PNTING: Detecting AI in the pAInting world

Mhar Tenorio
Stanford University
mhar@stanford.edu

Abstract

In recent years, generative models have become more advanced, and the line between AI-generated and human-generated images has slowly blurred. This has raised concern within artists, who argue about how this rise affects the value and work of real artists. In this project, we plan to train a model that detects whether an artwork is AI- or human-generated. We use the AI-ArtBench dataset, which contains images of real and AI-generated artworks across ten different art styles. We compare the performance of deeper CNNs and finetuning pretrained models using our dataset with our baseline models of simple MLP and shallow CNN networks. Our models all outperformed the baseline models, with the best model achieving an accuracy of 67.89% in this classification task on 20 labels (either AI or human, differentiating across 10 art styles). While our models have made mistakes in differentiating across the ten art styles, we found that they were able to differentiate between real vs. AI quite well, with accuracies for this binary task reaching 96.47%.

1. Introduction

As generative models like DALL-E become more advanced, AI-generated images have permeated popular culture and entertainment, from movies like Civil War and Late Night With the Devil using AI-generated movie posters and title cards in their film, respectively. [13] Sites like childbook.ai use AI to create illustrations for children's books. With text-to-image models and prompts, these generative models have been used to produce images of artworks mimicking the style of real-life, human-made artworks. [18] Artists argue that the rise of AI art depreciates the value of art and the work of real-life artists. [14]

In this project, we hope to create a model that verifies the authenticity of a painting. We will train a model that evaluates and differentiates whether an artwork is generated by AI or a specific artist. It will be trained on a combination of the works of artists from the 14th to the 21st century and works generated by AI in ten distinct artistic styles. The ten artistic styles we will explore are Art Nouveau, Baroque, Expressionism, Impressionism, Post impressionism, Realism, Renaissance, Romanticism, Surrealism, and Ukiyo-e.

Our model would take in an artwork image of size 32x32. This is a classification task, where the output would be one of the 20 possible labels (either AI or human, differentiating across 10 art styles for each category). The format of each label is `[human|ai]_[art style]`. We do art style differentiation because we recognize how users usually specify an art style when trying to recreate an artwork using AI. We also wanted to assess whether there are specific art styles that AI models currently recreate well and how well the models can differentiate from different art styles.

In this project, we will pass our dataset into simple MLP and shallow CNNs as our baselines. We also input our dataset into deeper CNNs with more convolutional blocks and fully-connected layers that we implement. Lastly, We will also finetune on pretrained models, such as ResNet, GoogLeNet, and EfficientNet, using our dataset.

2. Literature Review

Recent technological developments have led to the rise of AI art. One of these advancements is Neural Style Transfer (NST). NSTs employ CNNs to mimic famous painting styles in real images. [6] This demonstrates the ability of AI models to differentiate between different art styles and to generate artworks given a specific prompted style. Generative Adversarial Networks (GANs) [4] have also contributed to AI-art. By employing a generator and discriminator architecture, GANs are able to produce more convincing images that mimic an artwork, such as a painting. [3] More recently, OpenAI's Dall-E has introduced a text-to-image generation model, allowing users to write prompts for images to generate. [12] Users can then ask the model to generate an artwork following a specific style. As technological advancements increase the quality of these AI-generated artworks, artists express concerns over ownership, copyright, and ethical issues of such image generation. [3]

Humans are not as good at differentiating between real and state-of-the-art AI-generated images. [11] In a study

conducted by Lu *et al*., they tested each participant to classify 100 randomized images as real or not. Images contain diverse subjects, from people, animals, plants, to landscapes. They found that participants, on average, only got 61.3% of the questions right (with a misclassification rate of 38.7%), demonstrating the ability of SOTA AI-generated images to deceive a general audience. [11] They also observed that participants misclassified real images as AI 33.1% of the time. A similar study that evaluates how well people classify AI-generated and human-generated paintings, social media images, news photos, and anime corroborate this finding. [9] Lu *et al*. writes that this shows how increasing realism in AI images "erode people's trust in accurate information". Lastly, they found that people excelled in identifying AI-generated images of people, but struggled in AI-generated images of objects the most. [11]

