Project Report : Google Landmark Retrieval Challenge(Kaggle)

John Martinez
New York University
jzm218@nyu.edu

Vinay Bhapkar
New York University
vvb231@nyu.edu

Shivam Swarnkar
New York University
ss8464@nyu.edu

1 Abstract

Our team participated in two Kaggle competitions as Term project for course in Deep Learning:

1.1 Google Landmark Retrieval Challenge

- Dataset with more than a million index images and over 100K query images. The task was to cluster the index images around the query images. We reached leader-board position 51 with an accuracy of 0.037.

1.2 Google Landmark Recognition Challenge

Dataset with more than 1.2 million labeled train images and the same set of query images, used as the test set. The task was to classify these test images. We reached leader-board position 129 with an accuracy of 0.007.

2 Retrieval Challenge

While most images in the Google Landmark Retrieval challenge dataset are landmark-centric, dataset contains more realistic images with wild variations including foreground/background, clutter, occlusion, partially out-of-view objects, etc.. Ground-truth annotation was extremely challenging in this case, especially considering the facts that it is hard to predefine what landmarks are, landmarks are not clearly noticeable sometimes, and there might be multiple instances in a single image. Choosing cluster on the basis of top features of image is the approach we followed in this case.

(i) Used a pre-trained VGG-16 architecture for feature extraction of all index and query dataset.

(ii)Performed k-nearest neighbors with k=150, to get train images corresponding to each query images . For deciding optimal value of K , we used concept of Silhouette Coefficient score and elbow method.

(iii)Loading and using this massive data was also challenge. Many urls in the csv file provided were broken. We simply excluded these data items.

3 Recognition Challenge

Similar as of retrieval image dataset ,while most images in the datasets are landmark-centric, dataset contains more realistic images with wild variations including foreground/background including foreground/background clutter, occlusion, partially out-of-view objects, etc. Due to this state of the art global and local descriptions in the large scale setting were not enough. DELF algorithm significantly outperforms these methods. DELF involves instance retrieval. clutter, occlusion, partially out-of-view objects, etc.

From various state of the art papers in computer vision we came to conclusion that for classification for this kind of complex and massive data following approach can derive good results: Large-scale retrieval system can be decomposed into four main blocks: (i) dense localized feature extraction, (ii) key-point selection, (iii) dimensionality reduction and (iv) indexing and retrieval.

DELF repository we used for project identify the top K(= 150) nearest neighbors for each feature in a query and extract up to 1000 local features from each image. Each feature is 40-dimensional.

In the DELF method, when a query is given, the approximate nearest neighbor search for each local descriptor extracted from a query image is performed. Then for the top K nearest local descriptors retrieved from the index, all the matches per database image. aggregated. Finally, geometric verification performed using RANSAC [10] and the number of inliers as the score for retrieved images is employed.

First we performed k-nearest neighbor on vgg16 extracted features of train and test data. Top four images corresponding to least distance from centroid are selected for feature extraction and matching with delf.

4 Problem Formulation

4.1 Problem Type

The Google Landmark Retrieval Challenge is similar to a clustering problem, as we have unlabeled data that we must group into clusters of multiple classes using some similarity metric. The Recognition challenge has labeled data and is a classification challenge, but we can also apply clustering to group classes together. However, before clustering images, we must reduce them into feature vectors using a network, making the challenge a computer vision problem as well.

4.2 Data

The full dataset is stated to contain images of around 15K distinct landmarks. It is not specified whether all of them are contained within or how they are distributed among the test images - only that many contain multiple landmarks, which means that the subsets of the index data will have intersections. This means that an image can be of multiple classes, and thus be placed into multiple clusters. Test data have many images which do not represent any Landmark, whereas train data contain only landmarks. Due to this there are many images in test data which can not be assigned any label based on training data.

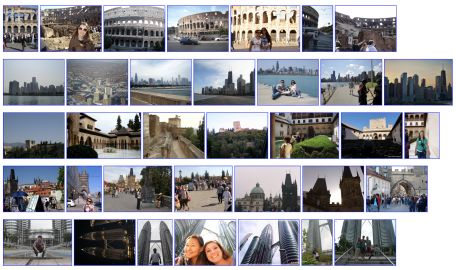

Figure 1: A sample of the training images: each row represents images for unique landmark.

4.3 Submission

For our submission, we had to generate a csv file with two columns. The left column contained the ids of test images and the right column contained space-delimited lists of ids of the index images that had the same landmarks as the test images.

