Zero-Shot Learning for Geolocalization in Restricted Image Domains

Team: Jason Figueroa, Charlie Gottlieb, Tom Holland, Arthur Pogosian.
Project Mentor TA: Harshwardhan Yadav

1) Abstract

For this project, we aim to build a model that can take an image, particularly from Google Street View, and predict the location in which it was taken. Being able to identify location from images can be very useful for applications that use geotagging, navigation, for emergency services, or for investigative work such as finding missing persons or criminals. We based this off of a pre-existing model, and our primary contribution was implementing OpenAI's CLIP model to pair image data with text embeddings for higher accuracy on new test data. Our secondary contribution was to include image data of other geographic regions not a part of the original analysis (the UK in particular). We successfully gathered about 117,000 new images with corresponding text labellings, that accurately described an identifying city, state and country, across both the US and the UK. In turn, we built a classification model for UK data that was able to achieve 27% accuracy with a low average distance of predicted-to-actual location of 131 km.

2) Introduction

Motivation:

The problem we are working on is being able to predict the location of an input image. This is an important problem to solve because of its many use cases. If authorities are trying to find a missing person or a criminal, being able to determine the location of a recent image could assist greatly in the investigation. Alternatively, if an application such as a social media platform wants to geotag an image but lacks GPS data, this model could be of use. A trivial use would be for implementation with the popular “Geoguessr” game, in which players are tasked with guessing where in the world an image was taken, and this model could be used either to assist struggling players or to add a ‘player vs. machine’ gamemode.

Inputs and Outputs:

The input of our system is an image, and the output is the location in text. Our model was trained to work for the continental United States as well as the United Kingdom; however, our model can be easily expanded to other areas with more training images.

3) Background

The prior work we were basing our project on was a 2020 project from students at the the University of Colorado, Boulder: “CSCI5922 Neural Networks Group Project: GeoguessrLSTM” by Nirvan S P Theethira and Dheeraj Ravandranath, https://github.com/Nirvan66/geoguessrLSTM/tree/master [1]. This work provided a basis for how to organize the image data. In particular, this work used geographic Shapely data to split the US into a grid of rectangular cells, and the output is a prediction for the grid cell that the image came from. The loss function is the haversine distance of the predicted cell to the actual image cell. The results of their trained model was an average of 1,200 miles away from the actual image location in testing (for context, the diagonal of the continental US, Florida to Washington, is 2,800 miles long).

The main shortcoming of this model is its weak accuracy, as stated above. On top of that, since the grid in which its predictions are made was generated for their project, the outputs are not immediately useful outside of this system (that is, since it outputs for example "grid #4", this is not meaningful without having information on the grid itself). Additionally, it only is trained on the US, and it is unclear how well it would do with data from other parts of the world.

Another work we used as inspiration was StreetCLIP. This model was a more generalized version of our model; they use CLIP to determine location by image, and also provided us the idea of getting the top most populous cities to classify [5].

OpenAI's Contrastive Language-Image Pretraining (CLIP) is a model that learns text and images simultaneously. This allows images to be understood in the context of natural language, linking image information with the vast 'comprehension' of an LLM [2].

4) Summary of Our Contributions

We wanted to build a model with stronger accuracy, and one method of doing so was to implement OpenAI's CLIP as part of our model to supplement image data with textual, locational data. We shifted the model from predicting grid cell (and therefore latitude/ longitude) to predicting location by city; therefore our model is a classifier. Additionally, we were interested in seeing how well the model would perform in other parts of the world, so we focused on the UK as part of our data.

5) Detailed Description of Contributions

Contributions in **Data**:

Overall we were able to create a more expansive dataset of **117,840 images and corresponding locations** (accurate text labellings per image) across the US and UK focusing extensively on data from cities.

Contributions in **Algorithm**:

Using OpenAI's CLIP model, we paired each image with a text embedding to train the model. For example, an image of London will have the embedding "a photo of London in England", and an image of Edinburgh will have the embedding "a photo of Edinburgh in Scotland". CLIP creates an encoding of its classes, and it's pre-trained on over 400 million text to image pairs. In this way, it can leverage a transformer's ability to extract semantic meaning from text to make image classifications out of the box without being fine-tuned on custom data. This means CLIP can perform much better than ResNet or any other model for our task. So, we were able to do very little fine tuning and can instead rely on our data, and minor changes as needed.

