Image Classification: CNN vs Random Forest

Shree Murthy

2024-02-19

Table of Contents

1. Problem Statement

The Intel Image Classification dataset contains thousands of images of natural scenery around the world. The images are divided into 6 categories: buildings, forest, glacier, mountain, sea, and street. The goal of this project is to build a model that can accurately classify these images into their respective categories.

Image classification is an ideal way to learn about CNN-based models and understand how it can outperform traditional machine learning models like Random Forest. Hence, this project will compare the performance of a CNN, specficially the VGG16 model, with a Random Forest Model.

While, the CNN model is expected to outperform the Random Forest model, we want to understand the extent to which it does so. Not every problem requires a complex CNN model, and understanding when it can be used effectively is vital. Nevertheless, based on my dataset, I expect the CNN model to be significantly better than the Random Foreset Model. Also, Deep Learning is required to properly classify the images in the dataset.

Deep learning is needed for this project because the data and relationships are complex. The pixels in the image are not independent features; thus, the pixels hold key spatial information. CNN models derive and understand the spatial relationships of images and understand generalized patterns such that they can classify new images perfectly. The Random Forest model, on the other hand, would not understand the spatial relationships because it requires the pixels to be flattened into a single row and then fed into the model. Essentially, every pixel would be treated as an independent feature, for the scope of a picture that is not ideal. However, I chose the Random Forest model because it is one of the better simple models when it comes to classification tasks. Random Forests use multiple decision trees to determine the best class for the input data. However, it is prone to overfitting when dealing with images because it focuses a lot on the training set and not generalize well to new data. This is because every pixel is independent so the model doesn't learn that a certain set of pixels would reveal a certain class with high probability. For instance, a set of 100 pixels that represent the sky may not be understood by the Random Forest because every pixel is an independent feature.

Therefore, using a Deep learning model is an ideal approach. This report will delve into the specifics of the models and their perfomances.

2. Data

The dataset contained roughly 24,000 images of natural scenery. However, I only used ~12,000 images for the scope of this project. I develop code on a virtual server used by other students. When copying over the files, I was unable to copy all the images. To copy the images I used the `scp` command. Nevertheless, ~12,000 images made for a solid sample set. The specific breakdowns of the train, test, and prediction sets are as follows:

- Train: 7335 images
- Test: 3000 images
- Prediction: 1785 images

After copying the data, I ran an `eda.py` file to visualize the classes and the images.

The first step was to visualize the train and test class distributions.

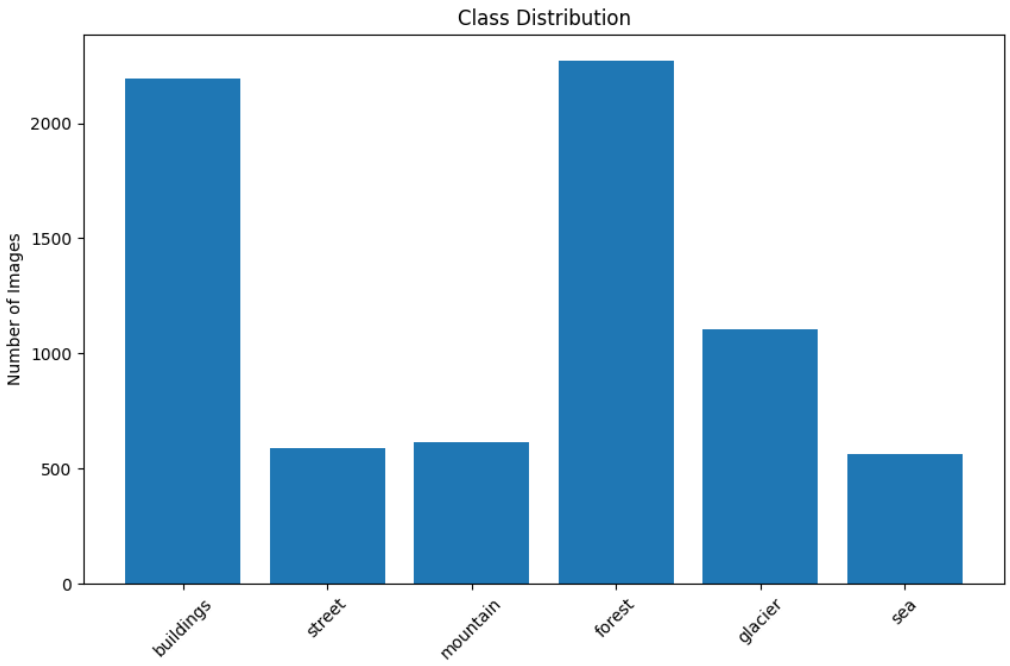

Figure 1: Train Class Distribution

This train class was imbalanced. The street, mountain, and sea classes had significantly fewer images than the other classes. The buildings and forest classes had the most images. This imbalance could lead to biased models. However, I chose not to address the imbalance issues because models need to be trained on a variety of sources. This imbalance could occur in the real world, albeit for a different situation, and the model must handle it and learn relationships. The test class distribution is as follows:

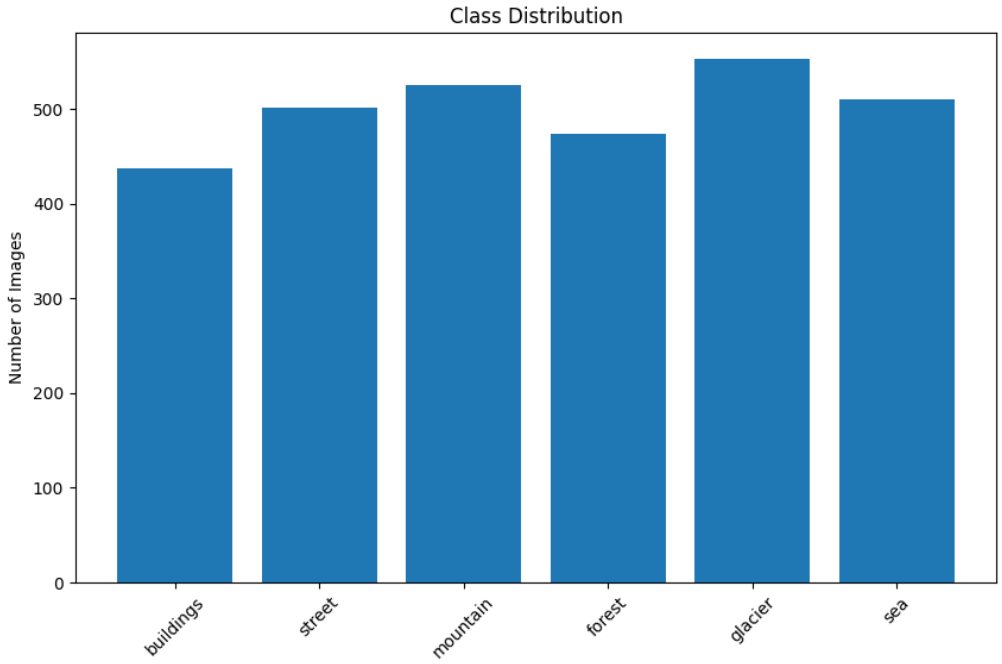

Figure 2: Test Class Distribution

The test class was not as imbalanced as the train class. The classes were more evenly distributed. While some classes had more than others, the difference was not large enough such that the model wouldn't have sufficient data to predict unseen data of a wide variety.

Next, I visualized the images of the train and test sets.

The images were of high quality and had similar sizes. Based on these sample images, the model should be able to understand where everything is located and not worry about areas being darker and unrecognizable.

The final step I took was to analyze the image sizes.

Both, the train and test had the same image size distribution; thus, I am only including the Train distribution. All the images were 150x150. While this is a good size to train with, I augmented the images to a 256x256 size. When scaling an image to a larger size, the pictures can become more unclear and tougher to analyze. This is done purposefully to ensure that the model is trained on grainer data and can handle unclear images.

