GeoGuessrAI: Image based GeoLocation - Midpoint Report

Nirvan S.P. Theethira
CSCI6502 001
Boulder, CO
nith5605@colorado.edu

ABSTRACT

Geolocation can be defined as the identification or estimation of real-world geographic location using location based data. The location based data used for geolocation includes but is not limited to satellite images, sensor data, traffic movement data, pollution data etc. Humans are not the best at guessing locations based on sensor data. But when it comes to location based image data, well travelled humans perform fairly well at guessing the location of where the image was taken. With google street view, it is now possible to obtain image data from most location in the United States. The game Geoguessr [6] presents street view data to human players who attempt to guess the location of the street view image. Using the recent advances in image recognition technology this project aims to emulated and hopefully surpass the scores of human players in the game Geoguessr for the United States map.

MOTIVATION

The geolocation of a real-world location is the latitude and longitude of that location. One specific application of geolocation would be inferring the location of a tracked animals based, for instance, on the time history of sunlight brightness or the water temperature and depth measured by an instrument attached to the animal. While this is a useful, real world application, the ability to just use images to identify a location would greatly help in a myriad of tasks. The ability to geolocate images can help in geotagging new unseen data thereby facilitation the creation of new dataset. Another application of this technology could be to aid in identifying location with low or no connectivity. A person who is lost in a low connectivity region could snap an image of the surroundings to get an estimate of their location.
The motivation for this project comes form the game called Geoguessr [6]. The game presents players with street view images (see Figure 1) which the user, using any pre-existing knowledge, has to guess the location of the image. The use of pre existing knowledge offers well travelled players an advantage in the game. Prior knowledge may include the use of language on street signs, motor vehicle type or in some instances the fact that certain regions (military bases) cannot be reached by google maps can help in elimination of those locations. The game scores the user based on how closed the latitude an longitude of the guess was to the actual latitude and longitude of the location of the street view image. While this is an extremely challenging task for a machine that has no prior knowledge that an avid traveller possess, the use of CNN's for other image recognition tasks have given astonishing results. While previous projects have aimed to geolocate prominent landmarks of locations such as the Eiffle tower, The Taj Mahal etc. this project aims to use generic images to extract locations. While humans can use prior knowledge as mentioned above, most humans who play the game use geographic features such as building styles, vegetation, mountains, weather etc to find locations. With the plethora of street view images available, a deep neural network can learn to notice these subtleties. As seen in Figure 1 on the left image, there are mountains and red stone buildings which can be used by a deep neural network to ascertain the image is from boulder.

RELATED WORK

There are three main approaches that have been researched:

- There are a few already existing approaches for the above mentioned task. One of the elementary approaches deals with the unsupervised k-NN algorithm [3]. It uses the clustering algorithm to cluster similar images. This idea is used to cluster images from a single location into a single cluster. A new image is run through the K-NN algorithm to find how close the image is to the centroids of the location cluster. The sofmax of these distances denotes how likely the image is from a particular location.
- There has been drastic improvements in the field of image recognition with the use of ResNet [4]. The next method therefore uses the CNN to geolocate images [8]. For the dataset, geotagged images scraped from Flickr was used in training and testing. One drawback of using Flickr images is that it results in indoor images and image of people that could clutter the dataset during training. The entire map is split into grids and images are collected from each grid. The model is trained to give a soft maxed output across the grids (see Figure 3).
- The third method takes into account that multiple images can be used in a single classification [8]. To process multiple images in one training epoch, the list of images are fed into CNN's that give the vector representations of the list of images. These vectors are then fed into a sequential LSTM and the output is soft maxed across regions from which data is collected (see Figure 3).

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

Figure 1. A comparison of a random location (Boulder,CO) on google street view to a famous location (Eiffel Tower) on google street view.