As stated above, unlike the model on which we were based, which uses haversine distance as a loss function, our model is performing label classification.

5.1 Methods

Gathering Data:

We started out by following the structure of the data collection from our source project. In the source project they first obtained a shapefile of US, then split the shapefile into 243 grids. Using google streetview API they found 20 random locations in each grid, and found 3 images per location, correlating with 0 degrees, 90 degrees and 180 degrees.

We then did the same by collecting a shapefile for the UK and splitting it up into 246 grids. Similarly we found 20 random locations in each grid, and found 3 images per location, correlating with 0 degrees, 90 degrees and 180 degrees.

After collecting this data we manually went through many images and found that the images were lacking information on cities and had a lot of rural locations with very little information to train on. To address this we created a CSV with the top 100 cities by population in the US and top 50 cities by population in the UK in order to augment our data. We found 20 random locations within these cities and found 3 images per location, correlating with 0 degrees, 90 degrees and 180 degrees.

After training on the above data in the UK we found that we were still plateauing in terms of capability. We decided to create another dataset focused entirely on cities. Here we found the top 150 cities in the UK by population. For the top 50, we found 200 locations, for the middle 50 we found 100 locations and for the bottom 50 we found 50 locations. We then found 3 images per location, correlating with 0 degrees, 90 degrees and 180 degrees. We also set the streeview api parameters to only include outdoor images to try to reduce noise (though still some indoor images slipped through).

Finally we realized that in our previous CSV's for the top UK cities we were missing some major cities such as Edinburgh, Leeds and Glasgow. This was a complete oversight on our part, but did give us an opportunity to expand our dataset even more. We decided to make a new top 50 uk cities dataset, with 200 locations for each city with a slightly tighter radius. We then removed our previous top 50 cities in the original Uk dataset. While it was hard to find a consensus top 50 cities in the UK, this dataset was much more accurate. We then used this dataset to try to achieve our most accurate model so far.

Grid US: 243 * 20 * 3 = 14580 Images
Grid UK: 246 * 20 * 3 = 14760 Images
Addition US cities: 100 * 20 * 3 = 6000
Top 150 UK cities: (50 * 200 * 3) + (50 * 100 * 3) + (50 * 50 * 3) = 52500 Images
Final uk cities: 50 * 200 * 3 = 30000 Images
Total: 117,840 images

Creating model:

Three dataset classes were created for the two different data formats we used to train our models. The first dataset 'UKClip' was formed similar to how data was initially gathered, with the United Kingdom being split into a grid and a fixed amount of street-view sampled images being gathered for each grid cell. In this model, each cell had a corresponding CSV for each respective text label that associated a town/city name with a latitude and a longitude. Alternatively, the dataset class 'UKCitiesClip' used data from only the top 150 cities in the UK, and loaded image labels through one csv that similarly associated a city name with a latitude and a longitude. Finally, a similar data class used data from only the top 50 cities. All classes used a parallelised data loader in order to speed up runtime.

The images were then encoded with a ResNet-50 convolutional neural network, and the text labels were encoded similarly using a DistilBert [3] transformer model, in order to reshape the dimensionality of these parameters ready for the CLIP model. Using another feed-forward neural network, ProjectionHead, these input embeddings were projected again into a lower dimensional space, while applying transformations to enhance the representation for downstream tasks. This neural network provides the final transformation mechanism needed using normalization and a residual connection in order to prepare the input data for a clip architecture, learning on the joint representation of images and text.

Now, we defined the CLIP model itself and the cross-entropy loss function. This was mostly boilerplate code and followed how OpenAI had recommended implementation of the architecture. At this point, we normalized the input data to the model and performed a train, val and test split. These splits were then loaded using DataLoader's.

Then, we were ready to train the CLIP model at hand. We managed a multitude of hyperparams; weight decays, patience, factor and respective learning rates for each encoder. These were tweaked accordingly, as is explained in the results section, and we measured loss on the different dataset splits.

5.2 Experiments and Results

Choice of Data:

We noticed that training loss varied largely depending on the data we used; whether or not it was an equal distribution of the entire UK, or a weighted distribution with more images based in cities. Largely, we noticed that the UK was predominantly generic countryside, which the model would find hard to accurately classify, or to determine some more specific location from. The huge amounts of countryside data led to a high bias within the model and hence underfitting, resulting in a higher loss than when using a dataset composed only of city data, with data for each of the largest cities. We trained a model on both data arrangements and noted that the models trained primarily on city data were the best performing. That being said these models also had much more data to train on.

