Detecting Anomalies from Roadside Video Streams

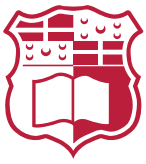
Nicole Bonnici

Supervisor: Prof. Adrian Muscat

Co-Supervisor: Dr Kenneth Scerri

July 2024

Submitted in partial fulfilment of the requirements for the degree of Master of Science in ICT.



L-Università ta' Malta
Faculty of Information & Communication Technology

L-Università ta' Malta

FACULTY/~~INSTITUTE/CENTRE/SCHOOL~~ of ICT

DECLARATION OF AUTHENTICITY BY POSTGRADUATE STUDENTS

Student's Code [redacted]

Student's Name & Surname Nicole Bonnici

Course Masters of Science

Title of Dissertation

Detecting Anomalies from Roadside Video Streams

Page/~~Word Count~~ 125

(a) Authenticity of Dissertation

I hereby declare that I am the legitimate author of this Dissertation and that it is my original work.

No portion of this work has been submitted in support of an application for another degree or qualification of this or any other university or institution of higher education.

I hold the University of Malta harmless against any third-party claims with regard to copyright violation, breach of confidentiality, defamation and any other third-party right infringement.

(b) Research Code of Practice and Ethics Review Procedures

I declare that I have abided by the University's Research Ethics Review Procedures. Research Ethics & Data Protection form code ICT-2024-00203.

As a Master's student, as per Regulation 77 of the General Regulations for University Postgraduate Awards 2021, I accept that should my dissertation be awarded a Grade A, it will be made publicly available on the University of Malta Institutional Repository.

[redacted]

Signature of Student

NICOLE BONNICI

Name of Student (in Caps)

03/09/2024

Date

Abstract

The interconnected nature of road networks implies that anomalies on narrow residential roads can ripple through the entire traffic system, particularly in high-traffic areas as common for the Maltese Islands. Detecting anomalies in such environments using roadside cameras is challenging due to the multitude of normal and anomalous events, changes in illumination, obstructions, complex anomalies, and difficult viewing angles. This thesis investigates anomaly detection methods tailored to the realistic road and data limitations typical of Maltese urban roads.

Classical anomaly detection, which identifies anomalies from structured data, and deep learning-based techniques, which detect anomalies directly from video input, were evaluated. The literature review revealed limited evaluations on realistic datasets for both methods.

The classical method was developed to filter out ID switch artifacts and identify specific anomalies using a combination of filtering, DBSCAN clustering, masking, and rule-based techniques. For the deep learning method, an AE model with the STAE [1] architecture was chosen for its ability to capture temporal representation. Both methods were evaluated on video datasets collected in Malta and a relabeled Street Scene [2] dataset.

The classical method demonstrated high reliability in detecting anomalies in structured data, achieving an 82% true positive rate and a 3% false positive rate for a local dataset. However, the data acquisition method did not accurately record all anomalies, reducing the true positive rate for actual video anomalies. The deep learning method showed strong performance across all datasets, achieving an 83% AUC and a 25% EER for a dataset recorded in the same location. Performance was slightly reduced for locations with heavy shadows, as shown on a second local dataset. Segmenting frames into tiles and augmenting datasets improved performance in shadow-affected conditions, as did masking irrelevant regions.

An event-level comparison showed both methods performed similarly in detecting *non-typical vehicle paths*. The classical method excelled at identifying *non-typical object locations* and was more robust against changes in scene dynamics, is more modular, and easier to debug. The deep learning method was better at detecting *non-typical slow-moving* and *non-typical vehicles* and was more resilient to variations in the data acquisition method within the Intelligent Traffic System (ITS). However, neither method effectively detected unforeseen anomalies. Overall, this thesis provides valuable insights and guidance for choosing the most appropriate anomaly detection methods tailored to different types of anomalies in complex urban road environments.

Acknowledgements

I would like to express my deepest gratitude to all those who have supported me throughout the process of completing this dissertation.

First and foremost, I would like to thank my supervisors, Prof. Adrian Muscat and Dr Kenneth Scerri, for their invaluable guidance, insightful feedback, and unwavering support. Their expertise and encouragement were essential in bringing this research to fruition. I would also like to thank Greenroads Ltd. for providing the data that made this research possible. I am sincerely grateful to my family for their constant support and encouragement throughout my academic journey. Lastly, I would like to acknowledge all the authors and researchers whose work laid the foundation for my research. Their contributions to the field have been an inspiration and a guiding light.

Thank you all for making this journey a rewarding and fulfilling experience.

Contents

List of Figures

List of Tables

List of Abbreviations

3D-GAN 3D convolutional generative adversarial network.

AAE Adversarial Autoencoder.
ADOC A Day on Campus.
AE AutoEncoders.
ALOCC Adversarially Learned One-Class Classifier.
AUC Area Under Curve.

BN Batch Normalization.

CDN Cascaded Deep Neural Network.
CNN Convolutional Neural Network.
Conv-VRNN Convolutional Variational Recurrent Neural Network.
ConvAE Convolutional AutoEncoders.
ConvLSTM Convolutional Long Short-Term Memory.
ConvLSTM-AE Convolutional Long Short-Term Memory autoencoder.

DBSCAN Density-based Spatial Clustering of Applications with Noise.
DNN Deep Neural Network.

EER Equal Error Rate.

FPR False Positive Rate.

GAN Generative Adversarial Networks.
GMM Gaussian Mixture Model.
GPS Global Positioning System.

HDP Hierarchical Dirichlet Process.
HMM Hidden Markov Model.
HOF Histogram of Optical Flow.
HOG Histogram of Optical Gradient.

ID IDentity.
IOU Intersection Over Union.
ITS Intelligent Traffic System.

KCF Kernelized Correlation Filter.
KLT Kanade–Lucas–Tomasi.

LCSS Least Common Subsequence.
LDA Latent Dirichlet Allocation.

LSTM Long Short-Term Memory.

MemAE Memory-augmented Autoencoder.
MSE Mean Squared Error.
MSSSIM Multi-Scale Structural Similarity Measure.

PCA Principle Component Analysis.
PSNR Peak Signal to Noise Ratio.
PTM Probabilistic Topic Models.

