Paintings recognition at the time of COVID-19

Andrea Maino
dept. of Finance
SFI PhD Program; University of Geneva and GFRI
Geneva, Switzerland
andrea.maino@etu.unige.ch

***Abstract*—This project applies machine learning and image recognition techniques to a unique dataset of paintings from the collections of the Rijksmuseum in Amsterdam. The objective of this project is to explore how image recognition techniques applied to artistic paintings are able to extract features useful for clustering algorithms. The features extracted relate to the colors, the texture and to the object represented in each painting. In this project, two clustering algorithms are used on the extracted features: KMeans clustering and Hierarchical clustering. The dataset is downloaded using the museum's API and contains selected paintings from the existing collection. The results show that the selected metrics are successful in extracting features able to allocate painting in clusters. The comparison of clustering with KMeans and Hierarchical clustering shows strong similarity between the two algorithms and supports the validity of the results obtained.**

***Index Terms*—Image processing, painting's features extraction, style, HOG, History, Paintings clustering, KMeans clustering algorithm, Hierarchical clustering algorithm**

I. INTRODUCTION

The recent events related to the spread of the COVID-19 has forced the global workforce to rethink its habits in terms of office work towards remote working and home office. This has been possible thanks to recent improvements in remote connectivity applications. However, the development of technologies allowing the transition towards remote experiences, different from remote working and home office, had started to develop before the COVID-19. For instance, in the context of cultural outreach, several museums around the world had started years ago to offer virtual visits to their collections allowing to develop new type of experiences.

Among others, the Rijksmuseum in Amsterdam (https://www.rijksmuseum.nl/en/from-home) offers several of these remote initiatives.

Of interest for this project is the possibility to fully access collection data (including some high-quality metadata) for developers, researchers and art enthusiasts. The data can be accessed via an API accessible once a (free) account is opened at https://data.rijksmuseum.nl/object-metadata/api/. The Rijksmuseum is the major museum in the Netherlands and has, among its collection, several Rembrandt and Vermeer, just to cite some of the most known Dutch painters.

The access to the metadata of the museum collection opens to interesting initiatives in several research fields. In this project the focus will be on applying computer vision techniques to analyse images of paintings and to extract features to use machine learning techniques for clustering paintings based on the extracted features. The applications can be multiple: clustering paintings can be used for recognizing common styles, attribution of newly discovered paintings, organizing collections based on ex ante criteria using a computer program or for teaching purposes, just to make some examples.

Compared to more traditional image recognition tasks, the analysis of paintings and art works in general, represent a different challenge. In fact, the colors, the texture and the style are prevalent compared to the content or subject represented on itself. In over simplified terms, the task of algorithms for visual recognition consist most of the time in recognizing an object and distinguishing it from other objects, for instance distinguishing a cat from a dog. In the context of art recognition however, the content is not the only important metric but other "unobservable" features such as style play a pivotal role. This opens to the obvious question of how to structure an algorithm of image recognition able to analyse art works and extract features with the goal to successfully applying machine learning techniques, for instance clustering algorithm.

In this project, I propose an implementation of image processing techniques to extract important features for paintings to address the above mentioned criticalities. The extracted features relate to color, texture and objects detection and are used to build a collection of features for each painting in the dataset. The extracted features are then used to implement two clustering algorithm: a KMeans clustering and Hierarchical clustering. Each algorithm brings interesting insights on how the clusterization works and on the validity of the extracted features to successfully allocate in clusters the paintings.

The choice of features is based on ex ante measures and are not selected by an algorithm. The color features are based on the encoding of each image in three layers of pixels, each representing the intensity of a primary color via a number between 0 and 255. The features are extracted by analysing the distribution of color intensities, by computing the first three moments, for each primary color and for each image. The texture features are based on the distribution of the intensity of grey, the matrix of co-occurrence of levels of grey and a set of measures proposed by Haralick (1973). Finally, the object features are extracted via the Histogram of Oriented Gradient (HOG), which is used to detect objects in images thanks to the distribution of the intensity of gradient in cells composing the image.

