CNN Plays Geoguessr: Photo Geolocation with Convolutional Neural Networks

Remy Zazo
remy.zazo@mail.utoronto.ca

Shaarif Ali Syed
shaarifali.syed@mail.utoronto.ca

Asser Ismail
asser.ismail@mail.utoronto.ca

Ishav Sohal
ishav.sohal@mail.utoronto.ca

Abstract

Several machine learning models have been developed to predict locations from an input image the same way a user would play a round of GeoGuessr. Many of these models, like DeepGeo, treat the task as a classification one, rather than a regression task and thus produce predictions with limited utility. The regression task is much more complicated than the classification one, as various zones from vastly different corners of the world share common attributes, such as weather, architecture, and nature. The proposed model aims to learn the minute differences and narrow down the precise location the input image is taken at. It is built on an EVA-02 backbone, with a prediction head for continent classification as well as a gated head for coordinate regression. This leads to a best of both worlds approach, as the classification head guides the coordinate regression by limiting it to a specific zone (and out of Oceans and unrelated geographical regions). The aforementioned model is trained on a custom dataset comprising over 38000 images, extracted from 12600 locations derived from the GeoGuessr map "An Equitable, Stochastic, Populated World". The success of the architecture is evaluated using a mean haversine distance between the predicted coordinates and the true label of the image.

1 Introduction

GeoGuessr is a geography game in which players try to guess locations from a Google Street View environment. The game has been rampant in popularity, and content produced on platforms such as YouTube has garnered global interest. Top players are able to immediately and precisely locate the environment, whereas novices struggle and predict very distant locations. The skill gap makes this task a very interesting one to develop a machine learning model for, especially as the quality of the model is perfectly represented by the haversine distance between the prediction and actual location.

Piggybacking off of this popularity, many machine learning models have been developed, attempting to simulate the experience of professional players whilst also being effective enough to generalize to imagery not previously seen on the GeoGuessr platform. Models such as PIGEON successfully do so but require immense pools of training data that is expensive to access, whilst other less popular models such as DeepGeo are unable to predict locations on a worldwide scale, nor generate a continuous output (coordinates). The proposed model attempts to strike a balance between modest model complexity and dataset size, whilst generating a precise, continuous output better suited to the game of GeoGuessr.

Geolocation is a very difficult problem due to various factors such as similar vegetation/architecture across different regions, ambiguous rural locations, and the common absence of critical metadata such

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

as road signs and license plates. Such factors tend to be the Achilles heel of coordinate regression models due to the loss being minimized by taking "averages" of far coordinates whose scenery resemble each other. Deep learning is the perfect tool for the aforementioned task. Its ability to learn intricate, multi-scale features like architecture, vegetation and subtle metadata accurately parallels the skills required by a professional GeoGuessr player.

The input of the model is a 448x448 image pulled from Google Street View, and the output is a set of spherical coordinates representing the prediction. This blends the ease of interpretability of continent classification with coordinate regression. The proposed model aims to combat this by including the classification task within the coordinate regression one.

2 Background and Related Work

GeoGuessr is a browser-based geography game in which players are provided with Google Street View imagery and must deduce where they are in the world. Players have to utilize a variety of features in the street view images they are provided, such as road signs, kinds of vehicles, landmarks, building design, weather patterns, vegetation, and much more.

A related paper is called "PlaNet - Photo Geolocation with Convolutional Neural Networks", by Tobias Weyand, Ilya Kostrikov and James Philbin. In this paper, the authors created a deep neural network that performed classification, called PlaNet. This model was trained on millions of geo-tagged images, and the surface of the earth was subdivided into thousands of multi-scale geographical cells to represent the output classes. So, the input to the neural network was an image of some location on Earth, and the output was a probability distribution over the surface of the Earth. The paper found that "PlaNet far outperforms other methods for geolocation of generic photos and even reaches superhuman performance." Weyand et al. [2016].

