CNN Plays Geoguessr: Transfer Learning on ResNet50 for Classifying Street View Images

Finn Dayton, Jeffrey Heo, Eric Werner

Introduction

Geoguessr is an online game where the player tries to guess the geographic location of a street-view image. Points are awarded proportional to the precision of the guess, so a close guess earns more points than a far-off guess. For this project, we sought to create a computer vision (CV) model to play a modified version of Geoguessr where, instead of guessing the coordinates of a given street view image, the model would learn to predict the country of the location. Our project tackled a multi-class image classification task where the input to our algorithm is a 224 x 224 RGB screenshot of a geoguessr question.

We used a convolutional neural network (CNN) to take the image input and output a predicted country label. More specifically, we used ResNet-50 CNN pre-trained on the ImageNet dataset and performed feature extraction and fine tuning using our data set to improve performance. Unlike the actual Geoguessr objective, which would be more closely modeled as a regression task of pinpointing exact geographic coordinates, our project classifies countries based on underlying traits of a country, such as climate, foliage, natural landscapes, street signs and road markings. Street views of heavily-urbanized and population-dense Korea, for example, will differ significantly with that of Dutch cities built around canals. The prospect of a neural network learning the nuances of street views of different countries inspired us to build this project.

Related Work

Literature studies were focused mainly on three themes: 1. Transfer learning, 2. Transfer learning applications using CNNs, and 3. Geolocation estimation.

In “Inadequately Pre-trained Models are Better Feature Extractors (1) the research seeks to understand the relationship between the extent in which the pre-trained model had fitted to ImageNet and performance in transfer learning. One weakness was that the datasets used for transfer learning - Stanford Cars dataset for example - did not contain data that differed significantly with that of ImageNet. This research is similar to our work as it utilizes ResNet-50 pre-trained on ImageNet. The difference is that our research is focused on the application of the model while the cited research is an empirical study.

Another related paper, “Transfer Learning with ResNet-50 for Malaria Cell-Image Classification,” (2) utilized a ResNet-50 model pre-trained on ImageNet for the binary classification task of classifying cells infected versus uninfected by malaria. The research replaced the final fully-connected (fc) layer with softmax activation with a fc layer with sigmoid activation. A weakness in this research is that the accuracy plots indicate that the model performance has plateaued after the first epoch. Nevertheless, the research does not seek to overcome this training plateau issue through tuning the learning rate.

“Classification of COVID-19 X-ray images using a combination of deep and handcrafted features,” (3) also deploys pre-trained CNNs as feature extractors. This research, given a chest X-ray image, concatenates hand-picked features to features extracted by pre-trained CNNs and feeds the feature vector to an SVM classifier that classifies the lungs as infected with COVID-19 versus pneumonia versus uninfected. Notable differences from our current research are 1.) generation of hand-picked features to supplement extracted features and 2.) SVM unit as output layer.

“Geolocation estimation of photos using a hierarchical model and scene classification,” (4) approaches the geo-localization task seeking to output the location of a scene depicted in an input image as a classification task where the earth is subdivided into cells that represent geographic regions. The primary difference in approach between this research and our project lies in the objectives of each research: for the prior research, the main goal is to estimate the physical location depicted by the scenery while for our research, the goal is to estimate the country. Moreover, the prior research utilizes pre-trained

ResNets to not only extract feature labels for classifying the input images' geographic region cells but also for classifying the scene type (e.g. urban, suburban, natural, indoors, etc.), while our research utilizes ResNets to output features used by the final softmax layer.

Lastly, a project by students at the University of Colorado, Boulder, called "Geoguessr AI: Image Based Geo-Location"(5) used ResNet to predict bucketed locations inside the United States. Our project was different for two main reasons. First, we include other countries than the US. Second, while this project used the haversine distance for loss, we use a custom loss function that both combines cross entropy classification error with physical distance.