AI models, on the other hand, may provide more insight. [11] CIFAKE [2] by Bird *et al*. is a dataset designed for identification of AI-generated synthetic images. They trained and tested on the CIFAR-10 dataset, consisting of 60,000 32x32 RGB images of real subjects (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck). [8] They use a stable diffusion model to generate synthetically equivalent images to the CIFAR-10 dataset, with 6,000 images for the ten classes. Each image is generated from the prompt 'a photograph of [subject]' with slight variations. They then used a Convolutional Neural Network to perform a binary classification task (either Real or Fake) on these images. Their best model achieves an accuracy of 92.98%. What they observed is that there is a different distribution of features for real and AI images. They found that classifying real images involved looking at majority of the image, whereas classifying AI images involved looking at select parts of the image that are considered "visual glitches" that depart them from reality. Such examples could be unreadable text, anatomical errors, or lack of certain details. This signifies how, despite the convincing nature of AI images, there are still visual gaps that will allow the model to differentiate them from real images.

GenImage expands on CIFAKE by creating a million-scale benchmark dataset of real and AI images using current state-of-the-art Diffusion and GAN models. [19] They generated 1.3 million fake images on the 1000 class labels found in ImageNet. They performed two tasks on their dataset – cross-generator image classification and degraded image classification, achieving best accuracies of 70.3% and 82.2%, respectively. This demonstrates how different image generator models lead to different and distinct visual features, introducing difficulty in creating an all-in-one AI detector for images.

They found that a pre-trained ResNet-50 model fine-tuned on the GenImage dataset is able to generalize to other image content, such as AI-face and AI-art detection, with best model accuracies of 99.0% and 95.0%, respectively. This demonstrates the effectiveness of transfer learning on image classification tasks. Models that have demonstrated effectiveness for image classification tasks include ResNet [5], EfficientNet [17], and GoogLeNet. [15]. This also shows the potential of improving these accuracies by training on dataset that is more domain-specific. We would explore how these pre-trained models would perform if trained on an art-specific datasets.

Such domain-specific datasets include ARIA (Adversarial AI-Art) [9] and ArtBench. [10] The ARIA dataset includes 140K images in five categories: paintings, social media images, news photos, disaster scenes, and anime pictures. [9] ArtBench is a class-balanced, annotated, and standardized dataset for benchmarking artwork generation. [10] It includes 60,000 real artworks across 10 distinct art styles, spanning from the 14th to the 21st century. They found that generative models finetuned on their dataset was more effective at creating AI artworks of landscape, cityscape, and marina artworks compared to creating portraits. Since ArtBench is more specific to paintings, we will incorporate the ArtBench dataset into the dataset we will be using.

3. Dataset

We are using the AI-ArtBench dataset on Kaggle.[1] This dataset contains 180,000+ artwork images. Around $\frac{2}{3}$ of the dataset (120,000) are AI-generated artworks using Latent Diffusion and Stable Diffusion. The AI-generated artworks are in 10 artistic styles, separately generated using Latent Diffusion and Stable Diffusion. These AI-generated works are 256x256 and 768x768 in resolution, respectively.

Around $\frac{1}{3}$ of the dataset (60,000) are real human-made artworks taken from the ArtBench-10 dataset, which is a dataset that contains artworks from the same 10 artistic styles as the AI-generated ones. These human-made works are 256x256 in resolution.

The training set contains 155,015 artworks (50,000 human + 105,015 AI), while the test set contains 30,000 artworks (10,000 human + 20,000 AI).

3.1. Pre-processing Methods

To start, we pruned the amount of AI-images in the training set to achieve a better balance between the human-made and AI-made artworks (so AI-artworks are not overrepresented in our training data). We pruned it such that there are a total of 5,000 artworks for each artistic movement in the training set – with 2,500 generated using Latent Diffusion and 2,500 using Stable Diffusion. (There are 5,000 human-made artworks for each art style) In the end, our training set contained 100,000 artworks, with a balance of human-made and AI-made artworks. We then took 10,000 images

[1] https://www.kaggle.com/datasets/ravidussilva/real-ai-art

from the training set as the validation set. (90-10 split) We also resized each AI-image generated using Stable Diffusion in both the training and testing set to be 256x256 pixels to match the other images.

We then extract the label for each image by parsing the filename and directories. We labeled each AI artwork with 'AI', while we labeled each human-made artwork with the name of the artist. We transform each image so they are all of the same 32x32 size and then feed this into Pytorch's ImageFolder to get the dataset. We then feed this dataset to the Pytorch DataLoader to get batch data. In the end, we had 20 labels for this classification task.