Another related paper is called "DeepGeo: Photo Localization with Deep Neural Network", by Sudharshan Suresh, Nathaniel Chodosh, Montiel Abello. In this paper, a "deep neural network based on the ResNet architecture [was] trained" on a training set of "0.5 million Google Street View images" Suresh et al. [2018] from the United States. The input to the model "contains four images taken at the same location, oriented in the cardinal directions." Suresh et al. [2018] This way, the "panoramic viewpoints that [were] provided in GeoGuessr" Suresh et al. [2018] were fully utilized. The output classes were the 50 states of the USA. As a result, "this model achieves an accuracy 20 times better than chance on a test dataset, which rises to 71.87% when taking the best of top-5 guesses. The network also beats human subjects in 4 out of 5 rounds of GeoGuessr." Suresh et al. [2018]

Another related paper is called "CNN Plays Geoguessr: Transfer Learning on ResNet50 for Classifying Street View Images", by Finn Dayton, Jeffrey Heo, Eric Werner. In this paper, a "ResNet-50 CNN pre-trained on the ImageNet dataset" Dayton et al. [2022] was utilized, along with "feature extraction and fine tuning using ... data set to improve performance." The "Geolocation - Geoguessr Images (50K)" Dayton et al. [2022] from Kaggle was used. The output classes were 20 of the countries with the most image data in the Kaggle dataset. As a result, the model "achieves over 70% test accuracy, which is almost a 10-fold improvement in ... estimated human performance of 7%. " Dayton et al. [2022]

Unlike PlaNet, our input data is restricted to Google Street View images, and unlike DeepGeo, it is not restricted to a particular country. Our output is a set of spherical coordinates, rather than just a country. That is, we implemented a continuous coordinate regression model, which is more challenging than a country regression model, but yields more precise predictions. Rather than using ResNet-50, we decided to use a pretrained EVA-02 model as our backbone.

3 Data

We are obtaining the images by querying a good balance of locations from the KML file from the Google Street-View Static API. This dataset consists of three images per location/pin, with images being taken at angles of 0, 120 and 240. This was done to familiarize the model with the location rather than just a single view (which may be obstructed or have few hints). The name of each image file is of the form "lat_lon_angle.png". There are a total of 38407 images. The dataset was split up such that 70% of the images are in the training set, 20% of the images are in the test set, and 10% of images are in the validation set. We transform the dataset using the following mechanism:

- We perform colour jitter to adjust the brightness, contrast, saturation and hue of the images
- We perform random horizontal flips on the images
- We convert the images to tensors
- We normalize the images
- We perform random erasing on the images

After splitting the images into the train/validation/test sets, we load these data sets into CSV files (one for each dataset). Each row in a CSV file consists of the data for an individual image file. It includes the image file name, the latitude, longitude, x position, y position, z position, and the continent id. The latitude and longitude is obtained directly from the image file's name. The continent id is obtained by first using the `reverse_geocoder` library, which gives us the country code for the pair of lat/lon coordinates, and then using the `pycountry_convert` library to convert from the country code to the corresponding continent id. The x/y/z coordinates are obtained by converting the lat/lon coordinates to radians, and then setting:

- `x = cos(lat_rad) * cos(lon_rad)`
- `y = cos(lat_rad) * sin(lon_rad)`
- `z = sin(lat_rad)`

Once the CSV for each dataset is created, we parse these CSV files into a dictionary, where the keys are lat/lon coordinates (separated by underscore), and the values are the corresponding rows from the CSV. Since there are three images per lat/lon coordinate, each key of this dictionary corresponds to three rows of the CSV, stored as a list. We then load the dictionary data for each of the data sets into `DataLoaders`, for which we set:

- `batch_size` to 32
- `shuffle` to true (which means that the dataset is shuffled before being organized into batches)
- `drop_last` to true (which means that the DataLoader drops the final batch if its incomplete)

4 Model Architecture

The proposed model architecture is a Gated Coordinate Regression model designed to localize images to accurate spherical coordinates. To address various challenges related to coordinate predictions falling into invalid regions (e.g., oceans), the task is split into two stages: Continent Classification and Continent-specific Coordinate Regression.

