High Level Design (HLD)

Insurance Premium Prediction

Document Version Control

Date Issued	Version	Description	Author
11/01/2023	1	Initial HLD — V1.0	Aakash Pal

Contents

Abstract

In this project, we aim to predict insurance premiums for individuals by analyzing their health data. We employed models like the Regression model, Tree algorithm, and Bagging and Boosting Algorithm to accomplish this. We used these models to compare and contrast their performance.

The training dataset was used to train the model, and the predictions made by the model were compared with actual data to test and verify the model's accuracy. After reaching the accuracy of all the models, it was determined that the GradientBoostingRegressor and RandomForestRegressor algorithms performed better than the remaining models.

Ultimately, we found that the GradientBoostingRegressor algorithm was the best suited for this task as it provided the best evaluation score compared to the other models.

1 Introduction

1.1 Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level.

The HLD will:

- Present all of the design aspects and define them in detail
- Describe the user interface being implemented
- Describe the hardware and software interfaces
- Describe the performance requirements
- Include design features and the architecture of the project
- List and describe the non-functional attributes like:
    - Security
    - Reliability
    - Maintainability
    - Portability
    - Reusability
    - Application compatibility
    - Resource utilization
    - Serviceability

1.2 Scope

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system.

1.3 Definitions

Term	*Description*
UGV	Unmanned Ground Vehicle
Database	Collection of all the information monitored by this system
IDE	Integrated Development Environment
AWS	Amazon Web Services

2 General Description

2.1 Product Perspective

The UGV based Surveillance solution system is a deep learning-based object detection model which will help us to detect the anomalies in the society and take the necessary action.

2.2 Problem Statement

To create an AI solution for surveillance using rover or UGV and to implement the following use cases.

- To detect mob (illegal) activities and inform police.
- To detect disasters (like fire, smoke, etc..) and send details to concerned authorities.
- To detect a medical emergency (accident, etc..) and take emergency action (call ambulance and raise an alarm to the nearest hospital) for help.

2.3 Proposed Solution

The solution proposed here is an UGV (Unmanned Ground Vehicle) based Surveillance (Unmanned Ground Vehicle) can be implemented to perform above mention use cases .In first case, if UGV detects any mob(illegal) activities at a particular location it will take photos or video for the evidence and send the police the current location where the mob activities are taking place, further in the second use case, if UGV detects any natural or human made disasters(fire, smoke, etc..) the UGV detects with its sensors and will send details to concerned authorities and lastly in the final use case of UGV, if it finds any medical emergency (accident, etc..)itwill take rapid action (call ambulance and alert the nearest hospital)for swift help.

2.4 Further Improvements

UGV can be added with more use cases like weather detection, live temperature to detect and record the temperature of that locality. UGV can also be synchronized with UAV (Unmanned Aerial Vehicle) for better and fast response or action, with help of UGV and UAV synchronized it can be implemented in the other domains like mining, agriculture.

2.5 Technical Requirements

This document addresses the requirements for detecting the anomalies in the society at early stages and recommending the necessary and rapid action to avoid imbalance in the harmony of the society. Mobile platforms like Ground robots should be used for this purpose. Ground Robots can be based on wheels, tracks, or legs.

- UGV should be able to move on steps and various terrains. Wheel robots or Wheel-Legged robots would better fit kind of tasks.
- These UGVs' should include many sensors like stereo vision cameras, panoramic camera, thermal and infrared detecting systems.

- These can be battery powered or solar powered.
- UGVs' should be equipped with proper computing power to process the images or video of anomalies it had detected.

2.6 Data Requirements

Data requirement completely depend on our problem statement.

- We need images data that is balanced and must have at least 1000 images.
- We require at least 30- 40 images for each class label with annotation.
- An image is nothing more than a two-dimensional array of numbers(pixels)
- Pixel value ranging between 0 to 255
- It is defined by the mathematical function (x, y), the value off(x, y) at any point is giving the pixel value at that point of an image
- Original image is in the format of (width, height, no of RGB channels).

