Informatics Engineering Laboratory

GROUP 23
Traffic Accident prediction using Deep Learning

University of Minho, Department of Informatics,
4710-057 Braga, Portugal

PG41073 - Hugo da Gião A64282 - Paulo Alves

July 20, 2020

Abstract

Machine learning using convolutional networks to predict traffic accidents and possible danger level of roads and off-roads in Portugal mainland.

Using statistics from the main civil protection authority in Portugal, specifically the ones related to traffic accidents, and satellite images that were divided into grids, two different datasets were created. The first dataset with all the accident's occurrences and the second one dividing all the occurrences into multiple classes related to how many accidents were reported in each specific area . Both datasets were then trained using well-known deep learning model provided by the **Keras Applications** . Models using both random weights or pre-trained from **ImageNet** were tested to predict accidents or danger levels. The highest accuracy when predicting accidents hover the 0.8 (80%) mark . The highest accuracy when predicting danger levels, using the second dataset, hover the 0.39 (39%) mark.

An initial analysis of these results for the second dataset suggests that we made a mistake, either on the input or most-likely on gathering the output metric as 'categorical_accuracy'.

Overall the prediction of whether or not a specific location/road is prone to having a traffic accident was successfully with good accuracy, specially considering our case study, yet further analysis will have to be done on the second dataset to locate the problem as the results were not acceptable

The code associated with this project can be found and downloaded here:

`https://github.com/h4g0/LEI-RoadAccidents`

Contents

Chapter 1

Introduction

1.1 Motivation

Figure 1.1: *Marques de Pombal* roundabout in Lisbon(image taken from `https://moniz.pt/engcivil/rotunda-do-marques/`)

Road accidents and injuries represent some of the leading causes of deaths, just in 2016 they have caused 1. 4 million deaths and are the 8th cause of death and the leading cause of injury-related deaths. Portugal alone registered 472 road accident-related deaths and 2288 serious injuries in 2019.[3][4]

There are currently numerous projects and deep learning approaches that aim to predict and mitigate the number of accidents and improve road safety and with the widely spread out use of GPS systems and the emergence of self-driving cars creating systems that can assist humans in improving road safety is becoming a crucial part of our day to day life.

1.2 Work done in this project

The work done in this project consists of the various steps necessary for the creation and deployment of deep learning models capable of predicting the road accident risk associated with different areas in the Portuguese territory.

The first step to achieve our desired goal was to collect useful data and then proceed to do a careful analysis and processing of said data.

We then divided the mainland territory into cells and created a dataset containing the number of accidents for each cell. After doing so we use created our datasets and gathered satellite images to be used in the training of machine learning models.

Finally, we implemented a couple of applications that exemplify possible uses of the trained models.

Chapter 2

Creation of the accident dataset

2.1 Gathering the data

The statistics used to construct the original dataset where made available by the *Autoridade Nacional da Proteção Civil* (Portugal) [1] .These statistics consist of the occurrences of various accidents, natural disasters and other interventions made by the authorities in mainland Portugal for a given date.

The data was collected using a script found in this repository [2]. This script allowed us to circumvent the limitation of being only able to query the statistics for a given date imposed by the API in question by automatizing the process of downloading the data for a given range of dates.The data collected for the dataset ranges from early 2016 to March 2020.

The result of the above queries is an ensemble of CSV files containing the occurrences of the various interventions for each of the dates specified in our query. To create the dataset containing all the raw information collected from the API we then merged all these files into 1 containing all the accidents spanning this period.

2.2 Transforming the raw accident data

Since the dataset created using raw data gathered from the *Autoridade Nacional da Proteção Civil* (Portugal) is not suitable to realize an adequate exploration given the domain of the problem and could not be directly used for the creation of the grid dataset since it contains considerable amounts of redundant data we needed to apply the necessary transformations to this data.

The raw accident dataset contains the following columns:

[1] `http://www.prociv.pt/en-us/Pages/default.aspx`
[2] `https://github.com/centraldedados/protecao_civil`

Column	Description
Numero	Process number for the incident in question
Data de ocorrência	Date in which the incident happened
Data de fecho operacional	Date relating to end of the operations related to said incidents
Natureza	Nature of the incident
Estado Ocorrência	Indicates if the proceedings related to said incident are concluded
Distrito	District in which the incident happened
Conselho	Municipality in which the incident happened
Freguesia	Parish in which the incident happened
Localidade	Locality in which the incident happened
Latitude	Latitude in which said incident happened
Longitude	Longitude in which said incident happened
Numero de meios terrestres envolvidos	Number of terrestrial vehicles deployed in response to the incident in question
Numero operacionais terrestres envolvidos	Number of personell deployed in response to the incident in question
Numero de meios aereos envolvidos	Number of aerial means deployed due to the incident in question
Numero de operacionais aereos envolvidos	Number of aerial personnel deployed for the given incident

Table 2.1: Features present in the collected dataset

At a first glance, we were able to notice that the vast majority of features present in the collected dataset would be of little to no use for the creation of a possible dataset associating a given area with its danger risk.

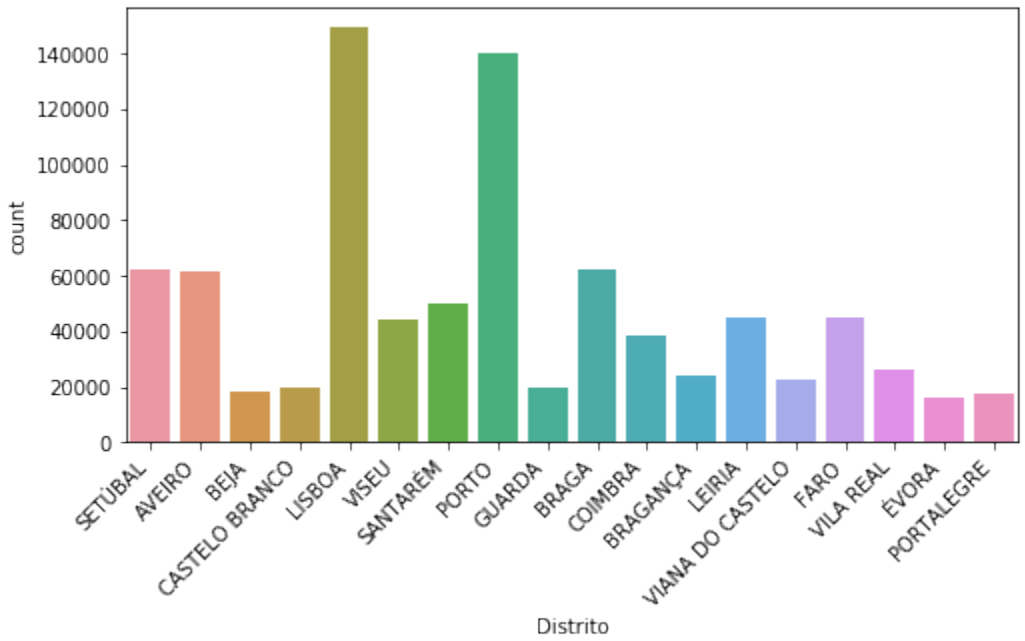

Figure 2.1: Number of incidents in the dataset per district(mainland Portugal only)

After visualizing the number of incidents per district we were able to conclude that this metric is strongly correlated with the populational density of each of the districts, that was expected since the occurrence of the majority of incidents in the dataset is strongly correlated with human activity. This analysis was done as a way to verify the comprehensiveness of the collected dataset and identify possible bias for one region compared to the others, this could arise due to various factors such as some areas having higher ratios of their incidents registered and possible losses of data.

'Protecção e Assistência a Pessoas e Bens / Assistência e Prevenção a actividades humanas / Limpeza de Via e Sinalização de Perigo'
'Riscos Tecnológicos / Acidentes / Colisão rodoviária'

Table 2.2: Example of elements from the camp *natureza* belonging to the class ROAD_ACCIDENTS

We where also able to notice that the vast majority of the incidents in the datasets are of no use for our project due to them not being directly correlated to the accident risk of the area in which they are situated.