4. Approach & Methods

4.1. Baseline Models

For our baseline models, we implemented two approaches: a simple Multilayer Perceptron (MLP) model, and a simple Convolutional Neural Network (CNN) model.

Simple MLP Model Our first baseline model is a simple MLP model. It first flattens the input to a size of $32 * 32 * 3$. We then pass it to a fully connected layer with 128 units. We pass the output to a ReLU activation layer before passing it to a final fully connected layer that has 20 units, one for each classification label.

Simple CNN Model Our second baseline model is a shallow four-layer CNN network. We first pass a batch of 32x32 images into a 2D Convolutional Layer with 32 filters, a padding of 2 pixels, and a 5x5 kernel size. We pass the output into a ReLU activation function. We then pass this into a Max Pooling layer with a 2x2 kernel size and a stride value of 2. We pass it into another 2D Convolutional Layer with 16 filters, a padding of 1 pixel, and a 3x3 kernel size. We then flatten this and pass it to a fully connected layer that has 20 units.

4.2. Proposed Model

We also created two, more complex CNNs.

Model #1 Our first model consists of two convolutional blocks, a flattening layer, a fully connected block, and a last linear layer to generate an output. This has a total of 16 layers. The first convolutional block consists of passing a 32x32 image into a 2D Convolutional layer with 32 filters, a padding of 2 pixels, and a 5x5 kernel size. We then take this output and pass it onto a 2D BatchNorm Layer with 32 features and an $\epsilon = 1e - 5$ for numerical stability. We apply a dropout layer with a dropout probability of 0.15. We apply a ReLU activation to the output, before passing it to a 2D MaxPool layer with a kernel size of 2x2. The second convolutional block follows this structure, but the 2D Convolutional layer has 16 filters, a padding of 1 pixel, and a 3x3 kernel size. and the 2D BatchNorm layer has 16 features. After the two convolutional blocks, we pass the output to a flattening layer. We pass it to our fully connected block, which first starts by passing it to a linear layer with 128 neurons or outputted features. We then pass this onto a 1D BatchNorm layer, before applying dropout and ReLU activation. The final step of this architecture is passing it to a linear layer that outputs of the 20 possible classes for this task.

Model #2 The second model is similar to the first model, but is deeper and more complex. This model consists of (in order) three convolutional blocks, a flattening layer, two fully connected blocks, and a last linear layer to generate an output. This has a total of 25 layers. The first two convolutional blocks are the same as Model 1's. The third convolutional block consists of a 2D Convolutional layer with 8 filters, a padding of 2 pixels, and a 3x3 kernel size. We then take this output and pass it onto a 2D BatchNorm Layer with 8 features and an $\epsilon = 1e-5$ for numerical stability. Similarly, we apply dropout with a probability of 0.15 and ReLU activation before applying a 2D MaxPool layer with a 2x2 kernel size. The first fully connected block consists of a linear layer with 128 neurons, a 1D BatchNorm layer with 128 features, dropout, and ReLU activation. The following, second fully connected block consists of a linear layer with 64 output neurons, a 1D BatchNorm layer, dropout, and ReLU. Similarly, we pass the output to a linear layer to output a label.

4.3. Transfer Learning

We also incorporated Transfer Learning by fine-tuning pre-trained models. To do this, we replace the last fully-connected layer of the model so that the output matches the number of classes in our dataset (20 labels). We do not freeze any layers, but rather we train all layers using the existing pre-trained weights as the starting point.

ResNet Given that this is an image classification task and the findings observed in GenImage [19], we finetune on a pretrained ResNet model using our dataset. ResNet models work well for image classification tasks. They introduce a residual block that allows for training deeper networks by mitigating the vanishing gradient problem. [5] Residual blocks introduce skip connections that allows to fit a residual mapping ($H(x) = F(x) + x$) instead of directly trying to fit a specific mapping ($H(x)$).

