Formatting Instructions For NeurIPS 2023

Guillermo García Pérez **Austin Teng** **Tony Tian** **Michael Zhong**

Abstract

GeoGuessr is a popular online game built on top of the Google Street View database. The game presents players with a random street view image and challenges them to guess the location in the form of coordinates or country names. Players recognize and interpret features relevant to geographic environments in the photo, such as vegetation, architectural styles, road markings, and signage. This project focuses on winning the game by developing a convolutional neural network (CNN) model that can accurately predict a country of origin of a photo.

First introduced by LuCun et al. (1998), CNNs have already been proven successful in image classification tasks. CNNs particularly are suited for this kind of task due to their ability to pick up on intricate patterns in visual data, as seen in numerous past studies such as AlexNet. We hope that our model will learn relevant features of the image's environment similar to a human player. As a result, we build our model from ResNet due to its effectiveness in identifying individual objects, plants, and animals from ImageNet. Our model's capability of identifying countries could potentially extend to various practical applications, such as improving the accuracy of location-based services in technology and social media, and enhancing data collection methodologies in geography-related research.

1 Building our Dataset

Our dataset was originally found containing 50,000 images labeled amongst 124 countries. The images were 1536×662 RGB and included segments of the GeoGuessr UI. A few notable flaws caused us to modify this dataset.

First and most importantly, the dataset is incredibly unbalanced. The United States accounts for roughly 12,000 or 25% of the total images. 53 countries contain less than 100 images, far too few given how diverse countries typically are. We decided to perform stratified sampling on countries with greater than 1,000 images to ensure a balanced 11,000 image dataset. Our threshold of 1,000 was selected due us believing that 11 potential labels was an appropriate amount. Secondly, 1536×662 was too large. All images were downsized to $224 \times 224 \times 3$. RGB was maintained as color could be significant for identifying things outside.

We considered a few other datasets as well that we ultimately did not choose for various reasons. Muninn Dataset provides 15k images across the world labeled by longitude and latitude coordinates. We believed coordinates would be difficult to learn as the potential output space is too large/varied and many identifying features of images would be difficult to associate with longitude and latitude. Additionally, 15k images is relatively small for the complexity of the problem.

City Street View provides 50,000 images taken from five US cities; this dataset was initially appealing by maintaining a small balanced set of potential image sets, but provided image labels as longitude and latitude coordinates. We believed coordinates, even within cities, would be

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

difficult to learn as it requires the model to have too intricate knowledge of specific city layouts to give accurate predictions.

2 Creating our Model

Dealing with a dataset comprising 50,000 images across 124 countries has posed significant hardware challenges. The sheer scale of this dataset exceeded the memory capacity of our operating system, leading to frequent system crashes. Even when attempting to load the images in batches, we encountered persistent OS, memory, and other unexpected errors.To address these issues, we opted to preprocess the data by selecting 1000 images from each country with more than 1000 images.

After some experiments, we settled on a pre-trained ResNet-18 model and fine-tune its parameters. This approach yielded an approximately 60% accuracy across the 11 classes. We did not attempt larger ResNet models such as ResNet-152 because ResNet-18 is smaller and computationally less intensive. Furthermore, the rapid decrease in Train loss and increase in Validation loss after epoch 1 suggested that ResNet-18 is potentially sufficiently complex.

3 Model Card

- Description
  - Based on ResNet-18
  - Cross Entropy Loss Function: We decided on cross-entropy as our loss function because of its ability to handle multi-class classification problems
  - Adam Optimizer: We used the Adam optimizer because it is efficient for large datasets as well as complex models such as ResNet-18
  - Hyperparameter tuning: We used trial and error to determine the best set of hyperparameters for our use case.
- Usage
  - Our model is intended to identify geographic locations depicted in photographs through the use of visual cues and learned patterns from a global dataset. Primary users will be researchers and geographic analysts involved in location based services or educational tools.
- Factors
  - The model's performance may vary across different landscapes, weather conditions, and urban vs rural settings. The model has been evaluated primarily on the diversity of geographic locations, spanning 11 countries across all the continents.
- Metrics
  - Training Accuracy: 100
  - Validation Accuracy: 61.53
  - Test Accuracy: 58.62
  - Test Precision: 0.592
  - Test Recall: 0.587
- Ethical Considerations:
  The dataset consists of publicly-shared images that are geo-tagged and were collected with the respect to privacy and data usage rights. Identifying information like license plates or house numbers were blurred out.

