National Park GeoGuesser

CSC 487

By: Damond Li, Dustin Tran, Tyler McCue

Project Description:

The purpose of our project is to develop a deep learning algorithm that can identify national parks based on corresponding images. To control the scale of this project, it will be a multiclass classifier with 8 chosen national parks to identify from. The 8 national parks are:

- Bryce Canyon National Park
- Grand Canyon National Park
- Yellowstone National Park
- Yosemite National Park
- Glacier National Park
- Everglades National Park
- Crater Lake National Park
- Hawaii Volcanoes National Park

Data Collection:

We were unable to find any existing databases that contain labeled images corresponding to their respective national park, so we decided to generate our own data. Google Maps API offers a monthly limit of $200.00 free credits which we can use to process street view requests. Each request costs $0.007 which leaves us approximately 28,000 requests. For our training, validation, and testing data, we decided to do a 700/150/150 split (1000 images for each class).

To collect our data, we wrote an API class in python that generates an encoded URL that specifies the coordinates, image size, and radius of search for each API request. For each national park, a random coordinate within the park region will be passed in. We then check the meta data for that one specified coordinate to see if a corresponding street view image exists and if we have previously collected that image as part of our dataset. This process is then repeated until we have gathered sufficient data.

As we were collecting our data, we found that we often ran out of images to collect for the specified region, so we would shift our coordinates around to gather more images from other nearby regions. This is important to note because if we do not randomly split our training, validation, and testing data, we could be introducing biases into our model (e.g. majority of testing data comes from a different region within the same national park compared to those in the training data).

Project Design:

The project was built using a Convolutional Neural Network and implemented using TensorFlow in Python. Two approaches were taken, a pre trained CNN known as VGG16, and a model created from scratch with a similar architecture to the VGG networks. The pretrained VGG16 network which can be seen in figure 1 below classifies 1000 objects, from this an additional classifier layer was added to then determine what national park the picture was. This allowed for higher accuracies with lower training times.

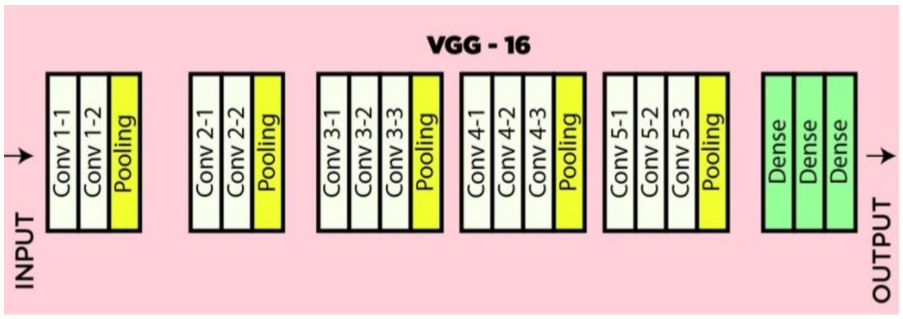

Figure 1. VGG16 Network Architecture

The model from scratch was modeled after the VGG networks where each base consisted of two convolutional layers followed by a pooling layer. At the end the layers were flattened, and two dense layers were added before the final classification layer. The final network architecture can be found below in figure 2.

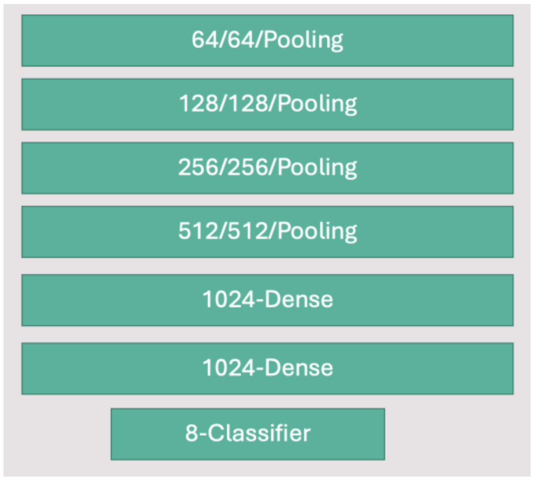

Figure 2. Model from Scratch

The final model from scratch had approximately 39 million trainable parameters and took around 12 minutes for each epoch to train on a dedicated GPU. This created a bottleneck as training easily took around 24 hours, thus slowing iteration time.