Dataset and Features

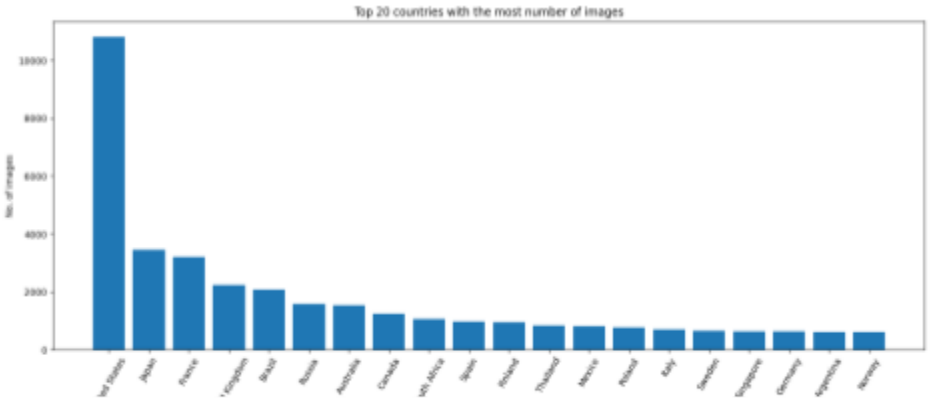

Figure 1: Data size for Top 20 Countries

The dataset that we used is the "Geolocation - Geoguessr Images (50K)" dataset on Kaggle (6). The dataset is organized into 124 sub-directories representing distinct country labels, totalling 50,000 images. Given many countries only had a dozen or so images, we cut the dataset down to the top 20 countries with the most images and designed a multi-class model to learn on and classify these 20 countries. After pruning the bottom 104 countries we were left with 39,441 images. The dataset was further split into train, validation and test directories with a size ratio of 90:5:5, or 35451:1983:2006. See Figure 1 for a distribution of the images in the train set by country. Before loading the data into ResNet50, we resized each image to 224 x 224 x 3, which ResNet50 is designed to accept . Figure 1 shows three examples of the resized images.

Figure 2: Three Sample Images

Normalization and data augmentation were further implemented using the transforms feature in the torchvision library. The input images were normalized with mean = [0.485, 0.456, 0.406] and std = [0.229, 0.224, 0.225] across the rgb channels, the mean/std of ImageNet, which were the normalizations applied to the training data of the pre-trained ResNet. Data augmentation was only applied in the training

data to prevent overfitting. We wanted the model to be evaluated by its performance with real-life images during validation and testing. For data augmentation, we used random resized cropping, random horizontal flipping, and random rotation of maximum 5 degrees were applied to augment the training set.

Lastly, we conducted a basic human-level benchmarking test, showing 50 images from our dataset of the top 20 countries to three players. The average human performance was only around 7%, barely better than random (5%).

Methods

(i) Overview - Transfer Learning

We used transfer learning to develop our country classifier. Transfer learning refers to using knowledge gained in a certain problem to facilitate learning for a distinct but related problem. Transfer learning through using large CNNs pre-trained on the ImageNet dataset either for fine tuning during training or as a feature extractor for subsequent trainable layers has become a standard practice in CV. In CNNs, the parameters learned by the model are filter weights where, as we go deeper into the multi-layered network architecture, the model learns increasingly complex filters. Early layer filters typically learn how to detect basic features in objects such as edges, corners, and other lines. Intermediate filters learn how to detect discernible features in objects; that is, if the model seeks to recognize faces, these filters would learn detecting eyes, nose, etc. Therefore, early and intermediate layers trained on the large ImageNet dataset composed of 1000 class objects end up learning detection techniques that are highly transferable across a vast array of CV tasks.

Transfer learning most often uses two main forms: 1. fine-tuning and 2. feature extraction. The former approach typically allows the gradient to flow to some later layers of the pre-trained model to tune the weights in these layers to perform well on the new dataset, while the latter approach freezes the weights of all the layers in the pre-trained model and except for a randomly initialized final layer. In essence, feature extraction on a pre-trained model extracts a feature vector representation of the input image to feed to the final output layer, the only layer the model is training.