4 Dataset

Our dataset is around 50,000 JPEG images from the GeoGuessr game collected from the Kaggle website. The images were scraped by a Python script. These images are street-view images of random locations around the world. It is divided up between 124 countries, but the distribution is not even. There is a significant imbalance with the number of images per country. This dataset can be

used to test machine learning techniques, such as neural nets, in geolocation tasks with multiclass classification of countries. It is linked here.

Some preprocessing was needed before this dataset could be used in our model. We had to size images down from 1536×662 RGB to 224×224 RGB to help with computational time due to our hardware limitations. Additionally, we set a threshold on the number of images from a country to ensure we have enough data points for a class. Thus, our training dataset consists of 11,000 images from 11 different countries of size 224×224 RGB.

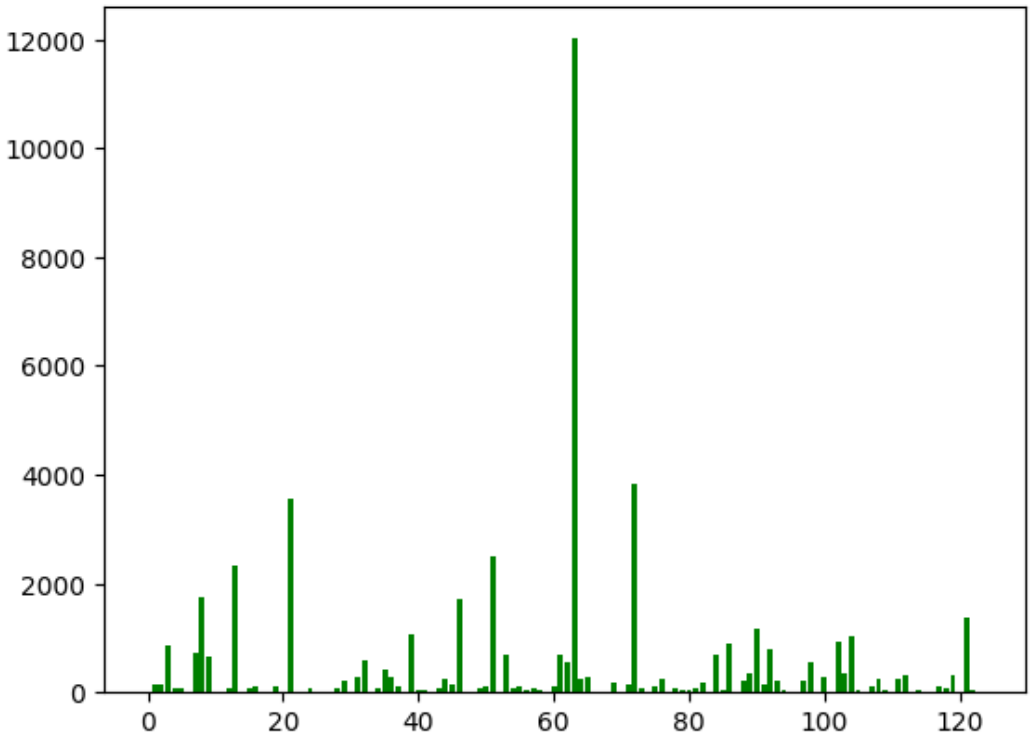


Figure 1: Amount of samples (on Y axis) per countries (mapped to integers on X axis)

