



MARMARA UNIVERSITY

FACULTY OF ENGINEERING

CSE4288

Introduction to Machine Learning

TERM PROJECT

Final Project Report

Group: 2

Table of Contents

1. Abstract	3
2. Introduction	3
3. Methodology	3
3.1 Preprocessing	3
3.1.1 Image Embedding	3
3.1.2 YOLO Preparation	4
4. Results	6
4.1 Decision Tree	6
4.2 Naïve Bayes.....	7
4.3 K-NN.....	8
4.4 Logistic Regression	9
4.5 YOLO.....	10
5. Discussion	11
5.1 Interpretation of Results	11
5.2 Challenges	13
6. Conclusion.....	14
7. References	15

1. ABSTRACT

This project compares different machine learning models for crosswalk detection in images by preselected evaluation metrics.

2. INTRODUCTION

Crosswalk detection models have huge importance for self-driving car systems. However, in those systems, most of the time focus is on detecting outer objects such as other vehicles, pedestrians etc. Although it has great significance to detect crosswalks to create a safe driving environment for many people, it is sometimes overlooked.

Therefore, in this project, we aim to understand the concept of object recognition with developing various models for crosswalk detection with using BDD100K dataset [1]. By doing this, we also aim to have a better understanding of machine learning studies.

3. METHODOLOGY

3.1 Preprocessing

3.1.1 *Image Embedding*

Image Embedding [2] is the process of converting an image to a high-dimensional vector of numbers. These numbers are obtained from models pre-trained on large datasets (the model that we used ResNet50 [3] trained on ImageNet) and represent information about the image, such as shapes, textures, and relationships between objects.

Classical machine learning models like K-Nearest Neighbors (k-NN), Naive Bayes, Decision Tree and Logistic Regression [4] cannot process raw images. Image embedding is done to convert high-dimensional, unstructured raw image data into numerical vectors. This allows simple models to effectively classify or analyse images without needing large datasets or computationally expensive deep learning training.

The image embedding process begins with dataset preprocessing, where the dataset was divided into training and validation sets, 80% of the images split for training and 20% for validation. The separation was made by saving the image names into two separate text files as train_images.txt and val_images.txt

For feature extraction, a pre-trained ResNet50 model from PyTorch [5] was used. The model's fully connected layer was removed to use it as a feature extractor. Images were pre-processed by resizing them to 224x224 pixels and applying normalization based on ImageNet standards. The pre-processed images were then passed through the feature extractor, and the resulting feature vectors were flattened and saved as .npy files for both training and validation sets.

This process successfully converted images into feature vectors, enabling their use in machine learning tasks. With using a pre-trained ResNet50 model, we achieved efficient and reliable feature extraction. Missing images and processing errors were displayed during execution, providing clarity for troubleshooting.

3.1.2 YOLO Preparation

“You Only Look Once” (YOLO) [6] preparation is performed because we need to process our raw data in a way that can be the input for our YOLO model training. YOLO dataset needs image labels in *x center, y center, width and height* format with this information in this specific order with normalization.

However, in our dataset, the crosswalk label includes the vertices of the 2D polygon, indicating the location of the object inside the image. In Figure 1, an example of crosswalk labelling is shown.

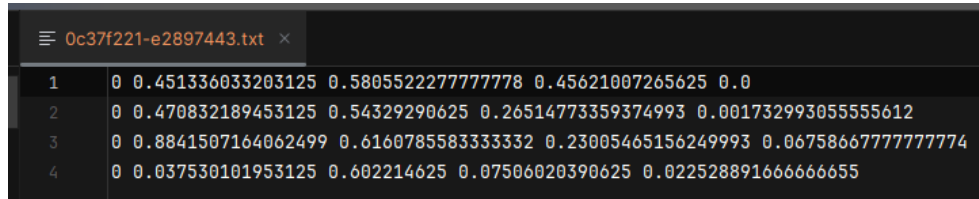
```
"category": "crosswalk",  
"poly2d": [  
  {  
    "vertices": [  
      [  
        490.49528,  
        419.568264  
      ],  
      [  
        576.406877,  
        416.571348  
      ]  
    ]  
  },  
]
```

Figure 1: Crosswalk labelling of an image

The labels are given as 2D polygons using vertices. However, YOLO's bounding boxes method is compatible with rectangular formed coordination as stated above. First thing to do is to convert these polygon vertices into rectangular form. This rectangle is defined by *x center, y center, width and height* of the crosswalk boundaries.