R-STAE Residual Spatio-Temporal AutoEncoders.
RBF Radial Basis Function.
ReLU Rectified Linear Unit.
ResNet Residual Network.
RNN Recurent Neural Network.
ROC Receiver Operator Curve.

SDAE Stacked Denoising Autoencoder.
SGD Stochastic Gradient Descent.
SPAE Spatiotemporal Autoencoder.
sRNN stacked Recurrent Neural Network.
STAE Spatio-Temporal AutoEncoders.
STAN Spatio-Temporal Adversarial network.
SVM Support Vector Machine.

TPR True Positive Rate.
TSC Temporally-coherent Sparse Coding.

VAE Variational Autoencoder.

YOLO You Only Look Once.

1 Introduction

Malta has seen a trend of increased road traffic-related fatalities and traffic congestion in recent times. In 2022, Malta witnessed an unprecedented surge in road traffic fatalities, with the rate skyrocketing nearly tenfold compared to any other EU nation, as highlighted by the EU Commission [3]. Between 2021 and 2022, the country experienced a staggering 189% increase in fatalities, a figure unmatched by any other nation. With 80% of the Maltese population agreeing that the frequency of serious traffic accidents is a concern [4]. Apart from the loss of life, these accidents are often the cause of road closures for many hours leading to heavy congestion in Malta's already congested network. Traffic congestion leads to significant frustration among the Maltese population, with 72% indicating that they feel fatigued by the congestion and 69% expressing frustration over the considerable time wasted in traffic [4].

1.1 The Problem

A significant issue contributing to traffic accidents is the occurrence of mistakes and errors in judgment by road users. While many of these errors go undetected, some lead to both minor and fatal accidents. Detecting these errors can facilitate the improvement of infrastructure design or the enhancement of law enforcement efficiency. However, it is essential to evaluate these methods within the specific context of Maltese roads. Malta, being an island with heavy traffic, features narrow roads that experience large volumes of traffic. Additionally, the interconnected nature of the roads means that an anomaly in a side road can have ripple effects throughout the entire system. Therefore, anomaly detection cannot be limited to large roads with only one type of normal behavior, such as driving on a predefined lane. Instead, it must be extended to residential, narrow, busy roads where multiple normal events occur, as well as multiple types of anomalous events

The widespread availability of video surveillance cameras has enabled deployment on many local arterial roads. Primarily used in traffic control rooms for manual observation of traffic conditions, these cameras pose challenges in identifying and analyzing errors in judgment due to the labor-intensive process of parsing extensive video streams. Consequently, there is a growing need for automated detection systems to identify errors in judgment by road users, which could be classified as anomalous behaviors.

Defining anomalous behaviors, in view of automatic detection, is inher-

ently challenging due to multiple factors. Firstly, anomalies are context-dependent. For example, what might be considered dangerous on one road, such as a right turn, could be perfectly acceptable on another. Secondly, the definitions of anomalies could also be subjective. For instance, encountering a bicycle on the road might be seen as unusual by some but completely normal by others. To overcome this limitation, the anomalies under investigation were categorized based on their features. This approach enables a more generalized investigation, accommodating a wider range of anomalies that a reader might find relevant or of interest, even if not represented explicitly in the datasets.

Another challenge in this work involves finding the data that allows adapting and investigating anomalous behavior detection within the context of Maltese roads. While anomalous video datasets exist in literature, few depict events relevant to this study. Additionally, many datasets lack realistic challenges for anomaly detection, such as shadows, changes in illumination, obstructions in the field of view, realistic anomalies of interest and difficult angles. Therefore, this study will focus on evaluating anomalies in realistic, challenging datasets with numerous artifacts that are relevant to the Maltese traffic landscape.

Classical anomaly detection methods offer a promising approach by utilizing multi-object detection and tracking algorithms to extract structured data, such as trajectories and velocities. This method enables the identification of anomalies within the extracted data, allowing seamless integration with existing Intelligent Traffic System (ITS), such as those provided by Greenroads Ltd.[1]. Although a wide variety of methods could be applicable to the structured data, the challenge lies in identifying suitable models for the data sourced from Greenroads Ltd. Since all multi-object detection and tracking methods have some level of error, the classical anomaly detection method must be able to distinguish between real anomalies occurring in the video and artifacts created by algorithmic errors.

Classical anomaly detection methods, however, exhibit inherent vulnerabilities. For instance, obstacles obstructing an object's presence in the data may impede anomaly detection. Additionally, the detector's limited ability to recognize all object classes on the road presents a challenge, particularly with new types of vehicles. Therefore, this work also considers deep learning video anomaly detection methods as a potential solution to these limitations. However, challenges persist in training deep learning models, primarily due to the scarcity of annotated anomalous data. Consequently, model training is often semi-supervised, encouraging the model to learn normal data patterns and flag deviations from the norm, thereby facilitating the detection of previously unseen anomalies.

Thus, the aim of this thesis is to investigate effective anomaly detection

[1] https://www.greenroadsmalta.com/

methods tailored to realistic road and data limitations, as would be typical if the technology is deployed on Maltese urban roads. Both structured data approaches and deep learning techniques will be evaluated using challenging realistic datasets.

1.2 Contribution

The contribution of this project is to develop and investigate anomaly detection methods on video data to identify and analyze events potentially leading to accidents or indicating their occurrence. The research will focus particularly on methods applicable to roads with characteristics similar to those of Maltese roads. Two main approaches will be explored: classical anomaly detection method based on the data provided by Greenroads Malta Ltd. and deep learning-based models for anomaly detection directly processing video data. These methods will then be compared based on performance, ease of implementation, computational time, and other relevant factors. The following objectives have been set to achieve these goals:

- Conduct a comprehensive literature review concentrating on anomaly detection methods, specifically emphasizing techniques relevant to road traffic data. The primary focus of this review is on structured data anomaly detection methods and deep learning video anomaly detection techniques. The objective is to thoroughly investigate existing methodologies and identify the most promising approaches for implementation in the study.
- Gather and annotate datasets as necessary, particularly focusing on acquiring labelled video data such that analysis can be performed with respect to the real anomalous event occurring.
- Utilize a structured data based anomaly detection method to analyze data obtained from the tool provided by Greenroads Malta Ltd. This involves distinguishing between artifacts created by the object detector and tracker and real anomalies that occurred in the video. This includes performing an analysis of the existing structured data acquisition tool to understand its limitations.
- Implement a deep learning-based approach directly on road traffic video data to detect anomalies. This method will involve the training of deep learning models to automatically identify anomalous events directly from traffic video footage.
- Engage in a comprehensive discussion and comparison of the results obtained from both methods. This analysis will involve a detailed evaluation per type of anomaly identified as well as other aspects which would effect

the choice of anomaly detector.

