THE ANDREW & ERNA VITERBI FACULTY OF ELECTRICAL ENGINEERING

VISL Vision and Image Sciences Laboratory

TECHNION Israel Institute of Technology

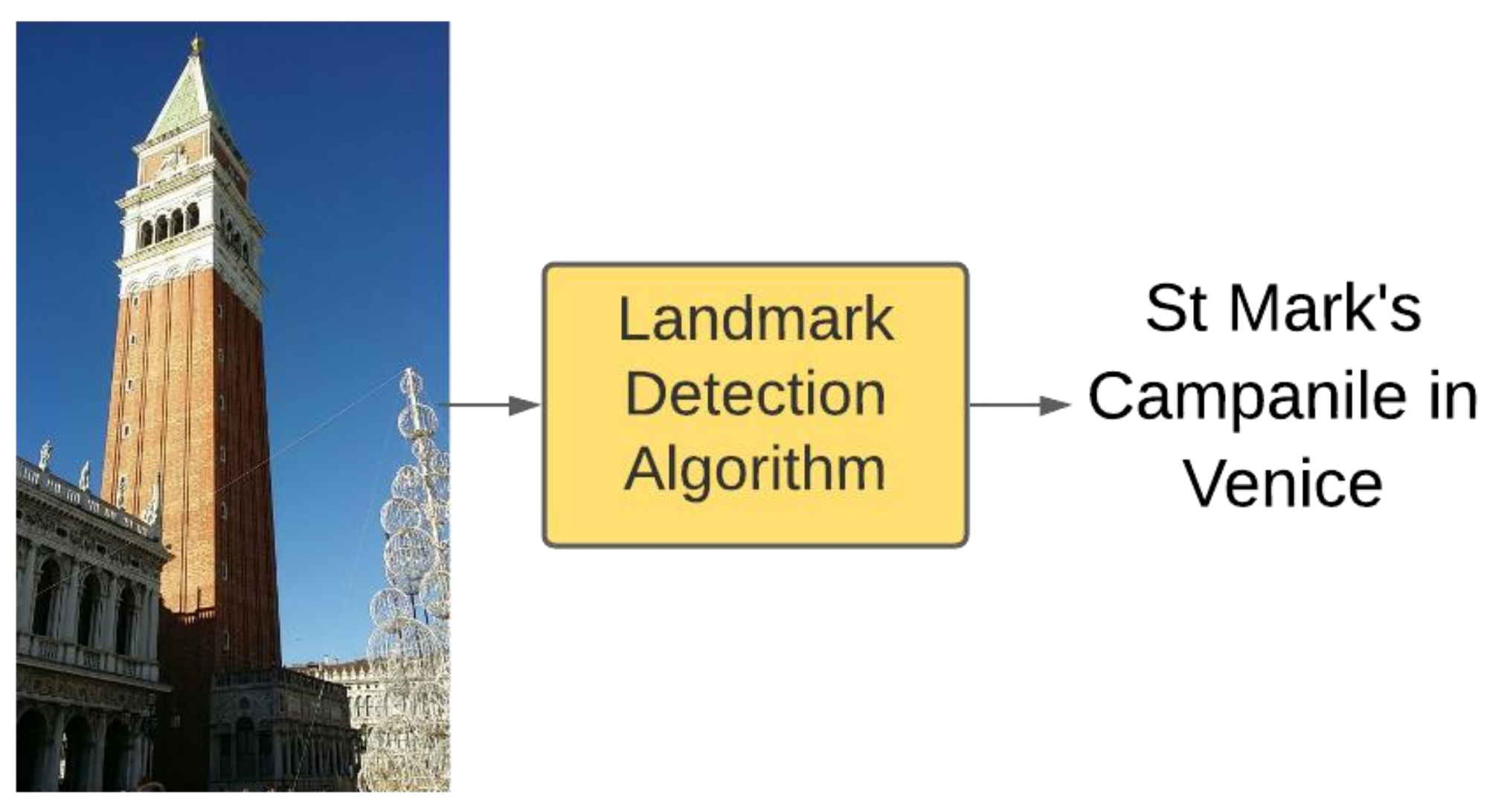
Identify Known Sites in Photo Album

Matan Kleiner and Yuval Snir, Supervised by Ori Linial

Introduction

This project is based on the "kaggle" competition "Google Landmark Recognition 2020".

The project goal is to recognize the correct landmark (if any) in a dataset of challenging test images.