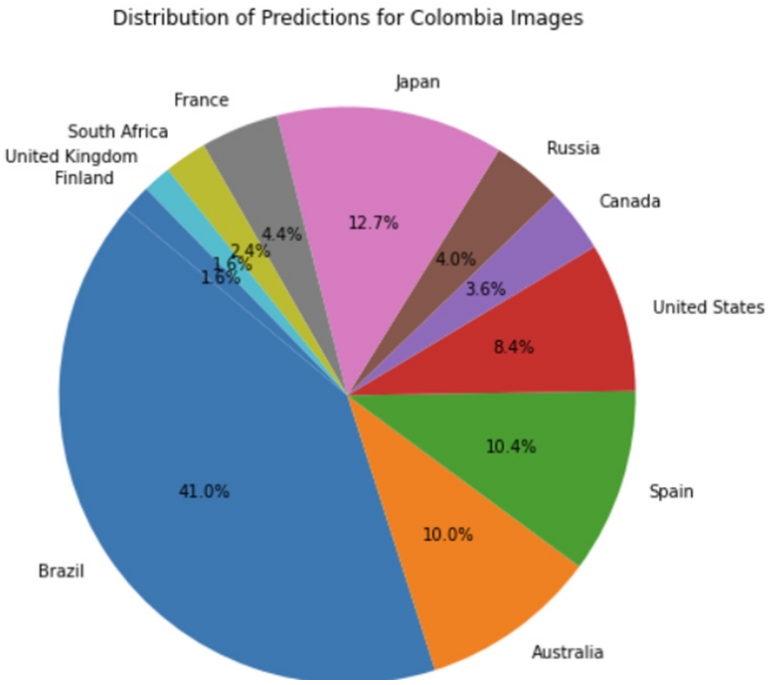
5 First Experiment

Initially, we attempted to copy the AlexNet architecture given in the lecture slide because we want to see how this groundbreaking discovery in the early days performs on this difficult problem. Our hypothesis was that AlexNet will still be able to learn something out of the dataset but not as well as the ResNet model which we settled on eventually. We were quite astounded to observe how incompetent AlexNet was, achieving a meager 9.09% accuracy after several epochs. Looking at its predictions we discovered that it was consistently predicting the same class for all tests, and since our train, validation, and test data were perfectly balanced across all 11 classes, $\frac{1}{11} = 9.09\%$, it was indisputable that the AlexNet struggled to learn anything. We believe that this result is likely due to a lack of pretraining, thus learning the significance of transfer learning and how effective ResNet is.

6 Second Experiment

With a test accuracy of around 60%, our model does not seem to perform well. We hypothesize our model is not general enough and designed an experiment to test this by feeding it images from countries outside our training set and looking at the similarity rates. Ideally, countries with similar geographic features, climates, landscapes, and urban architectural styles should be classified together. For example, most images from Colombia should be classified most like Brazil as opposed to

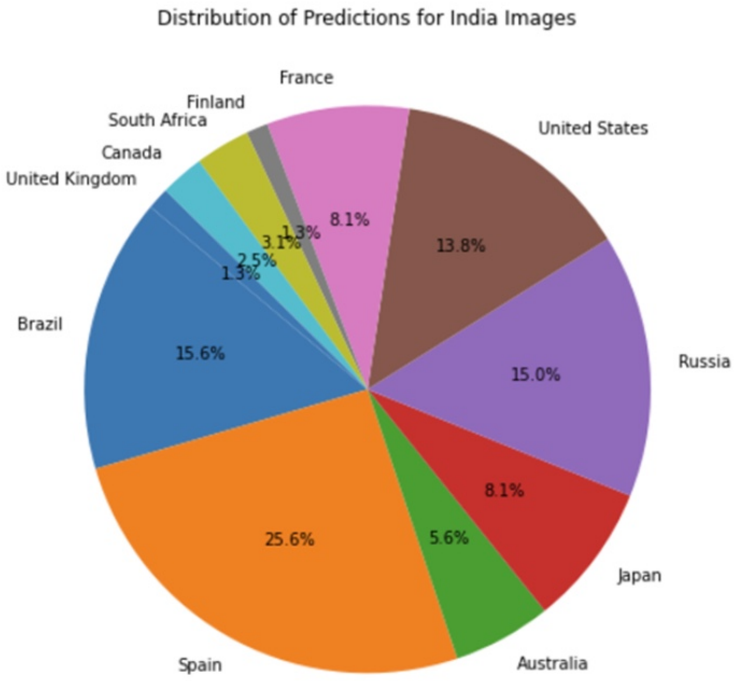
Russia. This is because both Colombia and Brazil share similar characteristics such as being part of the Amazon basin, having extensive coastal areas, exhibiting a blend of colonial and indigenous architecture influences, and reflecting modern architecture in bigger cities. The model then classifies the Colombian images as one of the eleven countries. Looking at a distribution of these images, we can then tell how similar Colombia is to one of the eleven countries.