4.1 Backbone and Input

The backbone chosen for the proposed model is EVA-02. This is used as the core feature extractor as it has been trained on the ImageNet-22k dataset and is familiar with a substantial number of features from therein. The pre-trained classification head is removed to access the final hidden state.

- Input: The model processes a stack of three 448×448 images (representing different viewing angles at the same coordinate) to generate a robust feature representation and circumvent bad starting angles.
- Output: The backbone outputs a 1024-dimensional vector (final hidden state) which serves as the input for the tasks heads to be discussed below.

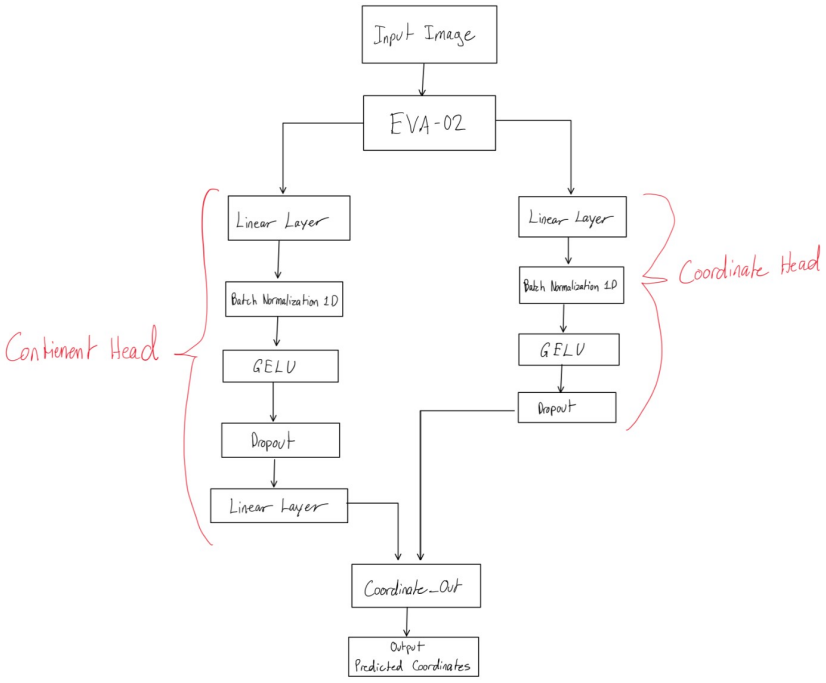

Figure 1: Model Architecture Diagram

4.2 Task Heads

The feature vector outputted by the backbone is fed into two distinct heads:

1. Continent Classification Head: This head predicts the probability distribution over 7 continent classes. It consists of a Linear lay, followed by BatchNorm1D, a GeLU activation, a Dropout layer, and a final Linear Projection to 7 logits.
2. Gated Regression Head: This head predicts specific coordinates. It processes the features through a Linear layer, BatchNorm1D, ReLu activation, and Dropout. The final output layer projects the 1024-dimensional output from the backbone model into 7 sets of spherical coordinates (x, y, z), one set for each of the 7 continent classes.

4.3 Gated Inference Mechanism

A key feature of the architecture is the gating mechanism used to ensure consistent outputs.

- Inference: The model first predicts the most likely continent using the Classification Head. It then selects the (x, y, z) coordinate set corresponding only to that predicted continent from the Regression head, discarding the predictions for the other 6 continents.
- Training: A teacher forcing approach is used, where the regression head corresponding to the ground truth continent is selected for loss calculation. This ensures that the model learns to predict coordinates accurately within the correct continent without being penalized for predictions in irrelevant, far away regions.

