Classifying Famous Artwork

Max Land, Son Do, Jeremy Pulmano, Victor Hua

Abstract

We perform image classification on different paintings by matching them to their respective artists. We utilize various subsets of the full dataset containing thousands of paintings from 50 of the most influential artists of all time and develop several classifiers to identify a given painting's artist. We train several classifiers -- Logistic Regression (LR), Support Vector Machines (SVM), Random Forests (RF), and Convolutional Neural Networks (CNN) -- to classify the test paintings. Furthermore, we use PCA and K-means clustering for extract latent variables, use Stratified K-Folds to split the data proportionally, convert each painting to grayscale, reduce the number of artists analyzed, and more. As an extension of our work, we performed neural style transfer. Thus, this project provides an interesting introduction to image classification and its creative applications.

Introduction

Image classification has an extremely wide range of applications, from facial recognition within apps such as Google Photos, all the way to recognizing cancer within images of brain scans. In this project, we attempt to extend our coursework and explore one such application of image classification: famous paintings.

One of main goals in this project was to become more familiar with some of the many applications of image/object classification. Computer vision, in particular, is essential to autonomous vehicle engineering, image and face recognition on social networks, and much more.

In our case, studying paintings' characteristics and attributing them to their creators can be of considerable interest to art enthusiasts and especially those in the field of art history. We put a unique emphasis upon using machine learning to understand the ways in which famous artists created their paintings, which can be crucial for determining the many features -- e.g. texture, line use, colors, subject, and much more -- that are indicative of that artist's unique style. Thus, utilizing machine learning models to analyze and classify paintings represents a creative and innovative way to approach the topic of image classification.

Figure 2. Results of style transfer where an image of the Lewis Arts Center is stylized with van Gogh's "Starry Night."

Methods and Materials

We are given a data set of 8354 images with varying dimensions, each one with an author name as a label. We reduce the dimensions of each image to a 40-by-40 image with nearest neighbor interpolation using the Python Image Library (PIL); then, we obtained the RGB values of each pixel.

Classifiers:
Logistic Regression (LR); Random Forest (RF); Support Vector Machines (SVM); and Convolutional Neural Networks (CNN)

Latent Variable Methods:
Principal Component Analysis (PCA); K-means Clustering (K-means)

Results

We decided to reduce the number of artists to the 5 artists with the most paintings, which drastically improves our results. SVM and CNN perform the best, with accuracy of approximately 0.61 and 0.54 on color images, respectively. These are much better results when compared to the initial 0.297 and 0.270 accuracy rates of SVM and CNN, respectively.

Converting our images from color to grayscale proved detrimental to our accuracy, which we attribute to the loss of information by removing color outweighing the benefits of reducing the color variance. We found that the highest color variance of 12,883 in the data set was a painting by Andy Warhol, while the one with the lowest variance of 165 was a piece by Albrecht Durer.

The two artists with the most number of paintings—Edgar Degas and Vincent van Gogh—had the highest prediction accuracies. We attribute this to our classifiers having more data to more accurately train for those two artists in particular.

Color: 50 Artists				
Classifier	Accuracy	Precision	Recall	F1
LR	0.161	0.096	0.092	0.089
KNN	0.154	0.169	0.102	0.083
RF	0.155	0.027	0.051	0.028
SVM	0.297	0.237	0.145	0.132
CNN	0.270	0.153	0.181	0.149

Grayscale: 50 Artists				
Classifier	Accuracy	Precision	Recall	F-1
LR	0.096	0.104	0.063	0.069
KNN	0.130	0.188	0.092	0.086
RF	0.137	0.024	0.038	0.018
SVM	0.234	0.175	0.105	0.093
CNN	0.260	0.128	0.199	0.118

Color: 5 Artists				
Classifier	Accuracy	Precision	Recall	F1
LR	0.461	0.431	0.456	0.440
KNN	0.375	0.448	0.423	0.352
RF	0.404	0.279	0.335	0.298
SVM	0.609	0.658	0.547	0.554
CNN	0.563	0.520	0.545	0.513

Grayscale: 5 Artists				
Classifier	Accuracy	Precision	Recall	F-1
LR	0.335	0.309	0.315	0.309
KNN	0.326	0.351	0.362	0.275
RF	0.375	0.256	0.292	0.260
SVM	0.515	0.557	0.455	0.458
CNN	0.548	0.496	0.529	0.500

Figure 1. Results across all classifiers and data sets, with color and grayscale, with 50 artists and with 5 artists. Accuracy, precision, recall, and F1 scores are reported.

