Capstone Project: Landmark Recognition

Rajan Patel

December 2018

1. Definition

Project Overview

During vacations or sightseeing trips it is very important to have more information about the landmarks visited or seen. For example, who built this sanctuary? What does this statue represent? Why is this temple such an important place in this country? Although this kind of information is usually found in digital libraries (e.g., Wikipedia) or by asking local tour guides, it would be very useful to have a tool that would allow you to have the information by taking a picture of the landmark of interest. In addition, this could help to organize extensive photo collections taken in the visits.

Problem Statement

In this project, the objective is to train a model that, given a picture of a landmark is able to predict its name and potentially gather more information, such as, when was it build? Who designed it? Why is it important?

There are thousands of landmarks around the world and obtaining their information manually (that too without knowing their name) can be very difficult, especially if the landmarks are not so popular. With advancement in computing resources, thousands of images can be processed in a fraction of second. Further, machine learning algorithms (especially convolutional neural networks – CNN) have surpassed human performance in image classification for e.g., skin cancer detection (Esteva et al., 2017) and object detection (He et al., 2015). Hence, machine learning using CNN – a proven technique for image classification – is a better choice for our landmark identification problem. Refer to Gu et al. (2018) which describes latest research on CNNs including improvements in loss function, regularization, layer design, and acceleration.

The pipeline to build a CNN for landmark identification is designed as follows:

- Download the images of different landmarks and preprocess the data
- Construct the CNN using transfer learning
    - Replace the last dense layers with appropriate fully connected layers
    - Initialize weights of other convolutional/pooling layers with pre-trained weights
    - Train the complete CNN
- Test CNN after calculating relevant bottleneck features
- Choose the best CNN among different available pre-trained networks

The final CNN is expected to identify the landmark in the image with higher accuracy.

Metrics

Because the data set is considered well-balanced (see Analysis>Data Exploration section), the performance of the considered classification models is evaluated using the following definition of accuracy:

$$Accuracy = \frac{TP + TN}{Total\ number\ of\ samples} * 100$$

Where, TP = true positives and TN = true negatives.

2. Analysis

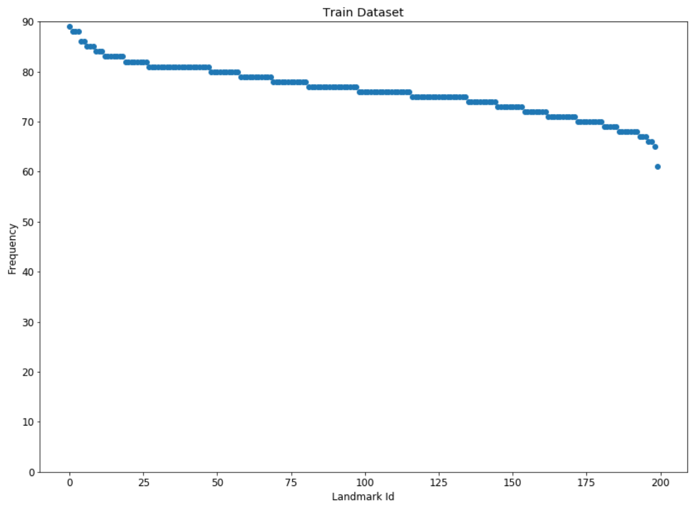
Data Exploration

To train the model the Google-Landmarks dataset is used; this dataset, in its original form contains more than one million landmark images from almost 13000 unique landmarks. The data was originally described in (Noh et al., 2017) and published as part of the Google Landmark Recognition Challenge. In this work, due to memory and processing limitations, the original dataset is reduced to 21816 images of 200 landmarks (Figure 1). This modified dataset is then divided into a training, validation and testing, consisting of 15270, 3273 and 3273 images, respectively.