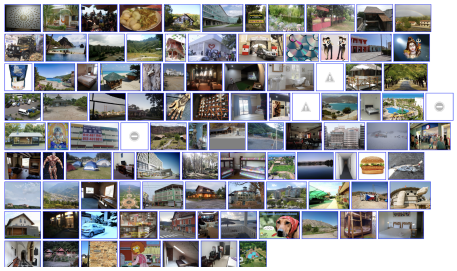

Figure 2: A sample of the test images used for both the Retrieval and Recognition Challenge: Most of then contain non-landmarks, some may contain multiple landmarks.

4.4 Evaluation

The sumbissions to the Google Landmark Retreival challenge are evaluated using mean Average Precision @ 100. (mAP @ 100, calculated as following).

$$mAP@100 = \frac{1}{Q}\sum_{q=1}^{Q}\frac{1}{min(m_q, 100)}\sum_{k=1}^{min(n_q,100)}P_q(k)rel_q(k)$$

Where

- Q is the number of query images that depict landmarks from the index set
- mq is the number of index images containing a landmark in common with the query image q (note that this is only for queries which depict landmarks from the index set, so mq0)
- nq is the number of predictions made by the system for query q
- Pq(k) is the precision at rank k for the q-th query
- relq(k) denotes the relevance of prediction k for the q-th query: its 1 if the k-th prediction is correct, and 0 otherwise

For the Recognition challenge ,if a submission has N predictions (label/confidence pairs) sorted in descending order by their confidence scores, then the Global Average Precision is computed as:

$$GAP = \frac{1}{M}\sum_{i=1}^{N}P(i)rel(i)$$

Where:

- N is the total number of predictions returned by the system, across all queries. In this case, it is at most 1.
- M is the total number of queries with at least one landmark from the training set visible in it (note that some queries may not depict landmarks).
- P(i) is the precision at rank i.
- rel(i) denotes the relevance of prediciton i: its 1 if the i-th prediction is correct, and 0 otherwise.

5 Methods

5.1 Algorithms

We used a pre-trained vgg16 network, implemented in Python with the Tensorflow library, to calculate feature vectors of size 1000 from the full dataset, which we preprocessed using zero-mean normalization. We then applied K-means clustering, which we also implemented in Tensorflow, to the feature vectors to assign a cluster ID to each image, thereby grouping them to generate our submission file. We made some minor modifications to the K-means code to turn it into K-Nearest Neighbors, and then applied DELF, which uses the RANSAC function to calculate image similarity, and Mask RCNN to find common shapes between images.

6 Architectures and Implementation

6.1 VGG-16

The VGG16 network consists of 13 2-Dimensional convolutional layers and 5 max pooling layers using same-padding followed by fully-connected layers. We used ReLu as the activation function. The final layer yields a size-1000 feature vector for each image. These are used as a data points for the K-means clustering. For the K-means clustering, we used Euclidean distance as the similarity measure and set the initial centroids to be the feature vectors representing the test set. Although we know that there are 15K landmarks in the index set, the number of classes contained in the test set was not given. Hence, it was necessary to determine K by other means. We chose this network for the Retrieval challenge because we had unlabeled data which we wanted to cluster, rather than classify. This pre-trained model converted images into unlabeled vectors whose similarity could easily

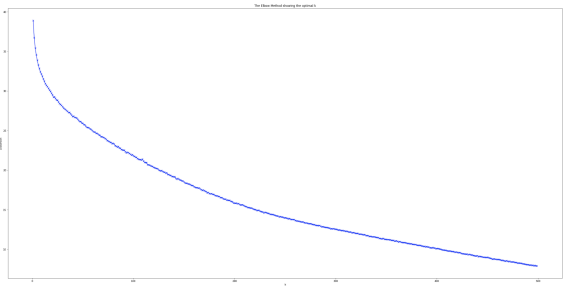

Figure 3: Distortions Vs Number of cluster plot

Figure 4: Evaluation of optimal K with Silhouette Coefficient for sample data

```
For n_clusters=245, The Silhouette Coefficient is 0.3195059895515442
For n_clusters=246, The Silhouette Coefficient is 0.31357792019844055
For n_clusters=247, The Silhouette Coefficient is 0.31629547476768494
For n_clusters=248, The Silhouette Coefficient is 0.3189690411090851
For n_clusters=249, The Silhouette Coefficient is 0.31565937399864197
For n_clusters=250, The Silhouette Coefficient is 0.3169248700141907
For n_clusters=251, The Silhouette Coefficient is 0.320673406124115
For n_clusters=252, The Silhouette Coefficient is 0.31612807512283325
For n_clusters=253, The Silhouette Coefficient is 0.31671789288520813
For n_clusters=254, The Silhouette Coefficient is 0.3154420852661133
```

be calculated and used for clustering, and made extremely few mistakes on a small sample dataset. We also used grouping images together for classification in Retrieval.