Final Project Report

After the conversion, the information about the image needed to be written in YOLO format, including the class ID, width and height of the crosswalk. In this project, the class ID of crosswalk is set as 0. Additionally, after calculating the width and height from the rectangular coordinates, these values should be normalized and recorded in txt file for each image. An example for YOLO format txt file is given in Figure 2.



1	0	0.451336033203125	0.5805522277777778	0.45621007265625	0.0
2	0	0.470832189453125	0.54329290625	0.26514773359374993	0.001732993055555612
3	0	0.8841507164062499	0.6160785583333332	0.23005465156249993	0.06758667777777774
4	0	0.037530101953125	0.602214625	0.07506020390625	0.022528891666666655

Figure 2: YOLO format txt file of training image

4. RESULTS

4.1 Decision Tree

Figure 3 gives the confusion matrix of our model with Decision Tree classification. In this case, the image is classified as either a crosswalk exists or not.

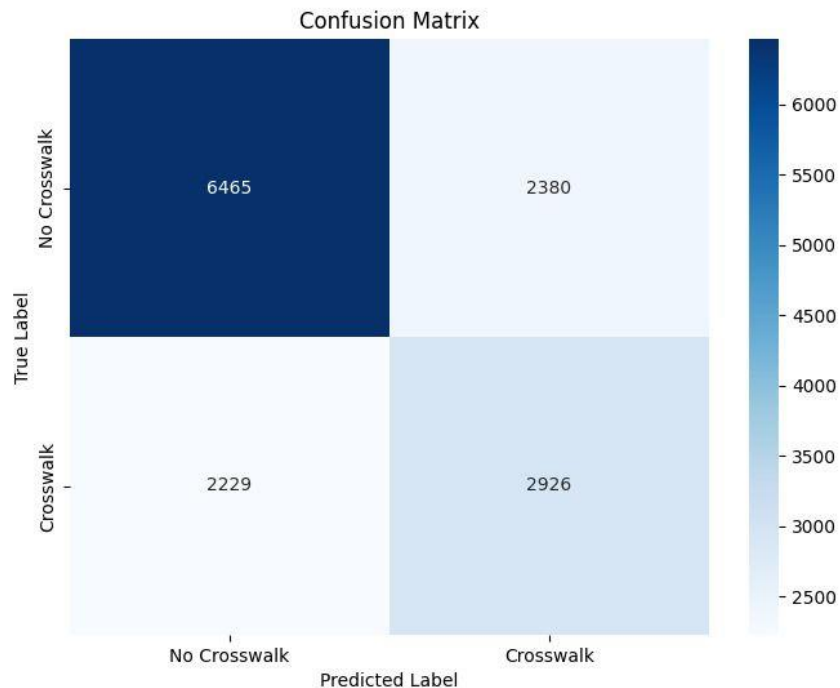


Figure 3: Confusion matrix of Decision Tree

Table 1 gives the evaluation results of our model with Decision Tree classification.

Table 1: Evaluation results of Decision Tree

	Precision	Recall	F1-Score	Support
No Crosswalk	0,74	0,73	0,74	8845
Crosswalk	0,55	0,57	0,56	5155
Macro avg.	0,65	0,65	0,65	14000
Weighted avg.	0,67	0,67	0,67	1400

Accuracy of the model with Decision Tree classification is 67,08%.

4.2 Naïve Bayes

Figure 4 gives the confusion matrix of our model with Naïve Bayes classification. In this case, the image is classified as either a crosswalk exists or not.