The ResNet architecture first starts with a convolutional layer with 64 filters, kernel size of 7x7, and a stride of 2 pixels. After passing this to a normalization layer and ReLU activation function, we apply a pooling layer. The ResNet architecture stacks then stacks residual blocks, with each residual block containing two 3x3 convolutional layers. At the end, we then apply an average pooling layer and flattening layer. We only have one fully connected layer at the end, which is used to generate an output. [5]

We will evaluate on two ResNet models: ResNet-18, a

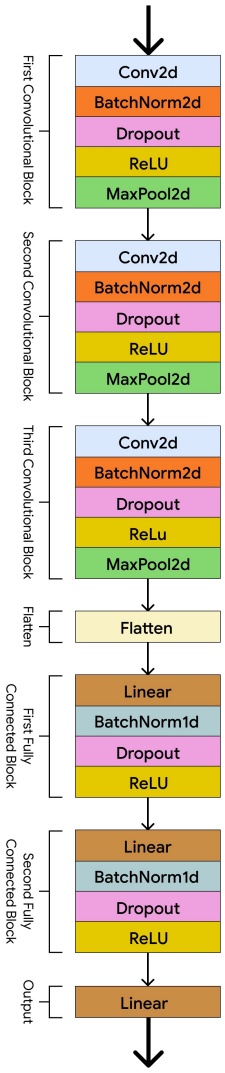

Figure 1. Model 2's architecture. It consists of three convolutional blocks, a flattening layer, two fully connected blocks, and a last linera layer to generate an output. Model 1 is similar, but with only the first two convolutional block, and only one fully connected block before the final linear layer.

shallower, less complex model, and ResNet-50, a 50-layer model. While both employ residual blocks and the general architecture above, ResNet-50 also utilizes bottleneck layers (three convolutional layers: 1x1 conv with 64 filters, 3x3 conv, and 1x1 conv with 256 filters) that helps with reducing the computational complexity of deeper models. Reducing the dimensions periodically (and then returning to the previous state) allows the model to learn with fewer parameters compared to just keeping a 3x3 convolutional block throughout, which becomes useful for more complex models. ResNet-18 does not have bottleneck layers due to its more shallow nature.

GoogLeNet We use a deep CNN pre-trained model called GoogLeNet. [15] This model is known for its deep, but computationally efficient, network. It consists of 22 layers, but with only 5 million parameters. It is able to do so through "inception modules." Given a previous layer, inception modules apply parallel filter operations with multiple convolutional receptive field sizes (1x1, 3x3, and 5x5) and a MaxPool layer with kernel size of 3x3. For computational efficiency, they employ dimension reduction using bottleneck layers of 1x1 convolutional layers before passing it to the 3x3 and 5x5 convolutional layers. They pass the output of the pooling operation to a bottleneck layer for efficiency. They then concatenate all the outputs before passing it onto the next block.

The overall architecture of GoogLeNet starts with a "stem network" with a 7x7 convolutional layer with stride 2, a MaxPool layer with kernel size 3x3 and stride 2, a 1x1 convolutional layer, a 3x3 convolutional layer, and another MaxPool layer with kernel size 3x3 and stride 2. We then stack the inception modules on top of each other. After the last inception module, we pass the output to a global average pooling layer with kernel size of 7x7, a fully connected layer, and finally apply the softmax function to generate the final output. [15]

EfficientNet Another CNN architecture that works well for image classification tasks is EfficientNet. [16] EfficientNet introduced model scaling. Model scaling involved scaling all dimensions–depth (number of layers), width (number of channels), resolution (input image size)– of the model uniformly by a fixed compound coefficient. This is useful because if resolution increases, then having a deeper network and higher number of channels will allow us to learn more from this increased resolution image. EfficientNet employed neural architecture search to create these models. These models also employ inverted bottleneck convolutional layers (MBConv). MBConv layers use a 1x1 convolutional layer to expand the input (rather than reduce it), then a depthwise 3x3 convolutional layer, and a 1x1 convolutional layer to reduce it again. The model has achieved state-of-the-art accuracy on CIFAR-100. [16]

4.4. Training Configuration

Each experiment was run for 10 epochs with a learning rate of 0.001. We employ an exponential learning rate scheduler that updates the learning rate each epoch:

$$lr = lr_0 \exp(-\gamma \cdot \text{epoch})$$

with a hyperparameter γ that dictates how much to decay our learning rate. We start with the larger learning rate to make larger jumps towards the solution in the beginning, but decrease it in further epochs so we don't oscillate towards

the optimal solution. This leads to a more stable learning process.

