Identifying Travel Stops from a Photo Album

Scott Ings (sings@stanford.edu)

Abstract

The goal of this project was to train a model that could accurately determine the presence of a specific landmark (for example, a specific tourist attraction) from a photo. The project focused on retraining and fine-tuning two existing models shown to be successful on the ImageNet classification problem: Inception v3[I] and MobileNet[II]. After some experimentation, the best MobileNet model obtained a test accuracy of 85.8% on a dataset with 118 landmarks. Analysis suggests that further efforts in increasing the training data set size, increasing image resolution, and further fine-tuning could lead to additional accuracy improvements.

Introduction

Consumer travel is a massive industry with a lot of opportunity for innovation. One major shortcoming with the current landscape is that a huge amount of valuable data about people's trips goes uncaptured. When planning a trip, people almost always look to source recommendations from friends and family who have already traveled to the same destination. The rise of things like the "Recommendations" feature on Facebook is evidence of this. However, there is no great way to capture and share these recommendations. Most people, after returning from a vacation, are unlikely to take the time to write a blog post, build a recommendations document, or write reviews on sites like TripAdvisor or Yelp. Indeed only 2% of TripAdvisor users write reviews. As a result, valuable data about what "people like you" liked and disliked about their travels is either shared informally or not at all.

This project seeks to improve on this problem by using a person's trip photo album to reverse engineer their travel itinerary. This takes the onus off of the traveler to document everything they did, and drastically reduces the friction in sharing valuable recommendation data. The scope of this project is to build a deep learning classification model across about 100 landmarks. This represents the order of magnitude of noteworthy top attractions within a single destination (e.g. a foreign city). The project is being scoped this way because within the context of the problem above, it is quite likely that we will be able to know the traveler's destination cities ahead of time (i.e. by making this a user input). This drastically reduces the complexity of the model which might otherwise need to correctly classify hundreds of thousands of landmarks across the world. Thus, this project attempts to demonstrate that a highly accurate model can be built for a single city, with the ultimate application building one such model for each city of interest.

Related Work

The broad area of "Computer Vision" is very popular and contains a large set of existing work. Much of the existing work that is most relevant to this project has been centered around the ImageNet dataset and yearly competitions.[III] These competitions focus on object identification, which has a lot of overlap with landmark detection. Many models over time have contributed to the latest understanding of image classification including AlexNet[IV], GoogLeNet[V], VGG[VI], ResNet[VII], and Inception v3[VIII]. In addition, the MobileNet[IX] model is an attempt to use these learnings to create a smaller-footprint model that can more feasibly be deployed in a mobile application. In addition to these object-detection models, Google has built a landmark detection model that they offer as part of their Cloud Vision API product[X]. Because it is a proprietary model, the details have not been shared publicly.

Data

Data Source

There are several interesting potential data sources for photos labeled with landmarks including through Google image requests, Instagram hashtags, and user-added photos on TripAdvisor. Conveniently, there is currently an active landmark identification Kaggle competition run by Google, where they have provided a somewhat curated dataset[XI]. In order to be able to save time on data collection and aggregation to get to the modeling component faster, I elected to use this dataset. It consists of images with a corresponding landmark label ID (note that the actual text name of the landmark was not provided, but that was ok for this experiment).

The dataset is quite large and beyond the scope of the project described above. To narrow it down from the 15,000 classes in the competition and sidestep issues with modeling classes with very small sample sizes, I elected to take the 118 landmarks with the highest number of labeled photos and use this subset of the data. In addition, due mostly to space constraints, I am starting by using about 25% of the available images for each landmark. The resulting dataset is about 140,000 labeled images. The raw images were a variety of sizes and resolutions, but as described below I experimented with different resolutions in trying to tradeoff between processing time and accuracy.

Data Cleansing

After downloading the images, I found that a small percentage of the images could not be read or processed. As a result, I had to flag and delete and images that seemed to be corrupt.

Data Splitting

Given the total resulting dataset of just over 140,000 images, I split training, dev, and test using a 70/15/15 distribution. This would leave me with 100,000 training samples, but still sizable dev and test sets to hopefully avoid overfitting either. The resulting dataset is summarized in Table 1 below. I also checked the distributions of each to confirm that they were indeed similar.