Testing:

Testing started with just classifying 3 national parks to speed up iteration time with the first model having an accuracy of around 90%. However, as the number of classes increased the accuracy decreased. The highest accuracy we could get for our from-scratch network was 75%. The highest accuracy we could achieve with our pretrained convolution base was 78%. Since our data consists of mostly street view images, many of the pictures have a similar composition between different national parks.

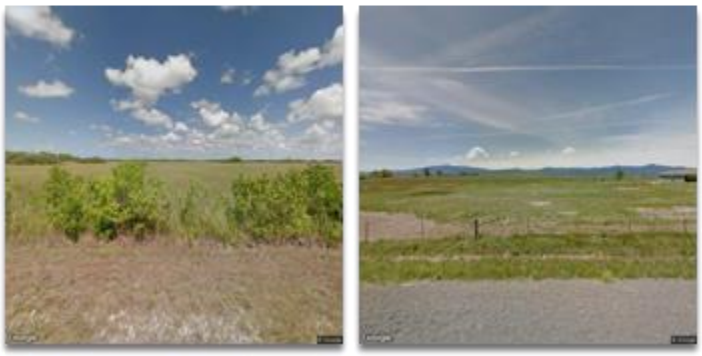
Figure 3&4. Everglades and Crater Lake Examples

For example, both images consist of a blue sky, scattered clouds, greenery, and a distinct horizon line. It would not be unimaginable if both images came from the same region only a few miles apart. However, the image on the left comes from Everglades National Park in Florida while the image on the right comes from Crater Lake National Park in Oregon.

Another edge case that showed similarities between two classes are the Grand Canyon and Bryce Canyon.

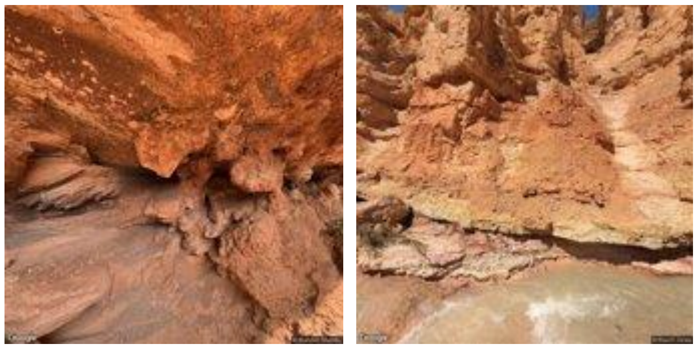
Figure 5&6. Grand Canyon and Bryce Canyon Examples

The left image comes from the Grand Canyon in Arizona and the right image comes from Bryce Canyon in Utah. Both images show rocky formations with similar color compositions.

During early testing of our model with only three classes (Yellowstone, Grand Canyon, and Crater Lake), we were able to achieve significantly higher accuracy of approximately 90%. This is likely due to how visually different the images were for these three classes. The Grand Canyon was rocky with a distinct red-orange color, Yellowstone had more trees/greenery, and Crater Lake has a huge blue lake in many of the images. The moment we add more national parks with similar features and characteristics, our accuracy will likely drop.

Analysis:

Although there were some difficulties achieving a reasonable accuracy with 8 national parks, the accuracy achieved from both the pretrained network and network from scratch both far surpassed the baseline accuracy of 12.5% or a 1/8 guess. Looking at the training data for the network from scratch, figure 7, the training accuracy starts to diverge from the validation accuracy at around epoch 35. While validation accuracy does increase slightly from there the returns are diminishing and disappear completely at around epoch 60 where validation loss starts to increase. Thus, adding drop out layers and increasing picture augmentation could be effective in creating a better generalized model.