There are numerous image file formats out there so it can be hard to know which file type best suits your image needs (on your requirement).

a TIFF — Tagged image file format
o BMP — Bitmap image file form
a JPEG - Joint photographic experts' groups
o GIF - graphics interchange format
a PNG — portable network graphics
a EPS — encapsulated post script
a RAW image files

- Tiffs are great for printing. These are lossless image files meaning they don't need to compress or lose any image quality or information. These format images are high quality images.
- bmp format developed by Microsoft for windows. There is no compression or information loss; this format is generally recommended for high quality scans.
- JPEG is a lossy format meaning that the image is compressed to make a smaller file but this loss is not noticeable.
- JPEG is a very popular format for digital cameras.
- GIFs are widely used for web graphics because they are limited to only 256 colours, can allow for transparency and can be animated. These types of files are typically small in size and very portable.
- PNG are a lossless image format; these files are able to handle up to 16 million colours unlike the 256 colours supported by GIF.
- EPS is a common vector type file.

- RAW images that are unprocessed that have been created by a camera or scanner. Digital cameras can shoot in raw, mostly used in photography.

If the data is in video format like (MP4) convert into images based on FPS (no. of frames displayed per second) in real time processing. There are number of tools to convert videos into images. Using cv we can convert video into images

2.7 Tools used

Python programming language and frameworks such as NumPy, Pandas, Scikit-learn, TensorFlow, Keras and Roboflow are used to build the whole model.

- PyCharm is used as IDE.
- For visualization of the plots, Matplotlib, Seaborn and Plotly are used.
- AWS is used for deployment of the model.
- Tableau/Power BI is used for dashboard creation.
- MySQL/MongoDB is used to retrieve, insert, delete, and update the database.
- Front end development is done using HTML/CSS
- Python Django is used for backend development.
- GitHub is used as version control system.

2.8 Constraints

The UGV based Surveillance solution system must be user friendly, as automated as possible and users should not be required to know any of the workings.

2.9 Assumptions

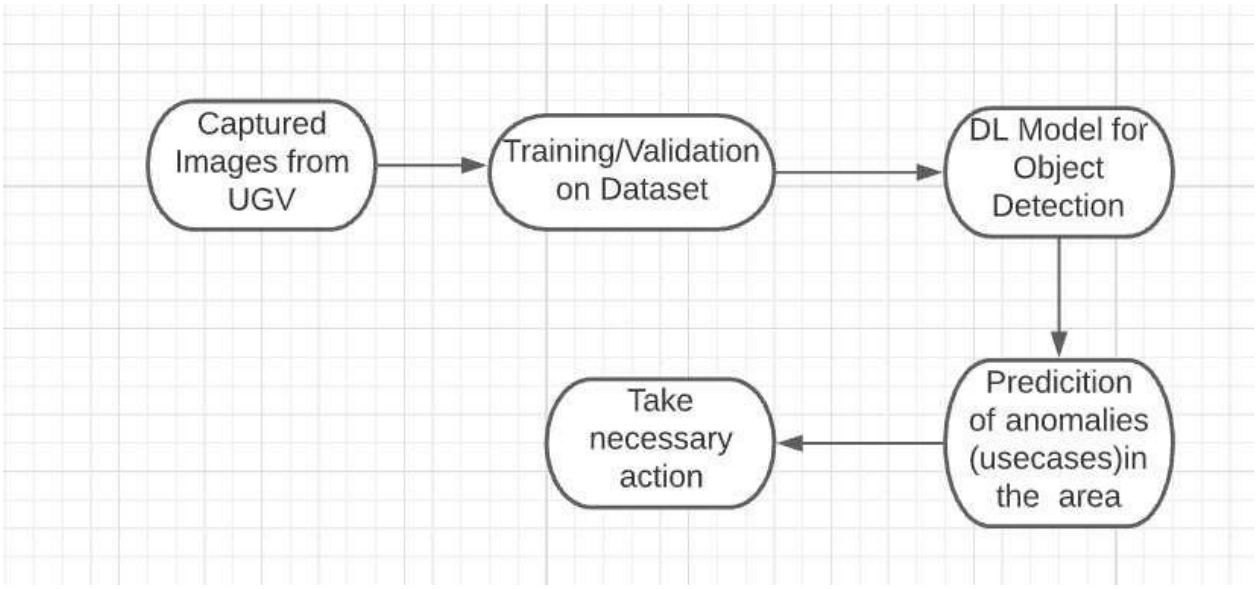
The main objective of the project is to implement the use cases as previously mentioned (2.2 Problem Statement) for new dataset that comes through UGV vehicle which has camera installed for capturing the live videos. Deep Learning based object detection model is used for detecting the above-mentioned use cases based on the input data. It is also assumed that all aspects of this project have the ability to work together in thewythe designer is expecting.