Table 1 – Summary of Project Dataset

	Train	Dev	Test
Images	100,978	21,640	21,642
Avg. Images per Landmark	856	183	183

Methods

Given the similarity with various other image classification applications, I chose to focus the project on using transfer learning from models that shown to be promising. In particular, I looked at two model architectures known from the ImageNet object detection competitions: Inception v3 and MobileNet. Inception v3 was chosen for its relatively high accuracy in ImageNet competitions, and MobileNet because it has a very small footprint and is designed for mobile applications which would be the ultimate use for this project.

Inception v3

The Inception v3 model architecture builds upon the original inception v1 GoogLeNet model, which introduced the implementation of multiple filter sizes at each layer in a convolutional neural net. Inception v3 comes from the same authors and refines their architecture based on additional learnings.

MobileNet

MobileNet is a Google project with the goal of enabling advanced modeling application on resource-constrained devices like mobile phones. The architecture is based on "depthwise separable convolutions" which is a way of separating standard convolutional filters into multiple components to improve performance.

Transfer Learning

For each of the above architectures, I used pre-trained model weights as a starting point. I then removed the final (top) model layer and replaced it with a new fully connected and then softmax layer. During the training process all model layers were locked with their already-trained weights except for this final layer.

Experiments, Results and Discussion

Experiment Setup

Given the problem definition, the key metric focused on was classification accuracy. We want to be able to correctly classify as many images as possible with the landmark they contain.

In addition to comparing the two model architectures described above, other key hyperparameters to investigate included: the learning rate, the mini-batch size, resolution of the input images, the number of model layers to retrain, and the size of the training set (while not strictly a "hyperparameter," it was important to understand how the size of the training set might impact accuracy).

Results

The best model results obtained, as measured by test accuracy, were for a full-sized MobileNet model with a 256-pixel image resolution input, a batch size of 100, learning rate of 0.01 and the full training set of 100k images. This model achieved a test accuracy of 85.8% with a dev set accuracy of 88.7% and a training accuracy of 99.0%. Figure 1 shows the accuracy of this model over 10k learning steps.

Figure 1 – Accuracy by Iteration for the MobileNet Model

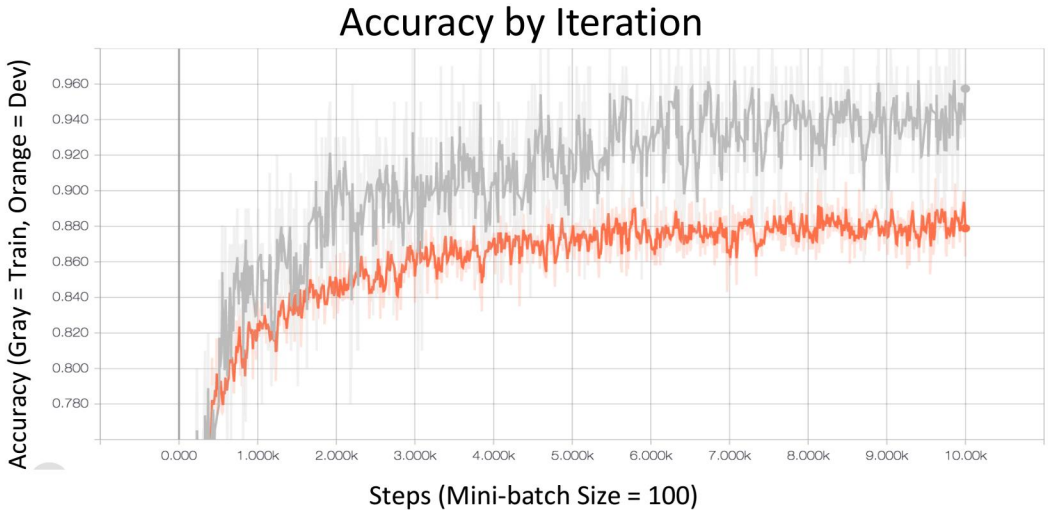
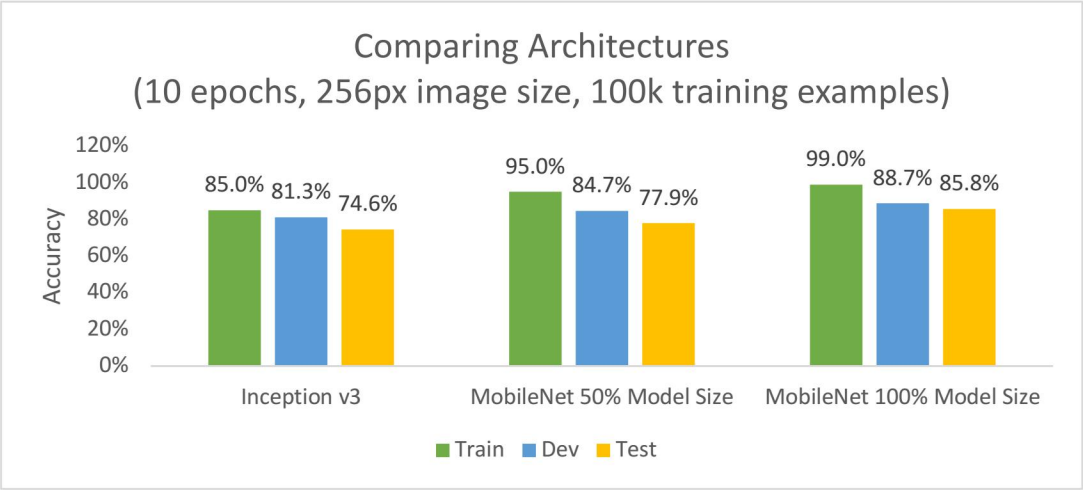