Building the model and hyperparameter choices;

We found the optimal values for hyperparams were image_encoder_lr = 1e-4, text_encoder_lr = 1e-5, head_lr = 1e-3, weight_decay = 1e-3, patience = 1, factor = 0.8, epochs = 6. When increasing the number of epochs, we found the model began to overfit.

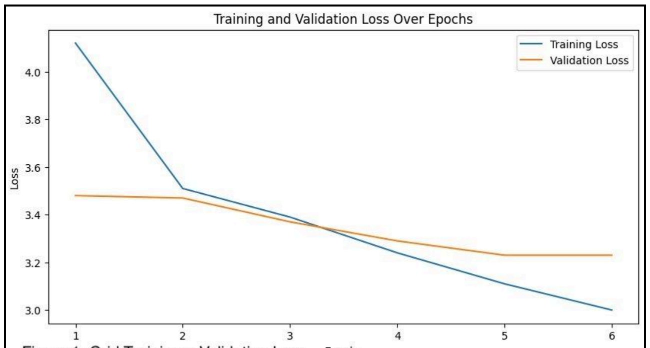


Figure 1: Grid Training + Validation Loss

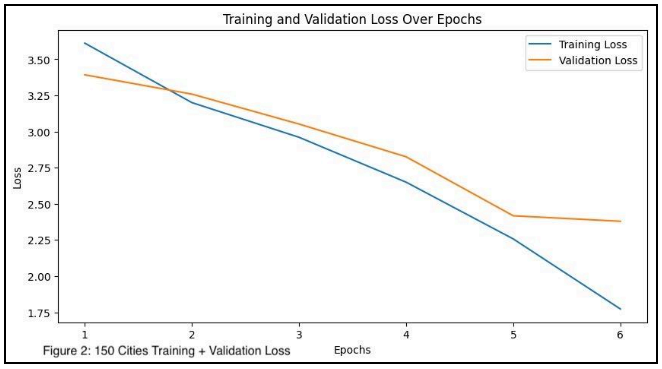


Figure 2: 150 Cities Training + Validation Loss

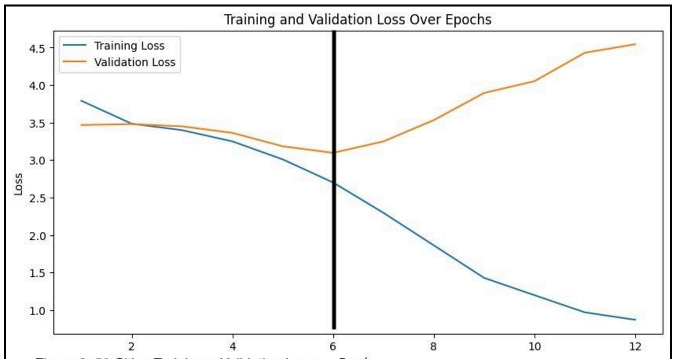


Figure 3: 50 Cities Training + Validation Loss

We used 70/20/10 train/validation/test split for all datasets.

Dataset	Accuracy	Avg Distance (km)	Num Predictions	Predictions within 25km	Predictions within 75km	Predictions within 150km
Grid	2.28%	292.23	1488	1.28%	9.54%	25.54%
Top 150 Cities	14.34%	127.88	1632	19.55%	47.00%	67.22%
Top 50 Cities	27.08%	130.76	192	22.22%	45.31%	62.50%

Comparisons with “GeoguessrLSTM” **model:**

The original model this paper was inspired by used a different loss function of haversine distance, whereas our model approaches the challenge as a classification problem. Additionally, their model was on the US, while our model is for the UK, which have vastly different sizes and cities. Therefore it is not well-defined to directly compare the two models. However, their paper did provide the average distance from their predicted location to the actual location, and we did the same. Despite the different countries, we can give a visual comparison, by showing the maps of each country along with the average prediction distance as the radius of a circle on the respective landmass. Below is that image.