Discussion/Conclusion

In conclusion, this assignment answered many of the questions posed to us after reading past related works on the topic of image classification of paintings using machine learning. For one, our results confirmed the theory suggested by one of the papers that using the original color of the images produced more accurate classification models. Second, reducing the number of artists labels from 50 to 5 dramatically increased accuracy of our models. In terms of the classification models themselves, SVM and CNN both had the best performances across all models, with SVM edging out CNN in terms of color classification.

However, despite these results, much more research needs to be conducted before these classification models can be used for real world applications, such as detecting if a painting is fake. A large part of the setback is the sample sizes of our dataset. Complex models such as CNN require a large amount of data, and we found that the relatively low number of paintings per artist was not enough to accurately classify on 50 artists. Furthermore, our classification models are biased towards artists with more paintings. Ultimately, the biggest takeaway from this assignment is feeding the classification models a quality data set. SVM and CNN are good models to use, but different methods of image processing should be explored in order to make the those two specific models as accurate as possible.

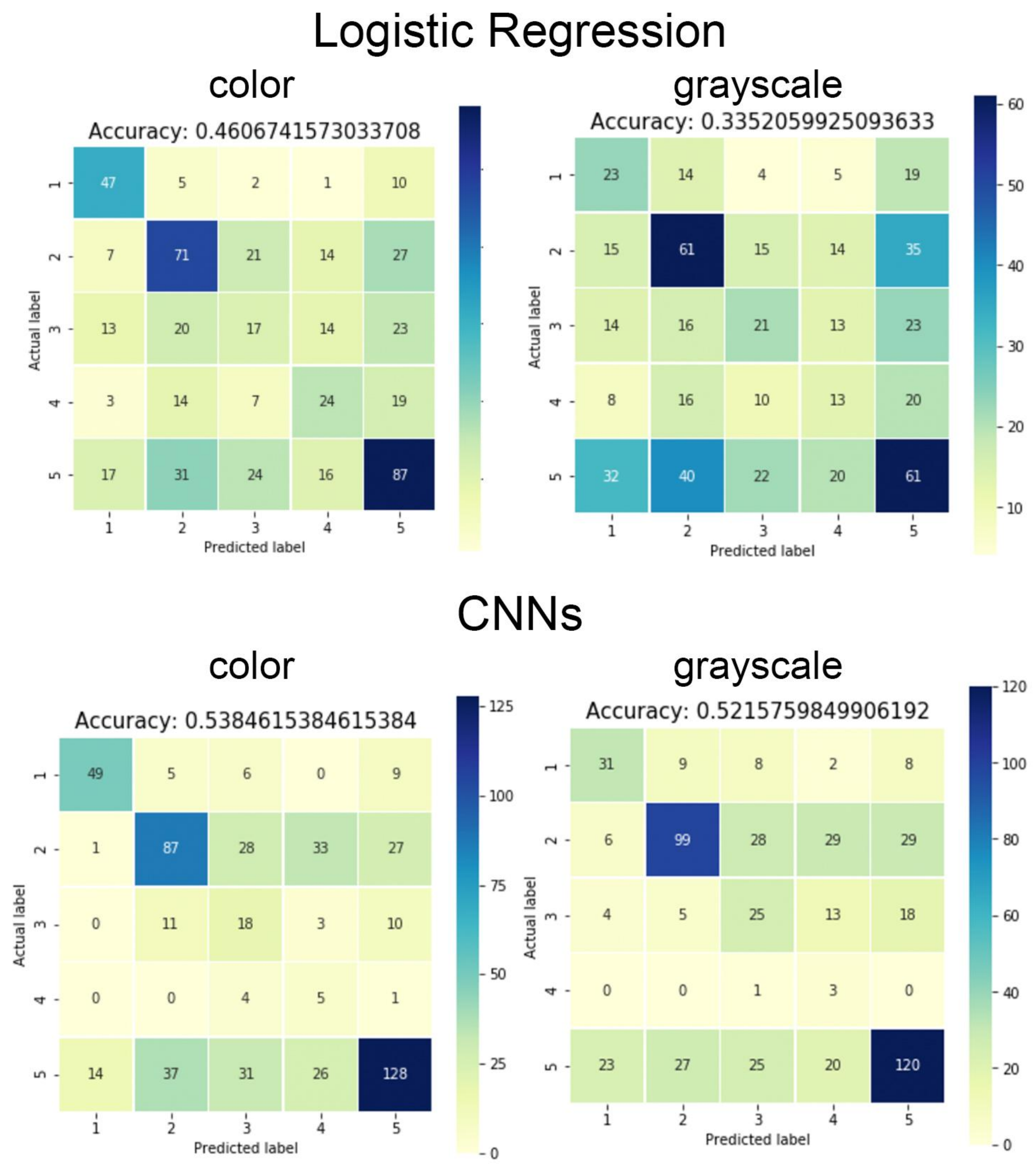

Figure 2. Confusion matrix results for our logistic regression and CNN models.

Contact

Victor Hua '21; Son Do '21; Jeremy Pulmano '21; Max Land '21
Department of Computer Science, Princeton University
vhua@princeton.edu; sqdo@princeton.edu; jpulmano@princeton.edu; mland@princeton.edu

References