A. The metadata

The metadata for this project has been accessed via the API of the Rijksmuseum at http://rijksmuseum.github.io. In particular, I focus on the paintings of the collection without including sculptures or other art pieces (such as pieces of furniture for example). This is done in order to focus firstly on a homogeneous sample of images with the intention in future works to expand it to include sculptures and other objects. Furthermore, among the available images, the focus is on those which can be downloaded (from the API, *hasImage=TruepermitDownload=True*). The data is retrieved in JSON format which needs to be parsed to select the url of the image, the title, the main author (principalOrFirstMaker), a long title containing other useful information (for instance the attributed year) and the size of the paintings. All these information are stored in a csv file named $museum\_data$. A second extraction is done to retrieve only paintings which are currently exposed at the museum (ondisplay=True). This second extraction, can be used to merge the information regarding the current exposition with the $museum\_data$ information.
In summary, the last version of the projects collects a total of 3006 paintings produced between the 13th and 18th century as can be seen from the Fig.1 distribution of painting's production by centuries.

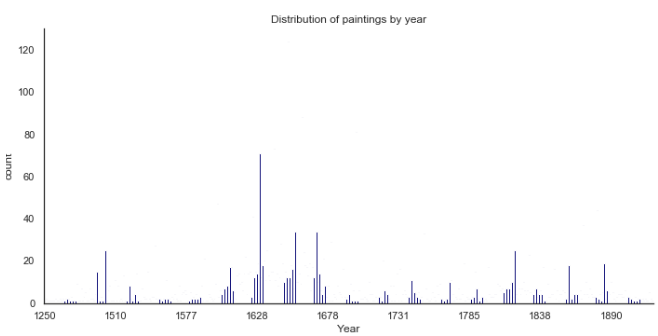

Fig. 1. Histogram of production of paintings per year in the dataset analysed.

The script "$download\_images.py$" retrieves the metadata for this project. First, the information regarding the paintings are downloaded via the API and stored on a csv file containing painting "id" and "url" which are then used to download directly the images using a procedure parallelized via `Multiprocessing`, a package that supports spawning processes using an API similar to the threading module.
Paintings are stored as ".png" for each image. For each painting in the metadata, it is possible to analyse several information, such as year of production, main author and if the painting is currently exposed. As illustration, Fig.2 represents some painting of the Dutch painter Vanmour.

II. Painting analysis:

In this section the focus is on the features considered for the analysis of a painting image: colors, texture and object

Fig. 2. Selected paintings from the dataset of the Dutch painter Vanmour.

detection. In the following sections I will discuss in more details each of these features and how they are constructed.

A. Color features: RGB and HSV format

The color is clearly one of the most important feature of a painting. Apart from being used to represent a visual aspect of an object, colors are integral part of the style of a painter and an unique signature in a painting. Often times indeed, colors are used to distinguish painters and artistic movements. In terms of how a computer represents a colored image, we need to refer to an image of size (x,y) as a collection of pixels recorded on three separate matrices: one matrix for each primary color. Indeed the "RGB" format stands for red, green, blue. Each pixel is associated to three numbers between 0 and 255, for each of the "RGB" matrices, which represent the intensity of the primary colors. Although RGB is the most common format for colored images visualization, it is not the only one. Another frequent alternative is the format HSV (Hue-Saturation-Value of Intensity) which represent an image in terms of its shade (saturation or amount of gray) and its brightness value. One way of analysing an image through its color format is by analysing the frequency histogram of the intensity in RGB or HSV format for each dimension available. This histogram represents a distribution which moments can be used to characterize the color features. In Figure 3, Figure 4 and Figure 5, I represent a given image in its intensity distributions in the "RGB" and "HSV" format. The comparison with a similar image in the Appendix section allows to show how different figures are characterized.

Fig. 3. An image selected from the dataset to which the distributions of intensity in the RGB and HSV format are extracted.