1.3 Structure of the Thesis

From here on the dissertation is organized in the following manner. Chapter 2 is the literature review which surveys classical and deep learning video based anomaly detectors as well as datasets that are relevant to the work. Chapter 3 defines the anomalies for this work. Chapter 4 describes the datasets annotated and used. Chapter 5 describes the methods used to detect anomalies from the data. Chapter 6 presents and discusses the results per method investigated then compares the results from the different methods. Finally, Chapter 7 concludes the dissertation as well as discusses future work that can be done based on the findings of the dissertation.

2 Literature Review

Vision based methods have rapidly evolved in the last decade [5] replacing the need for human supervision, which is time consuming and expensive. Accurately detecting anomalies could further reduce the need for human operators, and research to broaden the types of anomalies that can be detected is ongoing. This chapter reviews previously proposed methods for traffic anomaly detection from data collected from monocular vision systems from fixed road side cameras.

Other traffic data collection methods exist such as soft pneumatic tube based systems and Global Positioning System (GPS) tracking systems, which are not relevant to the problem that will be explored here for various reasons. Pneumatic tube based systems are used for counts and classifications of vehicles [6]. This method does not allow for vehicle tracking limiting the types of anomalies that could be detected however if only vehicle counts are required pneumatic tubes might be the most straight forward method. GPS vehicle tracking systems record the positions of vehicles and have the ability to track a vehicle journey from start to finish, unlike video systems which can only track a vehicle across the camera view. Traffic state can be estimated with a few probe vehicles driving and transmitting data through GPS [7] and hence collective anomalies such as traffic jams could be detected. However, each vehicle would need to be tracked to detect point anomalies such as U-turns and over speeding. The limitations of the installation of the soft pneumatic tube based systems and GPS tracking systems, contrast the simple road side installation required for camera data extraction method that will be reviewed here.

The methods presented in this literature review are split into two main sections: Classical Anomaly Detection Methods and Deep Learning Methods. The former section presents methods that could be used to detect anomalies from data extracted from a multi object detector and tracker. The intention was to investigate an anomaly detection system that would work well in conjunction with the already existing ITS from Greenroads Ltd. The latter section investigates deep learning methods for anomaly detection that directly process the video data. These types of methods extract features automatically through deep learning. Finally, a summary of relevant datasets found in literature is presented in Section 2.3

2.1 Classical Anomaly Detection Methods

This section reviews methods which use multi-object detection and tracking on a traffic video to extract structured data such as trajectories and velocity. Anoma-

lies are then detected from the extracted data. This type of pipeline allows the data extracted from the detector and tracker to be used in conjunction with other traffic control algorithms. Thus, the anomaly detection can be incorporated with existing ITS. The structured data also has the advantage of being less expensive to store when compared to the raw video data, thus anomalies can be detected offline on the stored data if required.

Anomaly detection can be set based on manually set thresholds, if one has information about the traffic scene. For example, in [8], Wang et al. detect anomalies based on speed and relative distance of the vehicles from each other by comparing the speed of the vehicle of interest to the speed of the neighbouring 20 vehicles. A vehicle is considered anomalous if its speed is significantly different from that of its neighbours. The threshold for determining whether a vehicle speed is anomalous is inferred from the percentage of vehicles that are stopped by the local police force for speeding. Ijjina et al. [9] also use prior knowledge of the scene to develop a system for detecting vehicular collisions. The system combines the change in acceleration, the angle of the trajectories of vehicles and the change in angle of a vehicle with respect to its own trajectory when the two bounding boxes of different vehicles overlap. If the result of the function exceeds a manually set threshold, the system detects an anomaly. The manually set thresholds are likely to not be valid when the data collected is different. For example in, [8], Wang et al., comparing to 20 neighbouring vehicles would result in many false positives if the data collected also captured parked cars,

The variety of the data that can be collected from traffic videos results in many methods being proposed for the anomaly detection. Methods that could be relevant to the data generated by the Greenroads Ltd. multi object detector-tracker will be reviewed in this section. Subsection 2.1.1 discusses the detection and tracking methods used prior to applying the anomaly detection methods to give a background on the extraction of the structured data. Subsection 2.1.2 describes clustering based anomaly detection methods, Subsection 2.1.3 discusses classification based methods, Subsection 2.1.4 reviews Hidden Markov Model (HMM) based methods and Subsection 2.1.5 describes Probabilistic Topic Models (PTM) based methods. Finally, Subsection 2.1.6 compares these methods and their perceived limitations.

2.1.1 Detection and Tracking Methods

Classical data based traffic anomaly detectors typically extract data from a traffic scene by first detecting objects and tracking them across the scene. Salient features are then extracted and used to detect anomalies. The choice for tracker and

detector determines the quality of the data extracted and which features can be accurately extracted from the data.

Vehicle detection is a method to identify vehicles on the road in video frames. The typical output of vehicle detection is the vehicle classification, position in the frame and bounding boxes [10]. Early methods relied on some form of background subtraction. For example Kamijo et al. [11], model the background of a scene by averaging the frames in a particular time period. Then a new vehicle is detected when a change in intensity greater than a threshold is noticed at the entrance of a road intersection. The block of pixels with this intensity is then given a vehicle ID. Recent advances in deep learning detection methods has greatly improved the performance of object detectors and thus fewer methods are opting to use handcrafted methods for object detection [10], choosing instead to use deep object detection.

