Architectural heritages elements classification

Sina Tayebi

Dataset

Architectural Heritage Elements Dataset (AHE) is an image dataset for developing deep learning algorithms and specific techniques in the classification of architectural heritage images. This dataset consists of 10235 images classified in 10 categories: Altar, Apse, Bell tower, Column, Dome (inner), Dome (outer), Flying buttress, Gargoyle, Stained glass, Vault. It is inspired by the CIFAR-10 dataset but with the objective in mind of developing tools that facilitate the tasks of classifying images in the field of cultural heritage documentation. Most of the images have been obtained from Flickr and Wikimedia Commons (all of them under creative commons license).

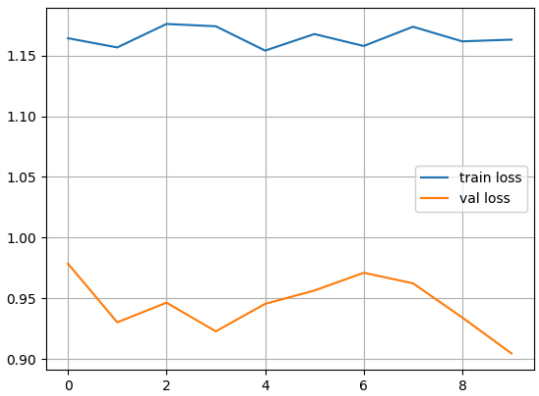
Problem statement

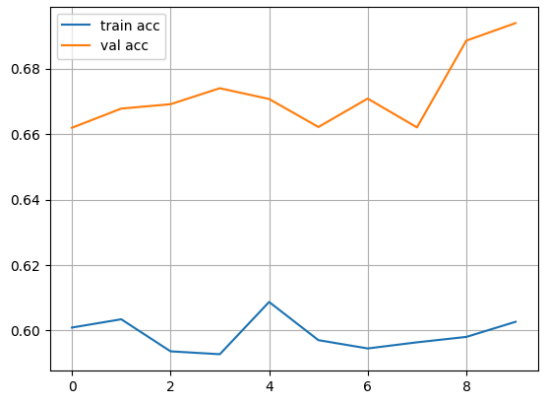
The main task is image classification, the number of classes are ten. On the other hand, the other tasks are, finding each convolution layer output(each filter focuses on what features in the input image) and gaining insights to improve the model performance. Also generating(or maybe reconstructing!!) an image from the last convolution layer output to examine how models interpret the input images.

Experiments

1- Image classification

We designed a CNN model with three convolutional layers without pooling layers to simplify the reconstruction part and stride equal 2 to reduce the image dimensions and padding equal to 1. We used ReLU as an activation function.
For the classifier part we used a fully connected network with 3 hidden layers and dropout.
The result of the model after 50 epochs of training is as follows.

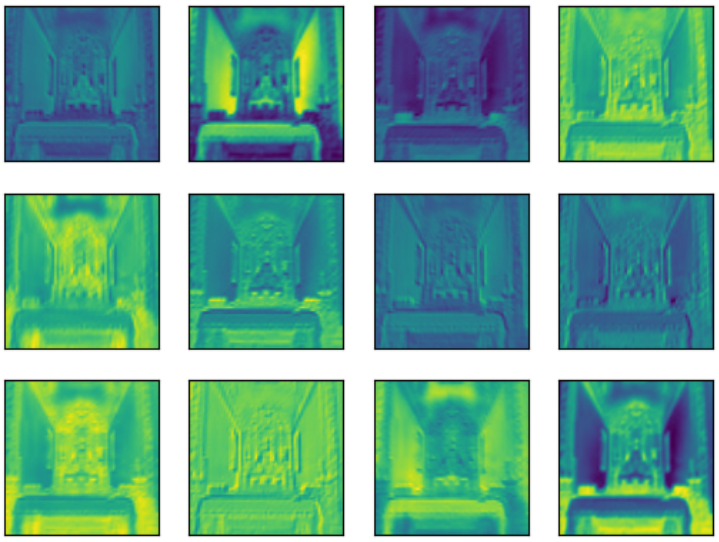




As we can see the model performance is getting better over the epochs. The result is not satisfying enough and it's because of that we make the model as simple as possible to handle other tasks, at last we can gain better results by using other CNN networks like ResNet and InceptionNet which performed well on this dataset(test accuracy more than 90%). So it means the classification problem on this dataset is already solved by using bigger pretrained models, and now the main challenges are visualizing the kernels feature maps and reconstruction images.