'Protecção e Assistência a Pessoas e Bens / Assistência e Prevenção a actividades humanas / Prevenção a actividades de lazer'
'Protecção e Assistência a Pessoas e Bens / Assistência em Saúde / Intoxicação'
'Protecção e Assistência a Pessoas e Bens / Assistência em Saúde / Trauma'

Table 2.3: Example of elements from the camp *natureza* belonging to the class NON_ROAD_ACCIDENTS

To better visualize the usefulness of the collected dataset we split the incidents according to their category being related or not to potential road accident risk in a given area.

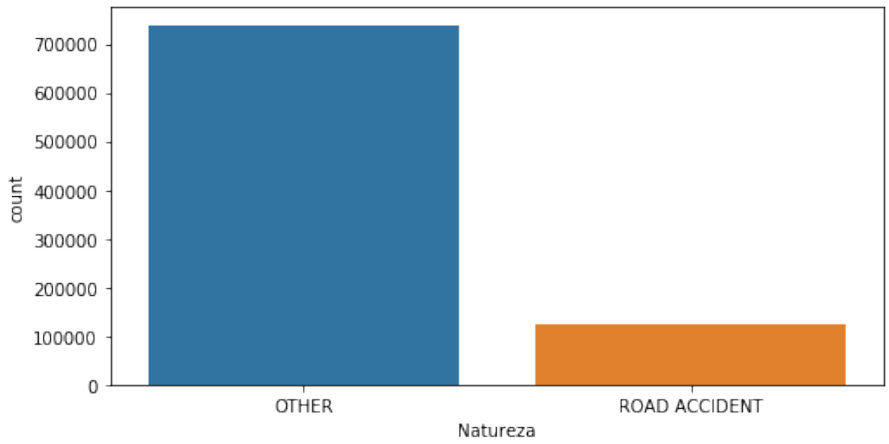

Figure 2.2: Elements of our dataset who's camp *natureza* belongs to the class ACCIDENTS or OTHER

As the graph above suggests the number of useful incidents in the collected dataset constitutes a very minority of the dataset their numbers being upward of 7 times smaller than their counterpart. To construct the dataset used for the assessment of the accidents for a given area we then dropped all the non-useful incidents from the current dataset. To help in the further analysis we also changed the original accident category names to more intuitive, simple, and easier to understand names.

Chapter 3

Exploration,visualization and analysis of the accident data set

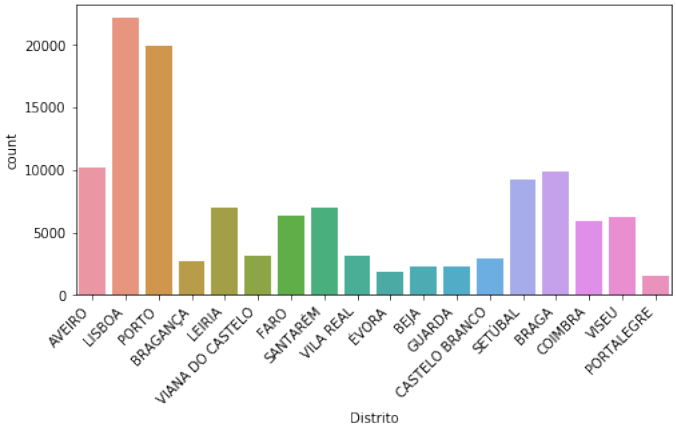

Figure 3.1: Number of road accidents per districts during the given time period

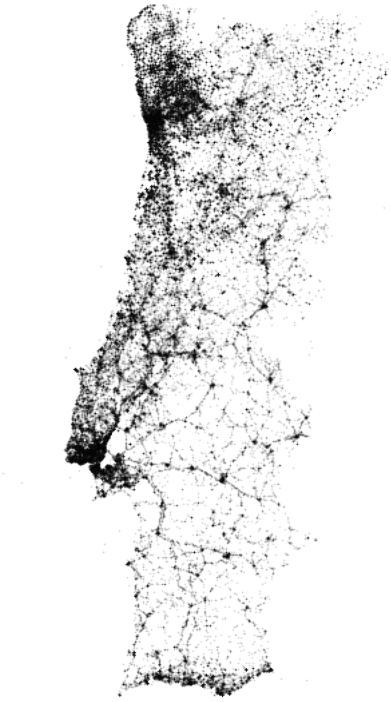

Figure 3.2: Location of road accidents

The graphs above show us a similar distribution of road accident-related incidents and the total number of incidents. We did as well plot the road accident-related incidents according to their geographical coordinates and a clear pattern or various roads and highways emerges from the resulting graph, from this we also observed that the accidents are concentrated in areas with higher population densities such as coastal regions and the Lisbon and Porto metropolitan areas.

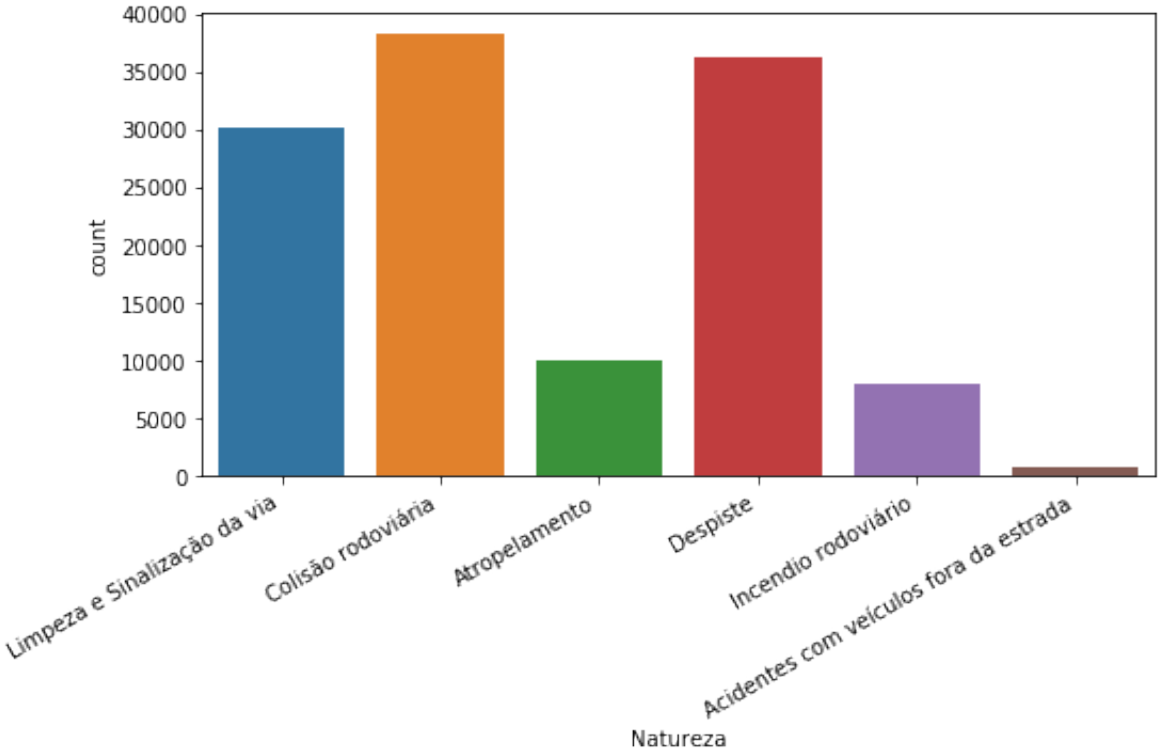

Figure 3.3: Road accident related incidents discriminated according to their cause

We used the created categories to visualize the number of road accident-related incidents according to their category and found that the vast 3 make most of the dataset this are collisions and maintenance tasks. We decided to not remove this sort of incident from the dataset since higher maintenance necessities for road and highways are correlated to various factors that could influence the risk of road accidents in a given area such as more use and higher deterioration of these areas.

Chapter 4

Creation of the grid datasets

4.1 Dividing the map into a grid

Figure 4.1: Approximation of what a 200mx200m cell looks like,to create this figure the *Estádio do Sport Lisboa e Benfica* official dimensions where used(dimensions taken from `https://www.slbenfica.pt/en-us/instalacoes/estadio/caracteristicas_zonas`)

We had to take into account various factors when choosing the size of each of the grid cells in our dataset, one of the factors we took into account was the viability of the results attained using models trained with cells of the chosen dimension taking into account the domain of the problem. More specifically we strived to find a size for which the results attained would describe the accident risk for an area with a big enough span to accurately represent road and terrain topologies, but also small enough for which the predictions can be used to describe risks associated with different streets, road topologies and not only neighborhoods.