Two main types of deep object detectors exist; two-stage and one-stage object detectors. Two-stage methods find regions of interests through selective search algorithms or region proposal networks. Then, classify the regions of interest proposed as detection results. The most frequently used object detection method as identified in the ITS survey by Zhang et al. in [10] is Mask R-CNN [12]. One-stage methods detect objects from images directly using dense sampling of all the possible locations of the objects. You Only Look Once (YOLO) [13], and its subsequent versions, are a popular, widely used object detector in the traffic analyses problems. It uses a Convolutional Neural Network (CNN) to predict bounding boxes of the vehicles detected and the probabilities of these bounding boxes being accurately detected in a single evaluation. One stage methods are typically faster than two stage methods. Table 2.1 provides a summary of the commonly used detection methods considered in this section.

To analyze the movement of vehicles over time, the vehicles detected in each frame, or a subsample of the frames, must be tracked so that the path each vehicle takes is identified. There are various algorithms that can be used for vehicle tracking. Early works like Kamijo et al. [11] developed an algorithm which works in conjunction with their intensity based tracker, and outputs the vehicle ID label of each pixel associated with a vehicle. Therefore, they modeled the vehicle detection and tracking as a labeling problem for each pixel in the frame. The position of the vehicle after a certain time period is calculated based on its current motion vectors of the pixels. In recent years, tracking methods based on handcrafted features have been mostly replaced with tracking-by-detection. For example, Wang et al. [8] use a multi-object tracker Kalman filter [14] where the next position of the object being tracked is estimated using the objects' parameters that are measured over time. Ijjina et al. [9] use a Centroid based [15]

Table 2.1 Summary of common object detection methods used

Used by	Method	Description
[11, 21]	Background Subtraction	If an intensity change larger than threshold is noted, on object is said to be detected in the frame
[22]	FasterRCNN [23]	Two stage deep learning based method
[9, 19, 20]	Mask R-CNN [24]	
[20]	Cascade R-CNN [25]	
[8]	YOLO [13]	One stage deep learning method
[17]	YOLOv3 [26]	
[27]	YOLOv5 [28]	

tracking algorithm where they assumed that as the vehicle moves between consecutive frames of the video the distance between the centroid of the same vehicles will be less than that between other vehicles. A Kernelized Correlation Filter (KCF) [16] tracker makes use of kernels to quickly calculate fast results for the motion and velocity of the object and thus determine its future position, as used in [17]. The most used tracking algorithm for visual traffic applications is the Deep SORT [18] algorithm [10] and used by Yu et al. and Zhao et al. [19, 20]. Whereas the Centroid tracking and Kalman filter only use the velocity and position of the object for tracking, the Deep SORT algorithm also considers the appearance descriptors through a trained CNN. The Deep SORT Algorithm has been shown to result in a lower number of ID switches caused by obstructions. ID switches refer to instances when the tracker confuses on object with another one thus part of a track would belong to one object and part to another. Table 2.2 summarises the commonly used detection methods as reviewed in this section.

Optical flow based methods are another type of detection and tracking method. The proposed methods in Song et al. and Jeong et al. [30, 31] use optical flow. Optical flow methods measure the motion vector of pixels between subsequent frames, and assume that neighbouring pixels have similar motion and that pixels are similar in subsequent frames. Unlike tracking-by-detection methods, where the data collected would be data about each vehicle, methods that use optical flow collect information about each pixel, group of pixels or salient pixels (such as corners) per frame.

Detection and tracking methods are not perfect and each method has some level of missed detection and incorrect tracking. Zhao et al. [20] have developed an interesting approach to address these issues. They use a combination of multiple detectors to lower missed detections. Specifically, they employ a Faster R-CNN as their main vehicle detector and Cascade R-CNN as a backup detector.

Table 2.2 Summary of the object trackers used in the methods reviewed

Used in	Method	Description
[9]	Centroid tracking [15]	Assumes that the distance between the centroid of the same object will be less than that of other objects
[8]	Kalman Filter [29]	Object position in the future is estimated based on its current motion and velocity
[17]	KCF[16]	High speed tracker which makes use of kernels to perform fast calculations
[19, 20]	Deep Sort [18]	Uses motion and velocity to predict the future position of an object and makes use of a CNN to extract appearances of objects
[30, 31]	Optical flow	Assumes neighbouring pixels have similar motion and pixels do not change between frames. The movement of pixels from frame to frame is calculated. Optical flow output is often the velocity of pixels movement.

They also incorporate a DeepSORT tracker to find a mask based on vehicle trajectories. Furthermore, Pixel-level information is recorded to detect anomalies and determine the exact time an anomaly occurs.

The methods used for vehicle detection and tracking in a video determine the type of features that can be extracted. The most commonly used features are the entire trajectory of vehicles, their speed, and overall movement, which can be represented by the start and endpoint. The most common anomalies detected are anomalous trajectories and traffic accidents. Table 2.3 provides a summary of the features extracted, the methods that used these features, and the anomalies that can be detected. The subsequent sections of the literature review will go into further detail on the methods used to detect these anomalies.

Table 2.3 Summary of commonly used features

Paper	Features	Method	Anomalies
Kamijo et al. [11]	direction of motion, speed estimate [1] , distance between objects	supervised HMM	Bumping, Passing, and Jamming [2]

Anjum et al. [32]	average velocity, mean of trajectory, directional distance, initial position, speed, acceleration, PCA of trajectory points, trajectory turns	per feature clustering then merging into one cluster.	Anomalous Trajectories
Piciarelli et al. [33]	trajectories	SVM based method	Anomalous Trajectories
Jiang et al. [34]	trajectories	HMM, and 2-depth greedy search algorithm	Anomalous Trajectories
Morris et al. [35]	trajectories, start, end and stop points	spectral clustering and HMM	Anomalous Trajectories
Jiang et al. [36]	location, moving direction, velocity, interaction an co-ocurance of vehicles	HMM decoding problem	Point anomaly detection
Song et al. [30]	per pixel location, direction and speed	LDA of feature vector and their interactions	Anomalous activity and interactions
Jeong et al. [31]	trajectory represented by the cells defined in the frame, velocity	LDA to cluster trajectories	Abnormal velocities, trajectory shape and location, and temporal relation
Brun et al. [37]	trajectory represented by the cells defined in the frame, speed, shape	string kernel-based clustering	Anomalous Trajectories
Cai et al. [38]	trajectories, start and end points, velocities	k-means clustering and HMM	Anomalous Trajectories

