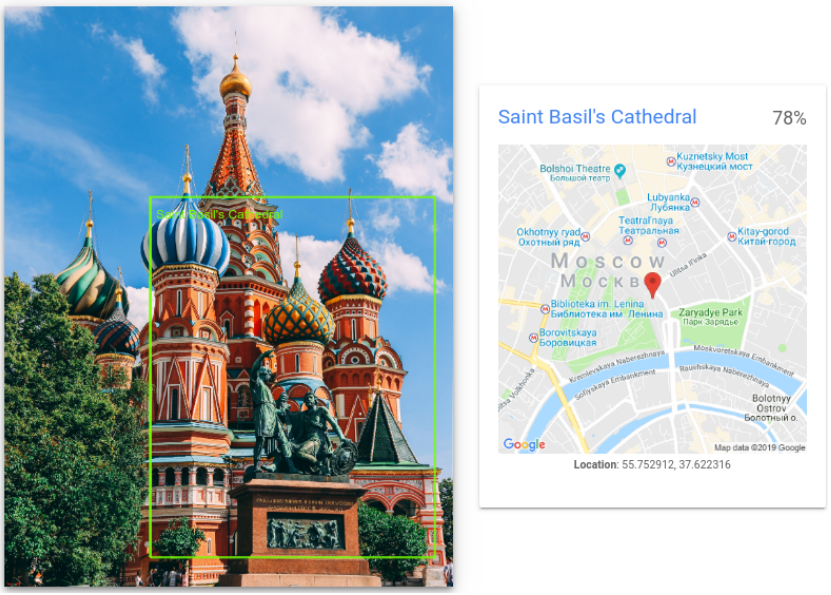
SAPIENZA
UNIVERSITÀ DI ROMA

AML final Project

Google Landmark Recognition



Trina Sahoo (1901254)
Paolo Falcone (1798233)
Debodeep Banerjee (1901253)
Dario Baraghini (1810586)

December 2021

Introduction

For this project we decided to take on the task of recognizing landmarks from a google dataset to be able to implement our image recognition knowledge into actual reality and important tasks, we did this by participating in this kaggle competition

Exploratory Data Analysis

The first thing we did was analyzing everything that was provided to us, we got a folder made up of a test, training and 2 csvs containing the image id and the landmark ID they actually belonged to, by analyzing the entire folder we found that the dataset was huge **100 gigabytes and more than 81 k classes**, we instantly realized that a skimming would have been necessary in order to process this data due to the lack of computational power.

As we can see from the plots above before we did the skimming there were many classes that had less

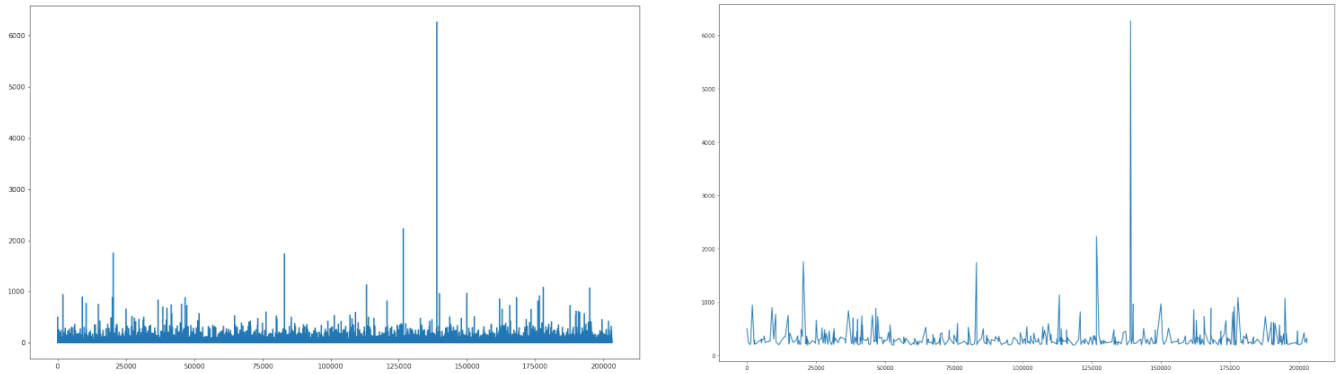


Figure 1: Exploratory Data Analysis

than 10 images associated, this was leading our models to having worst performances and also have huge running times (up to 8 hours for a single epoch of run), we cut down the classes to 500 and 200 (to be able to showcase different results) which is shown on the right.

Model Creation and Selection

We started with a CNN from scratch but we observed that it appeared to achieve less accuracy and it was computationally heavier than the other pretrained models which we tried. In this task we have considered 4 different types of pretrained models, viz, ResNet50, MobileNetV2, DenseNet121, Vgg16. To work with the models we have removed the last 3 layers of the model and added a pooling layer known as **GeM Pooling** layer and at last the dense layer with unit depending on the number of classes. The usage of GeM pooling layer increases the accuracy[1]. The actual increment happened after the usage of the **Ensemble model**.

GeM Pooling: GeMPooling computes the generalized mean of each channel in a tensor. Formally, the GeM embedding is given as $e = [\frac{1}{|\Omega|}\sum_{u\in\omega}(x_{cu}^{p})^{\frac{1}{p}}]_{c=1\ldots C}$ where x is the tensor computed from the convolutional neural network for a given image, C is the number of features or classes and $u \in \omega$ is the pixel depending on the height and the width for each channel[2]. So, gempooling reduces the dimentionality, performs normalization and finally the final output is the image deciptor. We have selected GeMPooling over MaxPooling or AvgPooling because it improves the implicit correspondences between the images and thus helps in image retrieval process.