Beyond the viability of the results, we also took also into account the computational needs involved in creating the cells and iterating over the dataset of road accidents to compute the number of accidents for the given cell. Taking all that into account we choose to divide the territory into 200x200 cells.

Algorithm 1 Grid generation algorithm

$grid \leftarrow initialize\ a\ dictionary\ containing\ one\ empty\ list\ for\ each\ district$
$lat_{curr} \leftarrow lat_{min}$
while $lat_{curr} < lat_{max}$ **do**
  $lat_{increment} \leftarrow compute\ the\ latitude\ increment\ for\ the\ current\ latitude$
  $lon_{increment} \leftarrow compute\ the\ longitude\ increment\ for\ the\ current\ latitude$
  $lon_{curr} \leftarrow lon_{min}$
  while $lon_{curr} < lon_{max}$ **do**
    $district \leftarrow compute\ district\ for\ the\ center\ of\ the\ cell$
    if $district| = "offbounds"$ **then**
      $grid[district].append(current\_cell)$
    $lon_{curr} += lon_{increment}$
  $lat_{curr} += lat_{increment}$

The strategy used to divide the Portuguese territory into a grid space was to first use a GeoJson file containing the ruff specification of mainland Portugal to attain the maximum and minimum longitude and latitude and then iterate from the lowest latitude to the maximum latitude in an outer loop and from the lowest to highest longitude in the inner loop, for each iteration moving 200m from bottom to top or right to left.

In each of the inner loop iterations, the district of which the center of the cell belongs is computed, and then if this point doesn't belong to any district the cell is discarded if not the algorithm saves the minimum and maximum latitudes and longitudes in a list containing the cells of the district of which the cell belongs.

To find the district for which the cell belongs we used a freely available GeoJson specification of all districts in mainland portugal found here [1].We used the formulas $111412.84 * coslatitude - 93.5 * cos(3 * latitude) + 0.118 * cos5 * latitude$ to aproximate the longitude and $111132.92 - 559.82 * cos(2 * latitude) + 1.175 * cos(4 * latitude) - 0.0023 * cos(6 * latitude)$ for the latitude.

In retrospective we found that using geospatial libraries such as LatLon[2] and geopy[3] could have been a more suitable alternative in terms of giving us better assurance and robustness comparatively to doing our meter to latitude and longitude conversions ourselves.

Algorithm 2 Create grid dataset

$grid \leftarrow compute\ the\ grid\ points\ for\ all\ the\ districts\ using\ the\ Grid\ generation\ algorithm$
for $district \in grid$ **do**
  $compute\ the\ number\ of\ accidents\ for\ each\ of\ district\ cells\ using\ the\ district\ dataset$
$grid\_dataset \leftarrow concatenate\ all\ the\ district\ grid\ datasets$

After computing the cells we used the road accident dataset to find the number of accidents occurring in each one of them. To allow for a reasonable computing time we divided the original dataset into districts and for each of the cells only look up the accidents occurring in the same district as its center point thus reducing the complexity of finding the number of accidents for all cells from $cells * dataset\_size$ to $\sum_{district \in districts} cells\_district * dataset\_district\_size$.To speed up this process we also used several heuristics such as during the process of finding a given cell district looking up the district of the previous cell.

[1] https://idealista.carto.com/tables/distritos_portugal/public
[2] https://pypi.org/project/LatLon/
[3] https://pypi.org/project/geopy/

4.2 Creation of the first dataset

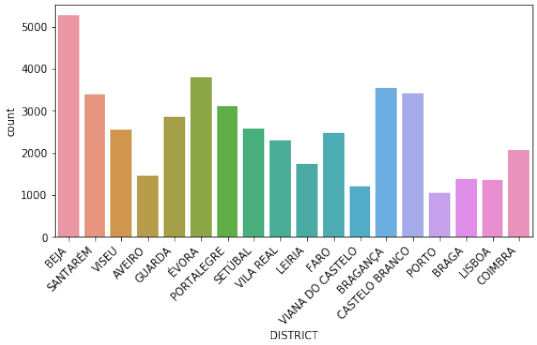

Figure 4.2: Distribution of the cells with accidents

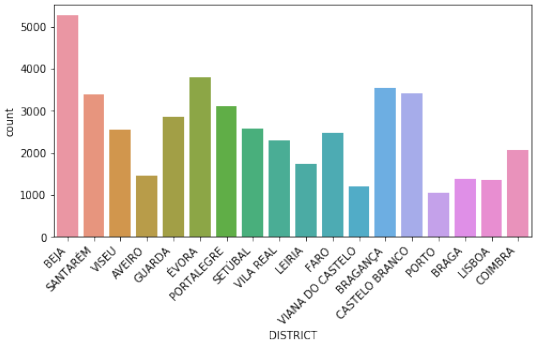

Figure 4.3: Distribution of the cells without accidents

The main goal behind the creation of the first dataset consists of differentiating between places having a very low probability of having road accidents such as fields, forests, some countryside roads, and places with moderate to high probability of accidents such as highways, roundabouts, and main roads. To do so we decided to divide our grid points into two classes one called SAFE comprised of cells where no accident was registered in our original dataset and ACCIDENTS comprised of the cells where at least one accident occurred in the given period.

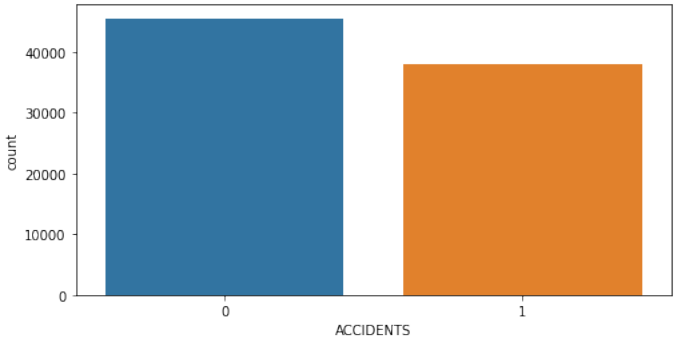

Figure 4.4: Number cells with(1) and without accidents(0) after doing a 1.1 to 1 undersampling

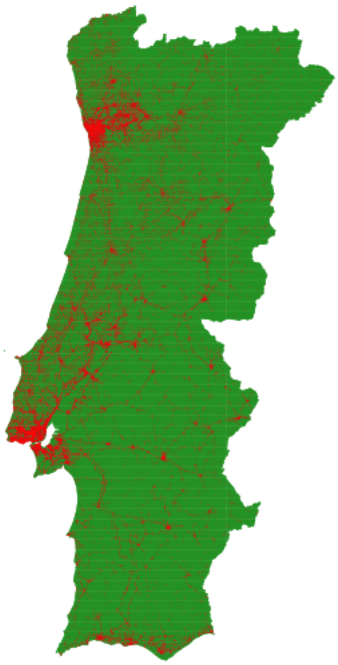

Figure 4.5: Places with 1 or more accidents in the accident dataset(red) and places with 0 accidents(green)

Since there where considerably more SAFE cells than ACCIDENTS we decided to balance the dataset, by sampling $1.1n$ cells n being the number ACCIDENTS cells from the SAFE cells, initially we decided to sample 1.1n cells instead n cells as a cautionary measure since at the time we assumed that we could eventually manually curate the images since some of the images in the SAFE dataset could be outliers and mislead the model into thinking some places such as roundabouts and other road topologies are safe and thus hindering the performance and training of possible models. In the end, we ended up balancing both classes by undersampling to have our training dataset have 50% of its elements of each class. We opted to not manually cure our dataset both for time constraints and to not introduce possible bias into our trained models.