We used an SGD optimizer with Nesterov momentum and a momentum γ value of 0.9. We use this optimizer because this works well for CNNs and outperforms Adam when employing Nesterov momentum. Momentum takes into account the previous gradients by keeping a running average. This helps smooth out the SGD updates, prevent oscillations, and help the model converge faster. With momentum, the update then becomes:

$$v_{t+1} = \gamma v_t + \eta \nabla L(\theta_t - \gamma v_t)$$

$$\theta_{t+1} = \theta_t - v_{t+1}$$

We use 64 as our batch size. We found that this number provides a good balance of computational and training efficiency compared to smaller batch sizes and improved generalization compared to larger batch sizes. [7]

Cross entropy loss served as our loss function because it generalizes well for multi-label image classification tasks. The loss is calculated as:

$$L(y, \hat{y}) = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

We then took our best model, and ran it for 20 epochs under the same hyperparameters. Each experiment was run on an NVIDIA T4 GPU.

We use accuracy as the evaluation metric. We thought that this is an appropriate metric given that the dataset is balanced on the amount of images per category. We evaluate on two metrics of accuracy. The first one is accuracy on a 20-label classification task, where the model predicts whether the art work is human or AI-generated AND the style that it was trying to follow. We also employ a more generalized accuracy, where we ignore the art style and only check whether the model outputs a given artwork as AI- or human-generated. Since labels follow *[human or ai]_[art style]*, we only check whether the first part of the output label for this binary task accuracy.

5. Results & Discussion

5.1. Results

Using the experiment configuration detailed above, we have achieved the following results, as shown in Table 1. We also have achieved the following average loss values across all batches per epoch, as shown in Figure 2.

Model	20-Label Task		2-Label Task
	Train	Test	Test
Simple MLP	35.40%	33.33%	68.08%
Shallow CNN	59.83%	47.16%	73.62%
Model #1	57.12%	53.77%	92.90%
Model #2	45.17%	42.68%	77.96%
EfficientNet	69.69%	63.94%	95.24%
GoogLeNet	69.70%	63.84%	94.93%
ResNet-18	87.51%	62.03%	93.77%
ResNet-50	93.52%	63.65%	94.39%
ResNet-50 (20)	**99.42%**	**67.89%**	**96.47%**

Table 1. The results of the models on the 20-label classification task (human or AI + the style of the work) and the binary task (human vs. AI, disregarding the art style). The last model was ran for 20 epochs.

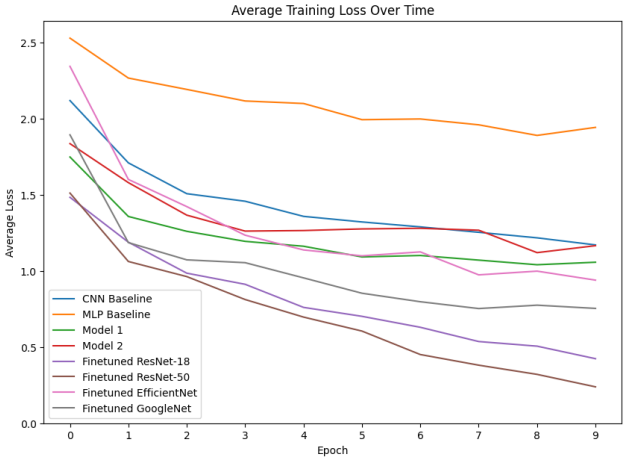

Figure 2. Average loss of each batch per epoch across all models.

5.2. Discussion

5.2.1 Quantitative Analysis

In the 20-label classification task, we can see that all our models outperform our baseline MLP and CNN models in the test sets. In the deeper CNNs that we implemented, we found that the more shallow network with one less convolutional block and fully connected block performed better than the more complex network. However, these models performed worse than the finetuned pre-trained models. These pre-trained models achieved the best results. We then took one of the best performing models and trained it for more epochs and found that the accuracy increased even more.

We can see that the baseline models often started with the highest loss values, while the pre-trained models often started with the lowest. We see that our baseline models