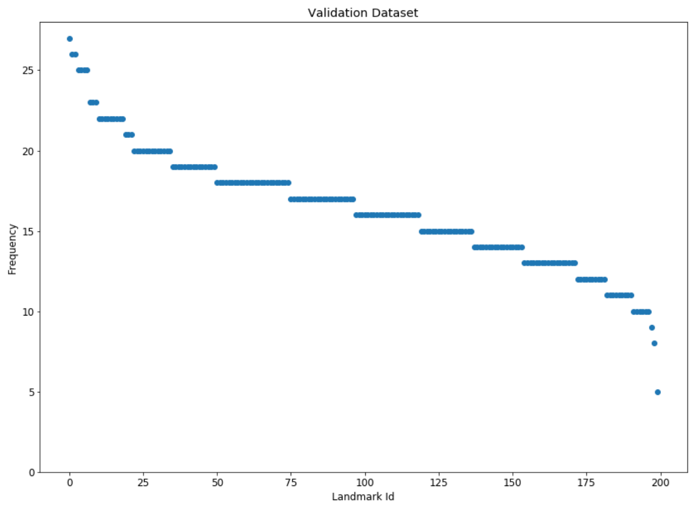
Figure 1. Sample images from the training data set of two (2) different landmarks

Figure 2 shows the number of images corresponding to each considered landmark for the training, validation and testing data. Note that the training data set the number of images per landmark is between 61 and 89, so we can consider it reasonably balanced.

a)



b)



c)

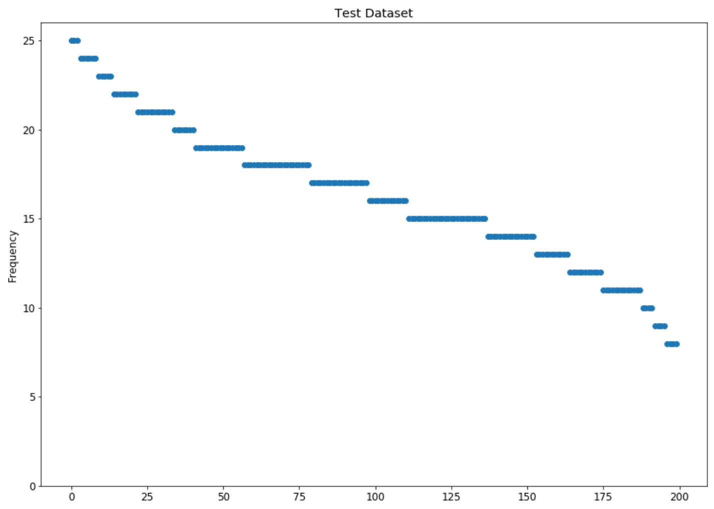
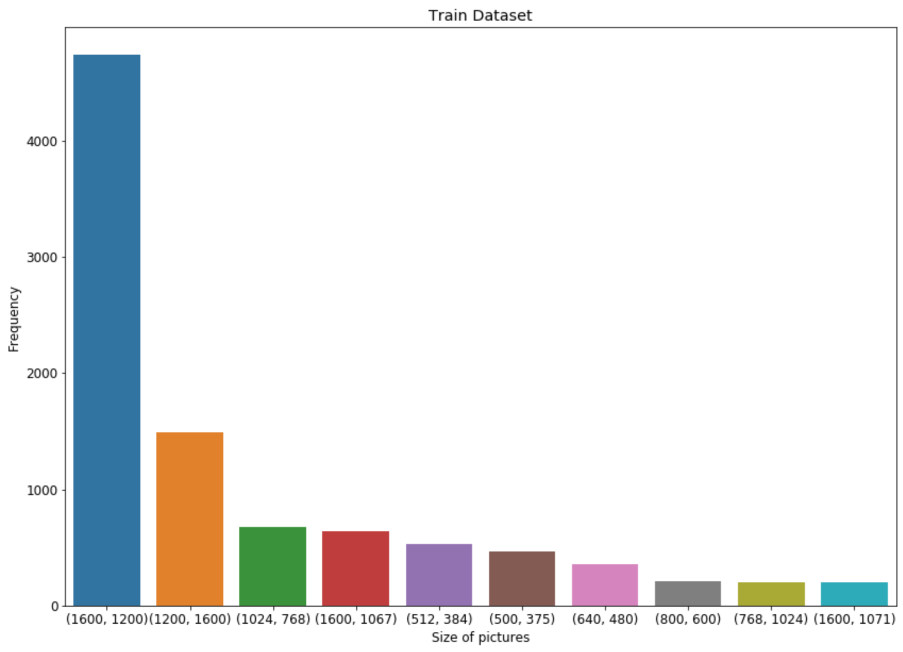