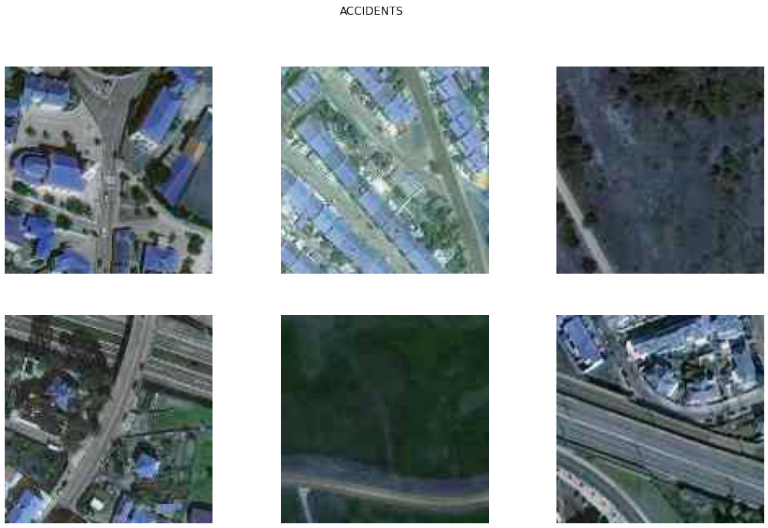

Figure 4.6: Images belonging to the ACCIDENTS category

Figure 4.7: Images belonging to the SAFE category

4.3 Creation of the second dataset

4.3.1 First version

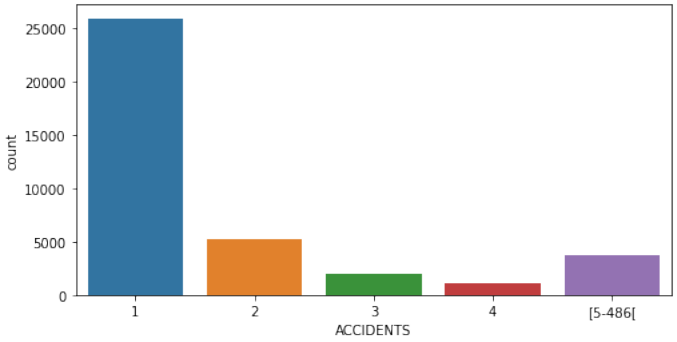

Figure 4.8: Number with accident count in a given range

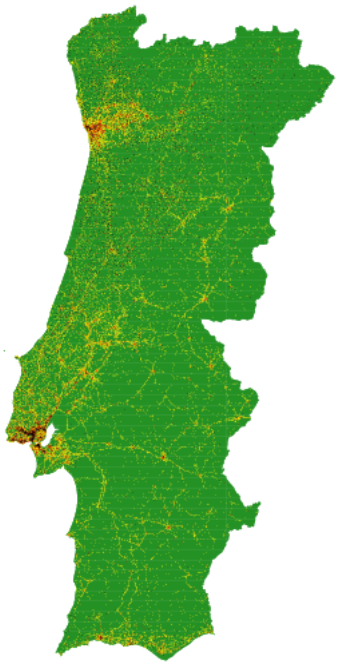

Figure 4.9: Places with 0 accidents(green),1 accident(yellow),2 accidents(orange),3 accidents(red),4 accidents(grey),5 or more(black)

The main motivation behind the creation of this dataset is to discern between the areas with a more than the moderate risk of an accident. To do so we initially took the elements in the ACCIDENT class of our first dataset and divided them into three classes according to the number of accidents in the given areas. To choose the range for each of the classes we first analyzed the number of accidents distribution and tried to find ranges for each of the risk levels for which the classes would be balanced unfortunately we found that there where considerably more cells containing for the lower number of accidents especially for 1 accident, this cells representing more than 60% of the total cells. We then divided the ACCIDENTS cells into three categories representing three different risk levels and having accident counts in the ranges $[1, 2[$,$[2, 5[$ and $[5, 486[$.Since the classes are still in inbalanced we balanced them using undersampling before feeding them to the models.

Figure 4.10: Images belonging to the LEVEL1 category

Figure 4.11: Images belonging to the LEVEL2 category

Figure 4.12: Images belonging to the LEVEL3 category

4.3.2 Second version

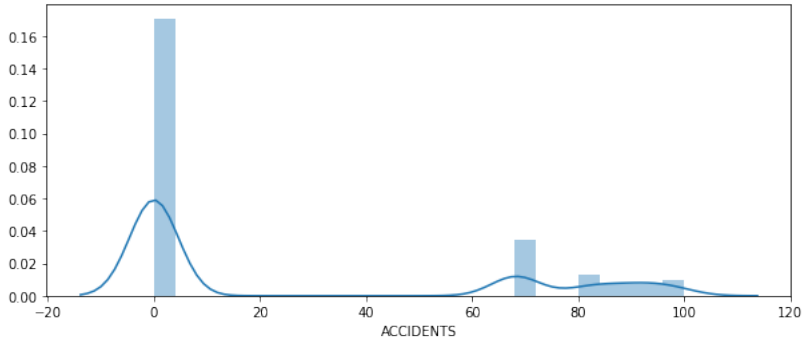

Figure 4.13: Distribution of the elements in the second version of the second dataset

We decided also to create a second version of this dataset using a continuous domain instead of dividing the dataset elements into categories to do so for each cell we computed a score comprised between 0 and 100 representing the risk level associated with the given cell. The risk level is computed according to the distribution of the number of accidents between the cells in the ACCIDENTS class of the first dataset.

For similar reasons as the first version of this dataset, this dataset is inherently imbalanced since it contains considerably more cells with a lower number of accidents than the others. In opposition to the first version of this dataset, we did not balance this one due to the difficulty associated with balancing a dataset with a continuous domain.

Chapter 5

Downloading and processing sattelite imagery

5.1 Downloading sattelite imagery

Originally we intended to source our satellite images using freely available satellite imagery, starting by using a WebMap service provided by the *Direção geral do território* containing Sentinel2 imagery spanning all mainland Portugal, but found that the images attained using this service were off to low resolution for the task, after further research, we found that maximum resolution for the areas of study to be of 30m for areas using images attained from Landsat, Sentinel2 using various sources such as sentinelhub[1], images with higher resolution from services such as planet[2], and others tended to only cover sparse areas of the Portuguese territory.

As an alternative, we tried sourcing our images from freely available aerial imagery such as the ones that can be found on google earth engine[3] and other services. This imagery is often used for monitoring crop growth and other agricultural uses so it could be a good fit four our use case since their resolution would allow for appropriate close-ups of the areas of study, unfortunately, we found that this images where nonexistent outside of north Americas and other very specific areas, none of them being in Portugal.

Figure 5.1: Area in the city of Lisbon,Sentinel2 imagery gathered using *Direção geral do território* WebMaps service

Having found that none of the previous options would have been appropriate we investigated similarly [1] [5] to other projects using the google maps static API[4]. Even though this API provided us with images of satisfactory resolution for the area of study we found that our work would go against Google terms of service. Since the use of static imagery API seemed promising for this project we looked into other such API's terms of services and found two whom both had imagery of high

[1] https://www.sentinel-hub.com/
[2] https://www.planet.com/
[3] https://earthengine.google.com/
[4] https://developers.google.com/maps/documentation/maps-static/overview

enough resolution and quality for our project and whose terms of service allowed for its use in our project. This API's being the bing maps API[5] and the Mapbox API[6].

Both having comparable imagery, but in the end, we decided to use the bing maps API due to its higher number of monthly requests. We also found that the bing maps API had a higher number of features such as Bird eye view and other features that might not be useful in this project but could be useful in a myriad of related projects.

In terms of figure size and zoom, we found that a 365x365 resolution and 19 zooms would be appropriate for our intended use and chosen grid size and at the time of downloading images we intended to either train our models using images of size 256x256 or 128x128 and as a preliminary measure we decided do download our original images using a higher size than what would be needed while preserving the images aspect ratio.Another reason for choosing the 19 zoom size was that images captured using this zoom size had shown better results comparativly to other zoom size in other works.[1]

Figure 5.2: Number of road accidents per districts during the given time period

Figure 5.3: Accident location

5.2 Image preprocessing

Figure 5.4: Original image taken from the bing maps API

Figure 5.5: Preprocessed image

[5] https://www.bing.com/maps
[6] https://www.mapbox.com/

After gathering the images we applied as it's custom for deep learning models minimal preprocessing to the images, the only preprocessing done to the images consisted in removing the from the BingMaps watermark from the figures and resizing the original figures from their 365x365 to 128x128, all this preprocessing was done using the OpenCV library.

The 128x128 size was chosen because we fount it to be a good compromise between graphical fidelity and detail of the image and allowing us to use more images to train our models, since using a higher size would have obligated us to use fewer images and increased significantly the training time. Smaller sizes such as 64x64 did sacrifice too much visual fidelity and thus could hinder the performance of the trained models.