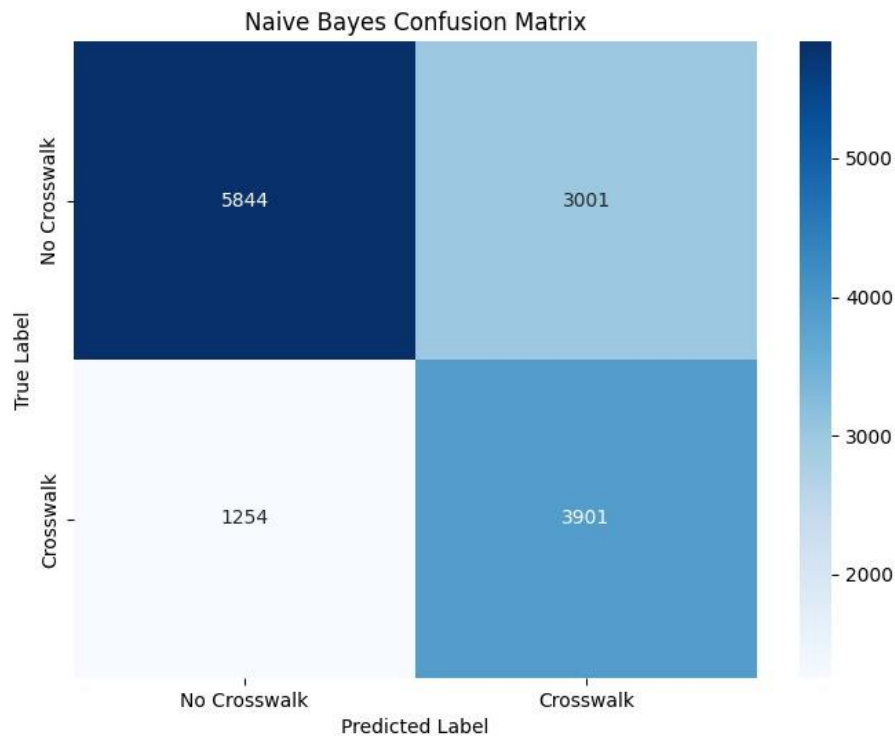


Figure 4: Confusion matrix of Naïve Bayes

Table 2 gives the evaluation results of our model with Naïve Bayes classification.

Table 2: Evaluation results of Naïve Bayes

	Precision	Recall	F1-Score	Support
No Crosswalk	0,80	0,66	0,73	8845
Crosswalk	0,57	0,76	0,65	5155
Macro avg.	0,69	0,71	0,69	14000
Weighted avg.	0,73	0,70	0,70	1400

Accuracy of the model with Naïve Bayes classification is 69,61%.

4.3 K-NN

Figure 5 gives the confusion matrix of our model with k-NN classification. In this case, the image is classified as either a crosswalk exists or not.