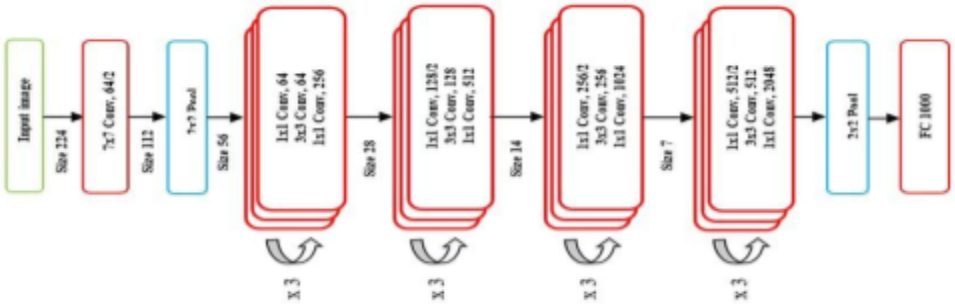

Figure 3: Diagram of ResNet50 (7)

(ii) Feature Extraction

Our first training method was performing feature extraction on ResNet50. ResNet50 is a residual network that incorporates a “deep residual learning framework,” characterized by additional connections called “short-cuts” between layers that allows activations from the prior layers to propagate directly to the subsequent connected layers. Residual frameworks enable the use of larger and deeper neural networks advantageous for fitting more complex data without needing to face the vanishing gradient problem inherent in deeper networks. Short-cuts are implemented through identity matrices; thus, gradients would simply be multiplied by 1 and passed directly downstream to earlier layers. See Figure 3 for a diagram of ResNet50. For feature extraction, we removed the final fc layer, which classifies 1000 classes. We

replaced this layer with a softmax layer with a fc softmax layer with 20 units to classify the countries. All other weights in the model were frozen.

(iii) Custom Loss Function

While previous projects have focused on either a categorical penalization (cross entropy loss) or a distance penalty (Haversine distance), we combined these two losses into a novel loss function. The reasoning for this is neighboring countries tend to have similar street views than geographically isolated countries due to similar climate and architecture. Thus, the custom loss function would penalize predicting Singapore more heavily than Canada when the ground-truth is the US, for instance.

$$\text{loss} = L_{\text{CE}} + L_{\text{Haversine}}$$

$$= \sum_{i=1}^{n}(-\sum_{k=1}^{K} y_k^{(i)} \log \hat{y_k^{(i)}}) + \alpha \sum_{i=1}^{n} \text{dist}_{\text{Haversine}}(\hat{y}^{(i)} - y^{(i)})$$

Cross entropy is a common loss function for classification problems for data of multinomial distribution. Haversine distance is the geographic distance between the true label and the prediction.

(iv) Fine Tuning

For fine tuning, we enabled the gradient to flow back to later layers in ResNet50. As its name suggests, ResNet50 contains 50 layers. Of these, two are input and output layers, while 48 are convolutional layers. These 48 convolutional layers are split into four blocks called Cfg[0], Cfg[1], Cfg[2], Cfg[3]. Since earlier layers in Resnet50 learn to extract features likely to be shared between the ImageNet dataset and our dataset, while later layers learn more complex features likely able to help differentiate between countries, we choose to only learn the weights of the last block, Cfg[3]. This block has 9 layers.

Results

Our model achieved **40.3% accuracy** on the test set from feature extraction and **72.5% accuracy** on the test set from fine tuning.