The project's goal

Data Set

In the project we're using a clean version of Google Landmark Dataset v2 (GLDv2). GLDv2 is the largest landmarks (natural and man made, from all over the world) dataset to date. The clean version contains over 1.5M images that divide into over 81K classes of landmarks.

The dataset has several challenging properties inspired by real world applications:

- **Big amount of data**
- **Long-tailed class distribution**

  There are more images of famous landmarks than non famous therefore some classes have more than 1000 images, some only 2.

- **Large intra-class variability**

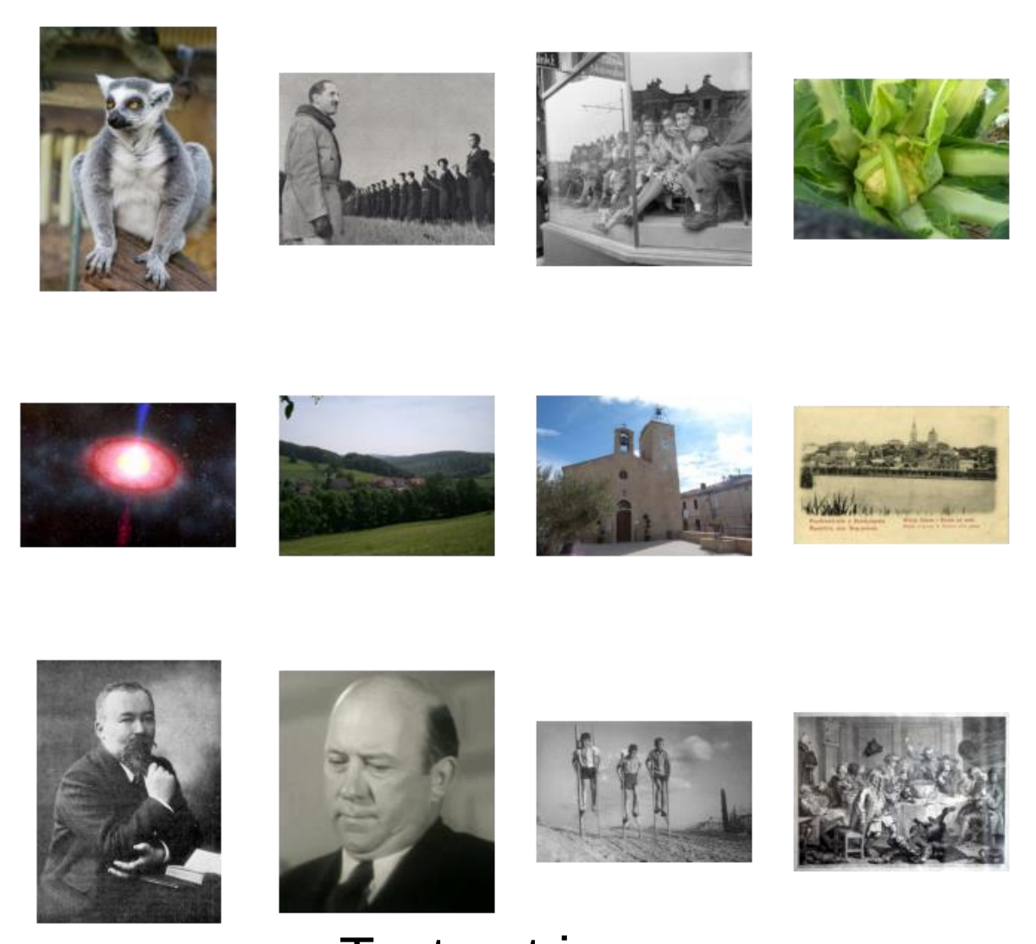
  Due to the fact that images from the same class were taken by different people, in different times with different devices and due to the fact that an image of the same landmark can be taken from different places (i.e., the front and the side of London Bridge) images from the same class may not look at all like each other.

  In addition, because of this large intra class variability test set images and train set images may also not look like each other.