Chapter 6

Deep learning models

All models used to train the network were either set to have a random initialization or an initial initialization from pre-training on ImageNet. This chapter will cover both datasets using these two scenarios. Besides this, all models have the same input shape for a fair comparison, all using 128x128 images with 3 channels for RGB. Different sizes were used but will not be covered here since they didn't improve the results in any significant way.

6.1 First dataset

For this dataset, we added a dense layer with 4996 units and 'relu' activation and a dense layer with 2 units and 'softmax' activation function the models presented in this section were used for training our dataset due to having shown good performance on similar datasets for the ResNet50 and InceptionV3 models, and for their reduced size and faster training times for the mobilenetv2 model. Due to memory limitations, we only used 50% of the training and testing portions of the model around 30 thousand images for training and 8 thousand for testing.

In terms of metrics taking into account the domain of this, we can assume that the possible consequences of our model mistaking a dangerous cell with a safe one is far greater than the converse. Knowing the above we choose an ensemble of metrics that would allow us to choose to determine which of our models possesses the best performance using performance metrics that go beyond accuracy as well as better diagnose potential overfitting and performance problems in our models.

6.1.1 Random initialization

Results from models having the weight parameter set to 'none', this mean they will have a random initialization of weights.

Architecture	accuracy	precision	recall	f1-score	epochs
mobilenetv2	0.623	0.83	0.31	0.45	40
ResNet50	0.788	0.7826	0.7989	0.790	40
InceptionV3	0.782	0.81	0.738	0.7725	40

Table 6.1: Results attained using different deep learning architectures

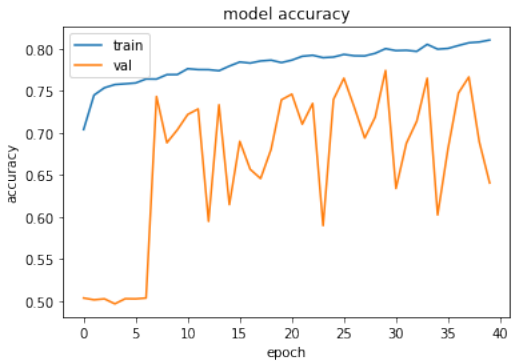

Figure 6.1: Accuracy mobilenet train and test

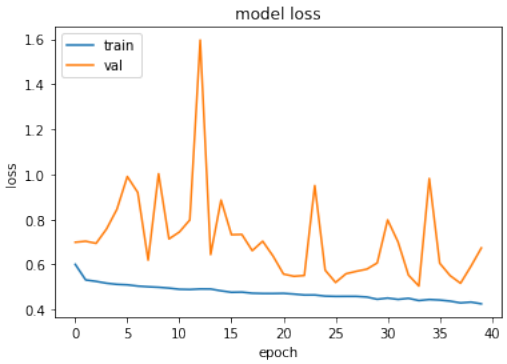

Figure 6.2: Loss mobilenet train and test

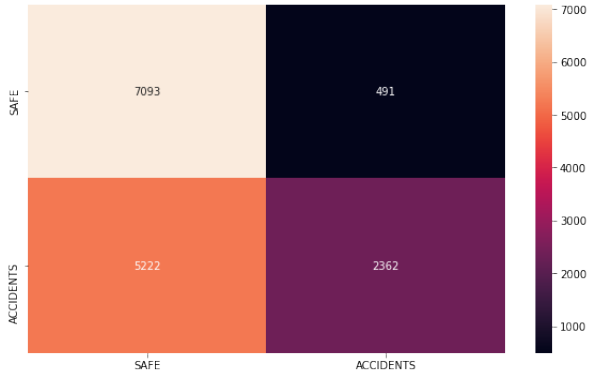

Figure 6.3: Confusion matrix for the mobilenet model

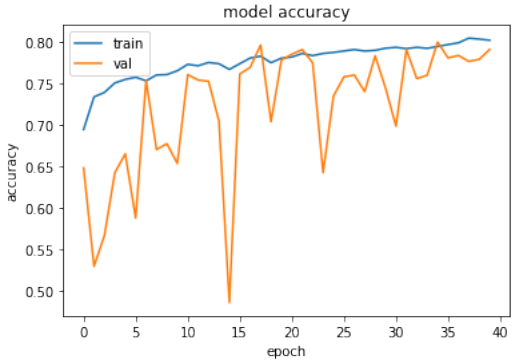

Figure 6.4: Accuracy ResNet50 train and test

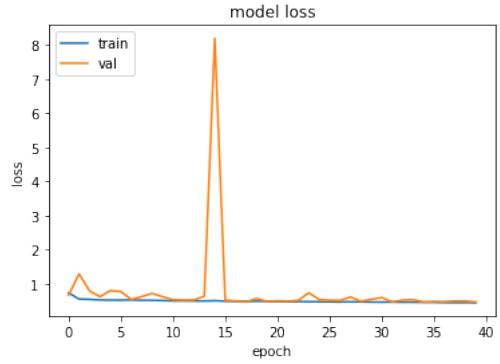

Figure 6.5: Loss ResNet50 train and test

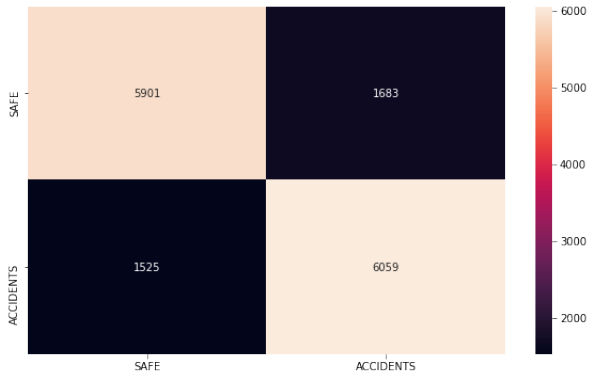

Figure 6.6: Confusion ResNet50 for the mobilenet model

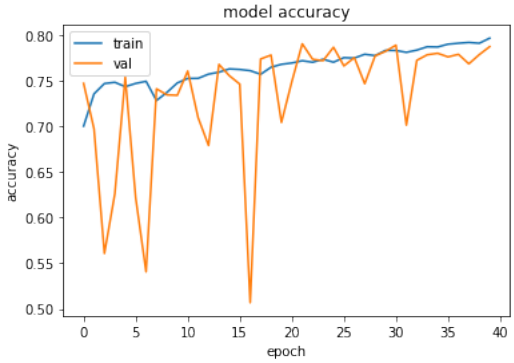

Figure 6.7: Accuracy InceptionV3 train and test

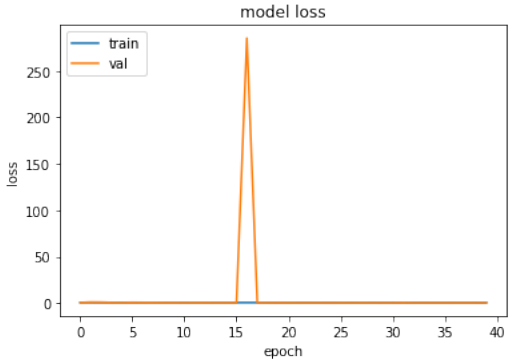

Figure 6.8: Loss InceptionV3 train and test

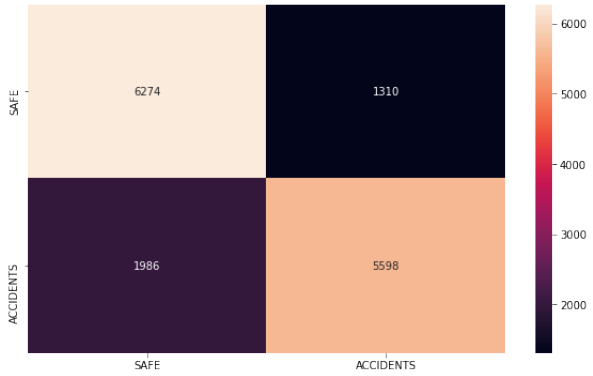

Figure 6.9: Confusion matrix for the InceptionV3 model

6.1.2 Transfer learning

Results from models having the weight parameter set to 'imagenet', this meaning their initial weights will be loaded from pre-training on ImageNet.We originally wanted to test the results of using transfer learning using the best-trained model, in this case, the Resnet50 but we where met whit frequent memory crashes when using this model with transfer learning with the same number of images as before. Since we didn't want to reduce the size of the dataset portion used for training since it could skew the results we decided to use the second-best model instead. We decided to train the model for 10 more epochs to better evaluate possible improvements with more training.

