Alabama Roads Project

Background

This is a 'proof of concept' exercise that will attempt the predict road safety using the Google streetview api and a retrained (transfer learning based) version of Tensorflow to enable mass prediction of road safety on large datasets. The ultimate output would be a heat map/scoring of predicted road saftey across Alabama.

Take streetview photos

Import roads

To do this we will use the road shapefile in the raw project data folder
~/Dropbox/pkg.data/alabama_roads/raw/us82erik/us82_dissolve. If you do not currently have access to the folder, you can download the folder by going to this Dropbox web link. Ideally, you should be able to dynamically link locally by using Dropbox desktop…

To build the streetview folders, the first step is to discretize the road shapefile and then use the google streetview metadata api to get locations of nearest google streetview picture (henceforth called pano_id). In this case the picture will be manually tuned to have the car driving in the 'correct' direction. In the future, this can easily be done with tensorflow as well, by simply ensuring that the photo taken is on the right (not the left) side of the road. Again, this can be easily automated in tensorflow in the future for bigger data applications.

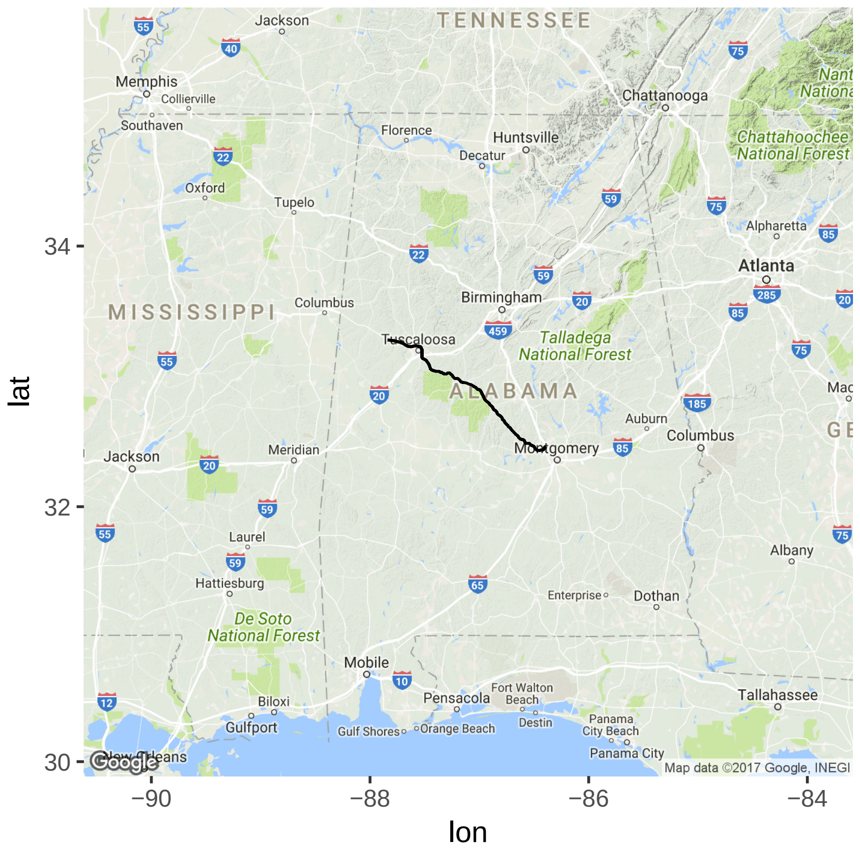
```
if(!(file.exists(roads.clean.location))){
  roads <- rgdal::readOGR(dsn = path.expand(roads.raw.location), layer = 'us82d', verbose = FALSE)
  roads <- sp::spTransform(roads, CRSobj =CRS(proj.env))

  saveRDS(roads, file = roads.clean.location)
} else {
  roads <- readRDS(roads.clean.location)
}
# Plot
map.center <- as.numeric(geosphere::centroid(rgeos::gBuffer(roads, width=0.1)))
```

```
## Warning in rgeos::gBuffer(roads, width = 0.1): Spatial object is not
## projected; GEOS expects planar coordinates
```

```
map <- ggmap::get_googlemap(center=c(lon=map.center[1], lat=map.center[2]), zoom=7)
```

```
## Map from URL : http://maps.googleapis.com/maps/api/staticmap?center=32.890998,-87.106853&zoom=7&size=
```

```
roads_fortify <- ggplot2::fortify(roads)
p <- ggmap(map)  +
  geom_line(data=roads_fortify, aes(x=long, y=lat, group=group))
print(p)
```



Roads Discretize

This discretizes the roads into points creates the points to find the panoids

```
if (!(file.exists(road.points.location))){
road.points <- suppressWarnings(sample.line(roads, sdist=0.001))
road.points$lat <- road.points@coords[,1]
road.points$long <- road.points@coords[,2]
saveRDS(road.points, file=road.points.location)
} else {
  road.points <- readRDS(road.points.location)
}
kable(head(road.points))
```

1	-87.84017	33.28436	-87.84017	33.28436
2	-87.84016	33.28436	-87.84016	33.28436
3	-87.84015	33.28436	-87.84015	33.28436
4	-87.84014	33.28436	-87.84014	33.28436
5	-87.84013	33.28436	-87.84013	33.28436
6	-87.84012	33.28436	-87.84012	33.28436

Pano_ids and meta data

The goal of the road discretization is solely to build a set of points with which to find the nearest pano_id. This can be automated in the future to adjust the sdist parameter in the road.points chunk to optimally collect all panoids while minimizing the number of repeats samples. This will speed up the meta_data api calls. For now we will just fill in the data data with the 183965 road points that we have sampled.

```r
road.points$lat <- road.points@coords[,1]
road.points$long <- road.points@coords[,2]
```

Pano_ids and camera direction

Snap street view picture

Tensorflow