Ensemble Model: As we have observed in many cases given a certain dataset various models behave in various ways, we found that none of the models we used performed up to the mark. As we do not meet the heavy hardware requirement to do hyper parameter tuning, an ensemble set up was the

best alternative. We observed(see figure 2) that for 200 classes, while all other models were giving a highest accuracy of 78%, the ensemble model helped us to elevate the accuracy to 86.05%. For 500 classes(see figure 3), while all the other models accuracies where ranging around 70%, the ensemble model accuracy spiked up to 81.15%.

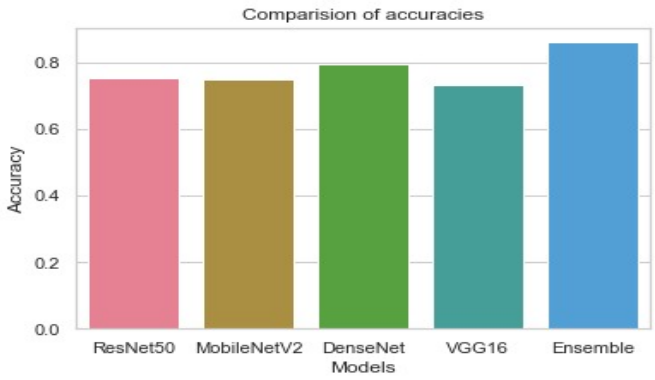


Figure 2: Model Accuracy on 200 classes

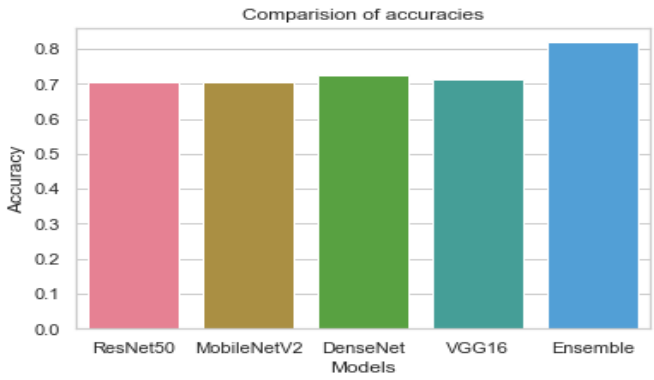


Figure 3: Model Accuracy on 500 classes

Experimental Results

In this section of the report we will discuss about the results obtained and some interpretation from the models used.

1. The table below shows the predicted test label and original test label for 200 and 500 classes.

true labels	predicted labels
13471	13471
41648	41648
13471	80147
11971	11971
30640	30640
35691	35691
75005	75005
10419	10419
73300	73300
39938	61553

true labels	predicted labels
16658	16658
138982	138982
67929	67929
30048	115979
139706	139706
162569	162569
28641	28641
108472	108472
6138	6138
190216	190216

   The first table shows the label prediction of test images for 200 classes and the next table shows the label prediction of test images for 500 classes.

2. The predicted test image for 200 classes and 500. For both the images we can observe that the prediction finds the similar images based on same label. For 500 classes 5, it shows the images of smoke in the hilly areas. For 200 classes 4, it clusters the image of wind mills.

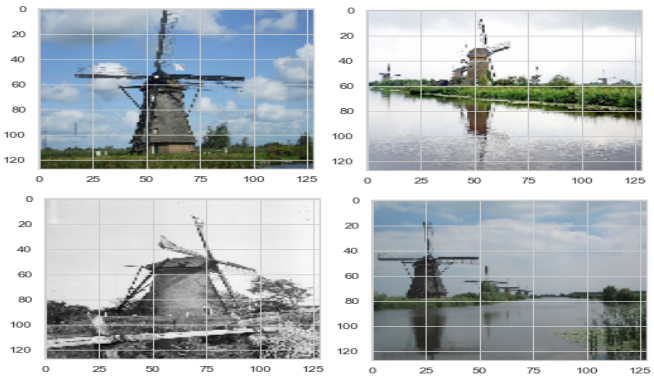


Figure 4: Test images for 200 classes

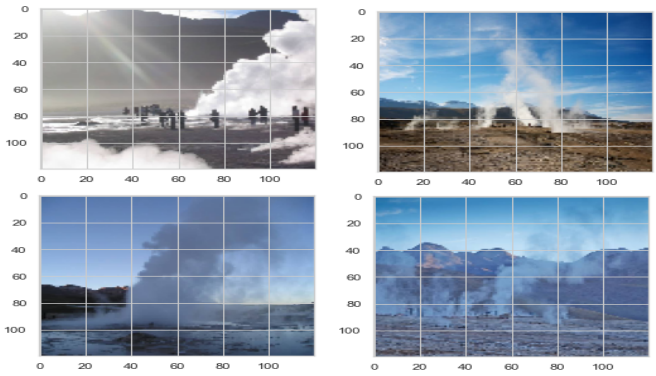


Figure 5: Test images for 500 classes

Conclusion and Future Work

In this project we have considered various models but the use of GeMPooling layer and ensemble model gave the highest accuracy. Another important remark obtained from this dataset is the lesser the number of classes the better the accuracy is because of the variation in the number of images per class.
Further as a part of the project we can implement dual path network which basically combines ResNet and densenet. The intuition is that ResNets enables feature re-usage while DenseNet enables new feature exploration, and both are important for learning good representations[3]. The other one that can be explored further is the grid search for weighted averages in Ensemble model but it is computationally very heavy.

Reference

1. https://amaarora.github.io/2020/08/30/gempool.html
2. https://arxiv.org/pdf/1902.05509v2.pdf
3. https://arxiv.org/pdf/1707.01629v2.pdf
4. https://www.kaggle.com/aniket286/mobilenetv2-google-landmark-howard