Architecture	accuracy	precision	recall	f1-score	epochs
InceptionV3	0.782	0.81	0.738	0.7725	50

Table 6.2: Results attained using different deep learning architectures

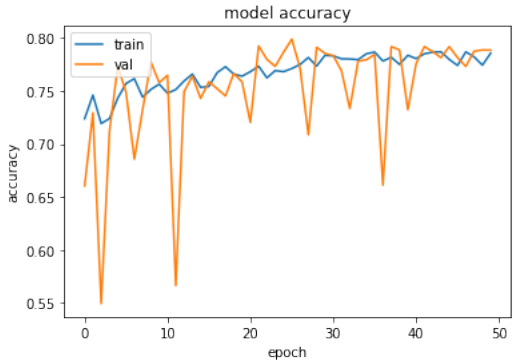

Figure 6.10: Accuracy InceptionV3 train and test

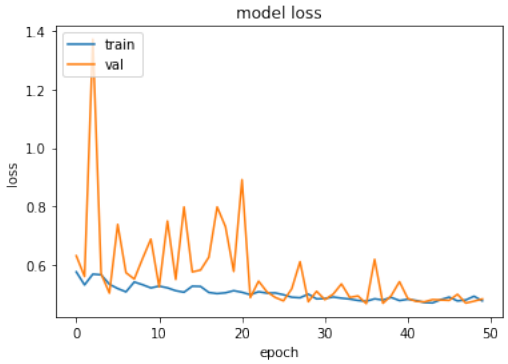

Figure 6.11: Loss InceptionV3 train and test

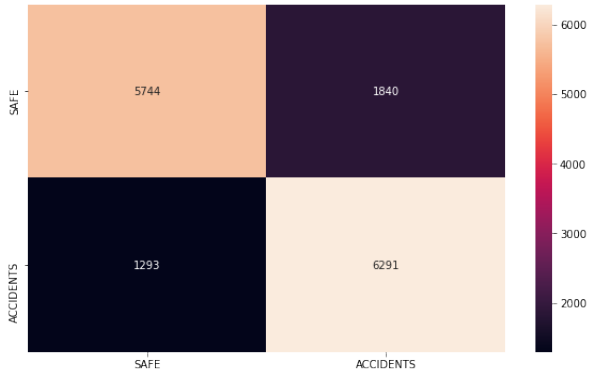

Figure 6.12: Confusion matrix for the InceptionV3 model with transfer learning

6.1.3 Analysis

Both networks ResNet50 and InceptionV3 show good results at classifying the images as places where traffic accidents could occur or not, using pre-trained (transfer learning) weights from ImageNet as well as using random initial weight values. MobilenetV2 didn't yield as good results overall, this is to be expected since the model is much smaller and faster to train and is directed to mobile environment usage.
Accuracy for the other two models sits at around 80% with the other metrics being very similar as the dataset was balanced before starting. Looking at the learning history we can predict that further training could potentially improve metrics a small portion, yet this could lead to overfitting of the model, so sitting at around 50 epochs may be the way to go .
Further data augmentation or tweaking the models could potentially yield better results, that being said 80% is close to an average Top-1 accuracy of these on ImageNet.
Comparatively to other works some of them possess similar results to ours [2] [1], we also found that the performance of our models can be partially blamed on our dataset since we were able to get better results using London imagery used in this project[2].

6.2 Second dataset

6.2.1 First version

For the first version of the second dataset, we only changed the last layer comparatively to the fist dataset using a layer with 3 units instead of 2.The size of the dataset is of around 8 thousand images for training and 2 thousand for testing.

In terms of metrics, we choose metrics that would allow us to evaluate the error in the categorizations and not only the number of wrong categorizations.

Initial results

Architecture	accuracy	mean absolute error	mean squared error	root mean squared error	epochs
MobileNet	0.357	0.75	0.984	0.99	40
ResNet50	0.357	0.75	0.984	0.99	40
InceptionV3	0.3511	0.821	1.16622	1.07	40

Table 6.3: Results attained using different deep learning architectures

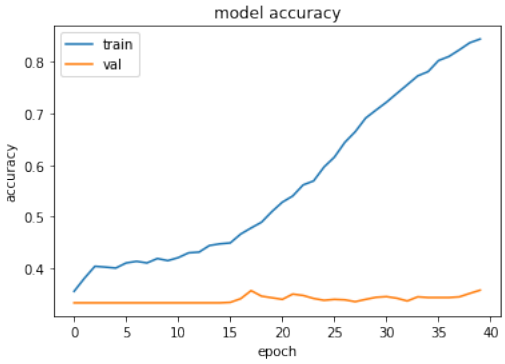

Figure 6.13: Accuracy mobilenet train and test

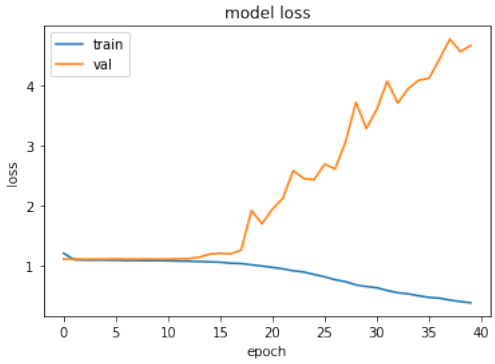

Figure 6.14: Loss mobilenet train and test

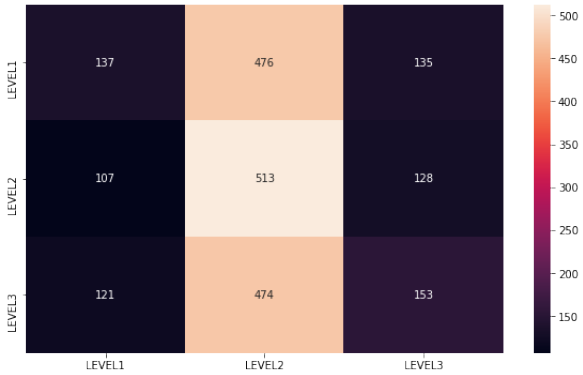

Figure 6.15: Confusion matrix for the InceptionV3 model with transfer learning

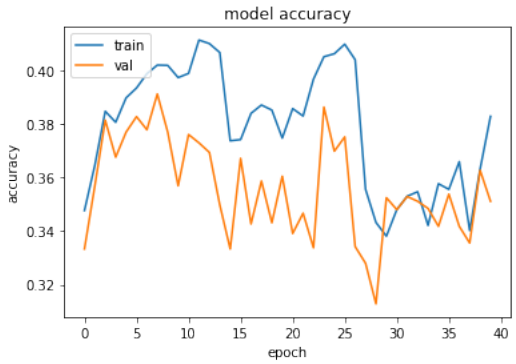

Figure 6.16: Accuracy InceptionV3 train and test

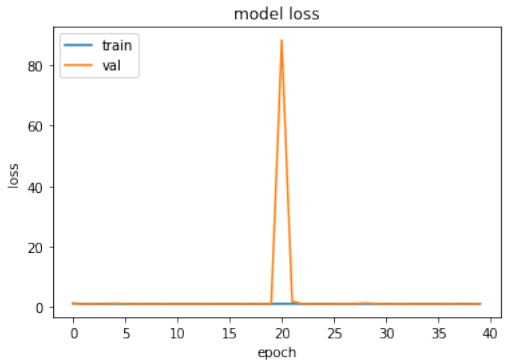

Figure 6.17: Loss InceptionV3 train and test

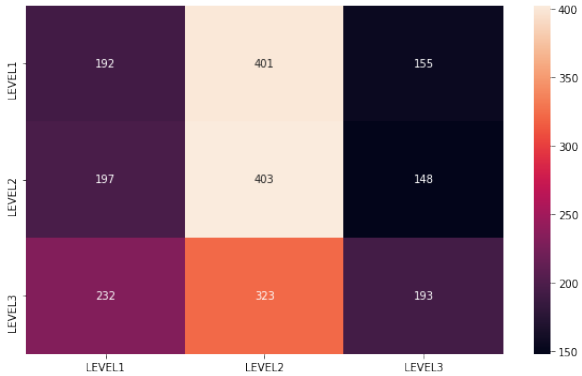

Figure 6.18: Confusion matrix for the InceptionV3 model without transfer learning and data augmentation