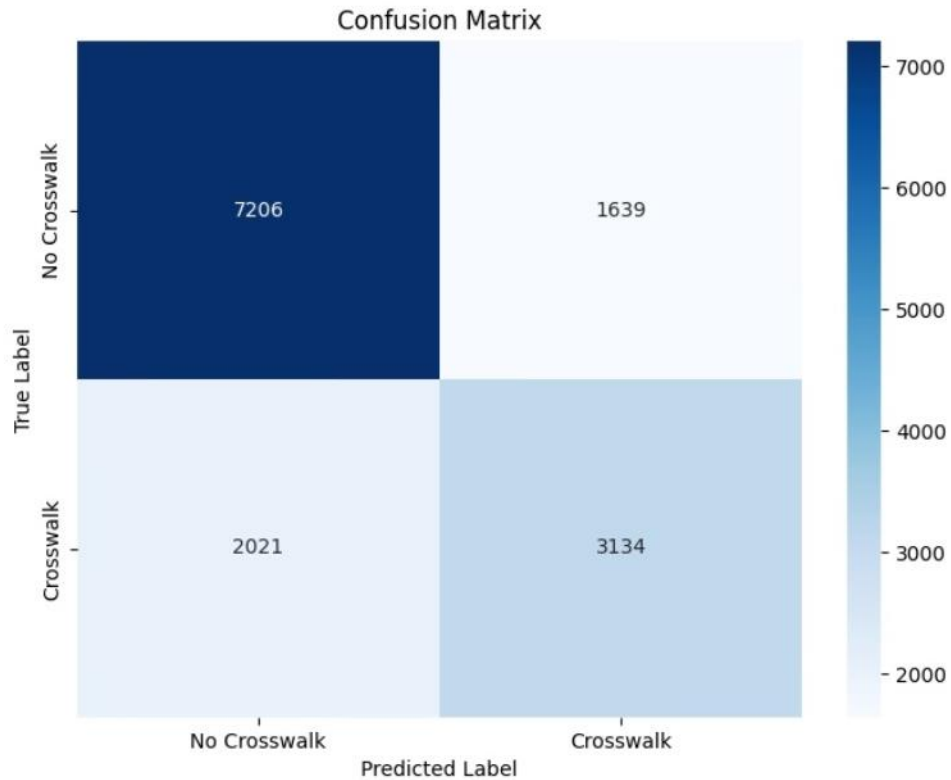


Figure 5: Confusion matrix of k-NN

Table 3 gives the evaluation results of our model with k-NN classification.

Table 3: Evaluation results of k-NN

	Precision	Recall	F1-Score	Support
No Crosswalk	0,78	0,81	0,80	8845
Crosswalk	0,66	0,61	0,63	5155
Macro avg.	0,72	0,71	0,71	14000
Weighted avg.	0,74	0,74	0,74	1400

Accuracy of the model with k-NN classification is 73,86%.

4.4 Logistic Regression

Figure 6 gives the confusion matrix of our model with Logistic Regression classification. In this case, the image is classified as either a crosswalk exists or not.