[1] "Rohan & Lenny #2: Convolutional Neural Networks." Medium. https://ayearofai.com/rohan-lenny-2-convolutional-neural-networks-5f4cd480a60b

[2] C.R. Johnson, Jr., E. Hendriks, I.J. Berezhnoy, I.J., E. Brevdo, S.M. Hughes, I. Daubechies, J. Li, E. Postma, and J.Z. Wang. Image processing for artist identification. Signal Processing Magazine, IEEE, 25(4):37-48, 2008

[3] Lecoutre, Adrian Negrevergne, Benjamin & Yger, Florian. (2017). Recognizing Art Style Automatically with deep learning.

[4] Han, Dianyuan. (2013). Comparison of Commonly Used Image Interpolation Methods. Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering. 10.2991/iccsee.2013.391.

[5] Sci-Kit Machine Learning Libraries. https://scikit-learn.org/

[6] TensorFlow Machine Learning Libraries. https://www.tensorflow.org/

[7] Ben Fraj, Mohtadi. "In Depth: Parameter Tuning for SVC." Medium. https: /medium.com/all-things-ai/in-depth-parameter-tuning-for-svc-758215394769

[8] Le, James. Convolutional Neural Network Models." Medium. https://towardsdatascience.com/the-4-convolutional-neural-network-models-that-can-classify-your-fashion-images-9fe7f3e5399d

[9] "CNN: 1 Convolutional Layer." Github. Repository created by user "khanhnamle1994." https://github.com/khanhnamle1994/fashion-mnist/blob/master/CNN-1Conv.ipynb

[10] Yuan, Raymond. "Neural Style Transfer: Creating Art with Deep Learning using tf.keras and eager execution." Medium & TensorFlow.https://medium.com/tensorflow/neural-style-transfer-creating-art-with-deep-learning-using-tf-keras-and-eager-execution-7d541ac31398

[11] "Neural Style Transfer." Github. Repository created by user "titu1994." https://github.com/titu1994/Neural-Style-Transfer

[12] "Building Autoencoders in Keras." *The Keras Blog*. https://blog.keras.io/building-autoencoders-in-keras.html