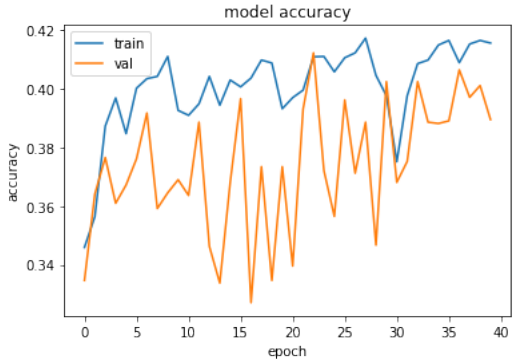

Figure 6.19: Accuracy ResNet50 train and test

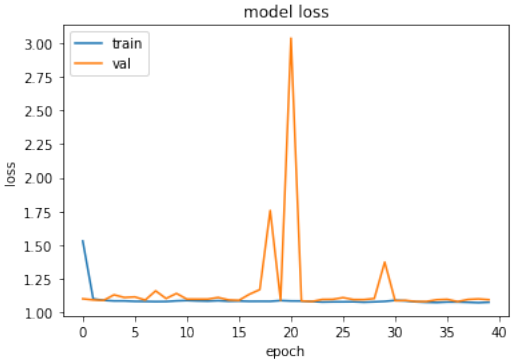

Figure 6.20: Loss ResNet50 train and test

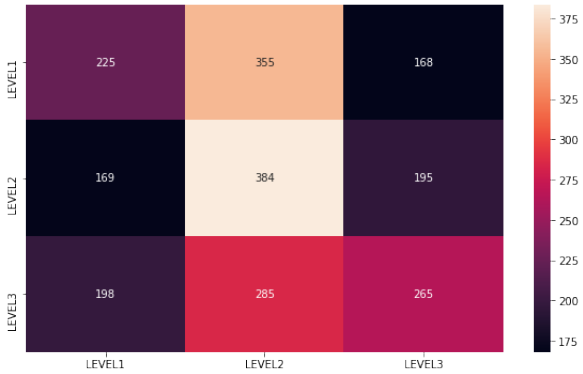

Figure 6.21: Confusion matrix for the resent model without transfer learning and data augmentation

Results with data augmentation and transfer learning

We then tried using transfer learning and data augmentation on the best model in this case the ResNet50 and trained it for 10 more epochs.

Architecture	**accuracy**	**mean absolute error**	**mean squared error**	**root mean squared error**	**epochs**
ResNet50 with data augmentation and tranfer learning	0.37	0.79	1.11	1.05	50
ResNet50 with tranfer learning	0.38	0.81	1.20	1.09	50

Table 6.4: Results attained using different deep learning architectures

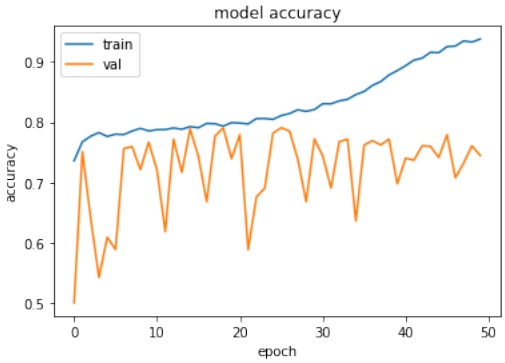

Figure 6.22: Accuracy ResNet50 d.a t.l train and test

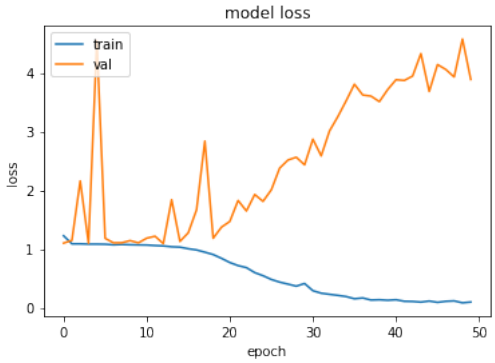

Figure 6.23: Loss ResNet50 d.a t.l train and test

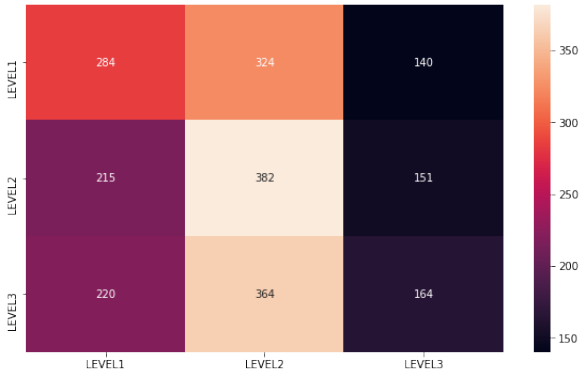

Figure 6.24: Confusion matrix for the ResNet50 with data augmentation and transfer learning train and test

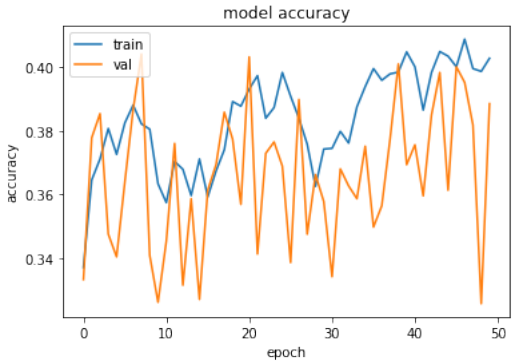

Figure 6.25: Accuracy ResNet50 d.a train and test

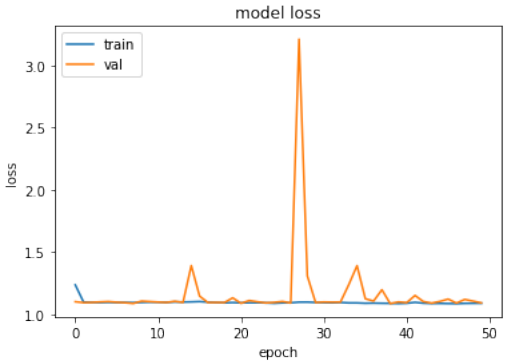

Figure 6.26: Loss ResNet50 d.a train and test

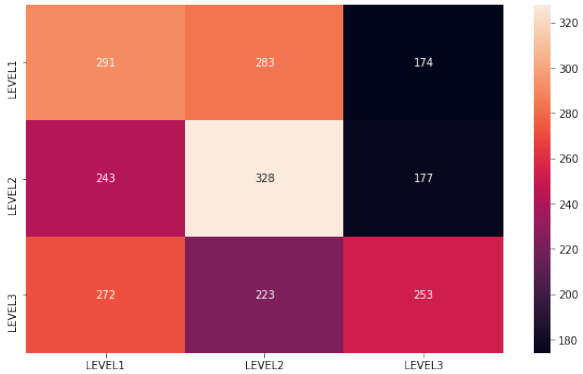

Figure 6.27: Confusion matrix for the ResNet50 model without transfer learning and with data augmentation

Analysis

As we can see the performance of the models trained with this dataset is bad across the board not reaching 40% accuracy even when using data augmentation and transfer learning accuracy, numerous factors could contribute to the poor performance being them the reduced dataset size, the similarity between images of different categories and possibly the fact that regression should have been used for training with this dataset.

6.2.2 Second version

For the second version of this dataset, we changed the last layer activation to linear instead of softmax and the size from 3 to 1.We changed the optimizer from 'adam' to 'rmsprop' also experimented using both the mean squared error and mean absolute

error loss functions and InceptionV3 and ResNet50 models, we found meager variances during training in both validation and training datasets, using both models and metrics. The dataset used had around 30 thousand elements and the model could have had overfitting problems due to the dataset being heavily imbalanced,but we still found the lack of learning with various metrics and loss functions to being something concerning and decided to not include the results due possibly of them being inaccurate and misleading.