The 4 images to the left (train set) are from the same class as the left image (test set)

- **The test set contain mainly out of domain images**

  In order to resemble the real world, the test set images are mainly out of domain images (only 1.5% are landmarks)

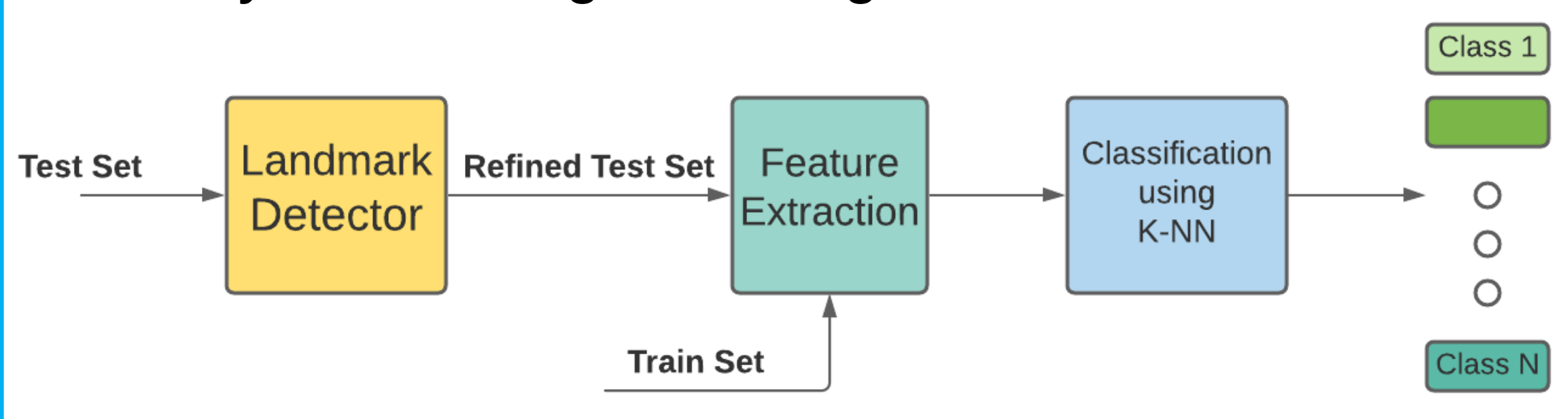


Test set images

Our Solution

Our solution is a retrieval bases solution which takes on the main challenges of the data set.

The solution made of two parts. The first one is to clean the test set (as we mentioned before, it contain mainly out of domain images). We'll do it using a landmark detector. The second part is to extract the features of the data set images and classify them using K-NN algorithm.



Our solution pipeline

Landmark Detector

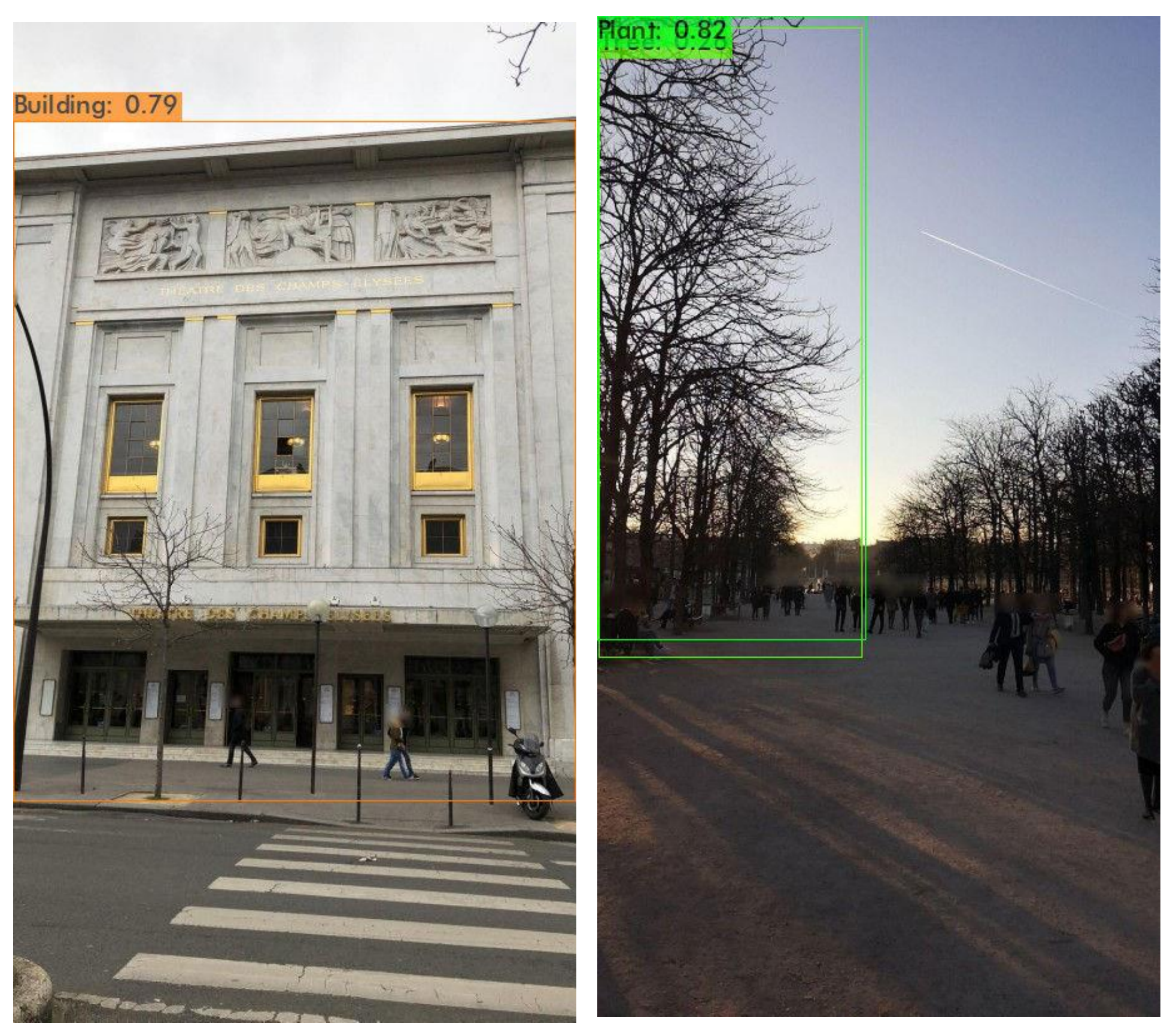
In order to clean the test set we'll use YOLO (You Only Look Once) pre trained object detector. Using this object detection we'll detect objects in all the test set images.