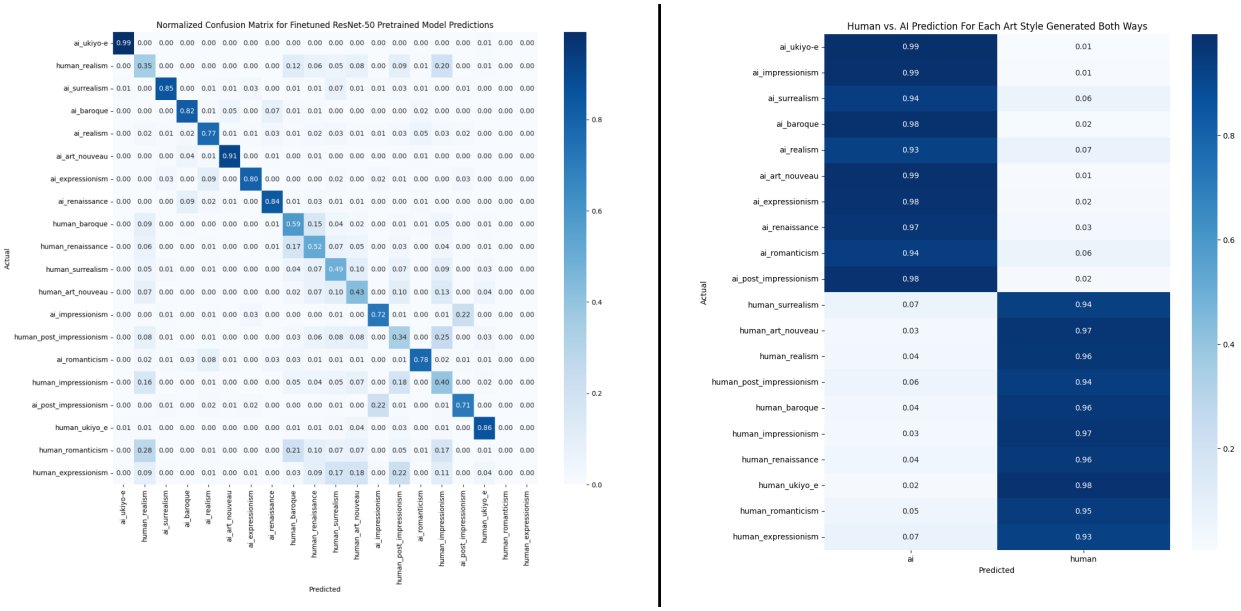

Figure 3. Confusion matrix of our best-performing model, ResNet-50 ran on 20 epochs, on the 20-label classification task. **(L)** Confusion matrix on the same model predictions, but only evaluating on classifying each label for each art category as human or AI and disregarding the art style in the prediction (binary classification task). **(R)**

decrease in loss values the slowest, indicating slower learning. Our own implemented CNNs (Model 1, 2) showed some small fluctuation in the loss values. We can also see that the pre-trained models have a faster decline in loss values, indicating faster learning. This indicates how leveraging pre-trained weights to initialize our model can help with making our training faster and more accurate compared to learning the weights by scratch. However, we should also be wary of overfitting, given the low initial loss values and their faster loss decrease rate. As we can see from Table 1, the ResNet models have a tendency to overfit to the training data, with training accuracies reaching close to around 90%. When trained under more epochs, ResNet-50 almost perfectly overfit to the training data with a training accuracy of 99.42%.

While employing transfer learning by fine-tuning on pre-trained models outperformed our own models, the pre-trained models demonstrated different performance. For example, the ResNet models showed faster convergence compared to the GoogLeNet and EfficientNet. However, the ResNet models tended to overfit more to the training data, compared to GoogLeNet and EfficientNet. This might be due to ResNet's deeper architecture. For example, ResNet-50 is 50-layers deep, compared to GoogLeNet which is only 22-layers deep. It also might be the lack of enough regularization techniques, whereas GoogLeNet and EfficientNet employs the Inception architecture and compound scaling that can help with increasing generalization. Nevertheless, their overall performance on the test set are similar on both tasks.

Now, we will analyze the output of our best performing model, the ResNet-50 trained under 20 epochs, using a normalized confusion matrix (Figure 3). We found that the model was better at classifying AI-made artworks to its specific art categories than human-made artworks. This may signify how artworks produced by an AI model in a specific style mostly follow a similar pattern, demonstrating its limitations in mimicking real artworks. Human-made artworks are often classified as another art style. This makes sense, given how each art style can develop from one another and therefore contain connections and similarities.

Interestingly, it was never able to correctly classify `human_romanticism` and `human_expressionism`. However, it usually classifies artworks belonging to this category with other art styles within the human category. This further exemplifies the model's shortcomings in differentiating between different art styles created by humans.

Despite the model's occasional failures in distinguishing between the different art styles, we found that it often outputs the correct creator. As seen in the right matrix of Figure 3, our best performing model was able to categorize each of the 20 labels by their creator extremely well, with accuracies reaching greater than 90%.

