Detection in Art Style **JDR03**

Odhrán McGranaghan 40157094

Table Of Contents

Gitlab Repository
https://gitlab.eeecs.qub.ac.uk/odhran-mcgranaghan/csc-3002-art-styles-detection.git

Introduction

Art is defined as “the expression or application of human creative skill and imagination, typically in a visual form such as painting or sculpture” (“Art,” 2019) [1]. Artworks are a major defining part of one's culture and in particular painted artworks have had a significant impact on how future generations perceive and understand the past. A significant problem with the understanding and appreciation of artwork is the identification of the artwork itself. I believe that this should begin with the artist, as knowledge of who produced an artwork allows further research to be done to better understand the context in which it was created and learn more about what the art depicts or represents. Determining who produced a piece of artwork is a barrier of entry to the world of art I hope to overcome with this project.

Problem Description

This project will produce a piece of software that is capable of determining the artist of an artwork by analysing the artistic style of the art itself using a trained Machine Learning, ML, model. The main applications for this model would be the ability to classify unlabelled artwork and add these to a labelled dataset. The software will have two main components, the ML model which will be trained using an existing labelled dataset of artworks and their artist, and an intuitive web application that will allows users to submit their own images for analysis, change parameters of the model and receive a best estimation as to who the artist of the submitted artwork is. This tool will be useful to students studying artworks for the first time, curators of art galleries or private collectors alike, anyone who has an interest in learning more about a piece of art.

This tool has innate potential for expansion beyond just artist identification, it would be feasible to expand the analysis to include the type of visual art that is being portrayed whether that be ‘Representational, Abstract or Non-Objective’ [2]. Additionally it would also be possible to estimate the year in which the artwork was produced and furthermore to determine what period it belonged to in Western art History, e.g. Medieval and Renaissance through to Contemporary. There are few existing services in this space, most notably the mobile application Magnus [3], which focuses on identifying artworks for valuation, and locating nearby museums or galleries where you can buy, sell or purchase the artworks. My project is primarily an educational and administrative tool that empowers users to take their learning into their own hands and expand their knowledge of a particular artwork, artist and time period.

Goals and Requirements

The primary goal of this project is to produce a web application that leverages a trained machine learning model to give the best determination of the artist of a painted artwork provided by the user. The web application will ideally be cloud hosted and have a minimal, intuitive UI that allows the user to upload an image of a painted artwork, and receive feedback on the artist who created it, in simple terms - a self-service artwork classifier.

The Essential Functional Features required to make up the proposed solution.

1. A Machine Learning (ML) application that is capable of training and storing models. These models must be trained with a large dataset of labelled artworks, that are capable of classifying the artist of an image that it has never seen before to a certain degree of accuracy. A Persistence layer should exist here to store trained models.

2. A Web Application (WA), implemented in Java using the Spring Boot Framework, that is the front end interface for the ML application, allowing the user to submit their image for classification and display the resulting feedback to the user. The WA should be able to access the ML Persistence layer to select trained models for analysis.

3. Ability to retrain the active ML model using different parameters. User's should have the ability to change the parameters of the SVM to achieve this.

The Desirable Features that would further enhance the proposed solution.

1. Trained ML model that can provide information beyond just the artist's name e.g. Period, Medium, Category of Art, Style.

2. Generation of the detailed report on the art that was submitted by the user. This may be a plaintext html file or rendered pdf that is then downloadable from the WA.

Test Cases

Below I have listed a sampling of test cases for each essential functional feature that will later be used to determine my success criteria. These are subject to change as the project progresses and will be completed in full covering all aspects of the completed system in my final dissertation. Desirable features do not have test cases as of yet, these will be defined if time and resources allow for the features implementation.

Machine Learning Model Application

1. The application must be able to take input of a labelled dataset used to train a model and a further test dataset to evaluate the model.
2. The application must be able to accept a dataset of images from the format jpg, .jpeg, .bmp,.png.
3. The application must be able to accept a set of labels for a dataset in the format of a .csv file.
4. The application must be able to submit the user submitted image from WA of the format jpg, .jpeg, .bmp,.png.
5. The application must be able to persist a trained model to the database, where it can be accessed by the WA.
6. The application must be able to take a trained model and user-submitted image and output a label for this image. I.e. the Artist name
7. The trained models must be retrainable based on parameters selected by the user, and these new models persisted to a database.
8. The trained models must be pre-trained and persisted in a database available for selection from the user within the GUI of WA.

Parameter Tuning

1. The user must be able to access a GUI enabling them to change the SVM model parameters to produce a different model within the GUI. e.g. SVM error, kernel type, degree etc.
2. The user must be able to select the feature or combination of features which are extracted from the training data to train the model e.g. Colour Histogram in the RGB, HSV space, increasing the number of bits.
3. The user must be able to model from a list of pre-trained models to classify their uploaded image.

