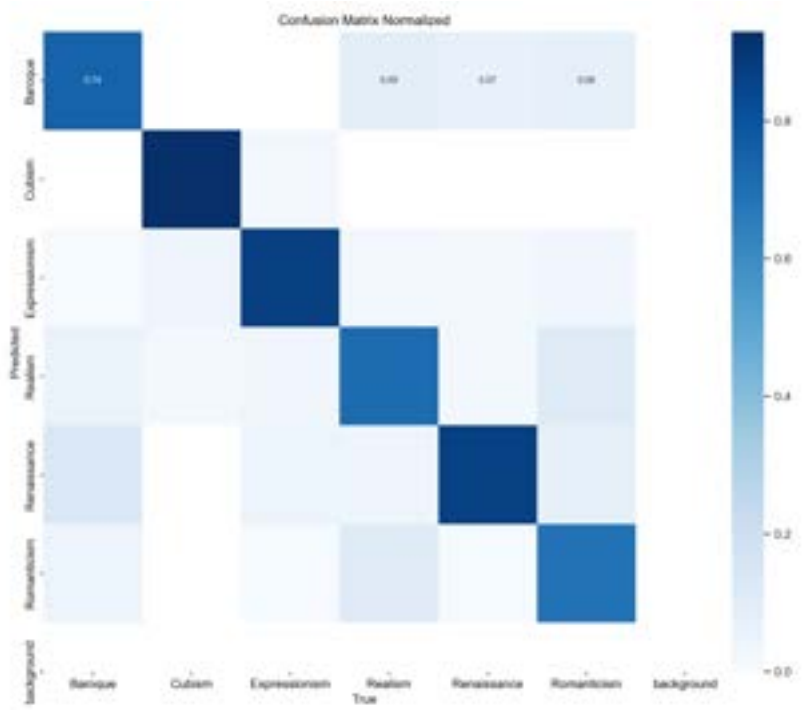
Model 5 with 5 epochs batch-size 12

	train/loss	metrics/accuracy_top1	metrics/accuracy_top5	val/loss	lr/pg0	lr/pg1	lr/pg2
1	1.3625	0.67583	0.99583	1.4574	0.00023741	0.00023741	0.00023741
2	0.89684	0.75417	0.99833	1.3484	0.00038127	0.00038127	0.00038127
3	0.7815	0.79917	0.99917	1.3053	0.0004309	0.0004309	0.0004309
4	0.68774	0.81333	1	1.277	0.00028988	0.00028988	0.00028988
5	0.59603	0.8325	1	1.2605	0.00014851	0.00014851	0.00014851

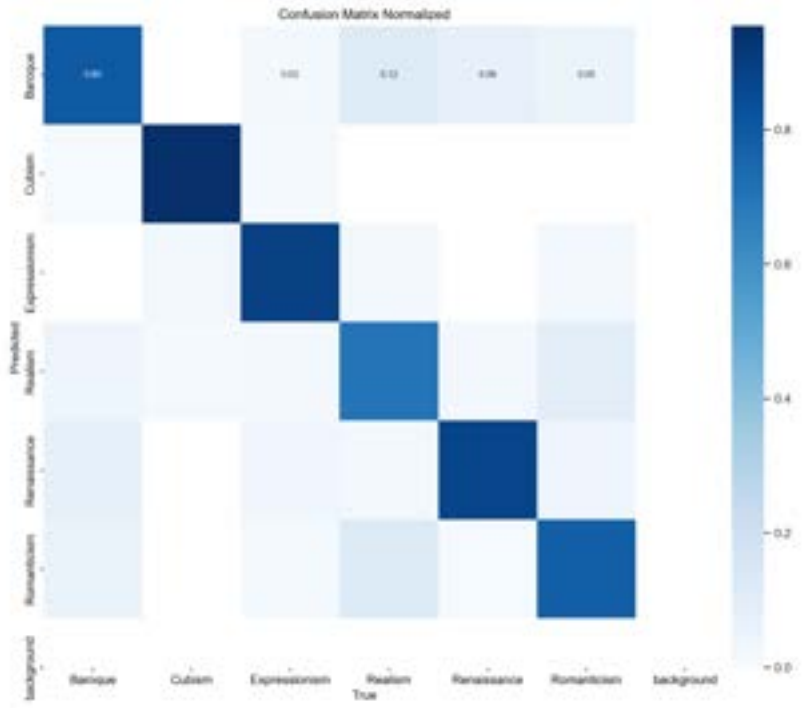
Model 5 with 5 epochs batch-size 12

Model 5, had an initial top-1 accuracy rate of 67.6% which peaked at 83.3% at epoch 5. The top-5 accuracy also increased, from 99.6% to 100%. Unlike our team's prediction, model 5 has a lower accuracy rate than both the models that shared the same epochs (with batch size of 16 and 20), making **Model 4** the model with the highest top 1 accuracy rate.