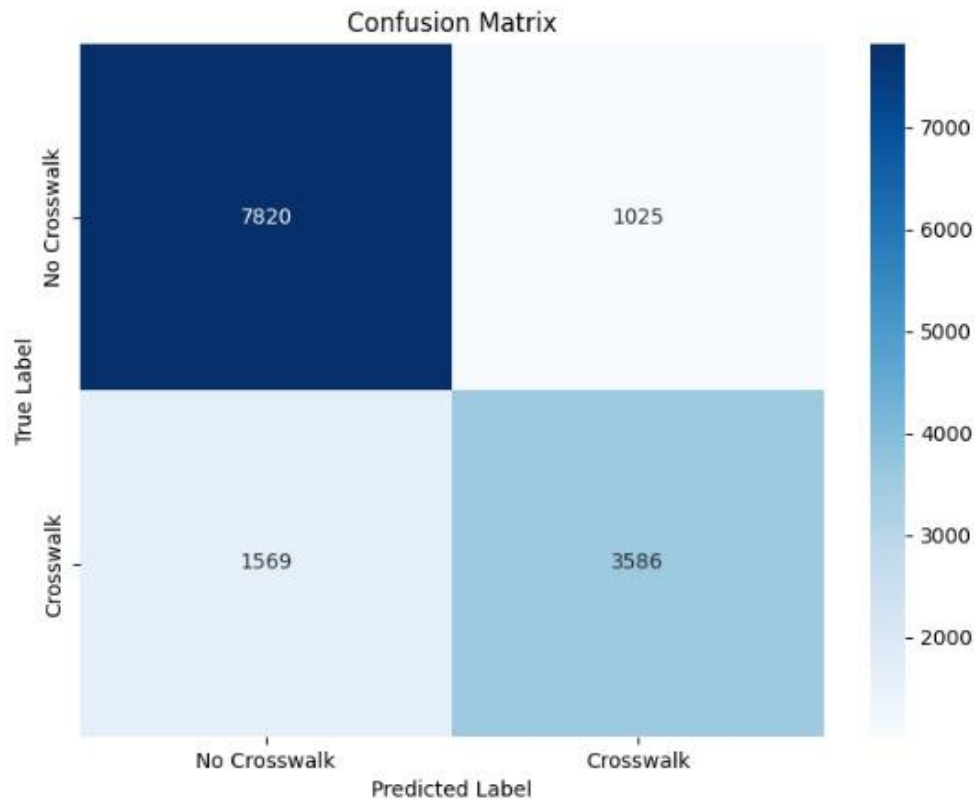


Figure 6: Confusion matrix of Logistic Regression

Table 4 gives the evaluation results of our model with Logistic Regression classification.

Table 4: Evaluation results of Logistic Regression

	Precision	Recall	F1-Score	Support
No Crosswalk	0,83	0,88	0,86	8845
Crosswalk	0,78	0,70	0,73	5155
Macro avg.	0,81	0,79	0,80	14000
Weighted avg.	0,81	0,81	0,81	1400

Accuracy of the model with Logistic Regression classification is 81,45%.

4.5 YOLO

Figure 7 indicates the confusion matrix of our YOLO model. Crosswalk is the area of the image that either has a crosswalk or a crosswalk is detected. Background means either the image has no crosswalk or no crosswalk is detected, meaning the part of the image is detected as background.

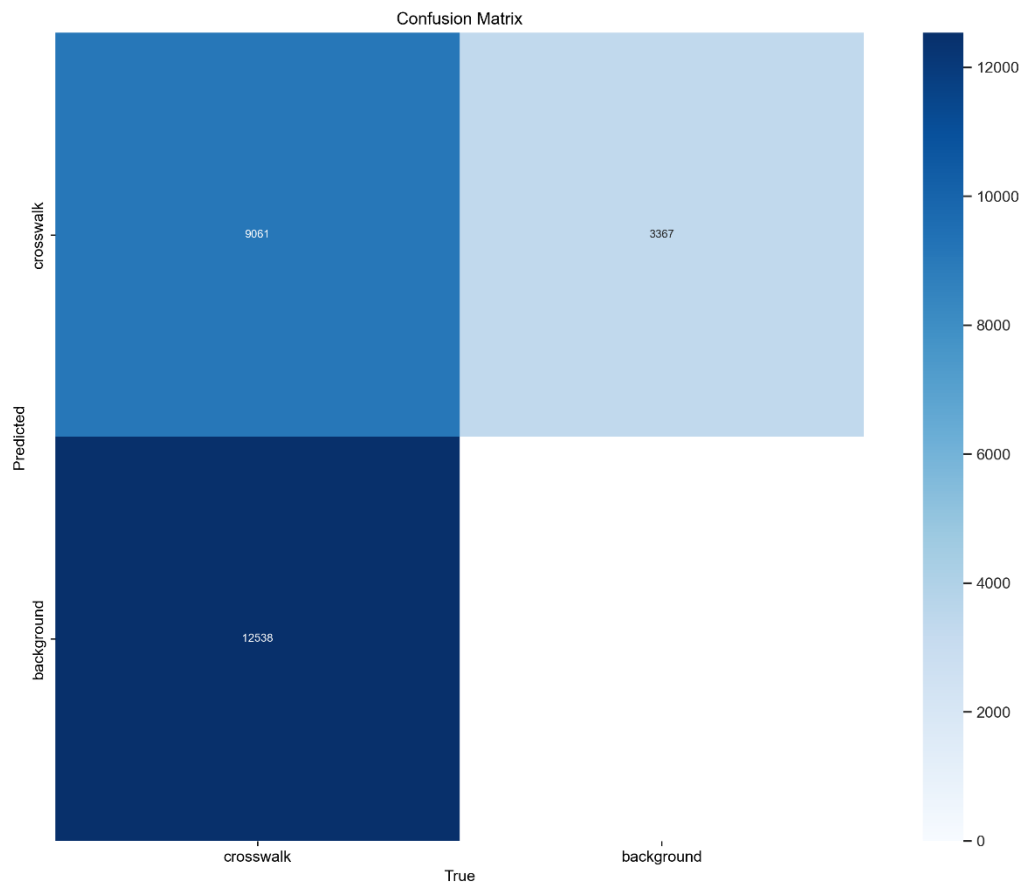


Figure 7: Confusion matrix of YOLO

Figure 8 indicates the confidence curve of the YOLO model. Currently the confidence level is quite low with a value of 0,5. This means that the model needs further optimization. The optimization steps are stated in Section 3.