6.2 Finding K

We assumed K to be equal to the size of the test set (query images). We reached this assumption by performing the elbow method, which plots distortions (the minimum Euclidean distance for each image vector and centroid) VS the number of clusters. The elbow point of the graph, or the point of steepest change in distortions is taken as the optimal K value, which we found to be near the number of query images.

Along with Elbow method we have used Silhouette Coefficient Method. A higher Silhouette Coefficient score relates to a model with better-defined clusters. The Silhouette Coefficient is defined for each sample and is composed of two scores: a: The mean distance between a sample and all other points in the same class. b: The mean distance between a sample and all other points in the next nearest cluster.

The Silhouette Coefficient is for a single sample is then given as: s = (b-a)/max(a,b)

6.3 K-Nearest-Neighbors

After this attempt, we applied the K-Nearest Neighbors algorithm to the same feature vectors produced by the VGG16 Net. We did not need to know the number of classes. For the Retreival Challenge, we

chose 150 for our value of K and calculated, for each query image, the K most similar index images, still using Euclidean distance as our similarity metric. For the Recognition Challenge, we knew the number of classes to be 14951. As such, we rounded the square root of this number and used 123 as our value for K. For each query image, we calculated the number of votes for each class by counting the number of times they occurred in the cluster given to that query image. We then picked the class that had the most votes and calculated our confidence by taking this number and dividing by K. We chose this method to improve upon our K-means clustering method, as it was grouping far too many index and train images to a few query images. With this method, we were able to control the number of predictions and have a reasonable confidence metric.

6.4 DELF

We then applied Google's DELF (DEep Local Features) Repository.

This code takes two images as input, and by extracting local features and descriptor matrices for each, produces corresponding points, such as edges or corners between the two images, known as inliers. We used the number of inliers as a similarity metric and attempted a nearest neighbor search. Due to computational limitations, we used sampling to find these nearest neighbors. Using the output yielded by the KNN algorithm, we determined the classes of the four most similar train images for each query images, and randomly sampled two images from the training set for each class. We then picked the class with the largest total number of inliers with the query image and divided it by the total inliers for all four pairs of training images to calculate the confidence. We used this repository because it was specifically designed for images containing landmarks, and works very well in determining whether two images contain the exact same landmark.

7 Results

7.1 Accuracy

The accuracies for Retrieval and Recognition submissions are 0.037 and 0.007 respectively. These latest scores and rankings can be found here (Our team name is NYU_DL) :

- Retrieval: https://www.kaggle.com/c/landmark-retrieval-challenge/leaderboard.
- Recognition: https://www.kaggle.com/c/landmark-recognition-challenge/leaderboard

Figure 5: Query image(left),training image(right) Here we see an image of a non-landmark given a classification. Although the images are entirely irrelevant to each other, the model we used made some prediciton for every query image. This may very well have contributed to our results.

Figure 6: Query image(left),training image(right). Here we see an image of a non-landmark given a classification. The images are similar and have common objects, but they are not of the same landmark.

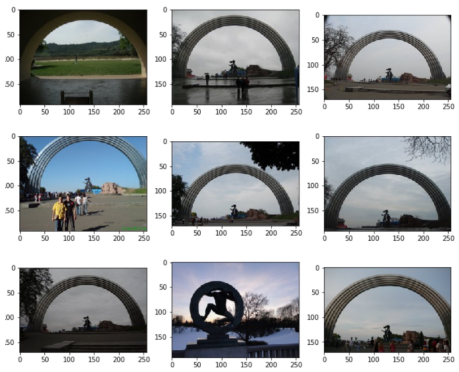

Figure 7: Cluster of Retrieval images. The top left image is the query image. We can see here that most of the images in the cluster are the same landmark, but they have been grouped with a similar image of a different landmark.

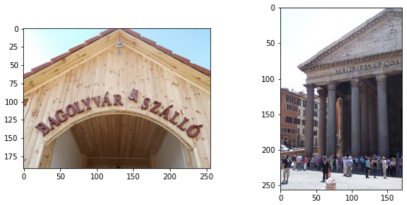

Figure 8: KNN: Query image(left),training image(right). Here an image is shown of the landmark that the model classified believed the query image to be. Once again, however, they are not the same landmark, but from their shapes it is understandable why the left landmark would be classified as the right. This shows that the model can group together the objects of the same general type and shape, but has trouble identifying the exact same object.

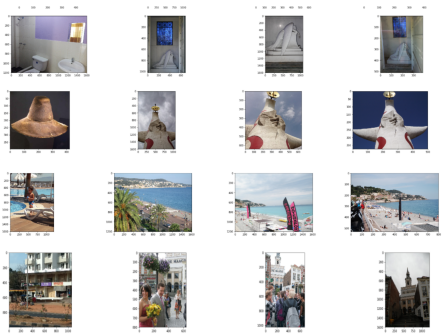

Figure 9: Clusters of images from KNN and DELF. Query image(left), 3 train images (right). We can see here that for each of the query images, the 3 nearest neighbors were the same landmark, and a landmark with interesting commonalities with its respective query image. On the first row, we see a white room with a window. On the second, we see a statue with a similar shape to the hat to which it was classified. On the third, we see a beach setting and on the fourth we similar buildings.