Figure 2. Number of images corresponding to each landmark for training (a), validation (b) and testing (c) data

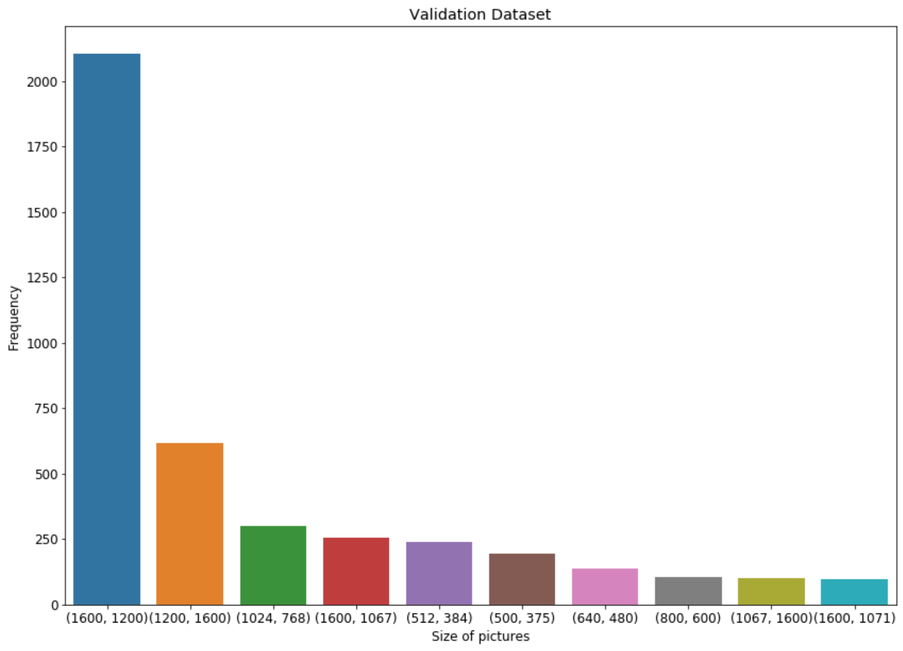
Exploratory Visualization

Regarding the dimensions of the images, Figure 3 shows the number of images according to their size for the training, validation and test data set. Here, the top ten (10) dimensions are shown, note for example that in the test data set for dimension there are 4741 images of the size (`1600,1200`), this represents 31.04% of the total test data set. In general, these ten (10) dimensions group 62.34% of all the images in this data set.

a)



b)



c)