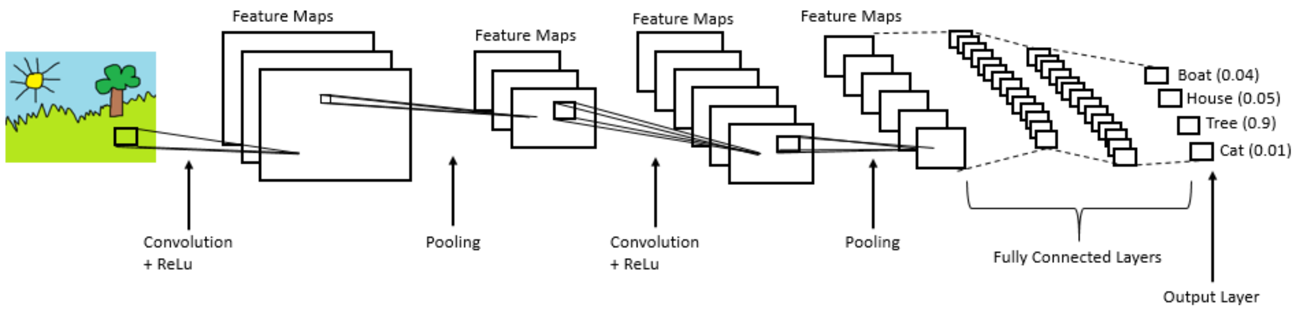

Figure 2. The CNN with soft maxed output across location grids.

PROPOSED SOLUTION

The proposed solution takes the following steps:

- The first step involves the map and defining geographic girds to collect data from. For this project the map of the United States will be chosen. This is because the country has a diverse geographic landscape with definitive features in each region. This can be leveraged by the CNN to give good results. Another reason for this choice is because a plethora of street view images are available for most regions in the country. Confining the area to just the united states will also make comprehensive training possible as using data from the entire planet will require a lot more computing power for training than is available for this project.

- The next step is to split the map into logical grids. The size of these grids will be a hyper-parameter to be tuned. A view of the US map split up into grids that are the size of states is show in Figure 3. All geometry related tasks will be handles using the python package called Shapely [2]. Data points will then be scraped from each grid with the number of data points being scraped being a hyper parameter.

- Data will be scraped from google street view. This will be done using the google street view API [7]. Data scraped from a single location will act as a data point. Since google offers a 360 degree street view, multiple images will be taken at each location to act as a single data point for training. This list of images will be processed sequentially using the LSTM technique mentioned above.

- Once the data collection pipeline is ready, training will be conducted using three machine learning techniques mentioned in the related work section. As the first technique involving K-NN's is unsupervised, an extremely large training set will not be needed. The computational power required will be low as there will be no training required for the unsupervised approach.

- The next technique involves using single images in training a CNN. After training, given a test image, the CNN will have to produce a soft maxed output across grids of locations Figure 3. The training of this network will be computationally heavy. To make training efficient the network will be trained in batches with images being scraped during training. Google Colab GPU's will most likely be used for training.

- The final solution involves using a CNN/LSTM combination. Here a single training epoch will consist of multiple images obtained from a single 360 degree sweep of a location. The number of images collected from a single location will be a hyper-parameter. The list of images collected from a single location are processed sequentially using the CNN/LSTM logic mentioned in the related work section. The output will be a softmax across the grid of locations across the map. To make training efficient the network will be trained in batches with images being scraped during training. Google Colab GPU's will most likely be used for training. **Note: As discussed over email, due to time constraints this method (CNN/LSTM) will not be implemented in this course project. It will instead be implemented in the CSCI 5922 Neural Networks course project. For this project, the previously discussed CNN method will be implemented along with the K-NN for baseline accuracy metrics.**

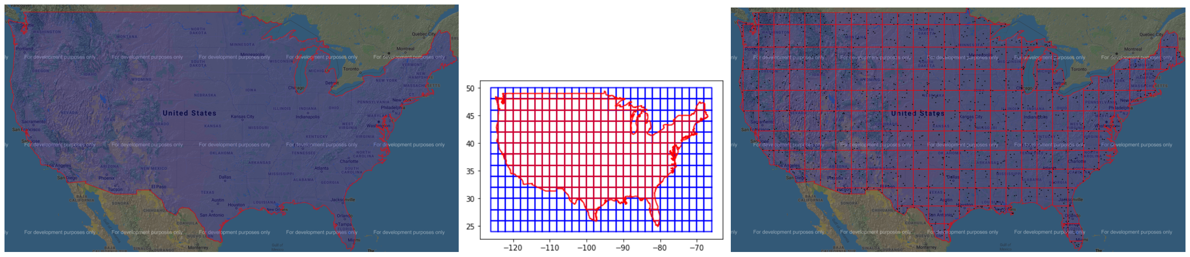

Figure 3. Left: Shape file of mainland US. Center: Fishnet used to split us into grids. Right: The map of mainland US split up into grids.

Figure 4. Sample images collected at grid 0 (Oregon).