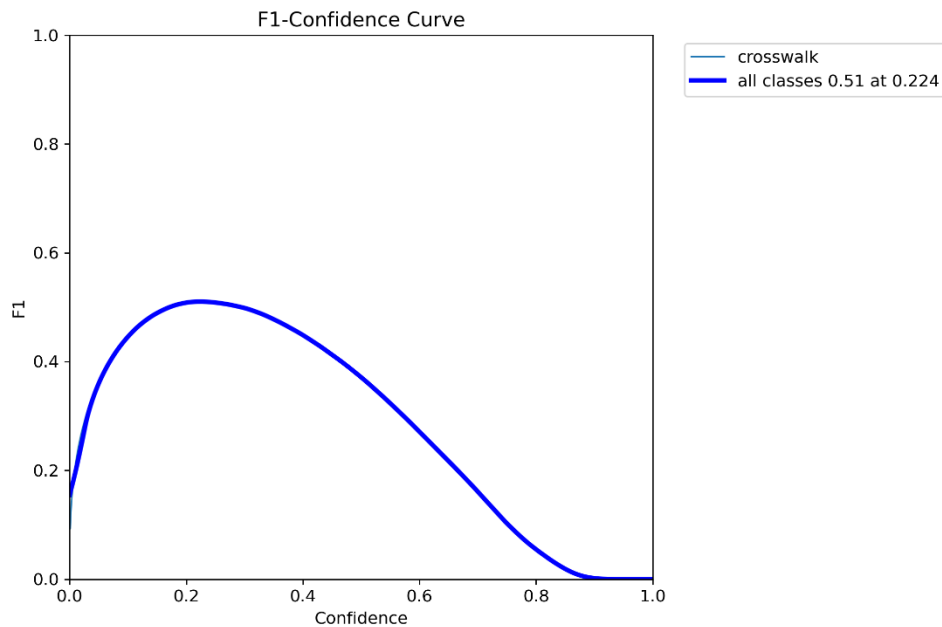


Figure 8: Confidence curve of the YOLO model

5. DISCUSSION

5.1 Interpretation of Results

In this section the models are grouped as Decision Tree, k-NN, Naïve Bayes and Logistic Regression as the first group; and YOLO alone as the second group. The first group of models determines whether the image includes a crosswalk or not. However, the YOLO model tries to find where the crosswalk is if there exists any. Therefore, the first group is classification, and the second group is detection.

Among the classification models, the Logistic Regression performed better than other models with the highest accuracy of 81,45%. The models can be ordered from best performance to lower performance as stated in Table 5.

Table 5: Classifications model comparison by accuracy

Classification Model	Accuracy
Logistic Regression	81,45%
k-NN	73,86%
Naïve Bayes	69,61%
Decision Tree	67,08%

In terms of classification, these currents are performing binary classification: deciding whether the image consists of a crosswalk inside or not. The possible reasons for Logistic Regression Classifier to perform better than other classifier can be listed as below:

- Logistic Regression can provide an advantage for binary classification where the data is approximately linearly separable, which we tried to achieve with Image Embedding in Data Preprocessing phase.
- Logistic Regression applies techniques for preventing overfitting, which is a common issue for our models. The dataset includes several frames of in-car videos. Therefore, some similar images can be learned more than needed, which can result in overfitting. Preventing this situation to a level can improve the performance of our model.