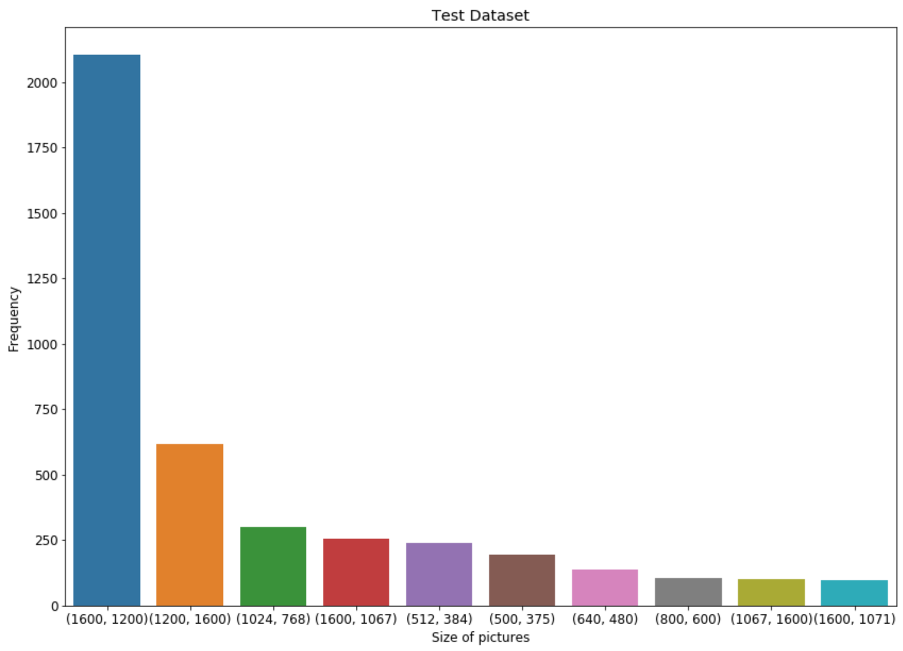


Figure 3. Partial histogram of the number of images according to their size for training, validation and testing data

Algorithms and Techniques

Deep CNNs are used as classifiers due to their ability to handle big data sets. In particular, CNNs can be divided in two parts (Figure 4): feature extraction and classifier. The feature extraction part is the heart of CNN as it captures the important characteristics of an image which can later be helpful in classification part. For feature extraction, CNN uses convolutional layer which can accept matrices as input thereby retaining the image structure. Each node in convolutional layer represents a group of pixels in input image and by moving the convolutional window vertically and horizontally, it tries to learn important features while exploiting the fact that nearby pixels in the image have higher correlations. Convolutional layers are considered as an input for pooling layers in the network which restrains the size of feature maps. To identify multiple features in the image, many combinations of convolutional and pooling layers can be used. It is common to add fully connected layers at the end of the CNN to classify the images based on the features identified by the convolutional layers.

Designing the CNN and training it with large image datasets is computationally expensive and cumbersome. An alternative is to use "transfer learning" approach in which pre-trained CNNs (e.g., VGG19, ResNet50) with pre-established architecture (i.e., number of layers and neurons) are considered. The classifier part is then added which consists of a global average pooling layer and a fully connected layer, where the latter contains one node of each landmark in our case and is equipped with a `softmax` activation function. The weights of fully connected layer are adapted to our landmark dataset by training them using the relevant bottleneck features. In total, three (3) pre-trained CNNs will be evaluated, these are: *VGG16, VGG19* and *ResNet50*.

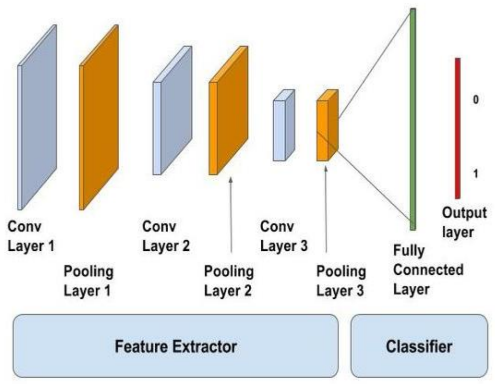


Figure 4. Deep convolutional neural network architecture.
Credit: https://www.learnopencv.com/keras-tutorial-transfer-learning-using-pre-trained-models/

Benchmark

The three pre-trained CNNs described in the previous section will be compared to a traditionally trained CNN (TT-CNN). This TT-CNN, in theory, will exhibit a much lower performance than the tree pre-trained CNN and will represent a sort of sanity check. In terms of the architecture, number of hidden layers, number of neurons in each layers, epochs and activation functions all represent design variables of the TT-CNN. In this context, a trade off between processing time and accuracy must be established.

3. Methodology

Data Preprocessing

The total range of dimensions in the training data set are between (1600,1200) and (11,15). Because the wide range in dimensions of the original images, in preprocessing the images are re-sized to 224x224 and no data augmentation is implemented. Furthermore, since we are working with color images, each image has three channels. So after the processing stage the resulting data structure for each image will be (1, 224, 224, 3).