4.4 Training and Optimization

4.4.1 Loss Function

The network is optimized using a composite loss function combining classification accuracy and cosine similarity.

$$L = L_{clas} + \lambda L_{coord}$$

- Classification Loss: Computed using Cross Entropy Loss
- Coordinate Loss: Computed using an Angular Loss function based on cosine similarity. The similarity scores are clamped between $-1.0 + 1e^{-7}$ and $1.0 - 1e^{-1}$ for stability. The loss is derived by subtracting the mean cosine similarity from 1.

The hyperparameter λ scales the importance of coordinate regression. $\lambda = 400$ gave the best results, assigning tremendously higher importance to the precise geolocation over the overall continent classification.

4.4.2 Implementation Details

The model was trained for 15 epochs with a batch size of 32. The Adam Optimizer was used with a weight decay value of 0.0001. For stable convergence, different learning rates for different levels of the model were applied:

- Backbone: $1e^{-5}$
- Prediction heads: $3e^{-4}$

5 Results

The model described above has the following performance on the test set:

- Continent Accuracy: 96.1%
- Mean Squared Error: 0.0713
- Mean Km error: 683.37 km
- Median km error: 251.19 km
- Mean km error when the predicted continent is correct: 366.68 km

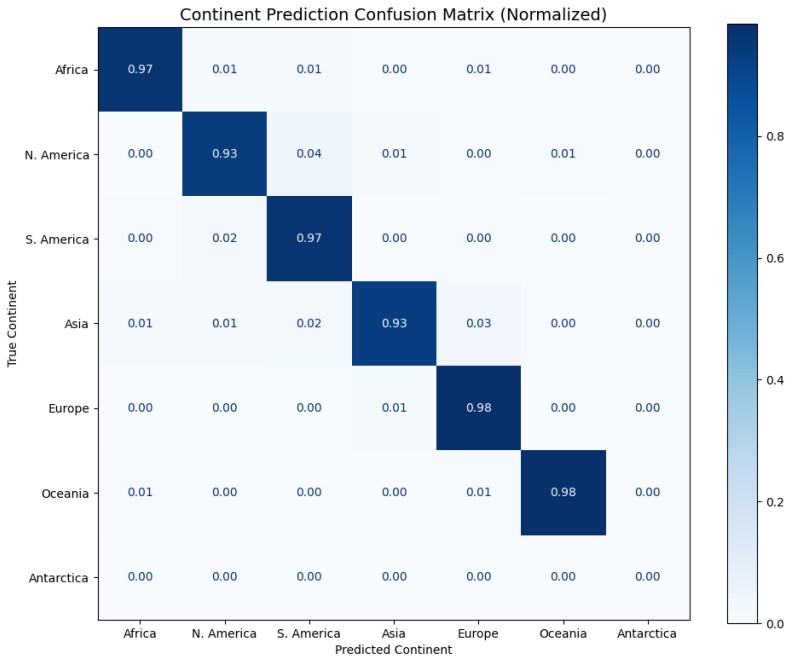

Figure 2: Confusion matrix of results from model evaluation

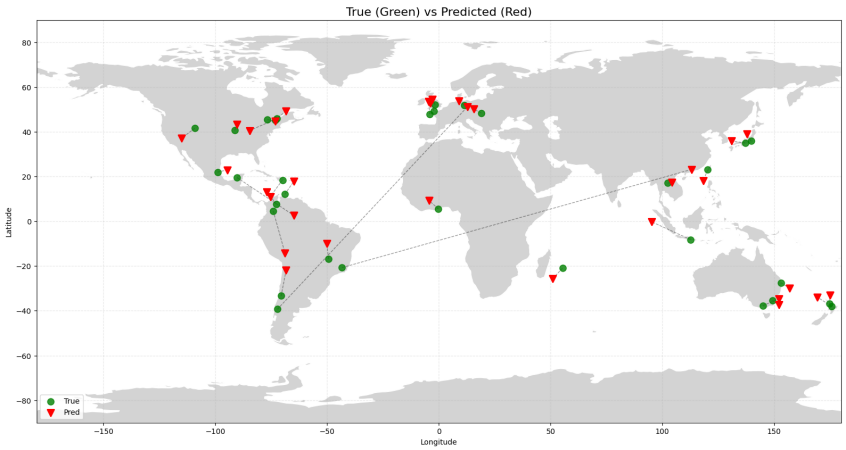

Figure 3: Map visualization of some predictions made during evaluation on the validation set

A baseline is the PlaNet model. This model has a median km error of 1131.7 km and a continent accuracy of 90.5% Weyand et al. [2016], whereas the proposed model has a median km error of 251.18 km and a continent accuracy of 96%.