The model thinks the majority of Colombian images are Brazilian. There are still a wide range of images classified as being from countries other than Brazil suggesting that our model is not general enough. Thus, we can not definitively conclude whether our model is general or enough. It does a decent job at correctly identifying similarities, but only in this specific curated case that we have built.

7 Third Experiment

Using the same structure, we decided to test our model on a very diverse country. We hypothesize that our model should struggle in the case of India because its learned patterns cannot effectively generalize to that region's complexity. India has a vast array of geographic features as well as distinct contrasts between urban and rural areas. It does not have closely correlated countries because of this. Here are the results:



The spread in the model's predictions could be indicative of its attempt to match features common in other, more familiar geographic locales within its training dataset to those seen in the Indian context. For instance, the model might confuse arid regions in India with similar landscapes in other parts of the world, or mistake urban high-density areas with other global cities if key distinguishing features are not adequately learned during training.

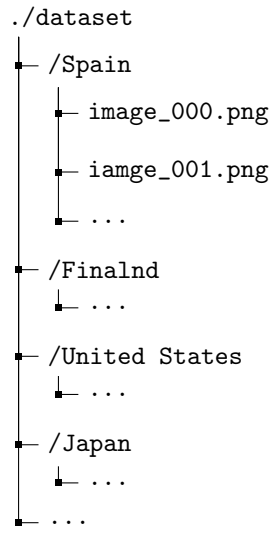
8 Fifth Experiment

In this experiment we tried using data augmentation to correct for the problem of an unbalanced data set. Data sets that have many samples for a certain class but few for others are quite common, so we believe developing techniques to counter the effects these unbalanced can have on a model during training is important.

The idea behind our experiment was using data augmentation on the classes that had less than a threshold amount of samples to increase the model's performance. Our goal was to obtain a model that did well on images from all of the different classes. In order to do this we decided to first focus on the problem of creating a test set and after focus on training different models and comparing their accuracy.

When creating a test set we had to deal with two main problems. The first one was the size of the data set. We realized it would not be efficient to store in memory all test images in tensors while testing. What we did instead was create a dictionary that had as keys different classes and as values a list with the names of the image files corresponding to that class. When testing we would fetch n images and test on them, then fetch the next n images and repeat this until there were no more images to fetch. This solved the memory problem but we still had to ask ourselves how we were going to choose the images to add to our test set.

Our data set was stored in our filesystem the following way:



We had to choose images from these folders such that the test set was representative of the data set. For example, the United States class had four times the amount of samples than the Japan class, we thought a test set should showcase this. What we did is to choose the first 10% images from each folder to form our data set. This allowed us to have a test set representative of our data set and ensure there was at least one image per class in it. The test set was 10% of our data set, 4946 images.

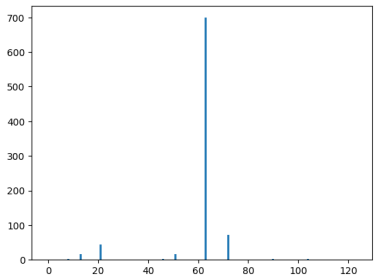
Once we had done this we decided to train ResNet in three different ways and compare the results. One way was by fetching n images (avoiding those in the test set) from the data set in proportion to the amount of images from each country. We would randomly choose a country from which to choose an image and repeat this n times. The random choice was made taking in mind a set of probabilities, these were calculated by dividing the number of samples a class has by the total number of samples of the data set.

