A pipeline for painting recognition and people localization in art galleries

NICOLA SALVATORE FRANCICA, GIOVANNI BONISOLI, GIOVANNI PANELLA

Dipartimento di Ingegneria dell'Informazione - Università di Modena e Reggio Emilia
{268351,204058,274832}@studenti.unimore.it

June 9, 2020

Abstract

This paper proposes a pipeline for paintings detection, rectification, recognition and people localization in a museum. The pipeline is divided in two main phases: the first one includes the use of a convolutional neural network which performs a semantic segmentation of a frame, the second one involves well know image processing techniques to extract paintings and people from the background according to the results given by the network and to rectify paintings in order to identify them.

1 Introduction

The challenge proposed concerns the fields of object detection, recognition and matching. Starting from a dataset of videos taken by different cameras from the interior of Gallerie Estensi of Modena, our goal was to correctly detect the paintings in a videos and match identify them using a database of images of the paintings exposed in the museum. More in detail, our goals were:

- *Painting detection*
- *Painting rectification*: correct the perspective distortion of each painting;
- *Painting recognition*: match each detected painting to the paintings of the DB;
- *People detection*
- *People localization*: assign each person detected to one room on the map.

To perform these tasks we have been provided:

- A set of videos taken inside the museum;
- A database of images containing most of the paintings exposed;
- A file containing the details of the paintings exposed and the reference to the corresponding image in the database;

To solve the tasks, multiple issues must be taken into account: a frame can contain from one up to twenty targets (people and paintings), objects and people shadows could interfere with the detection and the different cameras have different lenses and therefore different distortions (ex. GoPro videos have a pronounced barrel distortion).

To carry out the tasks listed above, we decided to combine a deep neural network for segmentation with image processing techniques to separate paintings and people from the background and with geometry techniques to rectify paintings.

2 Paintings retrieval and recognition

In this section we describe the pipeline we used to extract the paintings from a single video

Figure 1: Video frame example

frame and match them with the paintings in our database.

2.1 Semantic segmentation

In the first phase we want to detect the paintings visible in the frame. For this purpose, we decided to use *semantic segmentation*.

Semantic segmentation is the task of clustering parts of an image together which belong to the same object class. It is a form of pixel-level prediction because each pixel in an image is classified according to a category. This task is important for painting and people detection.

2.1.1 ADE20K Dataset

We discovered that CSAIL released a dataset called ADE20K[1], which covers a wide range of scenes and object categories with dense and detailed annotations for semantic segmentation[2]. Among the object classes it covers we can find both people and paintings.

2.1.2 The pretrained models

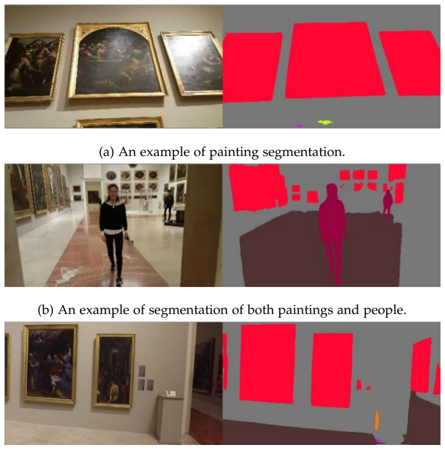
Along with the dataset, CSAIL published a variety of networks trained on the ADE20K dataset[2]. Each network has different accuracy and different computational requirements, covering a wide variety of purposes.

Each network is divided into two parts:

- *Encoder*: a convolutional network used as a feature extractor that transforms the input image into a multidimensional feature representation;
- *Decoder*: a deconvolutional netowrks that takes in input the generated multidimensional feature representation and generates a segmentation mask for the input image;

2.1.3 Example results

As aforementioned, all the pretrained models are able to recognize a large variety of stuffs and objects, including paintings and people.



(a) An example of painting segmentation.

(b) An example of segmentation of both paintings and people.

(c) Another example of paintings segmentation in more difficult conditions.

Figure 2: Some examples of segmentation using the pretrained Hrnet2 on the Ade20K dataset.

As we can see in the figure 2, the red color is associated to pixels predicted as belonging to paintings and the purple red label to the pixels predicted as belonging to the people[3].

2.1.4 Our choice

After different tests, because the frame analysis hasn't the constraint to be done in real-time, we decided to use HRNET2, which gives the best pixel accuracy results while being the most computationally expensive.