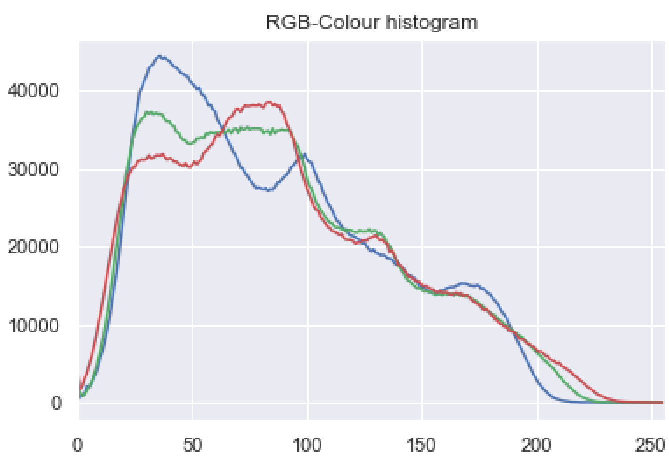

Fig. 4. Distribution of color intensity in the RGB format for the image in Figure 3.

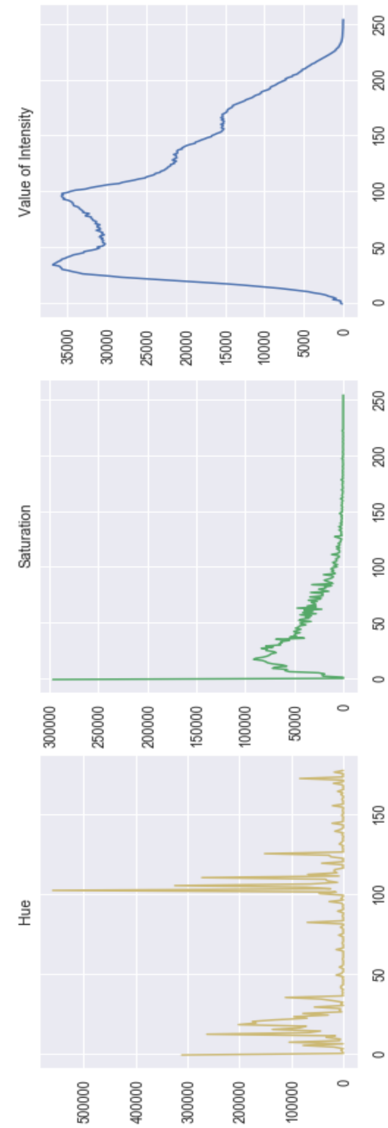

Fig. 5. Distribution of color intensity in the HSV format for the image in Figure 3.

B. Texture features: histogram of levels of grey

The texture analysis is given by the spatial arrangement of the intensity of grey, which is based on the distributions of intensity of grey of an image. The moments of this distribution can be used as features for texture. Furthermore, other measures can be used to analyse the spatial arrangement of intensities of grey. These measures are the matrix of co-occurrence of levels of grey which measures for each level of grey the number of times this has been close to another level of grey, and a set of measures defined by Haralick (1973): the second angular moment (ASM), the contrast, the correlation and the dissimilarity. The values obtained for these measures can be observed directly from the Jupyter notebook.

As an illustration, Figure 6 and Figure 7 represent two images from the dataset and the corresponding histogram of level of grey. Similarly to the color features, texture features are also extracted from the moment of this distributions.

C. Object features: HOG

The image analysis is completed with the object features given by the Histogram of Oriented Gradient (HOG). The HOG is a feature description used in computer vision for object detection in an image. In a first application, it was used for pedestrian detection in static images. The background idea is that local objects can be detected by the distribution of intensity gradients or edge directions. The image is divided in small connected regions called cells and for each pixel within each cell a histogram of gradient directions is compiled and the descriptor is given by the concatenation of these histograms. The use of HOG here is mainly done in order to characterize the image as a portrait (or more generally, containing an

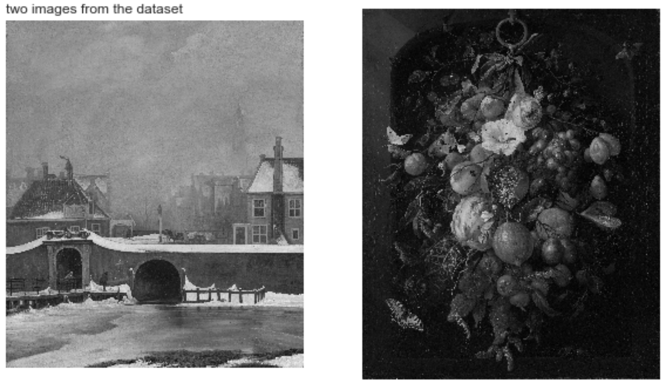

Fig. 6. Images from the dataset represented based on the intensity of level of grey