Implementation

Pre-Trained Deep Convolutional Neural Networks (CNN)

1) Pre-train stage

   a. Download the pre-computed bottleneck features of the three (3) considered CNNs, i.e., VGG16, VGG19, and ResNet50.

   b. The final layers of the networks are removed because they are too specific for the particular dataset. The remaining layers correspond to feature extraction section of the networks.

   c. Finally, the networks are shown the available data and the network is stored in an *.npz* format file.

2) Classifier stage

   a. The pre-trained networks in the form of the corresponding *.npz* are uploaded and fed as input to a global average pooling layer and fully connected layer.

   b. The training and validation data set are converted to a binary class matrix and are shown to the networks. Here the weights of the final two layers are calculated. In this process, the loss function, optimizer and metric used were categorical `crossentropy`, `rmspropr` and `accuracy`, respectively.

   c. Lastly, the network can now be used to test how well it identifies landmarks using the testing data set and a value of accuracy is calculated.

Traditionally trained CNN

The general architecture consists of the following elements and their corresponding parameters:

a. *Convolutional and pooling layers*. The combination of these two types of layers make up the feature extraction parte of this network, thus the number of layers used and their size must be selected.

b. *Dense layers (classifier).* These layers are used to convert the features into probability values. Key parameters include, activation function (e.g., `Relu`), dropout ratio. Additionally, similarly to the pre-trained CNNs, the last layer must use a `softmax` activation function to indicate the probability of the input image showing a particular landmark.

c. *Number of epochs*. The value of the loss function is constantly monitored and the number of epochs must be adjusted if this value is constantly decreasing.

In this process, the loss function, optimizer and metric used were categorical `crossentropy`, `rmspropr` and `accuracy`, respectively.

Refinement

Since we are using pre-trained CNNs to build our classification model, the typical tuning parameters in conventional CNNs have already been pre-established, i.e., number of hidden layers, number of neurons in each hidden layer and activation functions in these layers. Hence, the refinement is done by evaluation three (3) different pre-trained CNNs: VGG16, VGG19 and ResNet50. The training is done ten (10) times and the average accuracy for each CNN is shown in Table 1. Note that the Resnet CNN exhibits an average accuracy of 96.82861, compared to 84.20714 and 89.34617 of the VGG16 and VGG19 respectively.

Table 1. Accuracy of the three (3) considered CNNs

Pre-trained CNN	Average of Accuracy
VGG16	84.20714
VGG19	89.34617
ResNet50	96.82861

4. Results

Model Evaluation and Validation

As mentioned in the previous section, the CNNs where trained a ten (10) times to evaluate the randomness in the training process. Figure 5 shows the values of accuracy of each CNN in each training run; note that the variability exhibited by each CNN is low, especially for VGG19 and Resnet. In particular, the variance values were of 6.53, 0.13, 0.04 for the VGG16, VGG19 and Resnet, respectively.

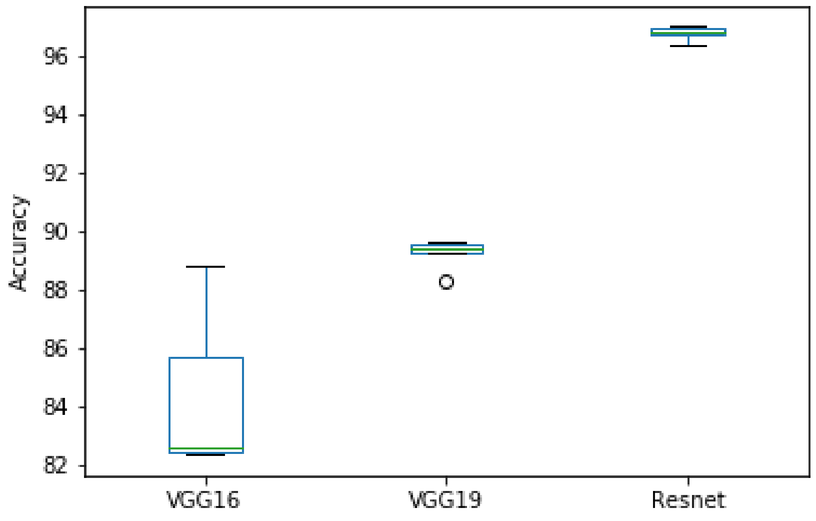


Figure 5. Accuracy values of each considered CNN in each training run

Justification

As shown in the previous sections, the Resnet model clearly outperformed VGG16 and VGG19, exhibiting an accuracy of over 96%. In addition, we can compare this performance to out cited benchmark model corresponding to a traditionally trained CNN consisting of:

- Five (5) layers of 3 by 3 convolutions
- One (1) 2 by 2 max pooling layer
- Two (2) dense layers. The first uses a Relu activation function and a dropout ratio of 30%
- 25 epochs

The performance of these two (2) models are shown in Figure 6, note that Resnet exhibits a much higher performance of more than 96% of accuracy while the traditionally trained CNN exhibits an accuracy of 39.2.

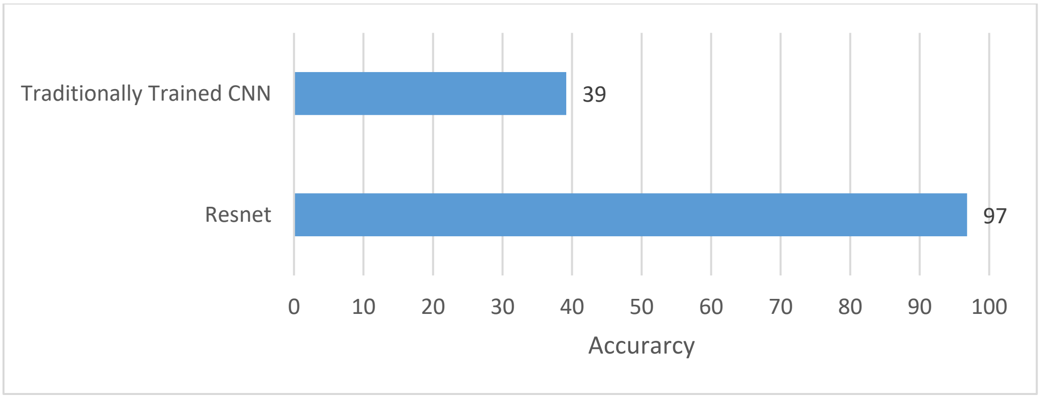


Figure 6. Accuracy exhibited by Resnet and a traditionally trained CNN

5. Conclusion

Free-Form Visualization

Quality of the final ResNet model was further confirmed with different type of images (for e.g., blurred, old, scanned, bird-eye view) of the three known landmarks (i.e., ceasers palace hotel, CN tower, and colosseum) downloaded separately from the Google (See Figure 7). The figure also shows whether they were correctly identified or not using green checkmark and red cross mark.