Another of our models was trained similarly to this last method but slightly changing the probabilities such that the difference between them was not as big as previously and adding data augmentation to thos classes that had less than 500 samples. The third model we trained was the same as the last except that we completely flattened the probability distribution.The chances of choosing country A were always the same as the chances of choosing country B.

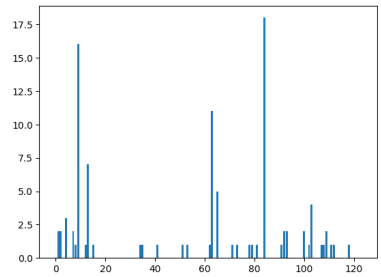
The results we got from these models were not great. In part we attribute this to the hardware limitations we had, running ResNet on a laptop without GPU access was not ideal. The results from the first model were as expected, it did ok predicting the United States, but for the countries that didn't have a lot of samples it never got them right. We believe that this model would be biased to choosing the countries with most samples even if we had had more training time.

We were expecting the results from the third model to not be great as well, and indeed they weren't. This model did worse for countries with a lot of images since we trained for the same amount of time yet and the model received less amount of images for those countries per epoch: in total it trained on less images for us then before. We did observe how it did better for countries with fewer samples, however the results were still pretty negligible. Even though we have not had time to test this model after training for long our hypothesis would be that for the classes with very little images it would not generalize great: it would have been trained on small variations of the same image over and over again.

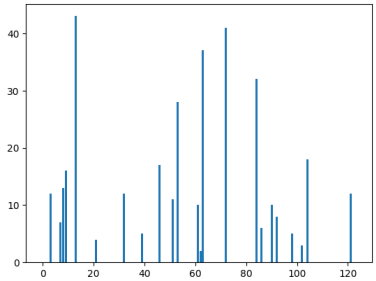
We were expecting the results from the second model to be the best, and they were although they were still pretty underwhelming. The model did better for countries with more images, as to be expected since it is training on more images. It also did better for countries with less images than the previous model. The results can still be improved significantly since the accuracy was 7%.



(a) First Model



(b) Second Model



(c) Third Model

Figure 2: Amount of correct guesses (Y axis) per country (X axis)

9 Sixth Experiment

For this experiment we decided to see how the previous models generalized on a data set containing images from GTA V. Since the world in GTA V strongly resembles Los Angeles we thought that our models should predict the class United States for all images.

The results for our first model mentioned above, the one that was biased toward choosing United States, got 100% accuracy for this GTA V data set. This was not a surprise at all. The second model we trained, the one that had a more flattened probability distribution, got an accuracy slightly above 70%. We believe this was a good result keeping in mind the poor accuracy this model had on the GeoGuesser data set and also the fewer amount of images this model had been trained on of United States. The third model, that trained on an near equal amount of images from all countries, got 0% accuracy. We believe this was also not a surprise, since this model did not train on many United States images it wouldn't do well on this experiment. For reasons we don't know, this model guessed Albania for most of the GTA V images.

10 Contributions

- Michael Zhong: My main contributions were around the two experiments of testing how general our model is. I went through the entire design process, from initial hypothesis, testing, and conclusions, by myself. I also did the correlated parts of the write-up and presentation for the experiment. Other contributions include the datasheet and model card parts of the write-up
- Tony Tian: Data preprocessing and model selection/experimentation, including testing of the AlexNet performance, image resize and loading, bootstrapping the data while splitting it to train and test sets, and setting up the test $\&$ train functions.
- Austin Teng: Implementing various architectures to facilitate the data processing processes
- Guillermo García Pérez: trained and tested the models of experiments 4 and 5