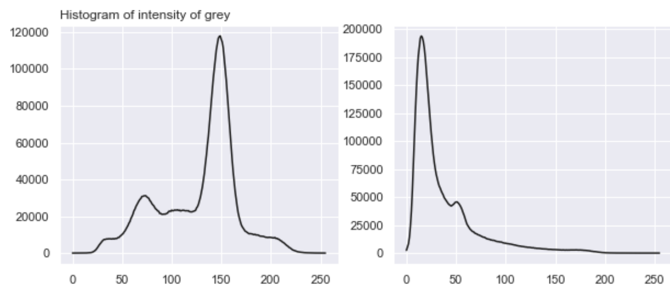

Fig. 7. Corresponding distributions of the intensity of grey based on the images in Figure 6.

object) or as a landscape. Figure 8 shows the HOG analysis to a image from the dataset.

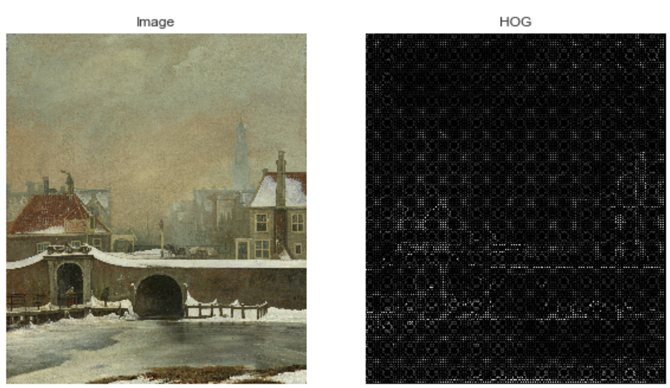

Fig. 8. HOG representation of an image in the dataset

The HOG is computed first on the full image and then on the image divided in half (horizontally) in order to distinguish portrait and landscapes. The HOG applied to the two half of the landscape images, as in Figure 9 and Figure 10, shows a clear separation between the upper (containing the sky representation) and the lower parts. This features help in distinguishing between landscapes and portrait.

Finally, in Figure 11, the HOG analysis is applied to a portrait and clearly shows how the body is detected with this

Fig. 9. HOG representation for the upper part of an image in the dataset.

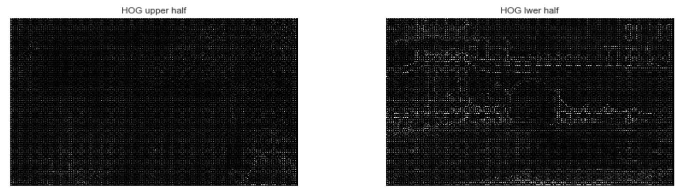

Fig. 10. HOG representation for the lower part of an image in the dataset.

technique which helps differentiating between landscapes and portraits.

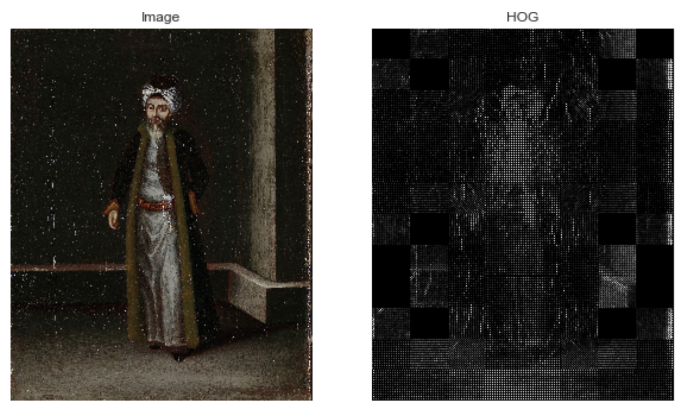

Fig. 11. HOG representation for a portrait in the dataset.