DATA COLLECTION

As mentioned in the Proposed Solution section the following steps were taken to collect data points from across mainland USA:

- The first step involved finding a shape file of the US mainland boundaries. The most recent shape file of the US published by the United states Census bureau [1] was downloaded. The shape file contained boundaries of not just mainland USA but also boundaries of Alaska and all the islands and union territories part of the US. To focus data collection efforts on just the mainland the boundary of mainland US had to be isolated. To do this, all shapes in the shape file were converted into polygons. The polygon containing the central geographic coordinate of mainland was found to isolate the boundaries of mainland US (see left Figure 3). The geometrical operations for this task were done using Shapely [2]. Once the latitudes and longitudes of the mainland US boundary were isolated, the coordinates were saved into a pickle file.

- The next step involved splitting the mainland polygon into grids. To do this a fishnet or mesh of square boxes were first overlayed on the boundary polygon (see center Figure 3).The squares the intersected with the boundaries of the polygon were clipped at intersection points to split the polygon into grids (see right Figure 3). Each grid had a maximum area of four square units which translated to an area of 138.265 sq units with grids at borders being smaller. Grids that were too small (less that 0.1 times area of the largest grids) were combine with the neighbouring larger grid. This was done to avoid some grids having no data. The entire process resulted in the map of the US being split up into 243 grids. The geometrical operations for this task were done using Shapely [2].

- The final step involved image collection from multiple locations in a single grid. Image from 5 locations were collected per grid in a single sweep. The date the image was take was also stored along with the image. The google street view static API [5] was used to collect image data. The images were saved into a file directory structured database. The structure format looked like: $data/ < grid - number > / < latitude, longitude > / < image - date.jpg >$. An example file directory structure looked like: $data/0/42.775957, -124.0667758/0 - 2009 - 07.jpg$. With the map being split up into 243 grids and 5 images collected per grid a total of 1215 images were collected per sweep. With each image averaging a size of 50 killo bytes, this lead to the collection of about 60 mega bites of data per sweep. At a price of 0.007 dollars per image, it cost about 9 dollars per sweep.

EVALUATION

The evaluation for the project will be done based on the euclidean distance (see Figure 5) between the predicted location grid and the actual location. The test set will be processed using the above mentioned machine learning techniques to get the softmaxed output. The top N softmaxed outputs are considered. Here N will be a hyper parameter. Out of the top N, the one that gives the grid with the least euclidean distance from the target location will be chosen. This way the models can be evaluated with the euclidean distance across the test set being summed up. The model with the least sum will be considered the best. Human player scores will be used as baselines to compare with model scores.

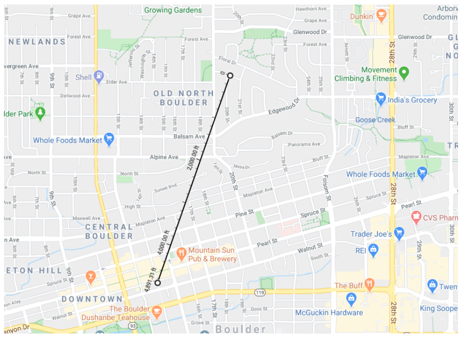
Figure 5. Using distance as an evaluation metric.

REFERENCES

[1] Unites States Census Bureau. 2018. Cartographic Boundary Files - Shapefile. (2018). `https://www.census.gov/geographies/mapping-files/time-series/geo/carto-boundary-file.html`

[2] Sean Gillies. 2020. Shapely. (Jan. 2020). `https://pypi.org/project/Shapely/`

[3] James Hays and Alexei A. Efros. 2008. im2gps: estimating geographic information from a single image. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. *CoRR* abs/1512.03385 (2015). `http://arxiv.org/abs/1512.03385`

[5] Google Maps Platform. 2020. Street View Static API. `https://developers.google.com/maps/documentation/streetview/intro`. (2020).

[6] Anton Wallén. 2013. Web-based geographic discovery game. (May 2013). `https://www.geoguessr.com/`

[7] Richard Wen. 2019. Google street view API. (2019). `https://pypi.org/project/google-streetview/`

[8] Tobias Weyand, Ilya Kostrikov, and James Philbin. 2016. PlaNet - Photo Geolocation with Convolutional Neural Networks. *CoRR* abs/1602.05314 (2016). `http://arxiv.org/abs/1602.05314`