3 Design Details

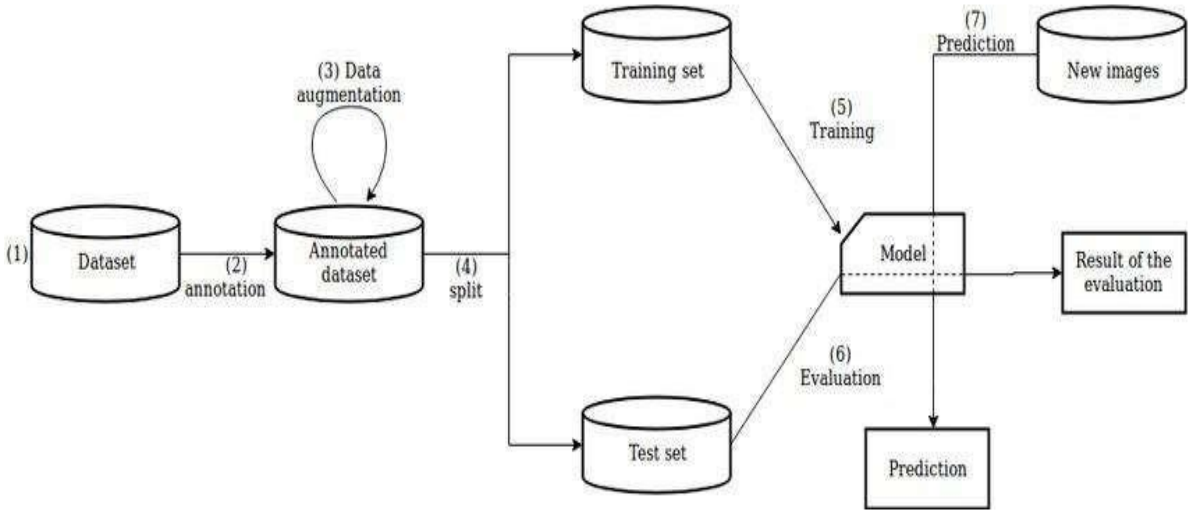
3.1 Process Flow

For identifying the different types of anomalies, we will use a deep learning base model. Below is the process flow diagram is as shown below.