Web Application

1. The user must be prompted to open their local file storage system to navigate and select their desired files for upload.
2. The user must be able to upload an image file of the format .jpg, .jpeg, .bmp, .png only.
3. The user must not be able to upload a file outside of these types .jpg, .jpeg, .bmp, .png.
4. The user must not be able to upload a file that has been archived or compressed such as .zip,.rar, .7z.
5. The user must not be able to upload a file size greater than 5mb.
6. The user must receive confirmation if their file is uploaded successfully.
7. The user must receive feedback if their file upload was unsuccessful and an error message.
8. The user must be taken to a results page upon completion of classification. This page will display their uploaded image and the name of the artist determined by the model.
9. The user must be able to navigate back to the homepage from any page in the application.
10. The user must be able to restart the user flow of uploading an image from the results page.

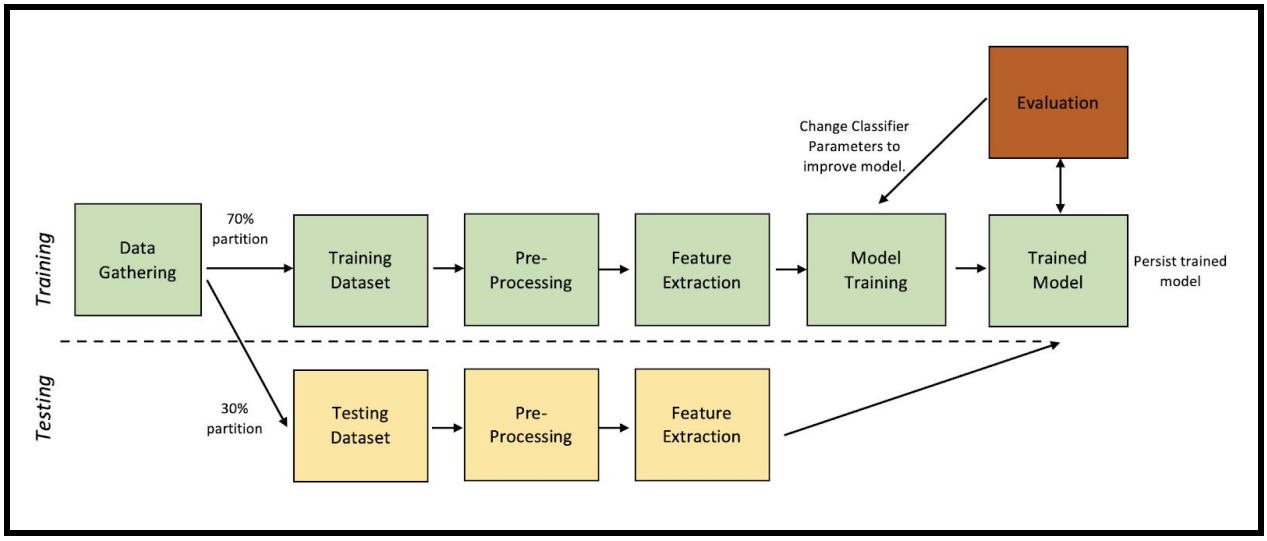
Project Technologies

The following is a brief overview of the technology stack that I will use to implement my proposed solution. These are my initial plans but are subject to change as the project progresses.

The system is composed of the following components:

1. The machine learning application that adds the intrinsic value of art classification.
2. The Web application that serves as an interface between the user and ML model.

Machine Learning Pipeline



The initial step of any Machine Learning Pipeline is Data Gathering of the dataset that will be used for training the model. I spent significant time trying to find a suitable dataset that contained a wide range of well-known artworks that were also already labelled with their artist, and if possible with further information such as artistic style, time period and country of origin. The dataset that I settled on was an open source set from the data science website Kaggle [4], this dataset contains high quality scans of many painted artworks from fifty artists ranging from Van Gogh to Pollock. A partition of the dataset is performed, taking a random 70% sampling of each artist to be used for training, and the remaining 30% to be used for testing and evaluating the model.

The next step is pre-processing of the dataset, for my initial model training I decided against any extensive pre-processing due to the feature extraction I was going to select later. This is subject to change as I seek to improve the model's accuracy and effectiveness.

Feature Extraction is a major part of this ML pipeline and is one step that will vary greatly over the course of this project. Upon reviewing several papers, in particular this paper published by researchers from Stanford University [5] found that by choosing a combination of features to