D. Feature Extraction

The above image analysis is the backbone structure of the features extracted from each image which are used to implement the clustering algorithms. In particular for each image the following features are extracted:

- *Color features*: three moments from the ”RGB” and ”HSV” distributions:mean, std dev, skewness;
- *Texture features*: extraction of features based on statistical approach on the matrix of co-occurrence of levels of grey. In particular, we extract the means (on each of the 4 main directions) of the first 4 statistics defined by Haralick (1973) and directly computed from the matrix of co-occurrence of levels of grey;
- *Object features*: the HOG features are computed for the full image and for the upper and lower part of the same image divided in two parts following the horizontal line. In particular, I extract an array of 3 dim corresponding to the mean of the HOG on the full image, the upper and the lower part;

[Insert Table I about here]

III. Clustering paintings

Once features have been extracted for the images in the dataset, these can be used to implement machine learning algorithms. The features are initially preprocessed in order to normalize each feature. Two clustering algorithms are implemented: a KMeans clustering and a Hierarchical clustering. Having two different clustering algorithm allows for comparing the clusterisation and validate the results. Furthermore, the Hierarchical clusterisation allows to illustrate how the clusters are related.

A. KMeans clustering

The first clustering algorithm is a KMeans clustering in which the output is the membership of each painting to one of the 14 clusters used. The number of clusters is determined looking at the painting allocation in several specifications and choosing the one which appears to reduce noise in each cluster.

The clustering results are visualized in several ways. In particular, for the 10 most frequent artists, the assignment of their works to the clusters is analyzed. Finally, the mean values of the features are computed in each cluster for the KMeans and for the Hierarchical clustering.

In Figure 12, Figure 13, some of the clusters obtained from the KMeans model are represented. The remaining can be found in the Jupyter notebook.

Fig. 12. Selected paintings from the cluster 0 produced by the KMeans algorithm.

B. Average image in each cluster

In order to visualize the way the clustering allocates similar images, it is helpful to compute the average picture in each cluster which highlights the average feature captured by the clustering model. In Figure 14 the average figure is showed and it is evident that images in each cluster have common features. Very explicit are for instance cluster 1, cluster 12 and cluster 14.

Fig. 13. Selected paintings from the cluster 1 produced by the KMeans algorithm.

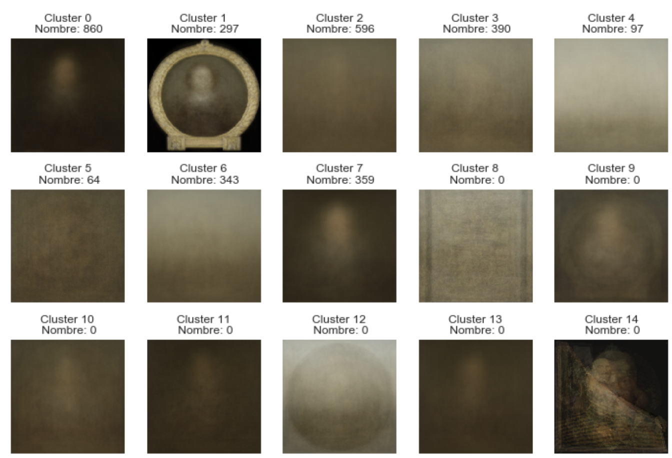

Fig. 14. Average figure representation for each cluster computed with KMeans

C. Allocations of painters per clusters and century of production

What is interesting to observe is that majority of main painters are also clustered in one or two clusters which goes back to the idea of style and subjects specific of an artist. Analysing the distribution of painters in each cluster we can clearly see from Table I, that painters are generally grouped in few clusters. This can be interpreted that the style typical of a painter is captured in few clusters.

[Insert Table I about here]

Similarly, the allocation of the century of production for each cluster brings to a similar result: most clusters capture similarities in styles which are concentrated across centuries. Figure 15 and Table 2 summarize the results.