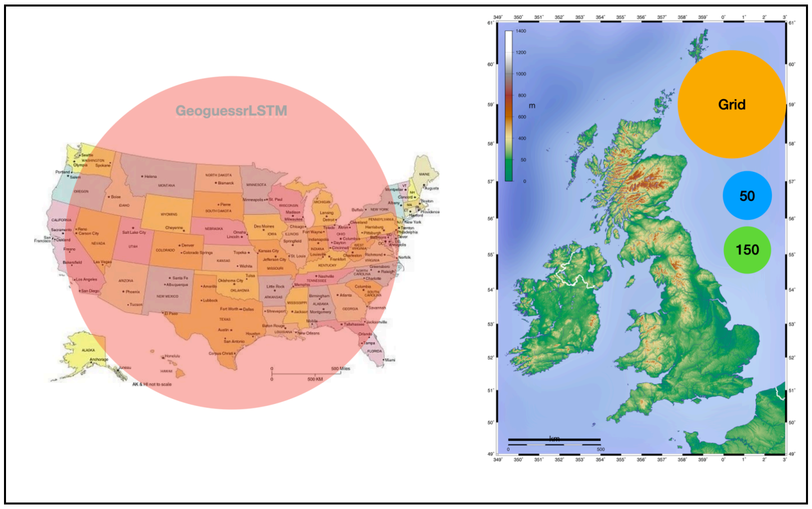


Figure 4: Model Average Distance Comparisons

6) Compute/Other Resources Used

We trained our model on Google Colab. We needed to use Colab Pro due to the high computing resources. We used an NVIDIA A100 GPU.

7) Conclusions

In this project, we were able to successfully build the foundation of a model that could predict location, given enough strong training data. It is important to consider the privacy concerns with this type of model; if someone (especially a celebrity) posts an image of themselves online they may not want to effectively broadcast their location.

Although our model is specifically tuned to data from the UK, we believe that it is essentially extrapolatable to any other country, provided a sufficient set of training images. Future models could try to apply this to other countries as well.

The implementation of CLIP seemed to add a lot to our model's capability, since it allowed the model to leverage the associative 'knowledge' of OpenAI's GPT and apply this to the image domain. Just as human Geoguessr experts are able to use real-world factual and associative knowledge to identify locations, so too would a high-quality model require such abilities in order to do the same. For future models, it might be advantageous to research the specific strategies that such human experts use and try to implement them into the model.

References

1. Nirvan S P Theethira and Dheeraj Ravandranath, “CSCI5922 Neural Networks Group Project: GeoguessrLSTM”, 2020. https://github.com/Nirvan66/geoguessrLSTM/tree/master.
2. Alec Radford, Ilya Sutskever, Jong Wook Kim, Gretchen Krueger, Sandhini Agarwal, “CLIP: Connecting text and images”, 2021. https://openai.com/index/clip/.
3. Victor Sanh, DistilBert model, 2019. https://medium.com/huggingface/distilbert-8cf3380435b5
4. M. Moein Shariatnia, “OpenAI-CLIP” from GitHub, 2021. https://github.com/moein-shariatnia/OpenAI-CLIP
5. Lukas Haas, Silas Alberti, Michal Skreta, “Learning Generalized Zero-Shot Learners for Open-Domain Image Geolocalization”, 2023. https://huggingface.co/geolocal/StreetCLIP/tree/main

Broader Dissemination Information:

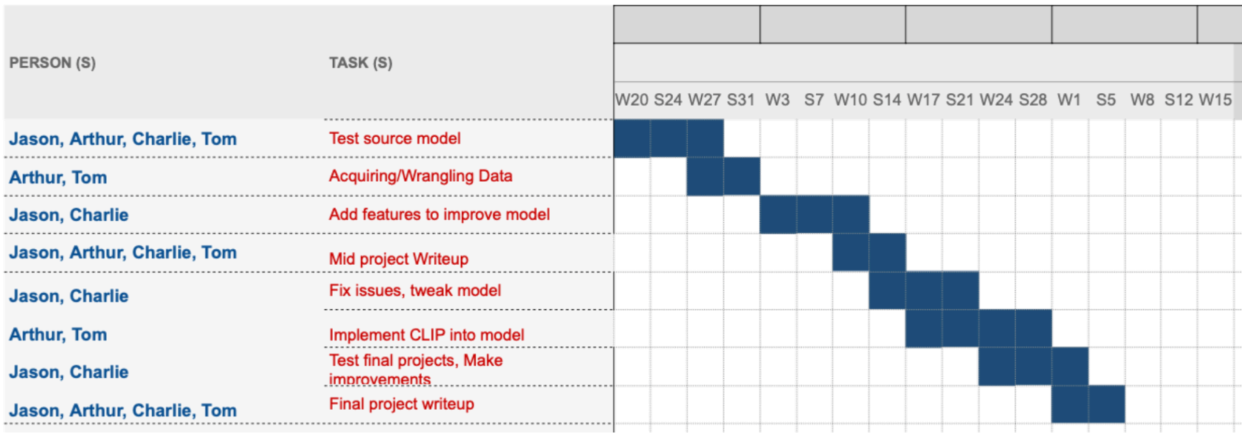
Your report title and the list of team members will be published on the class website. Would you also like your pdf report to be published?