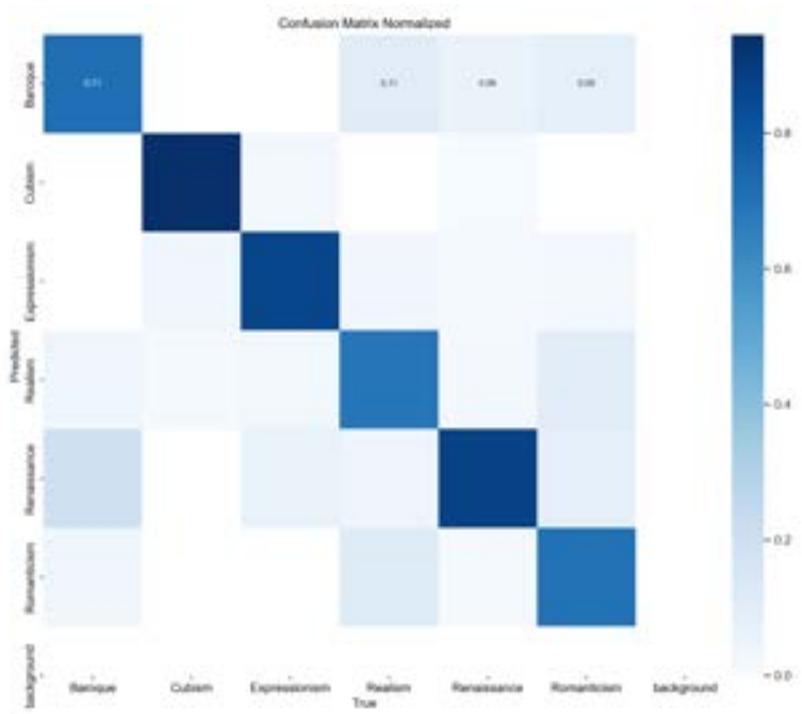
3. Confusion Matrix



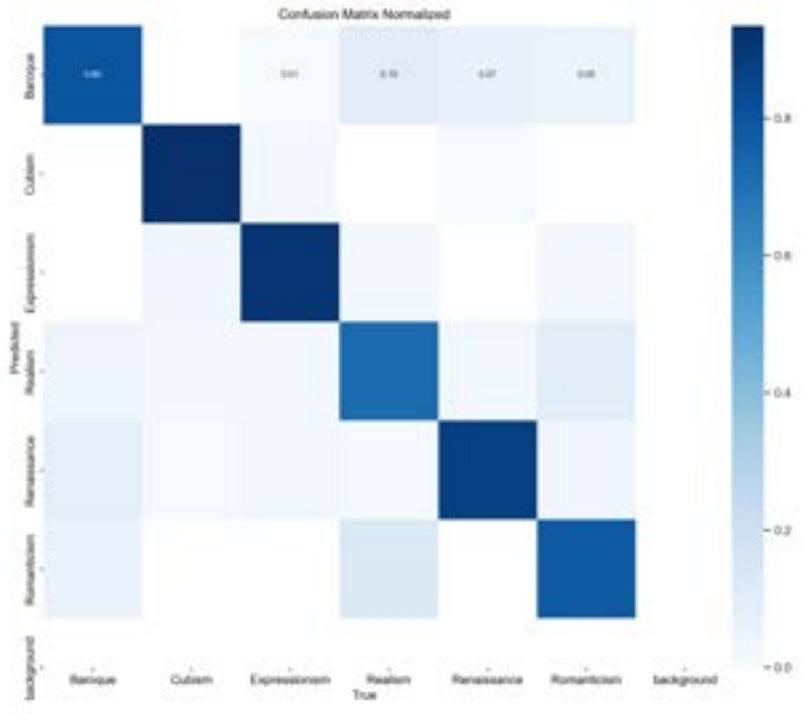
Model 1 with 3 epochs batch-size 16



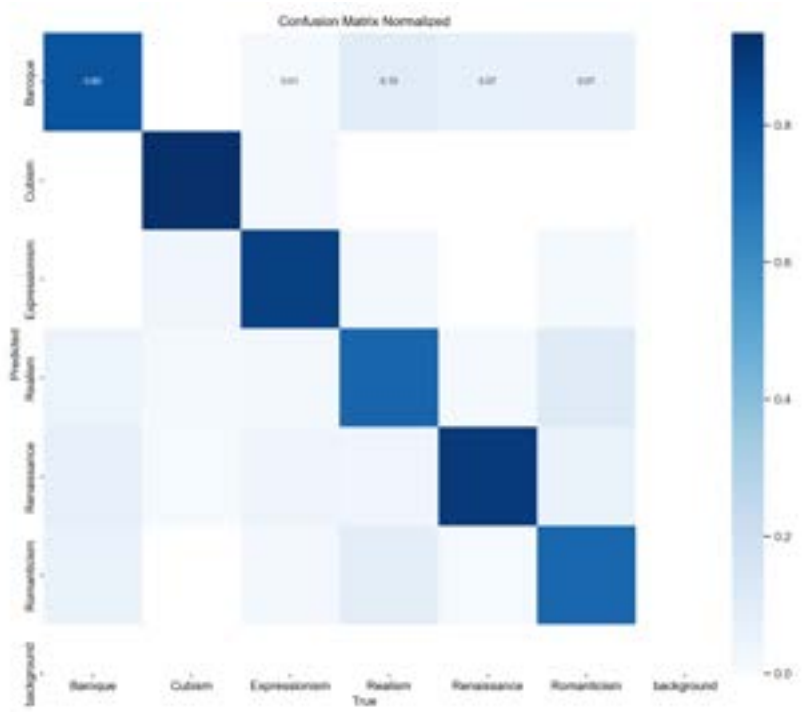
Model 2 with 5 epochs batch-size 16



Model 3 with 3 epochs batch-size 20



Model 4 with 5 epochs batch-size 20



Model 5 with 5 epochs batch-size 12

Considering the purpose of the model, which is to identify artwork correctly, the most crucial values are true positives and false positives. We aim to maximize true positives while minimizing false positives in order to increase the accuracy of item identification and avoid misinformation by incorrectly categorizing an artwork as a style it does not belong to.

Overall, the confusion matrices demonstrate that the true positives have the highest rates among other prediction results. This reflects that all models categorize each artwork correctly with the highest chances. **However, most counts are missing except for some numbers on the top row, possibly due to confusion that results in multiclass classification or issues related to thresholding or aggregation.**

<u>Model 1 with 3 epochs batch-size 16</u>

- True Positive: 74% of the actual Baroque instances were correctly identified.
- False Positive: 24% of actual Baroque instances were wrongly identified as
    - Realism 9%
    - Renaissance 7%

    - Romanticism 8%

Model 2 with 5 epochs batch-size 16

- True Positive: 80% of the actual Baroque instances were correctly identified.
- False Positive: 27% of actual Baroque instances were wrongly identified as
    - Expressionism 2%
    - Realism 12%
    - Renaissance 8%
    - Romanticism 5%

Model 3 with 3 epochs batch-size 20

- True Positive: 71% of the actual Baroque instances were correctly identified.
- False Positive: 26% of actual Baroque instances were wrongly identified as
    - Realism 11%
    - Renaissance 6%
    - Romanticism 9%

Model 4 with 5 epochs batch-size 20

- True Positive: 80% of the actual Baroque instances were correctly identified.
- False Negative: 23% of actual Baroque instances were wrongly identified as
    - Expressionism 1%
    - Realism 10%
    - Renaissance 7%
    - Romanticism 5%

Model 5 with 5 epochs batch-size 12

- True Positive: 80% of the actual Baroque instances were correctly identified.
- False Positive: 25% of actual Baroque instances were wrongly identified as
    - Expressionism 1%
    - Realism 10%