We trained our model on the training data for 10 epochs and ran validation a total of 30 different times, where, for each of the 30 iterations, we sampled each of our four hyperparameters randomly. The four hyperparameters we sought to optimize were: batch size (for mini-batch gradient descent) learning rate, scalar multiplier for custom loss and lambda (weight decay). Weight decay, also called L2 regularization, was employed to avoid overfitting to the training data. We followed a random search strategy for these four hyperparameters, as discussed in “Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS” (8) Randomly sampling each of the four hyperparameters independently ensured that within the 30 iterations, we saw 30 different batch sizes at most, 7 different learning rates at most, 30 different alpha values at most, and 6 different lambdas at most. Having four distinct hyperparameters made the hyperparameter grid search method too slow. The optimal hyper parameters discovered were batch size = 180, learning rate = .001, weight decay = .0001 and alpha = 2.5. Once these hyperparameters were found, we trained for 40 epochs.

The train and validation losses for feature extraction and fine tuning are shown in Figure 4.

Figure 4: Training/Validation accuracy vs. Epoch
Left: Feature Extraction Right: Fine Tuning

Figure 5: Confusion Matrix on Test Set. Y axis is true, X axis is predictions.

As can be seen, fine tuning achieves much higher train and validation accuracy than feature extraction.

Figure 5 shows the confusion matrix of the fine-tuned model being run on the test set.

The fine-tuned model was able to generalize to the test set remarkably well, showing it is possible to learn distinct features of countries–something human players struggle to do–with a CNN.

Conclusion

In our research, we utilized a ResNet50 CNN pre-trained on ImageNet as a feature extractor for a softmax classifier that classifies a scenery image into one of 20 country labels. Through fine tuning the last 9 layers of ResNet50 to our dataset, in conjunction with randomly sampling four distinct hyperparameters, we have built a model that achieves over 70% test accuracy, which is almost a 10-fold improvement in our estimated human performance of 7%.

In the future, we would explore creating a large, balanced dataset with all 124 countries in the original dataset possessing 12,000 images through web-scraping. This will ensure that whichever model we train, we will yield an unbiased classifier with greater probability. Moreover, we hope to explore using VGG and AlexNet, other popular image CNNs, to better predict unseen countries.

References

1. Andong Deng, Xingjian Li, Zhibing Li, Di Hu, Chengzhong Xu, Dejing Dou. “Inadequately Pre-trained Models are Better Feature Extractors” March 2022. https://arxiv.org/abs/2203.04668

2. Arrabelly, Sai Bharadwaj Reddy & Juliet, Sujitha. (2019). Transfer Learning with ResNet-50 for Malaria Cell-Image Classification. 0945-0949. 10.1109/ICCSP.2019.8697909.

3. Weihan Zhang, Bryan Pogorelsky, Mark Loveland, Trevor Wolf. Classification of COVID-19 X-ray Images Using a Combination of Deep and Handcrafted Features. Jan 2021. https://arxiv.org/abs/2101.07866

4. Eric Muller-Budack, Kader Pustu-Iren, Ralph Ewerth. “Geolocation Estimation of Photos using a Hierarchical Model and Scene Classification”, 2018, ECCV

5. GeoGuessr AI https://nirvan66.github.io/geoguessr.html

6. Rohan K. 2021. GeoLocation - Geoguessr Images (50K)

7. https://www.kaggle.com/datasets/ubitquitin/geolocation-geoguessr-images-50k

8. Ridha Ilyas Bendjillali , Mohammed Beladgham , Khaled Merit , Abdelmalik Taleb-Ahmed “Illumination-robust face recognition based on deep convolutional neural networks architectures” May 2020, Indonesian Journal of Electrical Engineering and Computer Science

9. Petro Liashchynskyi, Pavlo Liashchynskyi. “Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS” 2019. https://arxiv.org/abs/1912.06059

Contributions

Finn Dayton
- Training and test code for feature extraction.
- Data augmentation. confusion matrix visualization
- Write up editing

Jeffrey Heo
- code: dataset visualization, train/validation/test split, hyperparameter random sampling, model evaluation, confusion matrix visualization
- write-up: introduction, related work, dataset and features, methods, conclusion

Eric Werner
- GCP + collab management + spin up
- Data visualization
- Fine tuning code