Yu et al. [19].	distance measurement between vehicles	distance measure less than a threshold	Traffic Accidents
Giannakeris et al. [22]	speed, trajectories, optical flow characteristics in the object's bounding box	SVM based threshold	Anomalous Trajectories
Wang et al. [39]	positions, velocity	Dual-HDP to cluster trajectories and their features.	Anomalous
Trajectory Wang et al. [8]	speed estimate	relative speed between neighbours	Car Break Downs
Ijjina et al. [9]	overlap of bounding boxes, speed estimate, change in acceleration, trajectories, angle of overlap,	weighted function and manual threshold	Traffic Accidents
Zhao et al. [20]	pixel positions	time threshold if a foreground pixel is in a location for too long	Vehicle Collision and Stalling
Gao et al. [17]	trajectories, start point, end point, count of points, movement vector in Cartesian and polar coordinate system	combined into a vector and use methods such as SVM and K-means	Tracklets, ID switches

In addition to extracting vehicle position and motion, other techniques are employed to ensure accurate data collection from videos, namely camera calibration and video stabilization. Camera calibration can be utilized to obtain precise distance and speed data. Yu et al. [19] and Giannakeris et al. [22] employed a camera calibration method that detects vanishing points in the image. The vanishing point enables the calculation of perspective effects, thereby facilitating more ac-

[1] A distinction between speed estimate and speed was made; speed refers to cases where the camera was calibrated to find accurate speed values and speed estimate refers to cases where speed was calculated without accounting for perspective

[2] requires labeled data

curate distance and speed calculations. With accurate speed and distance data, Yu et al. [19] can predict future trajectories and detect danger (anomalies) as a function of the distance between two vehicles before they occur.

Camera stabilization aims to counteract the small jitters that a roadside camera can experience due to natural environmental factors such as wind and ground tremors. Zhao et al. [20] observed that camera shake can decrease the performance of their model, prompting them to deploy digital camera stabilization techniques to correct the camera's oscillatory motions. Initially, they detect the camera movements and then smooth them out. Subsequently, they employ a point matching method based on *good features to track* and calculate a sparse optical flow measure, which assists in identifying the transformations required for each frame.

2.1.2 Clustering

Clustering based methods are a very common road traffic anomaly detection method. Clustering is an unsupervised or semi-supervised technique that groups similar points of data into a collection or *clusters*. In the context of clusters, the anomalies would be the points that either do not belong to any cluster, lie the furthest away from the center of a cluster or belong to sparse and small clusters [40]. Clustering is often used as a step before applying another type of anomaly detection method as discussed Section 2.1.4.

A multitude of clustering algorithms exist with one of the most common being the K-means clustering [41] where the algorithm tries to group similar data points into K groups, with a data point belonging to one group only. The number of groups K needs to be predefined. The algorithm tries to assign each point to one of the randomly set centroids then updates the centroids to be the average of the data points belonging to the group. This process is repeated until convergence.

Another commonly used clustering algorithm is the Density-based Spatial Clustering of Applications with Noise (DBSCAN) [42] algorithm, which is more suitable to arbitrarily shaped clusters and noisy data. The algorithm clusters dense data as one cluster, based on a predefined *epsilon* value. *Epsilon* sets the radius which data points need to be within of to be classified as dense neighbours. The *minimum points* is another manually set metric which specifies the minimum number of dense points needed to form a cluster. The DBSCAN discards data points that do not belong to any cluster as outliers and the number of clusters does not need to be known before hand.

A widely used clustering algorithm for trajectory data based on DBSCAN is the TRACLUS algorithm proposed by Lee et. al [43]. The authors commented on

how typical clustering algorithms for trajectories fail to consider sub-trajectories. Thus, they proposed a partitioning and grouping framework that partitioned sets of trajectories and then clustered them. The sub-trajectories were split at the points a trajectory changes rapidly. After the trajectories are partitioned the line segments left are clustered using DBSCAN [42]. The distance metric used was the weighted sum of 3 different types of distances between each pair of trajectories; the angular distance, the parallel distance and the perpendicular distance. Anomalies can then be identified from the clusters as explained previously.

The structured data can also contain other types of features apart from the coordinate points which are referred to as trajectories, and these can also be considered during clustering. Anjum et al. [32] et al. noted that clustering using all the features available from a trajectory simultaneously leads to vague clusters. To address this, the authors proposed a partial trajectory clustering framework, where features are clustered individually before combining to form one crisp partition. The authors extracted the average velocity, the mean of the trajectory, directional distance, initial position, speed, acceleration, Principle Component Analysis (PCA) of the trajectory points, trajectory turns from the video data and each feature is considered as a probability density space. For each feature space the mean-shift clustering algorithm was used obtain clusters describing the separate behaviour. The separate clusters in each feature space are combined into one cluster by analysing the trajectories which are consistent in all feature spaces, then fitting the other trajectories which are not. Finally anomalies are detected from trajectories which lie in a dense region but have a significantly different pattern or belonged to clusters with only a few members.

Saggese et al. [37] proposed a clustering algorithm with the aim of reducing the computational requirements that can be associated with clustering high dimensional data such as trajectories. The method starts by clustering all data into one cluster and then splitting it along the main axis as shown in Figure 2.1. Then the cluster with the maximum squared error is selected and split again. This step is performed recursively until a stopping condition is met. This method uses a kernel based trajectory similarity measure which allows for less computationally expensive clustering. Saggese et al. [37] further reduced the computational expense by representing a trajectory by the areas in the frame the trajectory passed through, as opposed to coordinate points. The frame was split into areas recursively, with areas having a denser concentration of trajectories being split into smaller cells.