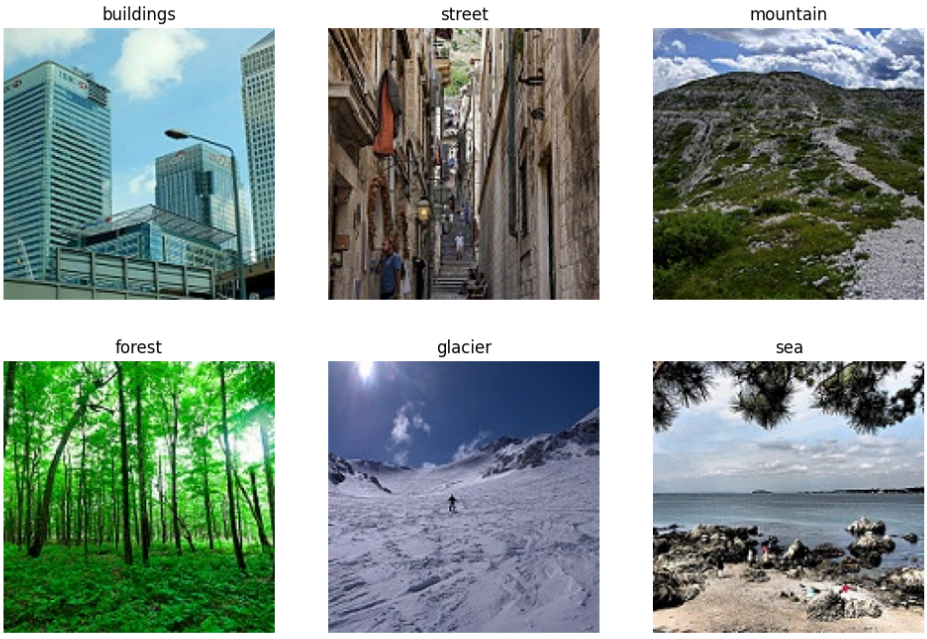

Figure 3: Train Images

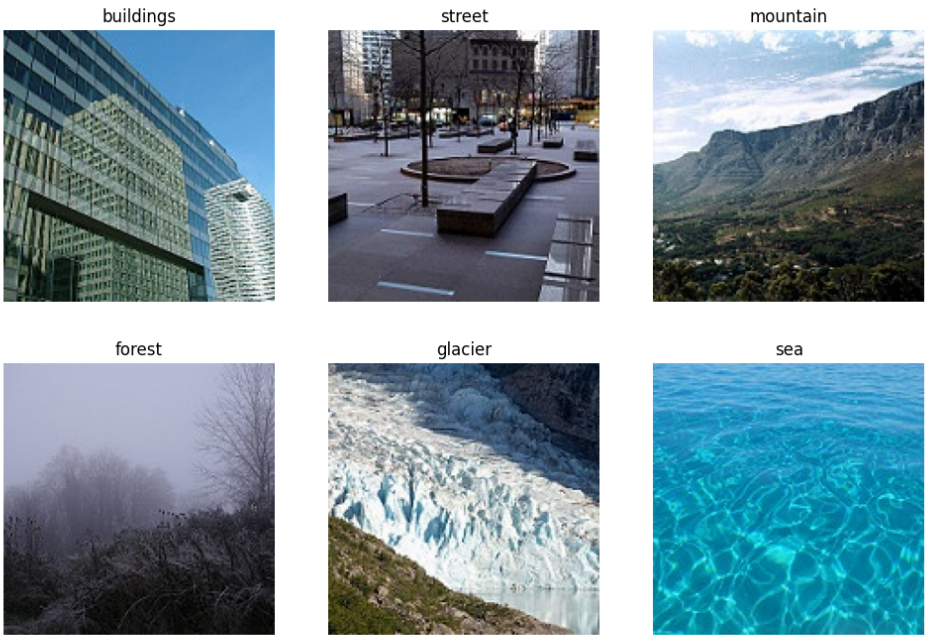

Figure 4: Test Images

3. Methods

3.1 Data Preprocessing

3.1.1 Random Forest Model

3.1.2 CNN Model

3.2 Model Building

3.2.1 Random Forest Model

3.2.2 CNN Model

4. Results

5. Discussion

6. Works Cited

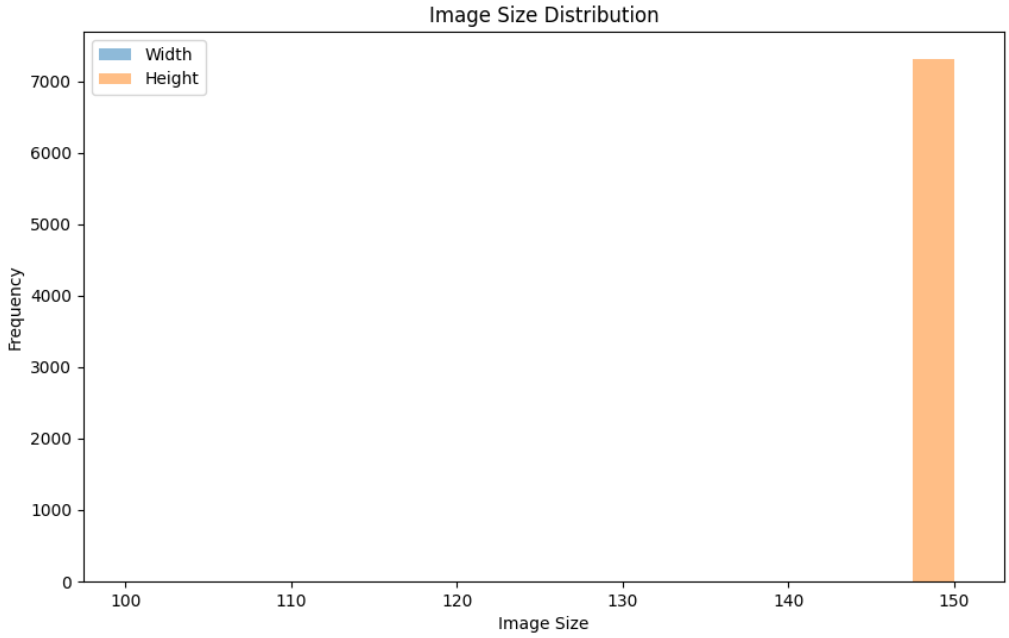

Figure 5: Train Image Sizes