[Insert Table 2 about here]

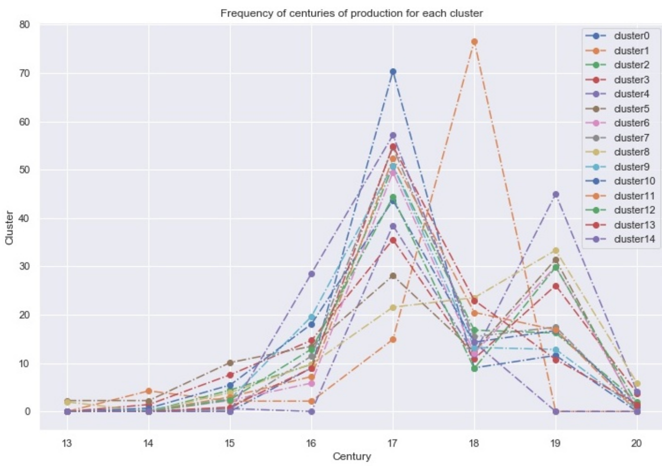

Fig. 15. Distribution of production per century for each cluster

D. *Hierarchical Clustering*

The second clustering algorithm considered is a divisive Hierarchical clustering, which implements a "top-down" approach where each observation starts from the same cluster and splits occurs as one moves down the hierarchy. The algorithm is implemented using 8 clusters and shows how each resulting cluster is related to the top head cluster.
The results are presented in a Dendogram in Figure 16.

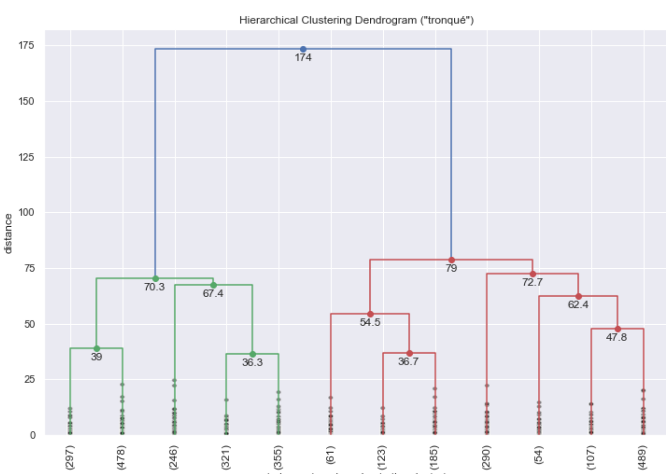

Fig. 16. Dendogram representation of the divisive Hierarchical clustering algorithm applied to the features extracted from the painting dataset. The algorithm uses 8 clusters.

E. *Clustering comparison*

It is interesting to compare the individual clusters obtained with the two clustering algorithm. In both cases, similar features should characterize similar clusters. It is possible to visualize them by comparing the allocation of features per each cluster. In this illustration, the comparison is performed for the Hierarchical clustering algorithm as illustrated in the previous section and with a KMeans algorithm with 8 clusters in order to ease comparability.

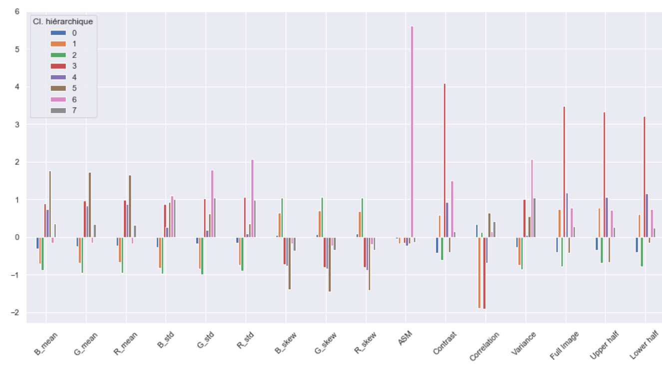

Fig. 17. Allocation of features per cluster using a KMeans clustering with 8 clusters.