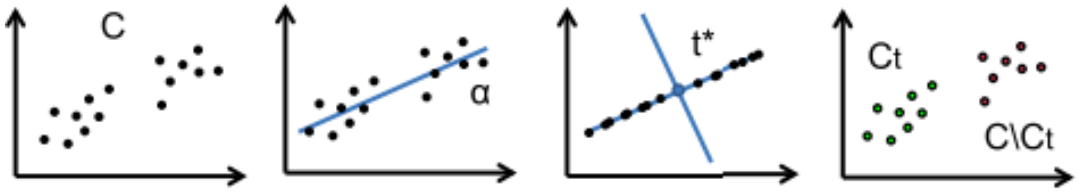


Figure 2.1 Clustering algorithm from [37]. The plots show the starting cluster C, the main axis used as a reference for splitting the cluster, the axis for splitting the cluster, and the split clusters Ct and (C/Ct) respectively

2.1.3 Classifiers

A classifier is an algorithm or model that is trained to classify input data into one or more predefined categories or classes. The goal of a classifier is to learn patterns and relationships in the input data that can help in predicting the correct class label for new, unseen data. A classifier is trained to be able to distinguish between normal and abnormal data in the context of a classification-based anomaly detection approach [40]. A single class Support Vector Machine (SVM) is a commonly used classification method for identifying anomalies. They can learn the boundary that separates anomalous data from the rest of the normal data. Batapati et al. [21] used SVMs to identify anomalous trajectories in a small-scale test bed that simulates traffic conditions. Their test bed provided them with prior knowledge, allowing them to manually select the SVM parameters. However, in real-life scenarios, manual selection of SVM parameters is not always possible.

Gao et al. [17] identified two types of anomalous trajectories that are commonly found in vision based object trackers. The first type of anomalous tracks are *short tracklets* which are short tracks, that result from short rapid movement of the object that causes the tracker to loose the object. The tracklets have a normal shape, but are too short to analyse. The second type of anomalous trajectories are caused by the identity switches of the tracker. Thus, the trajectories have a typical length but a strange shape.

The authors argue that typical anomaly detection methods ignore the *short tracklets* which leads to inadequate performance for tracks obtained from object detectors. To address this issue the proposed method extracted the *features of anomalous pose* which is a combination of the trajectory start point, end point, count of points, movement vector in Cartesian coordinate system and polar coordinate system. Then the *features of anomalous sub-tracks* are extracted such that anomalous tracks due to IDentity (ID) switches can be detected. This was done by splitting a trajectory into sub-tracks through thresholds on the angle of motion and velocity as shown in Figure 2.2.

The *features of anomalous pose* serves to detect the anomalous artifacts

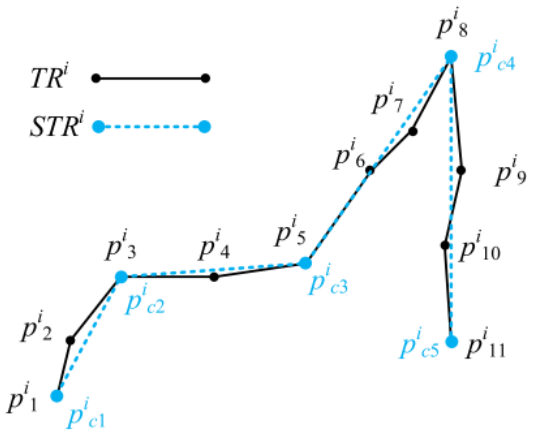


Figure 2.2 Figure from [17] showing the splitting of a trajectory into sub-tracks. Where TR^i is the trajectory and STR^i is the sub-tracks

from vision based object trackers whilst the *features of anomalous sub-tracks* serve to detect anomalous tracks due to ID switches. The *features of anomalous pose* and the *features of anomalous sub-tracks* are combined into an Eigenvector. This Eigenvector can then be used in multiple different anomaly detection models, such as K-means clustering. Gao et al. [17] found that the best method for detecting anomalies for the combined features was the single class SVM classifier, however, no details about the parameter selection process were given.

Piciarelli et al. [33] proposed a method to detect anomalies using SVMs without any prior knowledge of the outliers present in the dataset, which is often the case in real life scenarios. They address the challenge of estimating the upper bound for the number of outliers, which can vary for different datasets and difficult to estimate without prior knowledge, by hypothesizing that the hypervolume of the SVM shrinks rapidly if the removed data point is an outlier and slowly if it is not. Thus, the SVM is retrained multiple times, each time removing a data point, noting the size of the hypervolume. A rapid change in the hyper volume size, would indicate that the correct classification boundary was found. The process is repeated until all the outliers are identified. Although the method was effective in detecting the optimal boundary for detecting anomalies it required the calculation of multiple SVMs making it a computationally expensive process. Hence, the authors proposed a method where the angle of each trajectory with respect to the center of the hyper spherical cap containing the data is calculated. Each trajectory is sub sampled to give the same number of data points and transformed using a Gaussian Kernel prior to the angle calculation. After plotting the angles in order, the trajectories which are anomalous are identified by a sharp corner/elbow point in the plot. The data after the elbow point is considered anomalous and the rest of the data is normal. This essentially creates a labeled dataset to

train an SVM which can be used to detect anomalies.

2.1.4 Hidden Markov Models

A Hidden Markov Model (HMM) consists of a doubly stochastic process for which hidden underlying stochastic processes can be inferred [44]. HMMs are able to model typical normal behaviour from sequential data. The probability of a sequence of data occurring can be calculated by the trained model. An anomalous sequence of data would result in a low probability of occurrence. HMM requires that all the training data belong to one similar sequence. For example Kamijo et al. [11] take a supervised approach to using HMMs. The authors compute a danger vector for each pair of objects detected in the scene by first taking the difference in motion vectors between the two objects, aligning it to the positive x-axis and dividing by the distance between the two objects detected. These vectors are then used to train an HMM with examples of each type of anomalous events the authors would like to detect. If labeled data is required from each location for each anomaly type to train the HMM, the model would not be easily scalable to different locations. Thus, various methods have been proposed to bypass the need for labelled data.

Modeling Hidden Markov Models to Clusters

Although clustering can be used independently to identify anomalous trajectories, as explained in Section 2.1.2, the methods reviewed in this subsection involve training HMMs on the results of the clusters. This approach extracts the spatio-temporal properties of the trajectories and provides time normalization.