YES

If your answer to the above question is yes, are there any other links to github / youtube / blog post / project website that you would like to publish alongside the report? If so, list them here.

- Project Git: https://github.com/slackroad/restricted_geoclip/tree/main

Work Report



PERSON (S)	TASK (S)	W20	S24	W27	S31	W3	S7	W10	S14	W17	S21	W24	S28	W1	S5	W8	S12	W15
Jason, Arthur, Charlie, Tom	Test source model	■	■	■														
Arthur, Tom	Acquiring/Wrangling Data			■	■													
Jason, Charlie	Add features to improve model					■	■	■										
Jason, Arthur, Charlie, Tom	Mid project Writeup							■	■									
Jason, Charlie	Fix issues, tweak model								■	■	■							
Arthur, Tom	Implement CLIP into model									■	■	■	■					
Jason, Charlie	Test final projects, Make improvements											■	■	■				
Jason, Arthur, Charlie, Tom	Final project writeup													■	■			

Midway Report

1) Introduction

Set up the problem: what are the inputs and outputs of your system, what is the data you are/will be training on and testing on, and how will you/are you evaluating your proposed system for this task? (**Make sure to answer all these questions** unless they don't make sense. If you don't know the answer to something yet, say you don't know. 2 or 3 paragraphs. If you did it right, you may be able to use large parts of the T, E, P answers from your proposal.)

Throughout the report, use mathematical notation to explain clearly as appropriate. You may find add-ons like "Auto-LaTeX Equations" (https://workspace.google.com/marketplace/app/autolatex_equations/850293439076) useful, and you are of course welcome to use Google Docs' native equation mode.

Our goal is to give the model and image of a location and predict the location based on an image. We begin with a dataset which contains a set of images associated with a location. We will be training and testing on this dataset. We are starting with a model that has been trained specifically on images of locations in the USA. We will expand past that to include data on locations in the UK (and hopefully extend to as many countries as possible). After collecting image data we will retrain the original model on both sets of data. In terms of an evaluation metric we recognize the simplest will be distance from actual location. However this may cause issues when accidentally predicting the wrong country so we are not completely set on our evaluation metric. Instead, we set on using classification to solve our problem, instead classifying locations based on town names. We will use OpenAI's CLIP model, which pairs image data with text embeddings for higher accuracy on new test data. For instance, an image of London might be paired with the text: "a photo of London in England." This allows a transformer to find the context relation between the text and the image, and it generates output text that describes the location in the image.

Motivation: If you make progress towards solving this problem, what potential impacts might it have? (1-2 paragraphs. You may be able to use parts of your discussion of ethical considerations and broader social impact from your proposal.)

Regarding social impact, a ML model that can classify images as location could assist authorities at locating missing people, or preventing potential wrongdoings in disclosed locations. In a less significant light, the model could provide convenience to average people who have forgotten where they took a photo, or if, for whatever reason, a person needs to find out the location of what a photo is depicting. Moreso in the age of social media, this model could be used for military intelligence. The Ukraine war is a good example of a conflict that is documented quite closely on social media, and a model that could extract location from image could be of strategic importance and help prevent atrocities.

On a more trivial note, this ML model could help assist with solving GeoGuessr Puzzles and help struggling users of the platform achieve higher scores that they are more proud of. In addition, the model could be integrated into the current platform in order to implement some not impossible ‘player vs machine’ mode. The model could be implemented as some learning tool to help improve children's geographical knowledge.

2) How We Have Addressed Feedback From the Proposal Evaluations

Describe clearly the key feedback you received from proposal and any other interactions with your project mentor etc., and how you have incorporated it into your project plans. 1-2 paragraphs.