Another model used as a baseline comparison is the Claude-Sonnet 3.5 model from Anthropic. The average score of Claude 3.5 for a single round game of Geoguessr is 2268.97 (the maximum score per round is 5000) Wei [2025]. The formula for obtaining a Geoguessr score from a guess on the world map is approximately:

$$score = 5000 * exp(-distance/1492.7) \tag{1}$$

where `distance` is the km error Haas et al. [2024]. So, using this formula, the average km error of Claude-Sonnet 3.5 for a round of Geoguessr is 1181.2 km. In comparison, the proposed model has an average km error of 683.36 km, and, using the above formula, an average per-round score of 3,170 points. Claude-Sonnet 3.5 has a km error that is about 42% higher than the proposed model, which goes to show just how effective our approach was. One factor to consider is that Claude-Sonnet is not specialized in photo geolocation, however, the sheer size difference between the models makes it a worthy comparison.

6 Discussion

6.1 Model Reflection

Our model performs very well when compared to the baseline models mentioned above. It has a very high accuracy for all 7 continents, as displayed by the confusion matrix above.

Our model performs exceptionally well for street view images based in Europe. Europe tends to have quite distinct and unique characteristics when compared to the rest of the world, which our model has effectively picked up on. These include unique architectural styles (medieval stone, stucco, brick patterns), road signage conventions (EU-standard shapes, fonts, colours), dense infrastructure and consistent road markings, distinct vegetation and temperate climate cues.

Additionally, our model performs very well for locations in Oceania. The street view images of Oceania in our dataset are dominated by locations in Australia and New Zealand. In these regions, there are very distinctive architectural styles, unique signage and road markings, characteristic vegetation, recognizable road surfaces and lane patterns, left-hand driving indicators, distinct climate and terrain cues (eucalyptus forests, red soil regions, tropical Queensland, etc.), and high-quality imagery and consistent camera systems.

The model's performance slightly degrades when attempting to make predictions for images in Central and South America. This is likely due to the visual similarity across large latitudinal ranges (similar fauna, foliage, housing colours), mixed infrastructure quality that makes rural areas in Central and South America appear indistinguishable, street view sparsity in many interior regions, and fewer distinctive signs (compared to Europe or Japan). This is a classic problem for global geolocation models, as Central and South America contain large stretches of visually ambiguous terrain.

Our model performs quite well in North America, as well. This is due to strong street view coverage, highly standardized road infrastructure (ex., yellow center-lines), stop signs, wide-lane roadways, and suburban architectural patterns that differ sharply from other continents (ex., detached single-family homes with large front lawns, asphalt shingle roofing, attached garages and driveways leading to the street, etc).

Our model performs very well in East Asia (Japan and South Korea in particular). This is due to extremely distinctive signage (kanji, katakana, etc.), very unique road markings (ex., "X" Crosswalk Markings, lane arrow styles with long thin shafts, inverted triangle as a stop sign rather than an octagon, etc.), urban density, consistent architectural cues (tile roofs, vending machines, shopfronts), and highly structured and uniform infrastructure.

Although the continent accuracy of Africa is quite high, at 97%, the model did have some trouble when predicting certain regions of Africa. This is likely due to sparse street view coverage outside South Africa, Senegal, Kenya, and Ghana, as well as rural areas with minimal signage, making roads look similar across thousands of kilometres.