Cai et al. [38] propose a coarse-to-fine trajectory clustering and modeling approach. Initially, the authors cluster trajectories based on the main flow direction, grouping together trajectories that start and end at the same points within the camera's field of view. Each cluster of trajectories is then described using a Gaussian function. To enhance classification accuracy, the clusters are further refined using a robust K-means clustering algorithm. An anomaly is detected if a new trajectory's size exceeds twice the determinant of any cluster's covariance matrix. Additionally, the results of the K-means clusters are used to extract the spatiotemporal properties of the routes using HMMs. Each HMM is trained with all trajectories from a cluster and their velocities. When using this HMM-based anomaly detection, a path model that best describes a new trajectory is fitted, and if the probability of the path is less than a threshold, the trajectory is considered anomalous. This method allows the authors to detect abnormal trajectories corresponding to illegal U-turns, wrong exits and lane changes.

A similar method is used by Morris et al. [35]. The authors first separate clusters based on their start, end point and stopping points and model them using a Gaussian Mixture Model (GMM), at this point, short and broken tracks can be detected and removed from the dataset. The next layer of features are extracted through spatial clustering, whereby routes that pass through the same nodes are separated based on their length. A Least Common Subsequence (LCSS) method was used to measure distances between the trajectories of different lengths. Then a fuzzy C-means spectral clustering method is used to cluster together similar trajectories. This method was chosen to minimize the effects of outliers and obtain soft membership values. The clusters are further refined by merging clusters which have similar model routes, i.e. routes which passed through the same points or have a small distance between each other. Then a HMM is trained using the most representative trajectories in each cluster, considering its position and velocity. Since there is no guarantee that the training data encompasses all activities possible the HMMs are updated in an online fashion using maximum likelihood linear regression. To detect anomalies, when a trajectory has been completed, its likelihood of occurrence is calculated. Using Bayesian inference the likelihood of a new trajectory belonging to a cluster is calculated, and if the likelihood is less than a threshold, the trajectory is considered anomalous.

Jiang et al. [34] take a different approach and propose a dynamic hierarchical clustering method. An HMM is fitted to each trajectory and then a Bayesian information criterion based dissimilarity measure is used to compare the similarity of each HMM. If two HMM are similar enough, their models are merged. The merging procedure is repeated until there are no more similar HMMs. To prevent over fitting to the first HMMs in a cluster, the HMM is retrained with the new set of trajectories after each merging iteration. The most similar trajectory clusters are detected using a 2-depth greedy search. At each iteration, the algorithm identifies the two or three groups that would result in the lowest dissimilarity measure if merged. Some potential merges are rejected based on a table of simple exclusion criteria to expedite the calculations. After completing the merging procedure, a normal cluster is defined by its high probability rather than by having a large number of members.

Hidden Markov Model Decoding

Jiang et al. [36] decode a HMM to detect anomalies in traffic videos by representing events at three different levels. The first level is point anomaly detection, which involves identifying atomic events that describe the direction of movement and velocity of an object in each video frame. Point anomalies are detected if the

atomic event is infrequent. The second level involves sequential anomaly detection, which groups atomic events occurring in a trajectory to detect infrequent sequences of events. The third level is co-occurrence anomaly detection, which represents anomalies as an item set of the sequential and atomic levels.

To detect a co-occurance anomaly, the authors consider an observation item set as the output of an HMM, taking into account temporal constraints. The hidden states correspond to normal co-occurrence events, and identifying these hidden states reveals the most frequent activities. A distribution model based on the Viterbi algorithm is proposed to detect the parameters, facilitating the detection of anomalies by comparing the observed item set against the model.

2.1.5 Probabilistic Topic Models

PTM have typical applications in document and image analysis. They can be used to find the distribution of words across a large number of documents. The distributions are then referred to as topics, since they collect the general theme of the documents [45]. The models ability to cluster and summarize large amounts of data make them suitable techniques to cluster and describe road traffic data and are thus used to find anomalous events. A Latent Dirichlet Allocation (LDA) can decompose a set of documents into their salient topic, with a topic taking the form of a probability distribution over a set of words [45]. LDAs are able to summarize large amounts of data, however the number of topics need must be specified before hand, which is a serious limitation if no apriori knowledge of the data exists. In this case, the Hierarchical Dirichlet Process (HDP) can be utilized, a method based on Bayesian nonparametric models where the number of topics is set to be infinite [45]. Table 2.4 compares the methods proposed to detect road traffic anomalies using PTMs, which methods will be further discussed in this section.

Jeong et al. [31] and Song et al. [30] use an LDA model to extract information about the road traffic scene. Jeong et al. [31] use a Kanade–Lucas–Tomasi (KLT) feature tracker on the corner points to find trajectories after a background subtraction which results many broken trajectories. They define *words* as the grid cells that will split a scene and a *document* as a trajectory, which is denoted in terms of the grids that the trajectory has passed through. This would mean that the *topics* detected are the trajectory patterns, and the number of topics is set manually per dataset. They note that this value had minimal effect on performance. The velocity is also defined as vector of one point of a trajectory to another point after a time period. A two-stage greedy algorithm is proposed to learn the LDA model. Three sub-models are trained; the first model learns online trajectory clustering, the second learns the spatiotemporal dependency of activities

Table 2.4 Summary of the PTM based methods used

Paper	PTM	Words	Documents	Topics	Anomaly Detection Method
Jeong et al. [31]	LDA	grid cells that split a frame	trajectory defined in cells that it passed through	trajectory patterns	probability of occurrence based on 3 sub-models
Song et al. [30]	LDA	movement and speed vector per pixel	a snippet of video	co-occurring words and interaction patters	vector of traffic state per frame is extracted and used to train an HMM
Wang et al. [39]	Dual-HDP	quantized features	trajectory	semantic regions/activities	probability a trajectory occurred

and the third models the velocities. After the learning procedure, the distributions are considered fixed and an anomaly is detected if a frame has an atypical trajectory pattern in it, if a trajectory has an atypical spatial relation when compared to the rest of the trajectories, if the overall path of a trajectory is not normal or if it's speed is not typical.