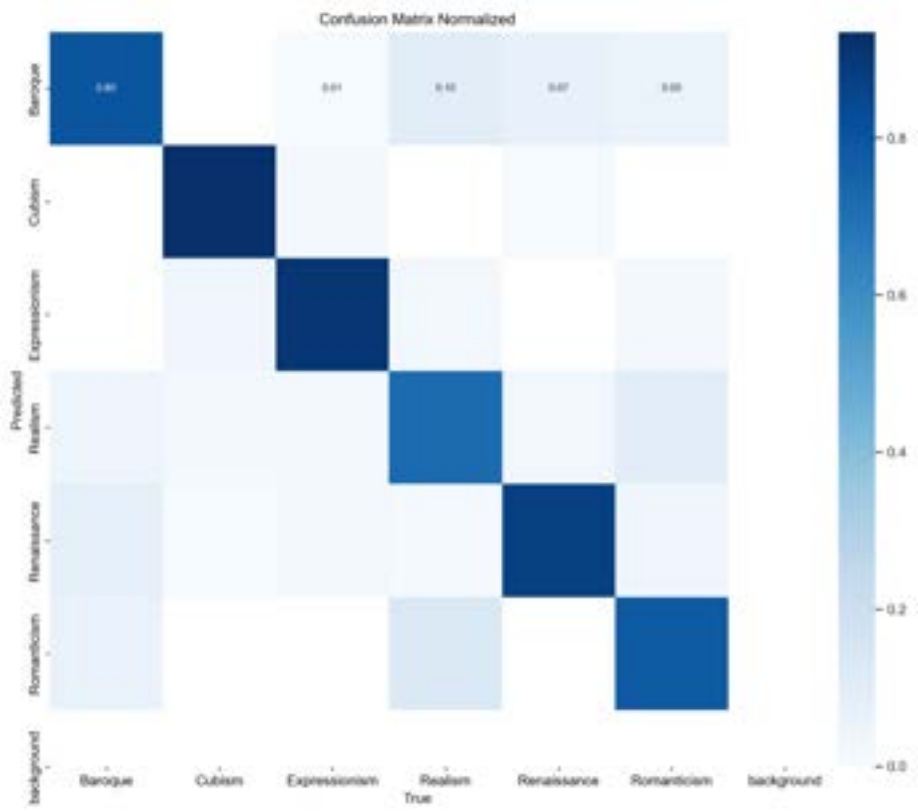
- Renaissance 7%
- Romanticism 7%

Models 2, 4 and 5, which have 5 epochs, have the highest rate of true positives at 80%. Among the models, Model 4 has the lowest false positive rate at 23% among the models, again proving it to be the best performing model, while Model 2 has 27% and Model 5 has 25%.

Evaluating the best model: Model 4

The analysis suggests that Model 4 has the best performance with the highest accuracy of 83.9%. Our team used Model 4 for our next step of the best model evaluation.

1. Confusion Matrix for validation



- True Positive: 80% of the actual Baroque instances were correctly identified.

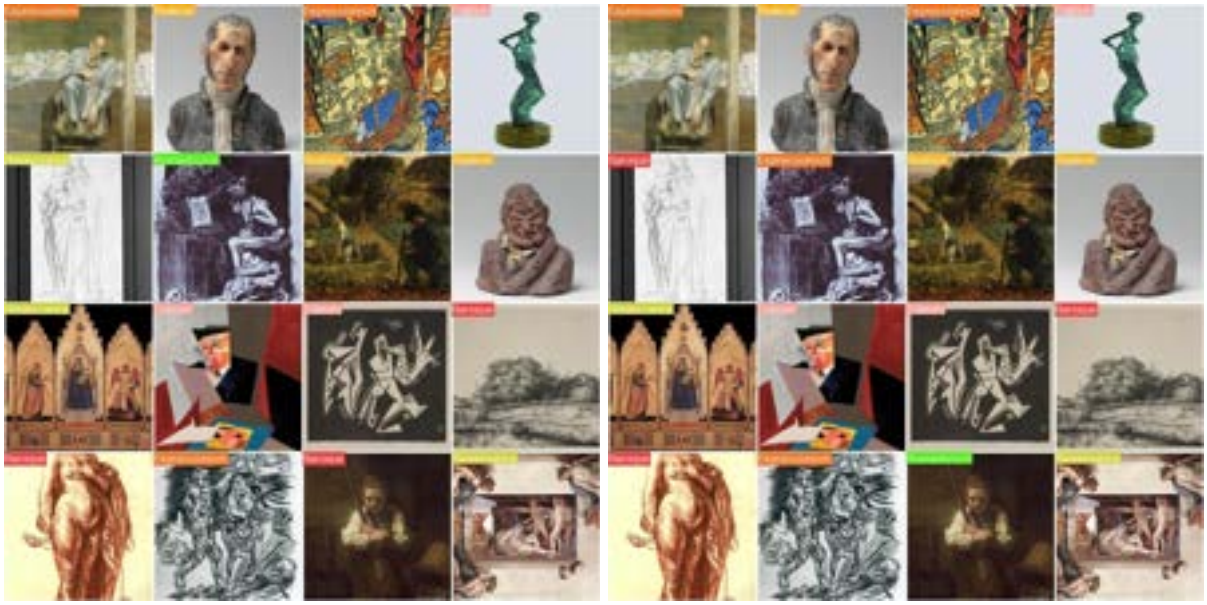
- False Positive: 23% of actual Baroque instances were wrongly identified as:
    - Expressionism: 1%
    - Realism: 10%
    - Renaissance: 7%
    - Romanticism: 5%

2. <u>Batch Prediction based on Labels</u>

Examining the performance of our model during the validation process using 3 batches of 16 images each to see if our model can correctly identify the true labels.