This is not only exclusive to our best model. In Table 1, we can see that most of the models are able to categorize a given artwork as human or AI over 90% of the time. Our best model outperforms the results in GenImage [19], with our model getting an accuracy of 96.47% on this binary

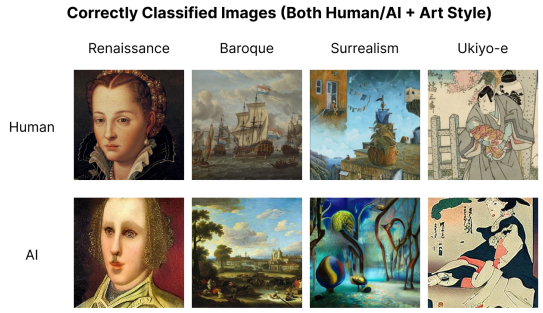

Figure 4. Examples of correctly classified artworks on the 20-label task (human or AI and the art style).

task over their 95.0% accuracy. This indicates that there are clear, learnable features that distinguish AI-generated artworks over human-generated ones. It also shows how we can improve on this task by using a more domain-specific dataset.

5.2.2 Qualitative Analysis

Figure 4 shows examples of correctly classified artworks on their creation type and art label. First, it is interesting to note how AI models are able to mimic and differentiate the general style of a specific art movement to some extent. For example, a Renaissance portrait is vastly different from an Ukiyo-e portrait. We can also see a surreal landscape is different from a baroque landscape.

Here, we can see that there are certain features that differentiate AI-generated artworks from human-generated artworks that helps the model distinguish between the two. For example, AI models are unable to capture faces well, as we can see from the AI-generated Renaissance and Ukiyo-e artworks. Even though they capture the distinct style, they are unable to get the detail of the facial features and proportions right. Other signifiers of AI vs. human for a specific art style is the quality of text. AI-generated artworks with text often have illegible text, both in English and non-English characters.

Sometimes generative AI models are not able to fully capture the style of a specific art movement, allowing our model to distinguish its AI nature. For example, the human and AI surrealist artworks are not very similar. Surrealism usually employs real objects and subjects, but placed in ways that don't obey physical laws. [1] For example, in the human surrealist artwork, there is a boat on top of a plant. While the AI-generated one still follows some of the bizarre elements of surrealism, it does not contain any clear object, subject, or even defined lines and edges. These AI generated surrealist artworks may have been grouped together because of these qualities that aim to mimic surrealism, but

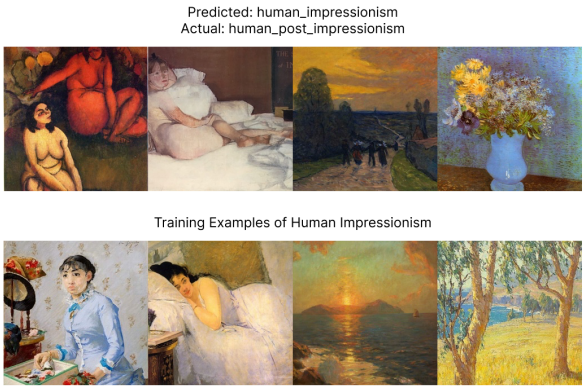

Figure 5. Human-made post-impressionism artworks that were mislabeled as human-made impressionism.

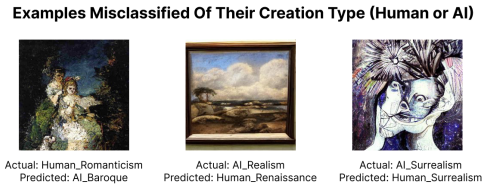

Figure 6. Examples of artworks that were mislabeled in their creation type (AI being classified as human, and vice versa).

fail to do so.

While the model performs well on the binary task of AI vs human, it can sometimes misclassify an artwork to its correct style, even if the predicted creator is correct. As we can see in Figure 5, the images above were classified as human-made impressionism artworks, despite being human-made post-impressionism artworks. Looking at the training examples for human impressionism, we can see the similarities between the two movements. For example, these training examples paint faces and landscapes similarly. The composition and subject matter are also somewhat similar. Historically, the connections between the two makes sense, given how post-impressionism follows impressionism. However, this demonstrates the model's shortcoming on distinguishing between different art styles.