The usage of this network brings some advantages especially in situations that could be

[1]ADE20K Dataset

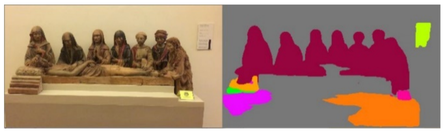
[2]CSAIL Vision pretrained models

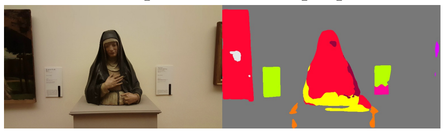
[3]Full colors/classes encoding file

quite problematic if tackled with a naive image processing approach. For example, in the case in which there is a strong light reflection on a painting, without using a deep network would be very difficult to identify that region as a part of a painting. Using a semantic segmentation network the illuminated part in instead often categorized as a part of the painting (2a). The very same reasoning can be applied for low luminance situations, in which segmenting paintings from the background could be very challenging.

We also tried to use other approaches for the paintings extraction task, such as the use of a mean shift algorithm to isolate the paintings from the wall. The algorithm is very computationally expensive to be used on each frame and the results are worse than the ones achieved with a convolutional network, especially in images with a wide perspective and with a large number of different paintings, people or different objects.

On the other hand, the network occasionally makes some mistakes when it receives a frame with busts or sculptures with a human subject. They are often wrongly classified as people (3a) or as a part of a painting (3b). Other similar errors occur when paintings with human subjects are very close to the camera: they are often wrongly classified as people (3c).


(a) Sculpture labeled as people.


(b) Sculpture labeled as painting.

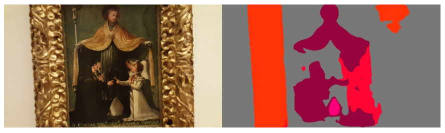
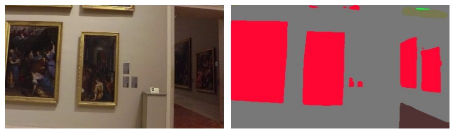
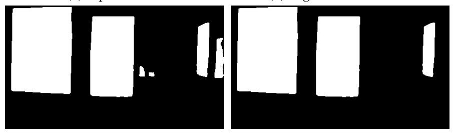

(c) Painting labeled as person.

Figure 3: Some segmentation error examples.

2.2 Paintings extraction

After passing the frame through the segmentation network, we need to extract the detected paintings from the original image.


(a) Input frame. (b) Segmentation result.


(c) Connected components. (d) Small objects removed.

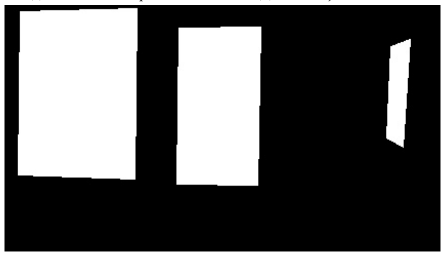

(e) Rectified approximated contours.

Figure 4: An example of the results of the various pipeline passages for the paintings extraction and rectification.

2.2.1 Binary image

Starting from the segmented image, we wrote a function (*extract_objects*) that given the name of the object we want to extract, it matches the corresponding color used by the network to encode that particular class and constructs a binary image containing just the interesting regions of the image. An example output of this stage is shown in figure 4c.

2.2.2 Small objects removal

Once we have obtained the set of paintings candidates, we labeled all the connected components, in order to be able to distinguish between different paintings in the image. We used the *connectedComponentsWithStats* function from OpenCV with an 8-way connectivity. If we look at figure 4c, you can see that sometimes the signboards are also labeled as paintings by the segmentation network. We therefore decided to remove all the components whose size is lower than a minimum size of our choice. After a few trials, we empirically found 2000 was the parameter that gave us the best results.

As we can see in figure 4d, the wrongly segmented signboards are correctly removed from the image. In other cases, if paintings are too small or too distant from the camera they are remove too. In any case, they would probably be too small to be recognized.

2.2.3 Contours rectification

At this point ideally we have a binary image with just the wanted objects. We need to remove all the possible shape imperfections from the contours. Therefore, we calculate the convex hull for each different component. In mathematics, the convex hull of a given set X of points is the smallest convex region containing X. By using the OpenCV's function *convexHull*, we obtain the contours of the regularized shapes.