In our original proposal we proposed working on an automatic essay grader. For that project we wanted to build off a model that performed essay grading for individual prompts and scale it to cross-prompt essay grading. That being said, many of our ideas were high-level, and we had not thought specifically about the contributions we wanted to make. Our TA recommended that we take a step back and figure out exactly what we wanted our contributions to be, along with how we planned on achieving those goals. After that feedback we realized we were not going to be able to achieve our original proposal.

We then pivoted to our new idea of predicting location based on images. We were able to find a helpful open source project and then come up with two contributions we are confident we can make. This also caused us to have to update our time table. That being said, thanks to the advice of our TA we were able to come up with a more achievable plan.

3) Prior Work We are Closely Building From

2-3 sentence introduction if necessary to set up the context for what follows. You may be able to reuse material from your proposal for this section.

A. Paper / blog post/ Kaggle submission etc.: Title. URL. What it does that is relevant to this project (2-3 sentences). Code link. Provide any links in text, like www.google.com, not like Google.

An open source project on geoguessr: https://nirvan66.github.io/geoguessr.html, which trains a similar model on the continental United States. This project provided a starting point for data collection and model creation. We have already begun work to expand that project’s data approach, splitting a map into boxes for training purposes, to include data from England as well.

B. More if needed. There should be at most 2 such closely related prior works in most cases.

4) What We are Contributing

2-3 sentence introduction if necessary to set up the context for what follows. You may be able to reuse material from your proposal for this section.

Our first contribution will be dataset collection and augmentation (specifically more image data). The project we found only includes the continental United States. We are expanding it to work with England as well, and perhaps other countries in Europe. Our second contribution is improvement of accuracy based on zero shot classification. We will utilize OpenAI's CLIP model, which pairs image data with text embeddings for higher accuracy on new test data. CLIP's advantage lies in its vast pre-training on over 400 million text-image pairs, enabling it to utilize transformer technology to interpret and classify images based on textual descriptions, surpassing traditional models like ResNet for our application. This is in contrast to a model that generates an integer output that it then assigns a text label to.

5) Detailed Description of Each Proposed Contribution, Progress Towards It, and Any Difficulties Encountered So Far

Describe in detail (about 1-1.5 pages expected). For example:

- If you are claiming an algorithm contribution, you may set up some notation in text, write out some high-level pseudocode, and discuss how your algorithm extends the prior work you have cited in Sec 2, and talk about how much progress you have made towards actually implementing and testing it, and any early results.
- If you are claiming a data contribution, you may talk about your data collection mechanism (e.g. crowdsourcing, or surveying experts, etc.), and show some examples of data you have collected, and how it has/will be cleaned up in preparation for ML, and tested for ML-readiness using common baseline approaches.

Format is somewhat flexible here, but for many of your projects, the following formatting would be the right starting point. (In any case, organize this section into useful headers as appropriate, to allow easy reading and evaluation.)

5.1 Methods

Describe your proposed approach. Be scholarly, and generously cite prior work, even aside from those in Section 3, if needed. Format them as [Author 1, 2020] or as [A].

5.2 Experiments and Results

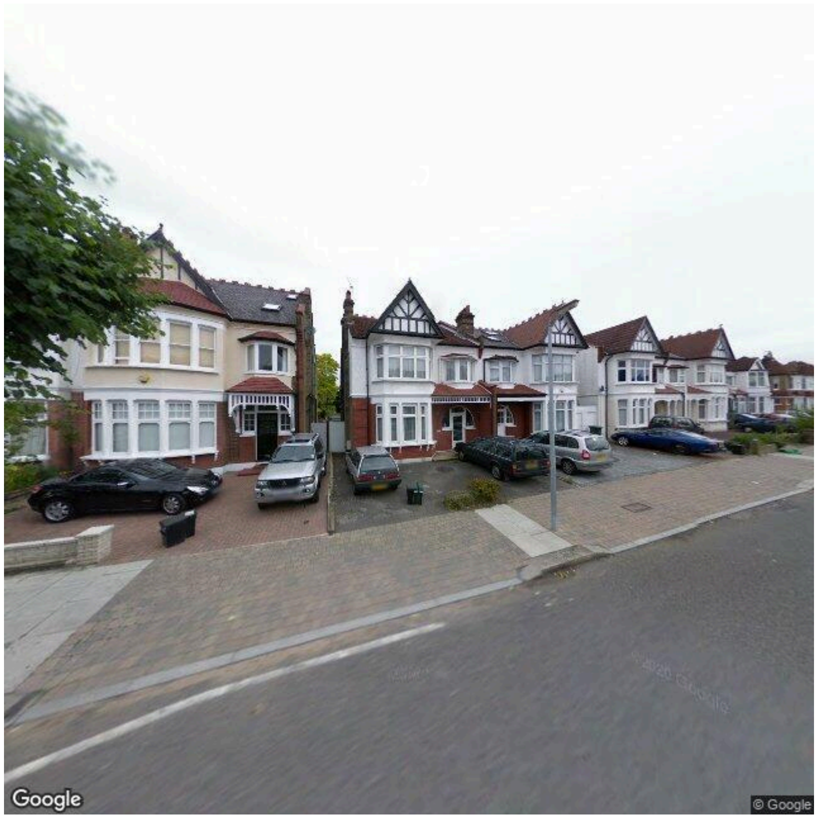
What are the key questions your experiments will try to answer, or hypotheses your experiments will try to validate? What will be the baselines for your approach? What datasets and performance metrics (may overlap with Section 1, this is okay)? Which of them have you run so far, and what are the results?