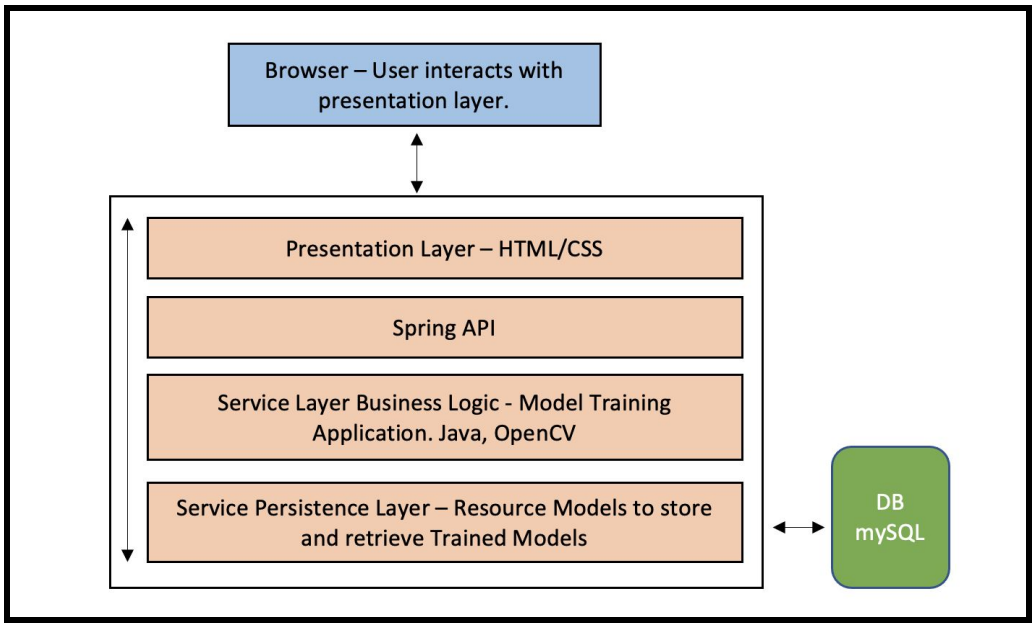
extract from their training data they found it boosted the overall accuracy of the model when run with the same testing dataset. My baseline model will extract an RGB Colour Histogram from each training image and compress this into a single row matrix vector.

Model Training will make use of an SVM learning algorithm to classify the dataset, the library in particular that I will be using is OpenCV for Java, which is an industry and research standard in the fields of computer vision. The SVM libraries of OpenCV [6]] are derived from libSVM [7] and are typically used for binary classification problems, as a workaround for this I will deploy a one vs rest strategy for each artist class. The SVM parameters will be tuned to find the best classification for the test dataset. The trained model will then be persisted to the DB.

Model Testing and Evaluation
The testing dataset will be processed identically to how the training dataset was and it will then be passed to the trained model for evaluation.
The machine learning model will be tested using statistical hypothesis errors Type 1 & 2, i.e. false positives and false negatives, an ROC curve will be constructed and this evaluation will inform future decisions on what feature extraction combinations are most effective and then how the classifier parameters can be tuned to produce the most optimal model.

Web Application Architecture

The Diagram below shows a high level overview of the Java web application that I will develop to serve and retrieve the trained models. It will be written in Java, using the Spring 2.0 framework. A simple API will be defined to allow the presentation layer to send requests to the ML Application and retrieve and send data to the persistence layer. The final architecture is to be determined as development is still ongoing.



Development Approach

The development strategy of this project will adopt an agile methodology, dividing up the Machine Learning (ML) model and web-app interface into manageable sections to then be developed over 2 week sprints as outlined in the Gantt Table below. The primary programming language of the User Interface (UI) software will be Java and this will be in the form of a local desktop, potentially deploying this as a cloud based web application given time constraints and the ability to deploy the ML model in a cloud environment.
The project will be programmed using a test-driven development (TDD) approach, pre-defining function behaviour in tests, and then implementing functions to satisfy and pass those tests. The JUnit testing framework will be used to write the tests as it is well-documented and an industry standard for Java applications.
Though an agile approach is taken, the initial Sprint is planned to be longer than the rest as it will take time to determine the requirements and design for the initial Minimal Viable Product (MVP).

The first milestone is to develop an MVP as soon as possible, this would include the following;

- A basic user interface where a user can provide images and receive back information on those images
- A machine learning model capable of taking images from user and determining with a who the original artist is;
- A method for testing the accuracy of the classification by the ML model. i.e. mathematical equation to model the accuracy

Gantt Chart

Gantt Chart can be viewed here:
https://docs.google.com/spreadsheets/d/1n7Gk4fjZmK_z7DdqCw1n_LelEwyZ1NCzu3F9O-JcFP0/edit?usp=sharing

I could not fit the Chart due to the page limit of this report. I will also email a copy of the chart to my supervisor separately.

References

1. Oxford Dictionary, *"Art"*. In Oxford Online Dictionary. Retrieved from https://en.oxforddictionaries.com/definition/art, 2019

2. M. Fussel, "Types of Visual Art", 2015. [Online]. Available: https://thevirtualinstructor.com/types-of-art.html. [Accessed October 8th, 2020]

3. Artface Inc, *Magnus.* [IOS, Android] Available: http://www.magnus.net/, 2020.

4. I. Caro, "Fifty Popular Artists and their works", 2020. [Online]. Available: https://www.kaggle.com/ikarus777/best-artworks-of-all-time. [Accessed October 5th, 2020]

5. N. Viswanathan, "Artist Identification with Convolutional Neural Networks", Stanford University, Stanford, CA, 2017

6. G. Bradski, "The OpenCV Library", Dr Dobbs Journal of Software Tools, 2000.

7. C. Chang and C. Lin, "LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology".Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2011.