This object detector was pre trained on Open Images Dataset, which is a big and diverse dataset that has 600 different classes of objects.

We divided this 600 classes to 3 groups:

- **Landmarks:** 'Tower', 'Fountain', 'Skyscraper', 'Building', 'Castle'
- **Maybe Landmarks: '**Sculpture', 'Tree' …
- **Non Landmarks:** everything else

If the network detect an object in the image it'll classify it as one of the 600 classes, set a confidence score (between 0 to 1) and draw a box around the object.



Examples of images after object detection

Based on the output of the object detector we'll tag each image as "Landmark" or "Non-Landmark".

If an object in the image classified as one of the classes we defined as "Landmark" we'll tag it as a landmark. If the objects in that image classified as "Maybe/Non Landmark" we'll check its confidence score and the relative size of the box are bigger than some threshold (we set it using ROC curve). If they're bigger than this threshold we'll tag the image as "Non-Landmark".

In addition to the object detector that was pre trained on the Open Images Dataset we used another one, that was pre trained on COCO dataset. It is a much smaller dataset with only 80 object classes, none of which we can identify as "Landmark" or "Non Landmark". Therefore we only use its output to discard even more images from the test set.

Using this method we managed to discard **47987 images** while classifying 68 images as "Non Landmark", where they actually are landmarks.