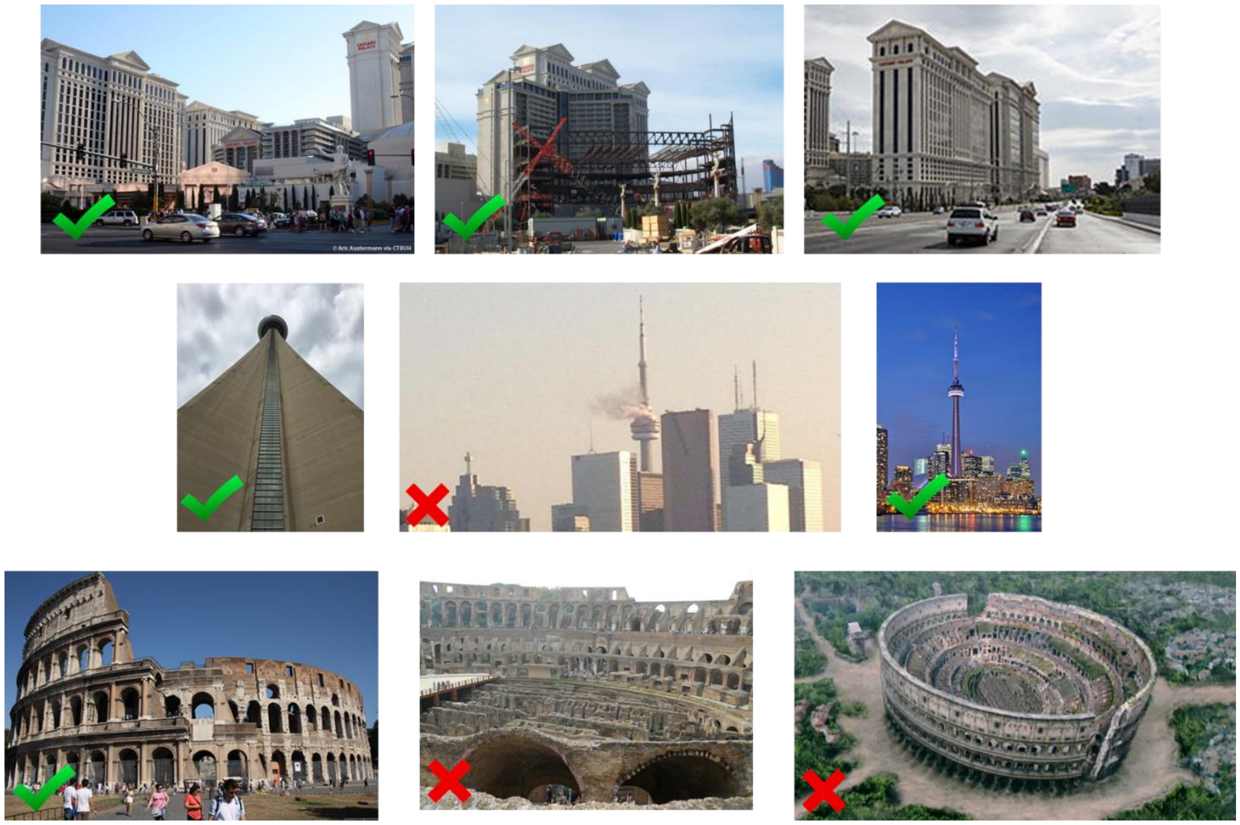


Figure 7. Examples of the images tested with ResNet model

From Figure 7, it can be said that only three images were misclassified probably due to following reasons:

- Other objects in the front (e.g., CN Tower is hiding behind other skyscrapers)
- Unusual/old image of a landmark (e.g., training data doesn't have inside or old images of colosseum)

Reflection

In this work the superiority of pre-trained convolutional neural networks is shown when compared to the traditionally trained alternative in image classification. In particular, even though all the considered pre trained models had an accuracy of more than 88%, the Resnet model shows impressive performance exhibiting values of accuracy of more than 96%. This high performance could also be explained by the fact the used data set shows significance balance and thus making it easier for the neural networks to make good predictions.

Improvement

Some improvements that should be made to this work include:

- Make use of the complete data set of the Google-Landmarks dataset that consists of more than one million landmark images from almost 13000 unique landmarks.
- In the case that the complete Google-Landmarks dataset is used, different metrics should be considered depending on how balanced this dataset is.
- Although the Resnet model showed impressive accuracy, other pre trained models should be evaluated to discuss its relative performance with respect to Resnet and draw conclusions as to when they would be more appropriate.

References

Noh, H., Araujo, A., Sim, J., Weyand, T., Han, B., 2017. Large-Scale Image Retrieval with Attentive Deep Local Features. Proc. IEEE Int. Conf. Comput. Vis. 2017–October, 3476–3485. https://doi.org/10.1109/ICCV.2017.374

Esteva, A., Kuprel, B., Novoa, R.A., Ko, J., Swetter, S.M., Blau H.M., Thrun, S. Dermatologist-level classification of skin cancer with deep neural networks. Nature **542**, 115-118. https://doi.org/10.1038/nature21056

He, K., Zhang, X., Ren, S., Sun, J. Deep Residual Learning for Image Recognition. arXiv:1512.03385

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., Chen, T. Recent advances in convolutional neural networks. Pattern Recognition **77**, 354-377. https://doi.org/10.1016/j.patcog.2017.10.013