Figure 2 shows a comparison between the different model architectures with other hyperparameters held constant. Somewhat surprisingly, the simpler MobileNet architecture outperformed the more complex Inception v3 architecture. While the reason for this isn't immediately clear, it may actually be that the Inception model is overly complex for this problem—which has a smaller set of classes than the ImageNet problem—and therefore may be more likely to over fit.

Figure 2 – Results for Different Architectures



Figures 3 and 4 show the effect of two other hyperparameters: the image resolution and the size of the training set. Moving from a small 64-pixel resolution to 256 pixels had a considerable impact on the accuracy and suggests that it would be worth exploring even higher resolutions. Training on only 20k randomly selected images only slightly reduced the accuracy of the MobileNet model, suggesting that gathering additional data would likely be helpful, but may not have a very large impact.

Figure 3 – Accuracy by Image Resolution

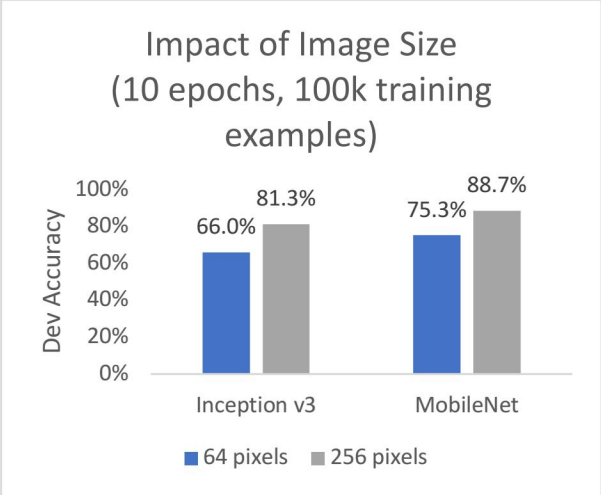
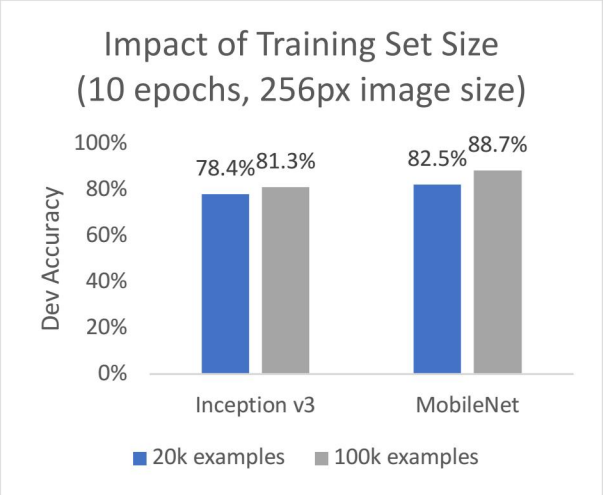


Figure 4 – Accuracy by Training Set Size



In addition, I experimented with the batch size and learning rate. Manipulating batch size showed almost no effect on accuracy, and so all models shown used a batch size of 100, which seemed to work slightly better than others. Experimenting with different learning rates showed that a learning rate of 0.01 was most effective, and this learning rate was used throughout the experiment results above.

