Machine Learning Algorithms - Project report

due at 01.03.2022

Florian Schager

1 Project proposal

In the game GeoGuessr the player is dropped on a random location within Google Streetview with the goal to determine his location as precisely as possible. In the simplest case, we are not allowed to move from our starting position, pan around nor zoom in the initial image.

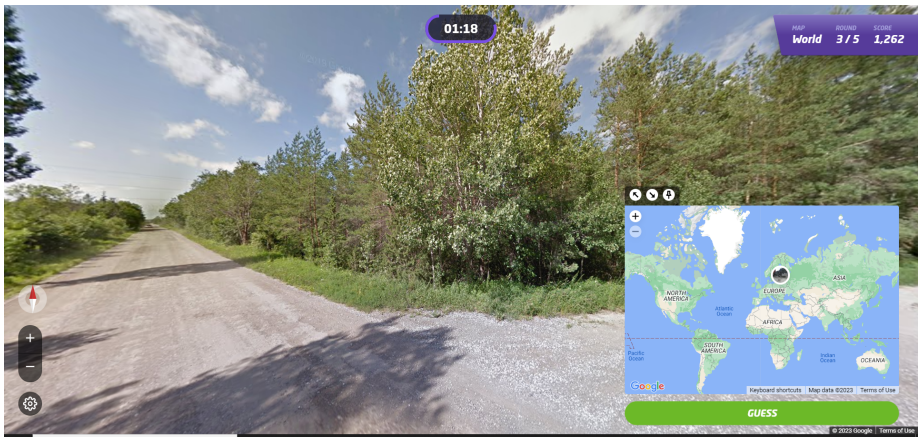

Abbildung 1.1: Example image from a game of GeoGuessr

Hence the goal of this Machine Learning project is to map Street View images to location data. This setting naturally allows for either regression, where we aim to predict the latitude and longitude of any given image, or classification, if we ask to classify the images into pre-defined geographic regions, like countries or continents.
To obtain our dataset, we use the osmnx library to create street network graphs for given geographical regions. This is necessary, because simply sampling geographically uniform points within a given region is unlikely to return coordinates with street view coverage. Instead, we can sample our points from these street network graphs, which in turn get fed into the Google Street View API. The API now checks whether there is an available street view image within a fifty meter radius and if yes, returns one such image.
I want to focus on the classification task, where images are classified into the countries, where they are taken in. A good starting point would then be to pick two distinct countries, like Austria and Australia to replace the `dogs_vs_cats` dataset. I would then propose to more or less keep the individual task of the project as is, except for the dataset. The only thing I am unsure about in this regard is, whether it makes sense for this task to use pre-trained weights from the imagenet dataset, since my particular problem seems to diverge too much from the idea of the imagenet dataset, where images are classified by so-called synonym sets. Hence I think it would make more sense to instead try, once the models perform well on the binary classification task, how well this approach extends to a multinomial classification, where I increase the number of countries the images are sampled from.

2 Generating the image dataset

We use the GoogleStreetview API to request street view images from a given location. The API takes a tuple of latitude and longitude as input and returns a street view image within a certain radius, if one exists. As a consequence, before we can start sampling images from the API, we first need a way to sample coordinates within our desired countries. As it turns out, simply sampling coordinates uniformly within a certain bounding

```
def request_image(self, location):
location = f"{location.y},{location.x}"
params = {
        'size': '640x640', # max 640x640 pixels
        'location': location,
        'source': 'outdoor',
        'radius': 100,
        'return_error_code': 'true',
        'key': self.api_key,
      }

# Check first, if an image is available at the given location
metadata = requests.get(self.metadata_api, params).json()

# If an image is available, save it and its coordinates in the respective folder
if metadata['status'] == 'OK':
  coordinates = metadata['location']
  response = requests.get(self.streetview_api, params)
  self.save_image(response)
  self.save_coordinates(coordinates)
  self.current_id += 1
```

Abbildung 2.1: Requesting images from GoogleStreetviewAPI

polygon representing a country does not yield many locations sufficiently close to a road with streetview coverage. Therefore it seems expedient to first get a model of the local street network and then sample locations from this graph, which can be done using the module OSMNX.

2.1 Sampling locations from street networks

2.2 Requesting images from GoogleStreetview API

3 Binary classification

3.1 Ridge Classifier

3.2 Logistic Regressor

3.3 Support Vector Machine

3.4 Neural Network

4 Adversarial Machine Learning

5 Model Explainability