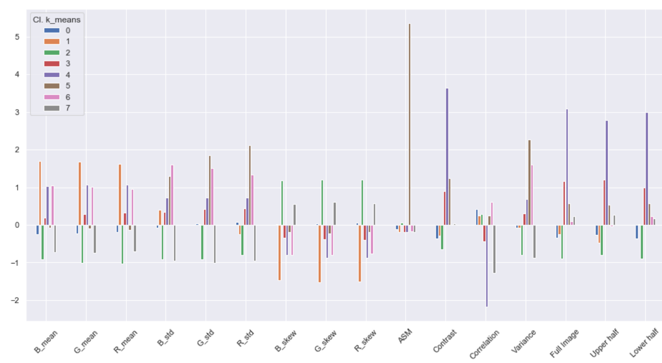

Fig. 18. Allocation of features per cluster using a Hierarchical clustering with 8 clusters.

IV. Conclusions

This project has focused in applying image recognition techniques to painting images to extract features related to colors, textures and object detection. These extracted features are then used in two clustering algorithm: KMeans clustering and Hierarchical clustering. The results show that the clustering algorithms are both effective in allocating similar paintings, for instance portraits with portraits or landscapes with landscapes, to the same cluster. Furthermore, both clustering algorithms deliver similar results in terms of features allocation in each cluster. Clearly certain aspects remain difficult to analyse via a digital image. For instance the type of paintings (oil,..), the depth in terms of painted surfaces, the frame etc.. The selected features in this project have tried to take all this complexity into consideration by selecting features related to the textures, colors and to the presence of objects. Possible extensions in future versions could be related to the selection of relevant features using neural networks, but which might loose the nice property of being interpreted, similarly to the one selected in this version of the project by selecting them directly.

V. Figures and Tables

A. Figures

Fig. 19. An image selected from the dataset to which the distributions of intensity in the RGB and HSV format are extracted.

Fig. 20. Distribution of color intensity in the RGB format for the image in Figure 19.

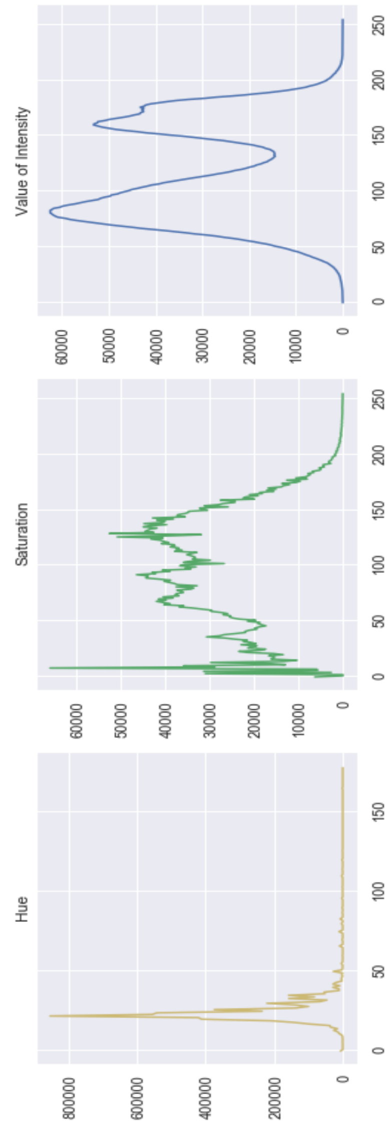

Fig. 21. Distribution of color intensity in the HSV format for the image in Figure 19.

B. Tables

TABLE I
ADD CAPTION

Features	B_mean	G_mean	R_mean	B_std	G_std	R_std	B_skew	G_skew	R_skew	ASM	Contrast	Correlation	Variance	Full Image	Upper half	Lower half
count	3006	3006	3006	3006	3006	3006	3006	3006	3006	3006	3006	3006	3006	3006	3006	3006
mean	0.58	0.77	0.86	0.35	0.41	0.44	0.01	0.01	0.01	0.00	1.55	0.01	18.69	0.01	0.01	0.01
std	0.32	0.35	0.35	0.15	0.13	0.13	0.01	0.01	0.01	0.00	1.48	0.00	11.94	0.00	0.00	0.00
min	0.04	0.08	0.10	0.04	0.08	0.13	-0.03	-0.03	-0.03	0.00	0.06	0.00	0.72	0.00	0.00	0.00
25%	0.34	0.51	0.59	0.24	0.32	0.35	0.00	0.00	0.00	0.00	0.63	0.01	10.17	0.00	0.00	0.01
50%	0.49	0.70	0.81	0.33	0.40	0.43	0.01	0.01	0.01	0.00	1.12	0.01	16.07	0.01	0.01	0.01
75%	0.75	0.99	1.08	0.45	0.50	0.52	0.02	0.02	0.01	0.00	1.90	0.01	24.33	0.01	0.01	0.01
max	1.88	2.08	2.14	1.05	1.00	0.98	0.13	0.06	0.05	0.00	15.93	0.01	98.93	0.03	0.02	0.03