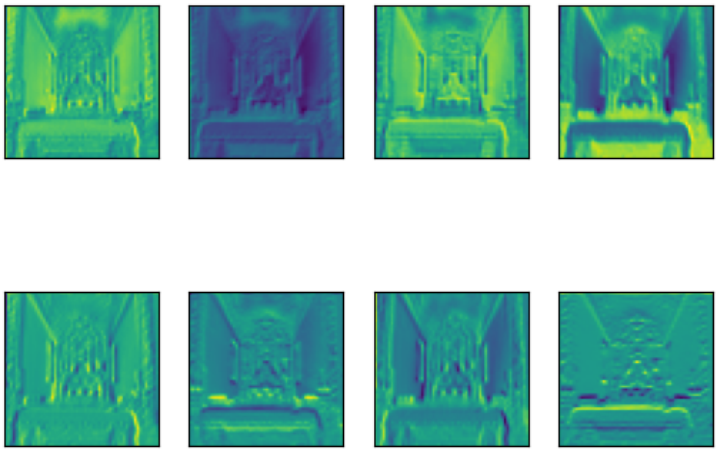
2- Visualising kernel feature maps

In this part, we applied a convolution kernel on the input image and then visualized each feature map. Let see the first convolution layer feature maps.



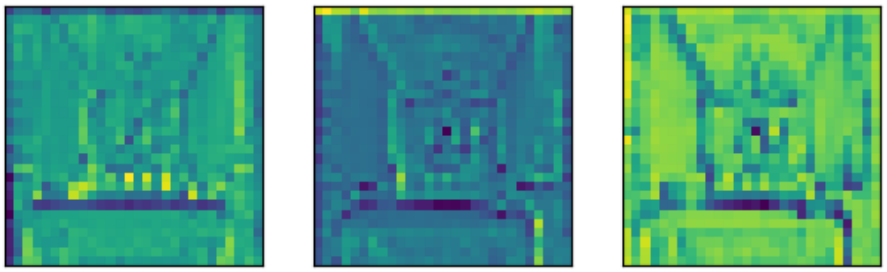
First layer kernel feature maps.

As we can see the first layer convolution extracted some basic features from the image. The abstraction level of features are not much and features are simple as edges and textures. Also the dimension and spatial information of the image reduced after applying the kernel.



Second layer kernel feature maps.

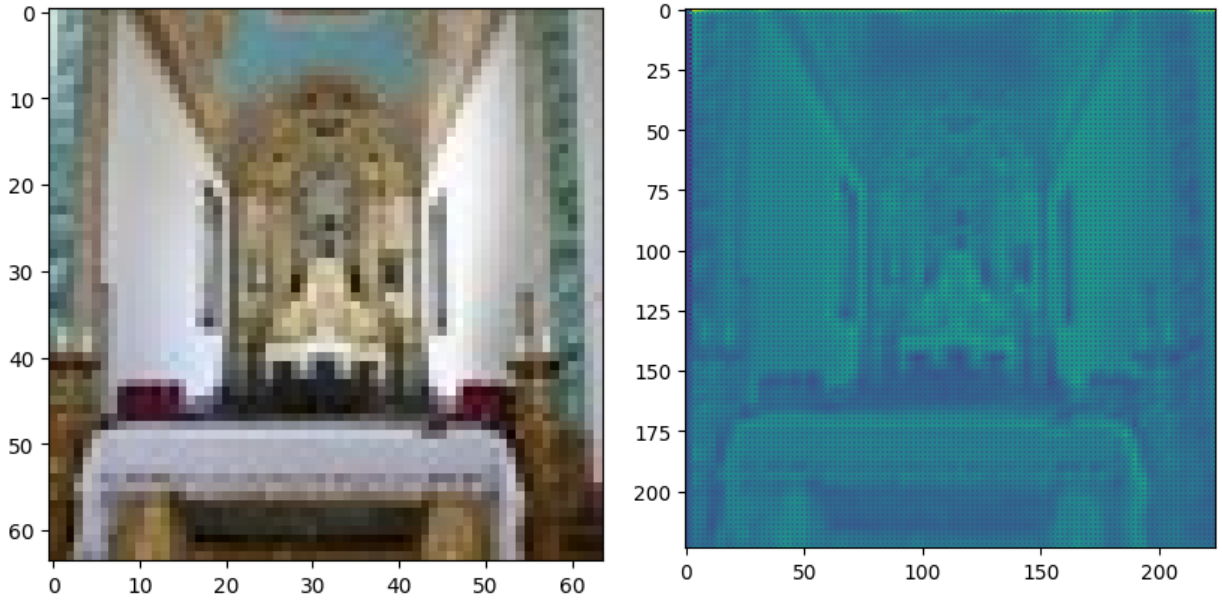
As we move in the depth of the convolutional layers, we can see the features become more abstract and become more semantic.



Third layer kernel feature maps.

3- image reconstruction

We use the idea of Convolution transpose to reconstruct images from the output feature maps. We use the transposed weights of the learned kernels to reconstruct the images. We randomly used an image from the test set.



The main image and reconstructed image.

As we can see, the reconstructed image has the whole shadow of the original image, but it lost some details and colors. This is because of the convolution kernel and the number of channels, in the task of the image classification the color and some other details are not necessary and ignored, but as we can see the main information is kept.