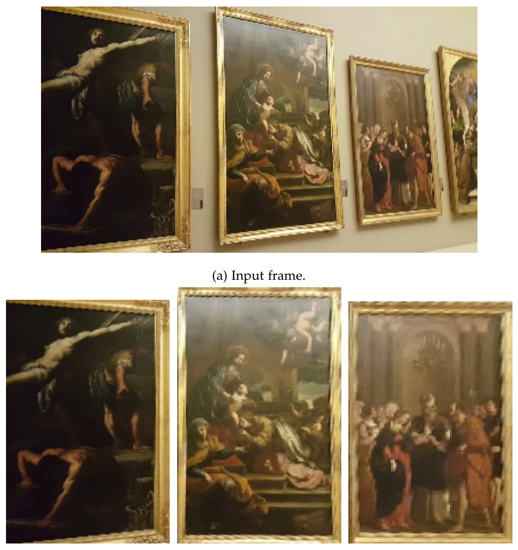
In order to extract paintings and do the rectification, we need all the region considered to be quadrangles.Therefore, we proceed analyzing the remaining contours and approximating their perimeter to a polygon with fewer vertices using the OpenCV's *approxPolyDP* [1]. The function takes as a parameter an epsilon, representing the maximum distance between the original contour and the approximated polygon. It gives in output the coordinates of the vertices of all approximated polygons. We decided to fix the epsilon distance to 0.03 so that most of the real painting shapes are approximated to quadrangles and the other objects (statues, sculptures, . . .) can be excluded from rectification.

As shown in figure 4e, at this stage for each detected painting with a regular shape we have the four points needed for the perspective rectification.

2.3 Rectification

To rectify the detected paintings used the just found vertexes of the rectified shapes as input of the algorithm suggested by *Zhang et al.* in his publication *Whiteboard Scanning and Image Enhancement*[3].

The algorithm first estimates the actual aspect ratio of the painting from the detected quadrangle based on the fact that it is the projection of a rectangle in space. Besides the aspect ratio, using the pinhole camera model it also estimates the focal length of the camera. From the estimated aspect ratio and by choosing the "largest" painting pixel as the standard pixel in the final image, we can compute the desired resolution of the final image. A planar perspective mapping (a 3×3 homography matrix) is then computed from the original image quadrangle to the final image rectangle, and the painting image is rectified accordingly.


(a) Input frame.

(b) First painting. (c) Second painting. (d) Third painting.

Figure 5: A frame extracted from a video and the corresponding paintings rectifications.

2.3.1 Results

After some tests, we achieved the following result: precision of 97% but a recall of 68%. Most of the false negative correspond to paintings that are detected by the network but discarded in the phase of small components' removal because they are too distant. In other cases, they are paintings seen so much in perspective that even the network is not able to recognize them as paintings.

2.4 Recognition

Starting from the rectified painting image, we decided to perform the paintings recognition task using *ORB*, a good and efficient not patented alternative to *SIFT*.

At the program launch, we compute and cache the ORB keypoints on the images of the paintings in the given database.

For each detected and rectified painting, we compute the keypoints and then perform a brute force matching with all the cached keypoints. ORB matching algorithm, though, does not implement any type of filter on the matches found, in contrast to the algorithm used by SIFT which discards weak matches.

As per the Lowe's SIFT paper [4], we implemented the distance ratio filter for the ORB keypoints matches. The filter, using pseudocode, can be described as follows:

```
matches = sort(matches) # Sort matches using distances
good_matches = []
for i, match in enumerate(matches):
  if match.distance < threshold_ratio * matches[i+1].
        distance:
    good_matches.append(match)
```

After computing the good matches for every painting in the database, we select the one with the highest number of elements. If this painting has at least 8 good matches and at least 5 good matches more than the second classified, it is considered a successfull match and it's associated with the detected painting in the analyzed frame.

Using this strategy we have been able to successfully recognize the recorded paintings in almost every case in which the luminance conditions were not particularly problematic.

In some cases, even using the filter described above, the right matching painting in the database does not give a strong result using the keypoints matching techique and therefore is not recognized in every frame of the video.

2.4.1 Results

We evaluate the performances of painting recognition assuming that: true positives are all the detected paintings that are recognized correctly; false positives are all the detected paintings which are confused with another painting in the database; true negatives are all the detected paintings that are not present in the database; false negatives are all the detected paintings that are in the dataset but are not recognized.

After some tests, we got a precision of 100% but a recall of 59%. The maximum precision is due to the fact that we don't have any false positive (no paintings are confused with other ones). Instead, we have a large number of false negatives. Most of them are both paintings that are only partially framed and paintings that are viewed so much in perspective that even the rectification is not so effective.

3 People detection and localization

3.1 Detection

After the paintings concerning tasks, we focused on the tasks concerning people.

Because the first stage of the painting's pipeline was build in order to be easily re utilized to extract every kind of object recognized by the chosen network, this task has been accomplished just calling again the function that extracted the paintings from the frame. We just passed a different object label to retrieve (*person*) and a different threshold to eliminate small objects.