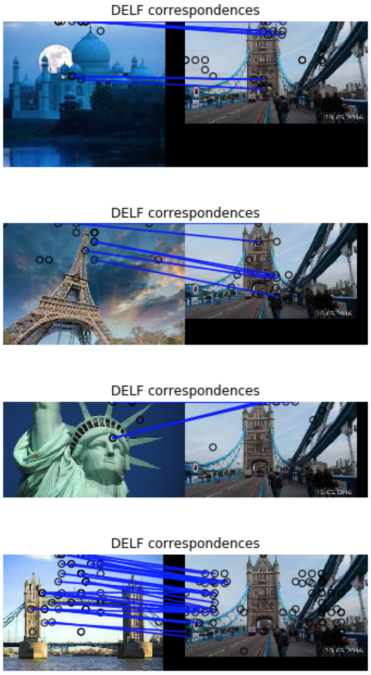

Figure 10: Four examples of DELF results. Here we can see on the first three pairs of images, that for different landmarks, there are less than 10 inliers. However, for the fourth pair of images of the same bridge taken from completely different angles, there are hundreds of inliers.

7.2 Code

Our full repository can be found here: https://github.com/JohnMartinez12/Landmark

8 Positive Aspects

- We were able to make improvements to both our Recognition and Retrieval scores and reach a nonzero accuracy for both.
- For some query images, we were able to cluster several images together with the same landmark, such as in Figure 7.
- The DELF method would likely yield some impressive results if we could perform a brute-force search, effectively performing the KNN algorithm, but with inliers as distance. This is well shown in Figure 10.

9 Drawbacks

- Even with 4 GPUS, the entire dataset is too large for any of the algorithms described here to generate submissions quickly enough. We will have to rely on a great deal of sampling and/or a different method of increasing the speed.
- as seen in some figures, the images in one cluster can contain the same landmark, yet all be attached to a similar looking but different landmark, or an irrelevant image, like in Figure 7.
- The method we used made a prediction for every image. This may very well have hindered our results. As we can now see with the data, we will have to make no prediction for most of the query images.

10 Future Work

To improve our results, we plan to further attempt (i) Increasing DELF's speed through different means of parallelization. Since we know that most images in the dataset are of non-landmarks, we will also attempt means of rejecting these images to greatly reduce the size of the dataset. We may build a binary classifier to remove these images or use clustering to remove outliers that are clearly non-landmarks.

(ii) Using Mask-RCNN for improved classification. RCNN(Rich feature hierarchies for accurate object detection and semantic segmentation) presents conceptually simple, flexible, and general framework for object instance segmentation.

Algorithms like Fast-RCNN over come computational drawbacks - like Training is a multi-stage pipeline, Training is expensive in space and time,Object detection is slow. Fast R-CNN builds on previous work of R-CNN to efficiently classify object proposals using deep convolution networks. Along with approach of Fast- RCNN , Mask R-CNN Use a ResNet-FPN backbone for feature extraction with Mask RCNN gives excellent gains in both accuracy and speed. As a general framework, Mask R-CNN is compatible with complementary techniques developed for detection/ segmentation, including improvements made to Fast/Faster R-CNN and FCNs.(Fully convolution Networks).

11 Related work and references

- The details and rules for the Google Landmark Retrieval Challenge: https://www.kaggle.com/c/landmark-retrieval-challenge/rules
- The details and rules for the Google Landmark Recognition Challenge: https://www.kaggle.com/c/landmark-recognition-challenge/rules
- Clustering And Querying Images From Unknown Classes Using Metric Learning (Chinmayee Shah - Electrical Engineering, Stanford University): http://cs231n.stanford.edu/reports/2017/pdfs/107.pdf
- Mozgalo2017 - Unsupervised content based image clustering: https://github.com/TKeesh/ImgClustering
- Finding the K in K-Means Clustering. We used this for our very first method, to determine the number of clusters we would use for the Retrieval Challenge: https://datasciencelab.wordpress.com/2013/12/27/finding-the-k-in-k-means-clustering/
- The Google DELF repository to apply inliers as a similarity metric. Small demonstrations of this code have performed exceptionally well. The challenge we now face is the speed: https://github.com/tensorflow/models/tree/master/research/d
- Large-Scale Image Retrieval with Attentive Deep Local Features - The paper associated with the DELF repository: https://arxiv.org/pdf/1612.06321.pdf