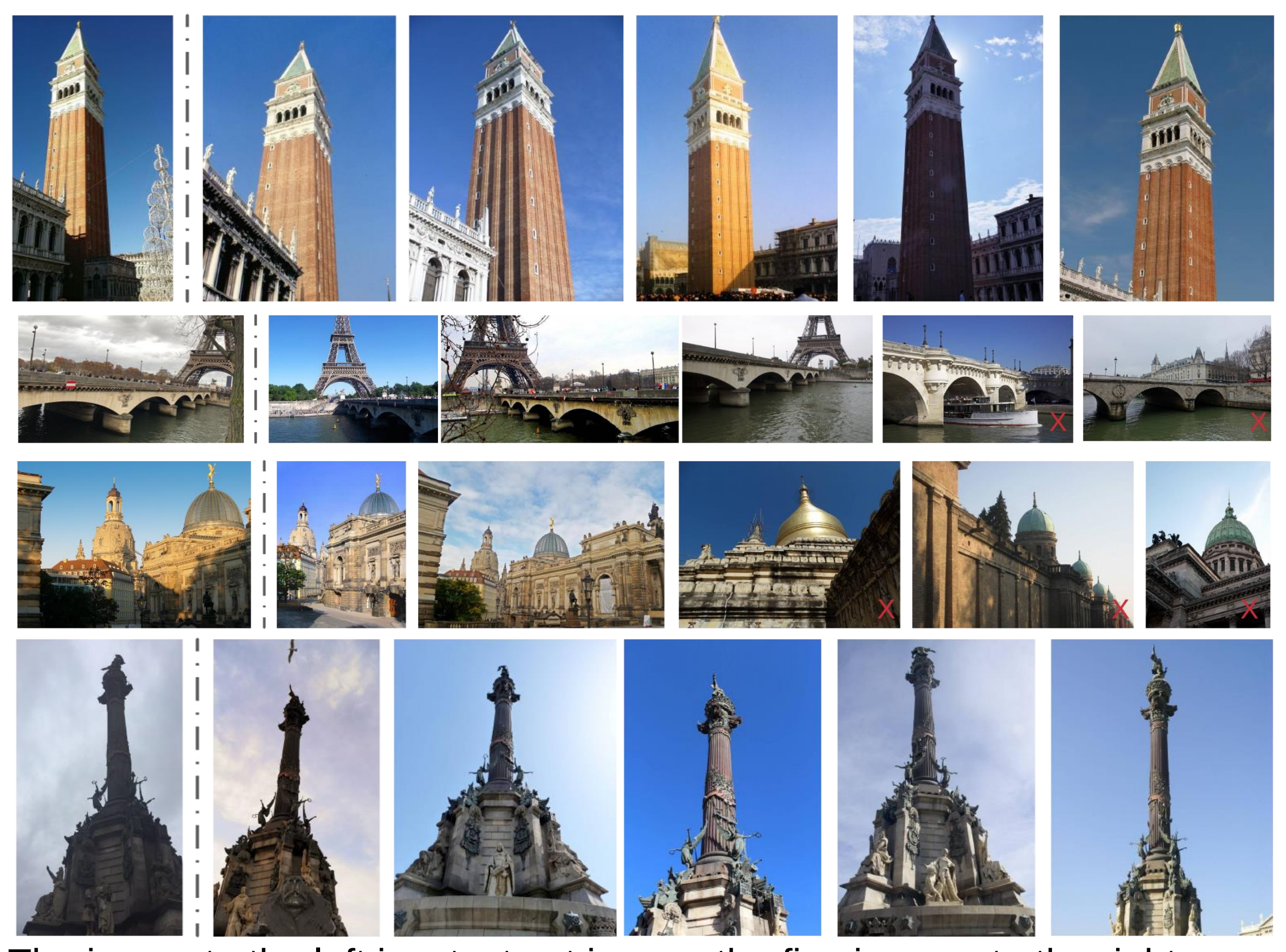
K-NN Classification

After cleaning the test set from out of domain images we'll need to classify the images. To do that, first we'll extract the features of all the train and test set images. We'll extract the features using the last layer of a pre trained CNN model.

After transforming the data from images (i.e. 3 dimensional matrices) to feature vector we'll preform K-NN classification, we'll find each test set items its K nearest neighbors from the train set. The prediction will be based on the most frequent class in the K nearest neighbors.

Using this method we classify correctly 269 landmarks, which is **17.31% accuracy** out of all the landmarks in the test set.

The K-NN classification using features vectors is quite powerful and it managed to identify the main important part of the image.



The image to the left is a test set image, the five images to the right are its nearest neighbors from the train set. If there is a small res X in the lower right corner of the image, it means that this image class is not as the test set image.

For example, in the two middle images there are some neighbors that are not in the same class as the test set image. However they are very similar and the main important part of the image (i.e. the bridge for the first image, the dome for the second image) is part of those images.

Results

Using only the K-NN classification prediction the results are quite poor, that is because the test set contain mainly out of domain images, The K-NN classification doesn't manage to classify those out of domain images correctly.

To classify those out of domain images correctly we'll use the K-NN algorithm output, the classes of each neighbor and the distance (L2 norm) between the image and each of its neighbors.

For each test image, we'll check the class of each neighbor and we'll predict an image as an out of domain if all its neighbors class are different from each other. Since out of domain images are not suppose to look like any of the landmarks (train set images) we assume that they will have nearest neighbors from different classes.

After that we'll check the average distance from each test set image to its nearest neighbors. If the distance is greater than some threshold we'll predict this image as an out of domain image. Since out of domain images are not suppose to look like any of the landmarks (train set images) we assume that the average distance between them and their nearest neighbors will be big.

Using this methods we achieved **76.68% accuracy** out of all the images in the test set while still predicting 225 landmarks correctly.