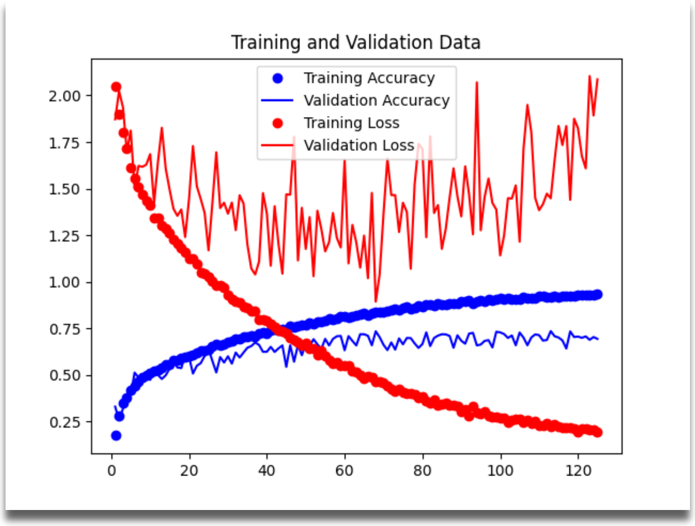

Figure 7. Model from Scratch Training Data

When compared to the pretrained networks training data, found below, not only is a slightly higher accuracy achieved but validation loss does not increase even though training occurred for twice as long. From this it could be reasoned that classifying objects within the image and then using that data to classify the region would be more effective. To further increase results the classifier could be plant specific or terrain specific. It would also stand to reason a larger network could better classify more national parks however this theory would require immense computational power.

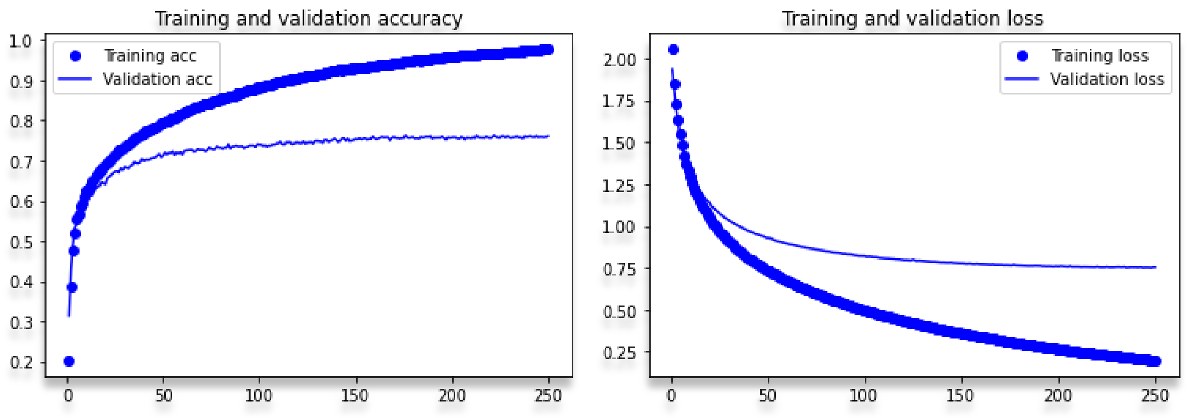

Figure 8&9. VGG16 Training Data

Since both approaches seem to converge around 75-78% accuracy, we think that the limiting factor is due to our dataset. Google Maps allows tourists and users to upload their own images which poses an issue for us. Not all images uploaded by users include the scenery of the national parks.

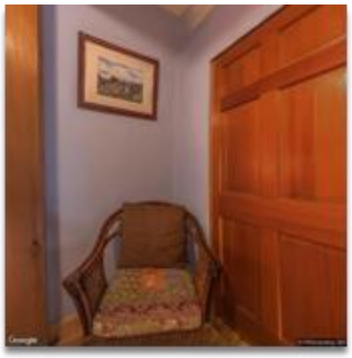
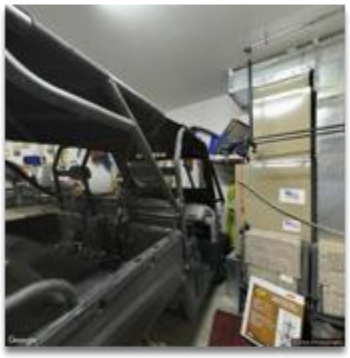
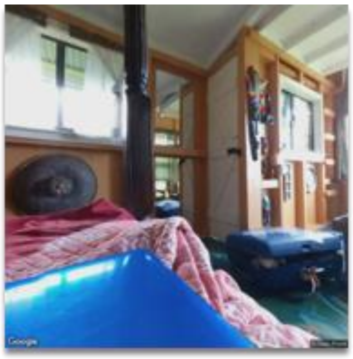

Figure 10, 11, & 12. Non-scenic Data

From left to right, their respective national parks are Glacier, Bryce Canyon, and Hawaii Volcanoes National Park. The images appear to show either a hotel room or a garage. Due to the timeline of this project, we could not implement anything to filter out these types of images from our dataset. We predict that doing so would increase our final accuracy.

Moving Forward:

To further the project, more data would need to be collected for not only new national parks but for each national park already trained upon. More advanced ways to classify would also need to be implemented to reduce confusion between two similar looking parks as talked about above. This neural network could also be expanded to include other classes such as cities or states.