For YOLO detection model, currently due to several reasons (such as some data that we cannot predict how they behave during training), although a crosswalk can be identified, the boundaries may not be set properly or the number of crosswalks in an image cannot be calculated perfectly

Despite the crosswalks not being identified accurately and the confidence is quite low for setting the boundaries, YOLO model has a great advantage when compared to classification algorithms, since YOLO not only classifies but also detects the object location in an image. During several testing, it is seen that YOLO has no problem with identifying the image alone by whether it contains a crosswalk or not. However, the value of the performance metrics being low is only caused by the inconsistency and inaccuracy of determining the boundaries of the crosswalk and location it.

5.2 Challenges

One of the main challenges faced during the Logistic Regression phase was handling the imbalance in the dataset, where images without crosswalks significantly outnumbered those with crosswalks. This imbalance led to the risk of the model favoring the majority class, potentially reducing its ability to correctly classify the minority class. To address this, we ensured the use of appropriate metrics, such as the classification report, to evaluate the performance beyond just accuracy, focusing on precision, recall, and F1-score for both classes.

Another challenge was ensuring that the model converged effectively during training. Logistic Regression's convergence depends heavily on the optimization process, and setting the maximum iteration parameter to 1000 helped address potential convergence issues.

Despite these challenges, the model achieved reasonable accuracy and provided clear insights into its predictive performance through metrics such as the classification report and confusion matrix. This helped in evaluating its performance against other classifiers used in the project.

The main challenge of training the YOLO model was the time needed for the training phase and the device requirements. The first training is done using CPU and with 10 epochs, the training took 2-3 hours for only 1000 images. The second training is done using GPU with 50 epochs, the training took 12 hours for 25000 images. Although we increased the parameters and size of the dataset, the training time did not increase as much due to GPU usage.

Second current main challenge of the final model is the variety of images in the dataset. The dataset we are currently using includes many different types of crosswalks. There are even images of barely seen, dark and blurry crosswalks, which probably is the main cause of the data defilement. We are expecting to solve this issue by doing a further cleaning of the dataset.

6. CONCLUSION

Summary of work and potential future improvements

In this project, we aimed to develop models for crosswalk detection and have an understanding in machine learning studies by comparing our models that are developed by using different classification approaches. These classifiers are Decision Tree, Naïve Bayes, k-NN and Logistic Regression. Additionally, we developed a YOLO model for crosswalk recognition in images and detecting its location.

Our results show that among the classifiers, the Logistic Regression outperformed others with an accuracy of 81,45%. Although we seem to have low performance values for our YOLO model, it detects whether a crosswalk exists or not. The main issue we are having with our YOLO model is the inaccuracy of detecting the location and setting the boundaries of the crosswalk inside the image.

As stated in Section 5, for further optimization, we were planning to clean our dataset to get rid of the data that we cannot predict their behaviour, that might be the cause of main uncleanliness among our images. Although this additional cleaning process would take us a huge amount of time, as we must go through each image to determine whether the image is sufficient to be a part of our training phase. Considering we have around 25,000 images, we would need around 8-9 hours to complete the cleaning.

7. REFERENCES

- [1] Yu, F. 2024. “BDD100K Documentation”. Date Accessed: 25.11.2024.
(<https://doc.bdd100k.com/index.html>)
- [2] Gallagher, J. 2023. “What is an Image Embedding?”. Date Accessed: 16.12.2024.
(<https://blog.roboflow.com/what-is-an-image-embedding/>)
- [3] Jing, Y. Li, X. “ResNet50”. Date Accessed: 19.12.2024.
(<https://www.sciencedirect.com/topics/computer-science/resnet50>)
- [4] Classification/Regression Algorithms and Evaluation [PowerPoint slides]. (n.d.). Google Classroom.
- [5] PyTorch. 2017. “ResNet50 Documentation”. Date Accessed: 16.12.2024.
(<https://pytorch.org/vision/main/models/generated/torchvision.models.resnet50.html>)
- [6] Ultralytics. 2024. “Ultralytics YOLO Documentation”. Date Accessed: 25.11.2024.
(<https://docs.ultralytics.com/tr>)