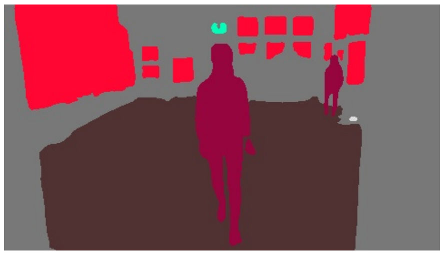
The latter parameter took us the most attention, because the network tends to detect as people also sculptures and painted human figures for paintings very close to the recording camera. Using an high threshold, even actual peoples often are discarded, increasing the number of false negatives. After some attempts, we opted for the threshold of 400, that allows us to detect most of the people avoiding without considering the smallest regions of bust and sculptures.

However, this does not allow us to exclude neither the busts and the sculptures when they are entirely considered as people nor painted human subjects incorrectly labeled by the network, as shown by figures 7c and 7d.

On our tests, we get indeed a good recall estimated around 80%, but a low precision of 33%.

The figure 6 shows an example of the results.

(a) Input frame.


(b) Segmented frame.

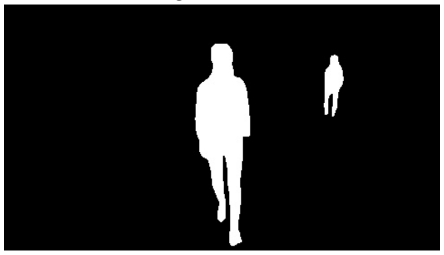
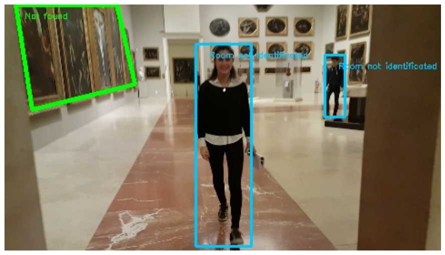

(c) Detected people.

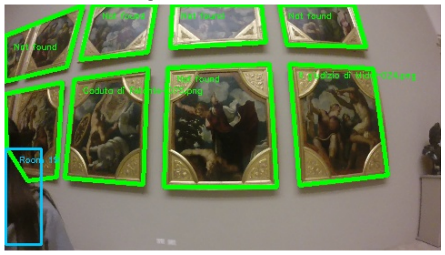
Figure 6: An example of the result of people extraction from a frame.

3.2 Localization

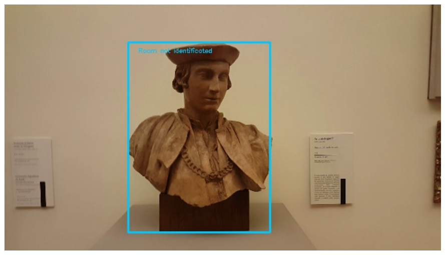
This task is basically performed using the results of the painting recognition.



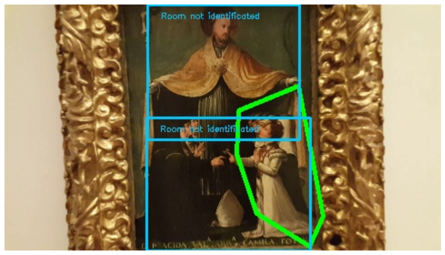
(a) Paintings and people detection, without successful recognition.



(b) Paintings and people detection, with successful recognition.



(c) Example of peolple detection error.



(d) Example of painting and people detection error.

Figure 7: Some sample final results of our work.

In the file containing the details of the paintings exposed in the museum, for each painting we also have the room number of the map in which the painting is located. It is possible to get the room only when one of the detected paintings in the room has been recognized^{7b}.

In the final output, all the detect people are highlighted with a light blue bounding box with a label indicating the number of the room in which they are. If no painting has been recognized, the label will say "Room not identified"^{7a}.

References

[1] D. Douglas, T. Peucker. *Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature*. The International Journal for Geographic Information and Geovisualization, 1973.

[2] Zhou, Bolei and Zhao, Hang and Puig, Xavier and Xiao, Tete and Fidler, Sanja and Barriuso, Adela and Torralba, Antonio. *Semantic understanding of scenes through the ade20k dataset*. International Journal on Computer Vision, 2018.

[3] Zhang, Zhengyou and He, Li-wei. *Whiteboard Scanning and Image Enhancement*. Digital Signal Processing, 414:432, vol. 17, 2007.

[4] David G. Lowe. *Distinctive Image Featuresfrom Scale-Invariant Keypoints*. International Journal of Computer Vision, 2004.