6.2 Baseline comparison

The PlaNet model performs very well on landmark-rich, visually distinctive locations. When the input image contains an obvious landmark (e.g. the Eiffel Tower) or other strong, location-specific visual cues, PlaNet tends to assign a very confident distribution spike over the correct geocell. Additionally, PlaNet performs very well when there is a sufficient amount of consistent local/regional cues, such as architecture, vegetation, weather, road/terrain styles and signposts. PlaNet, much like our model, performs poorly on ambiguous, "generic" photos with few distinguishing cues. All that said, these performance behaviours are extremely similar to our model. This goes to show that there tend to be common, fundamental challenges with geolocation models.

7 Limitations

A main limitation of this approach is that many Street View images are too vague for precise geolocation. Scenes like forests, highways, and beaches look almost identical across continents, so the model has very few geographic cues to rely on. As a result, small changes in lighting or camera angle can lead to very different predictions. The two-stage design also makes mistakes expensive: if the continent is predicted incorrectly, the coordinate output will be off by thousands of kilometres. In addition, the dataset is slightly imbalanced, which causes the model to perform better in regions with more samples and worse in underrepresented areas.

Some possible improvements include providing the model with more than one Street View angle per location, since multiple viewpoints would give stronger visual clues. Another extension would be adding simple metadata such as the camera direction or time of day, which could help separate otherwise similar-looking scenes. Overall, these limitations show why the model performs well in some settings but struggles in others, highlighting the difficulty of predicting a location from a single image.

8 Ethical Considerations

The Street View images used in this project are publicly available and anonymized, but individuals appearing in them did not give explicit consent for machine learning use. The model is designed to focus on environmental features rather than personal characteristics since using human-related traits would raise serious fairness and privacy concerns. Some images include cultural or religious landmarks, so care must be taken to avoid drawing conclusions that stereotype or link identities

to specific regions. Geolocation models also introduce broader risks. They could be misused for surveillance or for tracking individuals if combined with other data sources. There are also geopolitical concerns if predictions are wrong in conflict zones or culturally sensitive areas. Dataset bias is another issue: Street View heavily overrepresents urban and wealthier countries, and camera quality varies, meaning the model performs better in some regions than others, raising fairness concerns. This model is intended only for research and educational purposes. It should not be used for surveillance, profiling, or any application that could identify or target individuals or reveal private locations.

9 Conclusion

Our findings highlight that a balanced architecture, which combines continent classification with gated coordinate regression, can provide meaningful advancements in the difficult task of global geolocation from a mere street view image. Although our model operated with a modest dataset compared to larger-scale industry models, our model was able to learn the minute differences and narrow down the precise location of the provided input images, as demonstrated by our continent accuracy of about 96% and our average km error of about 251 km (when the predicted region is correct). While ambiguous and visually repetitive environments still pose challenges, our model was able to consistently learn useful geographical cues and thus demonstrate the feasibility of continuous coordinate prediction.

References

Finn Dayton, Jeffrey Heo, and Eric Werner. Cnn plays geoguessr: Transfer learning on resnet50 for classifying street view images. Cs229 final project report, Stanford University, Department of Computer Science, 2022. URL `https://cs229.stanford.edu/`. Stanford CS229: Machine Learning Final Project.

Lukas Haas, Michal Skreta, Silas Alberti, and Chelsea Finn. Pigeon: Predicting image geolocations, 2024. URL `https://arxiv.org/abs/2307.05845`.

Sudharshan Suresh, Nathaniel Chodosh, and Montiel Abello. Deepgeo: Photo localization with deep neural network, 2018. URL `https://arxiv.org/abs/1810.03077`.

Jerry Wei. Claude plays geoguessr, 2025. URL `https://www.jerrywei.net/blog/claude-plays-geoguessr`.

Tobias Weyand, Ilya Kostrikov, and James Philbin. *PlaNet - Photo Geolocation with Convolutional Neural Networks*, page 37–55. Springer International Publishing, 2016. ISBN 9783319464848. doi: 10.1007/978-3-319-46484-8_3. URL `http://dx.doi.org/10.1007/978-3-319-46484-8_3`.