6.3 Other models and work

We also trained an autoencoder to recognize images from our fist dataset. The motivation behind its creation was for use in the RiskMaps API since at the time we had intentions in creating those maps directly from reading GeoTiff files instead of reading a GeoJson file containing the specifications of a given area and using the bing maps API to gather images in that location and thus the autoencoder could have been useful in discerning aerial images from others.

We also visualize the activation in some of the trained models but found the results to not be useful.

Chapter 7

Deployment

7.1 Using trained models to predict the risk associated with a given area

Having trained our models we decided to create a couple of demo applications that would showcase possible uses of these models in the real world. The first step to produce use full data from our trained models was to create a pipeline that would allow for the prediction of a given location risk accident.

To do the above we created a library that allows users to get the accident risk associated with a given location whit its geographical coordinates. This is realized by using the BingMaps API to request the image associated with the area in the same way and with the same parameters that we used to first collect the images and then preprocessing the given image similarly in the same way we preprocessed the images before feeding them in our models. After these steps, we use the first model to predict if the image is in category ACCIDENTS or SAFE. If the image its in the SAFE category, both the low and high accuracy predictions equal 0 if not the low accuracy prediction equals 1 and the high accuracy prediction equals the risk level predicted by the second model. We used the first version of the second model for our risk predictions but could adapt our programs to output the second value in a continuous range instead of an integer associated with the risk level.

```
{
    "settings":{
        "model1": "trained_models/inceptionv3_tl.hdf5",
        "model2": "trained_models/inception_v3_da_dataset2.hdf5",
        "cell_size_x": 200,
        "cell_size_y": 200
    }

}
```

Listing 7.1: setting file allowing the specification of both models and cell size

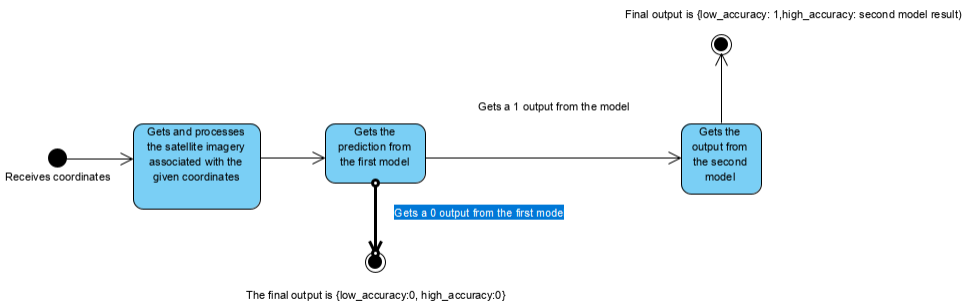

Figure 7.1: Risk calculating process for a given location

7.2 RiskMaps

One of the applications created using our models was a RiskMaps generating library this library allows users to generate GeoTiffs representing the predicted risk of the area regions using a color scheme associating each risk level with a specific color.

The first step to do the above consists of getting the specified are maximum and minimum latitude and longitude. After doing so we divide the area into squares with the specified area in the settings file. We then compute the associated high accuracy risk with the library created for this purposed described in the previous section.

After calculating the risks we find the corresponding areas in the figure RiskMap for each of the regions and save them in a dictionary. Then if the user wants the output the result as a GeoJson file containing the information used to generate RiskMaps the dictionary is written to a GeoJson file if not the *osgeo*[1] library is used to generate the GeoTiff file with the maximum and minimum longitude and latitude and the generated dictionary. The user can also, later on, use generated GeoJson files to create RiskMaps.

We included in our project repository a demo file containing detailed examples on how to use this library.


```
{"type":"FeatureCollection",
"features":[
{"type":"Feature","properties":{},
"geometry":{"type":"Polygon",
"coordinates":[[[−8.441662788391113,41.643894848472634],
[−8.424990177154541,41.643894848472634],
[−8.424990177154541,41.651350732638456],
[−8.441662788391113,41.651350732638456],
[−8.441662788391113,41.643894848472634]]]}}]}
```


Listing 7.2: GeoJson referencing the area in figures 7.2 and 7.3

Figure 7.2: Urban area in the district of Braga

Figure 7.3: Rural are in the district of Braga


```
{
    "2": [
        [
            0.0,
            0.0,
            247.31378301208196,
            147.44285197989652
        ],
        ...
        ],
        ...
    "1": [
            ...
```


[1] https://www.osgeo.org/

```
        [
            989.254665596184,
            589.8371013430248,
            1024,
            737.296376678781
        ]
    ]
}
```

Listing 7.3: Output resulting from a call to the RiskApi with the coordinates associated with the area represented in 7.5

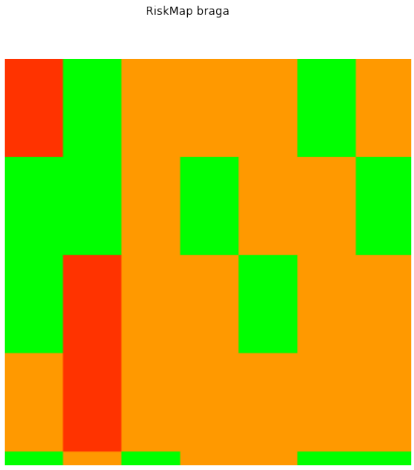

Figure 7.4: RiskMap for region associated with the region in figures 7.2 and 7.3

7.3 RiskApi

We found as well that one possible use of the trained models would be a REST API that would allow for users to get risks associated with given coordinates.

We implemented this functionality using the *flask*[2] library.The created API works by receiving calls from users with latitude and longitude coordinates and then computes the associated high accuracy risk for this coordinates using the same process used in the previous application and then sends the results back to the user.

We included in our project repository a demo file containing detailed examples on how to use this library.

[2] `https://flask.palletsprojects.com/en/1.1.x/`

Braga urban

Figure 7.5: Urban area in the district of Braga

Braga rural

Figure 7.6: Rural are in the district of Braga

```
{
    "low accuracy": "1",
    "detailed accuracy": "1"
}
```

Listing 7.4: Output resulting from a call to the RiskApi with the coordinates associated with the area represented in 7.5

```
{
    "low accuracy": "0",
    "detailed accuracy": "0"
}
```

Listing 7.5: Output resulting from a call to the RiskApi with the coordinates associated with the area represented in 7.6

Chapter 8

CONCLUSIONS

In terms of possible improvements, we believe that finding better data sources could yield better results, that said we did not find data related to road accidents for the Portuguese territory freely available and procuring better data could not be feasible since the data used already came from a governmental organ and other sources of similar data to the one used for our project might come from source as ours. We could also improve the formula that we used to compute the accident risk for a given area and instead of using a naive formula that accounts only for the raw number of accidents, we could also take into account other factors such as the population density and even give different weights to accidents according to various factors such as their date and severity, that said not all of these improvements where possible using our dataset.

We could have tried other deep learning approaches such as combining numerical and categorical data with our trained CNN's since this has being done in other projects with positive results.[2] Nonetheless, we find that this project greatest flaw entails in the results attained with both the second dataset versions.

Generally speaking, we found that the work was done as part of this project as a relatively high potential to be applied in real-world use cases and that this project despite some difficulties doing this project helped us gain insights and experiences in a variety of areas related to the process of collecting, analyzing and transforming data, deep learning models, satellite and aerial imagery, Geographic Information Systems and even some experience with the creation of REST API's.

Bibliography

[1] Yoshikazu Miyanaga Alameen Najjar Shun'ichi Kaneko. *Combining satellite imagery and open data to map road safety*. 2017. URL: https://dl.acm.org/doi/10.5555/3298023.3298224.

[2] Sabatino Chen Laura Lewis. *Teaching a neural network to see roads*. 2019. URL: https://towardsdatascience.com/teaching-a-neural-network-to-see-roads-74bff240c3e5.

[3] World Health Organization. *The top 10 causes of death*. 2018. URL: https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death.

[4] Público. *Número de mortos nas estradas diminuiu em 2019, mas houve mais acidentes e feridos graves*. 2020. URL: https://www.publico.pt/2020/01/02/sociedade/noticia/mortos-estradas-diminuiram-2019-acidentes-feridos-graves-aumentaram-1899039.

[5] Daniel Wilson. *Using Machine Learning to Predict Car Accident Risk*. 2019. URL: https://medium.com/geoai/using-machine-learning-to-predict-car-accident-risk-4d92c91a7d57.