TABLE II
ADD CAPTION

Custer Id	Jean Baptiste Vanmour	Arnoud van Halen	George Hendrik Breitner	Pieter van der Werff	Michiel Jansz van Mierevelt	Jacob Maris	Jan Maurits Quinkhard	Rembrandt van Rijn	Caspar Netscher
0	2.4%	0.0%	0.0%	56.7%	7.1%	0.0%	3.7%	23.1%	28.6%
1	0.0%	65.8%	0.0%	0.0%	0.0%	0.0%	40.7%	0.0%	0.0%
2	9.8%	5.3%	5.6%	0.0%	0.0%	17.9%	0.0%	7.7%	0.0%
3	7.3%	0.0%	25.0%	0.0%	3.6%	14.3%	3.7%	0.0%	4.8%
4	0.0%	0.0%	13.9%	0.0%	0.0%	25.0%	0.0%	0.0%	0.0%
5	0.0%	0.0%	8.3%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
6	4.9%	0.0%	11.1%	0.0%	0.0%	10.7%	3.7%	0.0%	0.0%
7	2.4%	0.0%	2.8%	20.0%	46.4%	3.6%	14.8%	15.4%	9.5%
8	0.0%	0.0%	5.6%	0.0%	0.0%	3.6%	0.0%	0.0%	0.0%
9	0.0%	0.0%	0.0%	6.7%	10.7%	0.0%	14.8%	7.7%	4.8%
10	29.3%	2.6%	8.3%	6.7%	14.3%	7.1%	14.8%	15.4%	14.3%
11	34.1%	5.3%	11.1%	0.0%	7.1%	7.1%	0.0%	11.5%	14.3%
12	0.0%	13.2%	2.8%	3.3%	0.0%	0.0%	3.7%	0.0%	0.0%
13	9.8%	7.9%	5.6%	6.7%	10.7%	10.7%	0.0%	19.2%	23.8%
14	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%

TABLE III

ADD CAPTION

Century	cluster0	cluster1	cluster2	cluster3	cluster4	cluster5	cluster6	cluster7	cluster8	cluster9	cluster10	cluster11	cluster12	cluster13	cluster14
13	0.0%	0.0%	0.0%	0.0%	0.0%	2.2%	0.0%	0.0%	2.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
14	0.0%	4.3%	0.0%	1.4%	0.0%	2.2%	0.0%	0.0%	0.0%	0.4%	0.7%	0.0%	0.0%	0.0%	0.0%
15	0.0%	2.1%	4.4%	7.6%	0.6%	10.1%	2.3%	0.5%	3.9%	2.6%	5.5%	3.0%	2.4%	0.9%	0.0%
16	9.0%	2.1%	9.7%	14.7%	0.0%	13.5%	5.9%	11.5%	9.8%	19.7%	18.1%	7.2%	12.9%	9.0%	28.6%
17	70.4%	14.9%	50.7%	35.5%	38.5%	28.1%	49.5%	54.6%	21.6%	50.9%	43.7%	52.4%	44.4%	54.9%	57.1%
18	9.0%	76.6%	16.8%	10.9%	11.8%	12.4%	12.1%	15.5%	23.5%	13.2%	14.3%	20.5%	8.9%	23.0%	14.3%
19	11.6%	0.0%	16.2%	26.1%	45.0%	31.5%	30.0%	17.5%	33.3%	12.8%	16.7%	16.9%	29.8%	10.7%	0.0%
20	0.0%	0.0%	2.1%	3.8%	4.1%	0.0%	0.3%	0.5%	5.9%	0.4%	1.0%	0.0%	1.6%	1.5%	0.0%