<u>First Batch:</u>

The model incorrectly identified 3 out of 16 objects, specifically misclassifying a Renaissance instance as Baroque (second-row first image), a Romanticism instance as Expressionism (second-row second image) and a Baroque instance as Romanticism (third-row third image).



Labels (left) and Predictions (right)

Second Batch:

The model incorrectly identified 3 out of 16 objects, specifically misclassifying a Romanticism instance as Baroque (second-row first image), a Baroque instance as Renaissance and an Expressionism instance as Realism (third-row first image).

Labels (left) and Predictions (right)

Third Batch:

The model incorrectly identified 2 out of 16 objects, specifically misclassifying a Baroque instance as Romanticism (second-row first image), a Realism instance as Baroque (third-row second image), and a Baroque instance as Realism (third-row last image).

Labels (left) and Predictions (right)

Interpretation: The model correctly identified a majority of the art styles in each batch, with only 3 out of 16 misclassifications in the first and second batches, and 2 out of 16 in the third batch. This suggests that the model has a decent base accuracy and can correctly identify art styles most of the time. We noticed that the model often mistaken certain art movements as others such as mistaking between Baroque and Renaissance, between Romanticism, Baroque and Expressionism, and between Realism and Baroque. Cubism even though had the least data, was the class that was easily classified by the model, mostly due to its distinctive features.

Prediction (Testing Images outside of the Dataset)

1. Baroque - Model incorrectly predicted test images most times (1/3)

2. Cubism - Model correctly predicted test images (3/3)

3. Expressionism - Model incorrectly predicted test images most of the time (1/3)

4. Realism - Model incorrectly predicted test images most of the time (1/3)

5. Renaissance - Model incorrectly predicted test images most of the time (1/3)

6. Romanticism - Model correctly predicted test images (2/2)

Potential Reasons Behind Model's Shortcomings

Through validation and prediction, we noticed that the model failed to distinguish between certain art styles, after further researching of each movement, there are some potential explanation:

- Similar characteristics of art styles:
    + **Baroque and Renaissance:** These styles are historically and stylistically adjacent, often sharing intricate details and dramatic expressions, which might be causing confusion for the model.
    + **Romanticism, Baroque, and Expressionism:** These styles, while distinct, can feature intense emotional expressions and dramatic compositions, potentially leading to overlaps in the model's feature recognition.
    + **Realism and Baroque**: This confusion might stem from the model's inability to distinguish between the realistic depiction of subjects in Baroque, which often includes dramatic, high-contrast lighting, and the everyday subject matter of Realism.
- Limited dataset: Given the extensive size of works for each movement, by limiting our files to a maximum of 1000 each classes, we also limited certain features and characteristics of the class where they can be useful to distinguish between each other.

Challenges

We faced significant challenges with handling our initial large dataset of approximately 5000-6000 files per class. OurJupyter Notebook application crashed multiple times, stating the kernel died and kept restarting during model training, leading to the loss of unsaved results. To mitigate this issue and stabilize our computing environment, we made the decision to reduce the dataset size by limiting each class to 1000 images. This

adjustment helped prevent further crashes and ensured smoother progress in our model training processes, although certain shortcomings as mentioned previously also occurred. We encountered a notable issue with our confusion matrix where it only displayed results for the Baroque class, failing to show any data for the other art style classes. This anomaly in the confusion matrix caused some problems as we did not fully have this piece of data, which can be helpful for the process and interpretation. However, we still had enough information while hypertuning and choosing the best model.

Reference List

Dataset

WikiArt Art Movements/Styles. (2023, January 3). https://www.kaggle.com/datasets/sivarazadi/wikiart-art-movementsstyles

Painting ERAS Detection Computer Vision Dataset by ArtAncestry. (n.d.). Roboflow. https://universe.roboflow.com/artancestry/painting-eras-detection/browse?queryText=class%3Acubism&pageSize=50&startingIndex=0&browseQuery=true