More dangerously, sometimes the model misclassifies an AI artwork as human and a human artwork as AI (Figure 6). While an unclear, deformed face may be characteristic of an AI artwork, as we've discussed in Figure 4, it may be the nature or style of a human artist. The first example shows Adolphe Joseph Thomas Monticelli's work, which usually features big strokes and unclear faces. This could potentially be the reason why it was misclassified as AI. We also have touched upon how surrealist artworks often em-

ploy clear edges and defined objects. Looking at the last example in Figure 6, we can see how this AI-generated artwork has these defined lines and perceivable objects (such as the eyes), which could be the reason why it was classified as human-generated instead.

6. Conclusion & Future Work

In this project, we investigated if we can build a model that can classify two labels: whether it was made by a human or an AI and the art style that the artwork belongs to. We pursued this task because we wanted to see how well generative AI models capture specific art movements and if there are specific characteristics that distinguish AI-made art from human-made art. To do this, we use the AI-ArtBench dataset on Kaggle that has a combination of AI- and human-made artworks across ten different art styles. We have 20 labels, with 10 art styles for both human-made and AI-made categories. We then evaluated on our own CNNs and on pre-trained models fine-tuned on this specific dataset. While we found that both of these methods outperform our baseline, we found that fine-tuning on pre-trained models performed the best, compared to building our models from scratch. In the original 20-class task, our best model was able to correctly classify the type of origin (human or AI) and the art style 67.89% of the time at the test set. In our analysis, we found that the models sometimes struggled in classifying the art style of a given artwork, specifically for human-made artworks. Nevertheless, all the models was able to distinguish an AI-generated artwork from a human-generated artwork (disregarding the art style), generating accuracies over 90% in this binary task.

Ways to expand this project could be expanding the dataset to incorporate other art styles. We can also experiment with deeper, more complex models, such as incorporating vision transformers. Given more computational resources, we can also see how training for longer could affect performance. We can also attempt to solve the overfitting problem to the ResNet models by employing more regularization techniques. For example, we could introduce data augmentation on the artworks or freeze more layers in these pre-trained models.

As the rise of AI image generation becomes more noticeable, artists have expressed their concern over how AI can disrupt and depreciate their livelihood and their craft. With our AI models being able to distinguish between AI-generated artworks and human-generated artworks quite well, our project demonstrates that there exists a striking difference between the two.

7. Contributions & Acknowledgements

I worked on and completed this project alone. Thank you for the CS 231N staff for the great quarter!

References

[1] Artchive. Surrealism art movement: History, characteristics, artwork. `https://www.artchive.com/art-movements/surrealism/`, 2023.

[2] J. J. Bird and A. Lotfi. Cifake: Image classification and explainable identification of ai-generated synthetic images, 2023.

[3] E. Cetinic and J. She. Understanding and creating art with ai: Review and outlook, 2021.

[4] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks, 2014.

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[6] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song. Neural style transfer: A review, 2018.

[7] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima, 2017.

[8] A. Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[9] Y. Li, Z. Liu, J. Zhao, L. Ren, F. Li, J. Luo, and B. Luo. The adversarial ai-art: Understanding, generation, detection, and benchmarking, 2024.

[10] P. Liao, X. Li, X. Liu, and K. Keutzer. The artbench dataset: Benchmarking generative models with artworks, 2022.

[11] Z. Lu, D. Huang, L. Bai, J. Qu, C. Wu, X. Liu, and W. Ouyang. Seeing is not always believing: Benchmarking human and model perception of ai-generated images. In *Neural Information Processing Systems*, 2023.

[12] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. Zero-shot text-to-image generation, 2021.

[13] R. Schwartz. Indie film's 'very brief' use of ai sparks backlash and calls for boycotts. *The Week US*.

[14] S. Shaffi. 'it's the opposite of art': why illustrators are furious about ai. *The Guardian*.

[15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions, 2014.

[16] M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020.

[17] M. Tan and Q. V. Le. Efficientnetv2: Smaller models and faster training, 2021.

[18] Y. Wu. Ai is outrageous — and wonderful. it's also prompting a new art form. *The Washington Post*.

[19] M. Zhu, H. Chen, Q. Yan, X. Huang, G. Lin, W. Li, Z. Tu, H. Hu, J. Hu, and Y. Wang. Genimage: A million-scale benchmark for detecting ai-generated image, 2023.