You will be expected to have made some non-trivial progress towards your project.

Data collection: We took a similar approach to the original project but instead of focusing on just the United States we expanded to the UK. We split the UK into a grid of 246 square. We then gathered a set of random coordinates for each square. From those random coordinates we used google street view api to collect 40 images for each grid, each one corresponding to a different location inside that grid. This will allow us to give our model multiple images corresponding to one location in the hopes that we can train a model to predict images from location. We encountered an issue with data collection, which was that we needed a place to store all the images we were collecting. We decided to store these images in google cloud so that we can store our large amounts of data while also being able to easily access this data in google colab. As well, we found town and city names associated with each image. This will be useful for our second objective of our project. This process also allows us to easily scale to include other countries as part of our location detection. We plan on using a letterbox filter on each of our pictures in order to remove the warping that comes with google street view images as a result of the 360 camera they use to take the images. We plan on training and testing the original model on our new data to serve as a baseline for accuracy that we will then compare to in our next contribution.

Example image collected:

51.62466836136767, -0.11914176280765626, London:



Model Architecture: After a detailed overview of our choices, we set on using Zero Shot Image Classification for this problem. Our other choice was to predict latitude and longitude through a continuous classification model (instead of discrete classification) with a CNN. We chose not to do this because we believe that there is higher accuracy in a closed domain, zero shot classification. Our data is selected from a localized area, currently the U.K. and soon more of Western/Southern Europe, allowing us to finetune the model to a very particular set of locations. Specifically, these areas have very diverse architecture and unique characteristics that will prove suitable for a zero shot model to classify.

In particular, we are looking to train our model to be able to classify based on town name, allowing certain areas that have certain aesthetic characteristics to be easily identified. Using OpenAI's CLIP model, we can pair each image with a text embedding to train the model. For example, an image of London will have the embedding "a photo of London in England", and an image of Edinburgh will have the embedding "a photo of Edinburgh in Scotland". CLIP creates an encoding of its classes, and it's pre-trained on over 400 million text to image pairs. In this way, it can leverage a transformer's ability to extract semantic meaning from text to make image classifications out of the box without being fine-tuned on custom data. This means CLIP will perform much better than ResNet or any other model for our task. So, we will have to do very little fine tuning and can instead rely on our data, and minor changes if needed.

6) Risk Mitigation Plan

Examples of questions to consider here: How will you build a minimum viable project within the remaining time? Will you start with a simplified setting where you can get some early results, so that you still have time to pivot if needed? If your approach doesn't work, how will you still turn in a useful project report? What if you find that you need too much compute? Will you try your algorithm on different, simpler data, such as a "toy" synthetic dataset you generated? Will you evaluate the reasons for failure, or find specific examples where your approach works better, even if it does not do better on the full dataset? We are looking here for evidence that you understand the risks associated with your plans, and have thought through these possibilities.

Regarding our first criteria, we could provide a MVP through just enhancing the original American data and adding more images to the previous dataset, even though our current plan was to enhance the dataset by adding another region. However it is likely we will be able to meet this criteria and build up a dataset for some foreign countries.

One risk is that we do not manage to make a model that can classify the difference in continents/countries, or a model that struggles in this regard, as the previous model has only had to classify cities and locations within the United states. Again, at least, we could look to just enhance the models ability to classify within the states, by reducing the loss metric.