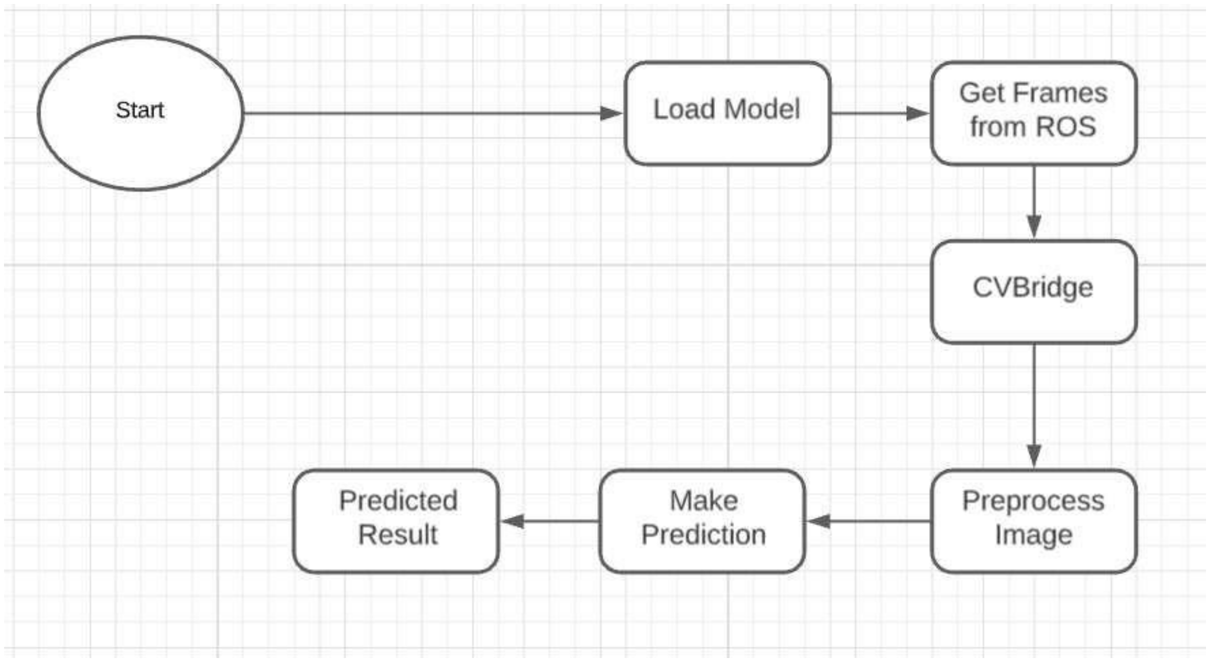
Proposed methodology


Captured Images from UGV
Training/Validation on Dataset
DL Model for Object Detection
Predicition of anomalies (usecases)in the area
Take necessary action

3.1.1 Model Training and Evaluation



3.1.2 Deployment Process



3.2 Event log

The system should log every event so that the user will know what process is running internally.

Initial Step-By-Step Description:

1. The System identifies at what step logging required
2. The System should be able to log each and every system flow.
3. Developer can choose logging method. You can choose database logging/ File logging as well.
4. System should not hang even after using so many loggings. Logging just because we can easily debug issues so logging is mandatory to do.

3.3 Error Handling

Should errors be encountered, an explanation will be displayed as to what went wrong? An error will be defined as anything that falls outside the normal and intended usage.

4 Performance

The UGV based surveillance solution is used for detection of anomalies in the society whenever UGV detects any anomalies (mob, medical emergency, fire, smoke, etc...) it will inform concern authorities and takes necessary action, so it should be as accurate as possible. So that it will not mislead the concern authorities (like hospitals, cops, etc..). Also, model retraining is very important to improve the performance.

4.1 Reusability

The code written and the components used should have the ability to be reused with no problems.

4.2 Application Compatibility

The different components for this project will be using Python as an interface between them. Each component will have its own task to perform, and it is the job of the Python to ensure proper transfer of information.

4.3 Resource Utilization

When any task is performed, it will likely use all the processing power available until that function is finished.

4.4 Deployment

Microsoft Azure

Google Cloud

amazon webservices™

5 Dashboards

Dashboards will be implemented to display and indicate certain KPIs and relevant indicators for the unveiled problems that if not addressed in time could cause catastrophes of unimaginable impact.

As and when, the system starts to capture the historical/periodic data for a user, the dashboards will be included to display charts over time with progress on various indicators or factors.

5.1 KPIs (Key Performance Indicators)

1. Key indicators displaying a summary of the anomaly detection in the society/area.
2. Time and workload reduction using the UGV based surveillance.
3. To detect mob (illegal) activities and inform police.
4. On time alert to nearest hospital on medical emergency (accident).
5. Taking adequate evidence of mob.
6. Send disaster details to concerned authorities.
7. Display of battery life and percentage of UGV.
8. Distance travelled by UGV.
9. Get the exact location of UGV.

6 Conclusion

The Designed UGV (Unmanned Ground Vehicle) will detect an anomaly in the locality based on various anomalies data used to train our algorithm, so we can identify the imbalance in the society in early stages and can take necessary action to stop them immediately, so we can have a pleasant environment in that area or location.

7 References

1. https://en.wikipedia.org/wiki/Unmanned_ground_vehicle
2. Google.com for images of UGV hardware.
3. https://www.ros.org/