Error Analysis

Given the motivation for this project, a test accuracy of 86% actually seems quite good. This could also be further improved by context currently outside of the model. For example, knowing from timestamps that two pictures were taken by the same person within 2 minutes of each other would likely allow us to make educated guesses and fill in some gaps in the model. However, it's still important to try to

understand what the model is getting wrong and whether there are any systematic biases we could try to correct for.

Figure 5 – Test Accuracy by Ground Truth Label

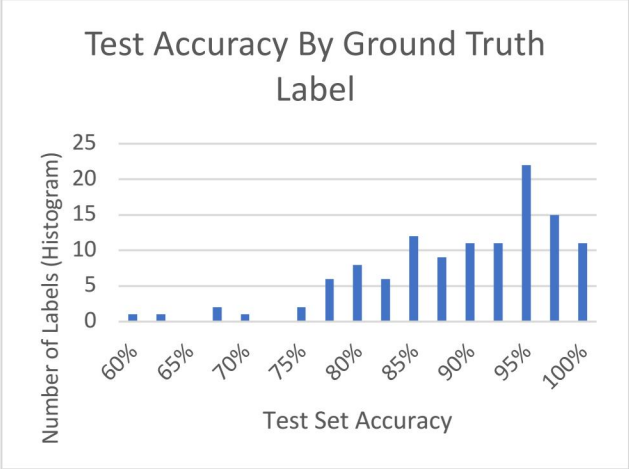
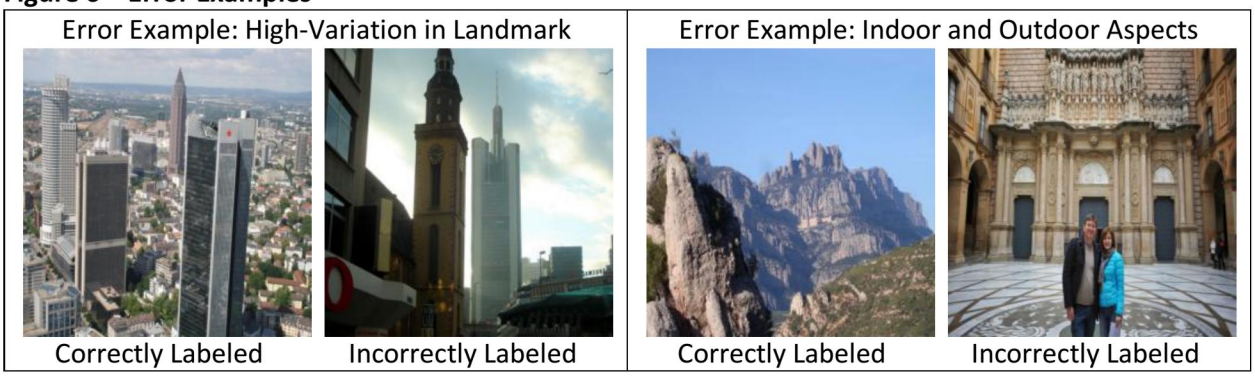


Figure 5 shows the distribution of test accuracy across the ground truth labels. In other words, it is looking to identify any landmarks that we are particularly good or particularly bad at classifying (this is essentially an aggregation of a confusion matrix across one dimension given the high number of classes). The chart shows that the vast majority of landmarks had an accuracy of at least 80%, but that a small set of landmarks were much less accurate.

Looking at those landmarks in more detail is informative for understanding why the model is having trouble. Figure 6 shows examples of the two landmarks with the lowest accuracies. The left example shows two pictures from the same landmark demonstrating that this particular city skyline has many different components and is likely hard to consistently classify as a result. The example on the right shows Montserrat in Spain which is a mountain landscape from a distance with a monastery on top. The "indoor" and "outdoor" aspects of this landmark are very different, which would be very difficult for a model like this to capture.

Figure 6 – Error Examples



Conclusion and Future Work

The retrained MobileNet model was relatively successful in accurately identifying a landmark in a given set of images. For the desired application, a test accuracy of 86% seems quite good. In addition, the analysis suggests there may be room for additional improvements by increasing the input image resolution and training with larger data sets (perhaps also by using random image distortions).

Given more time on this project, it would also be interesting to experiment with retraining additional layers of the MobileNet and Inception models as well as experimenting with different architectures. Given some of the error analysis, it may also be fruitful to explore whether separate labels should be used for the interior and exterior of certain landmarks. Finally, it could be very powerful to bring other image metadata, such as geo-tags and timestamp, into the process.