Song et al. [30] note how high level detectors and trackers are often problematic, thus opt for optical flow estimation using the multi resolution Lucas-Kande algorithm. Multi resolution refers to the fact that the algorithm down samples the image to find the best resolution even if the camera is mounted far from the vehicles. The optical flow outputs the displacement and velocity of a pixel per frame. A video of the traffic scene is split into small clips and a clip is regarded as a *document*. A visual *word* is defined as the movement and speed vector of a patch in an image. A LDA model is extracted at two levels; In the first level LDA the most frequent co occurring words are found (these model activities of the vehicles) and a second Level LDA aims to find interaction pattern between certain combination of activities. The two levels of LDAs are combined and a frame is labeled according to most dominant word interactions. To find the anomalous events, the frame labels are represented as vector and used to train a HMM to model the states in the traffic and a Viterbi algorithm is used to find the post probable traffic states. Anomalies are detected if there are activities (visual words) which do not correspond to all other detected activities or if the interaction of activities do not correspond to a previously detected pattern as modelled by the HMM.

The dataset used in Wang et al. [39] is a collection of trajectories from a car park. Thus the number of topics, i.e. activities or semantic regions, would be much harder to determine a priori due to the chaotic nature of the vehicle's trajectories. Whilst the number of topics would be easier to determine if a road had a clearly defined route. In their proposed methods a trajectory is treated as a *document*, with positions and moving directions as features quantized according to a code book. The quantized features are set to be the *words*. Thus, a trajectory is a bag of quantized features in a temporal order. The *topics* reveal semantic regions

used by trajectories. If trajectories pass through the same semantic regions, they belong to the same activities. A HDP does not cluster documents, which were defined as trajectories, which is required to find anomalies. Thus, Wang et al. [39] propose Dual-HDP which co-clusters both word (features) and documents (trajectories). An abnormal document (trajectory) is determined by the likelihood of the document occurring when the other documents exist and can be obtained using the samples obtained during Gibbs sampling.

2.1.6 Discussion

Through the literature reviewed one could note that the methods proposed were all evaluated somewhat differently. The anomalies labeled were not defined by a specific rule, there was no consistency in the datasets used and the evaluation metrics used were rarely the same. In addition the data extracted is rarely in the same form. In some cases vague information on how trajectories were extracted are given as for example in Cai et al. [38] and Jiang et al. [34]. Thus, a direct comparison of method performance cannot be made.

From Table 2.3 one can note that a variety of anomalies were defined as important for detection. The most common were anomalous trajectories and traffic accidents. Gao et al. [17] is one method that do not aim to detect anomalous trajectories, in the sense of a vehicle doing something weird, but rather detect anomalous trajectories which are caused by detector tracker error. Oftentimes the detector tracker errors are ignored or the methods do not have enough information on the tracks to tell apart detector errors and true anomalous trajectories caused by unusual traffic behaviour. Brun et al. [37] comment on the difficulty in labeling the MIT Trajectory [39] for a human operator due to the ambiguity of certain trajectories and lack of information. All methods reviewed achieved high performance metrics; however, they were evaluated only in terms of the anomalies present in the structured data. This demonstrates that these anomaly detection algorithms perform well on the available data. However, without also evaluating the anomalies detected in terms of those occurring in the video, which might not have been recorded during the data acquisition method, the actual performance in a real-life setting cannot be accurately assessed.

2.2 Deep Learning Video Anomaly Detection Methods

Section 2.1 delved into structured data anomaly detection methods that extract features from the structured data output of a vehicle detector and tracker, utilizing these features to identify anomalies. However, these methods inherently possess certain vulnerabilities. For instance, if the detector and tracker fail to detect an object due to obstructions, resulting in its absence from the structured data, the anomaly detection method may encounter missing data, thus impeding the detection of anomalies related to that particular object. Moreover, the detector's capacity is limited to recognizing a predetermined set of object classes on the road. Consequently, if a new type of vehicle emerges, the detector's classification accuracy may be compromised, rendering the anomaly detector incapable of identifying the novel vehicle as anomalous.

Furthermore, conventional hand-crafted feature-based methods only extract features considered representative of the scene by domain experts. There is no guarantee that these features encapsulate the complexity of all relevant data dimensions, potentially leading to the oversight of unforeseen anomalous events. Despite efforts to select features that anticipate anomalies, there remains a risk that these features may not accurately capture all types of unforeseen anomalies, potentially allowing the most unusual events to evade detection.

To mitigate these limitations, this section will examine Deep Learning Video Anomaly Detection Methods. The success of deep learning in various domains of computer vision, such as object classification and detection, has prompted researchers to explore deep learning-based approaches for anomaly detection. Deep learning methods typically require extensive and diverse annotated datasets for model training. However, collecting numerous samples of anomalous events is inherently challenging, as anomalies are expected to occur infrequently. Consequently, these models are often trained in a semi-supervised manner, utilizing datasets comprising solely of normal samples. Subsequently, the deep learning model can identify deviations from the norm, enabling the detection of previously unseen anomalies. In the realm of video data, deep learning models face the additional challenge of detecting anomalies temporally beyond the spatial scope of individual frames. This temporal aspect is particularly crucial for analyzing complex events in road traffic data, such as unusual trajectories and speeds.

Alternatively, supervised or two-class deep learning models have been proposed for addressing the deep anomaly detection problem. These models are trained on both normal and anomalous classes. However, training on limited

samples of anomalous classes may result in overfitting [46]. Additionally, these models can accurately detect only predefined anomalous classes present in the dataset [46]. Consequently, such methods may not offer significant advantages over traditional hand-crafted feature-based approaches.

Subsequent sections will delve into the current state of the art in semi- supervised Deep Learning Video anomaly detectors. Emphasis will be placed on distinguishing between models employing reconstruction, prediction, or generative-based approaches, as well as cascaded and metric learning models. Finally, a direct comparison of the aforementioned methods will be provided, facilitated by their benchmarking on a standardized set of datasets.

2.2.1 Reconstruction Based Video Methods

Anomaly detection poses a significant challenge due to the scarcity of anomalous data examples and the unpredictability of anomalous events. Consequently, many anomaly detection methods leverage a reconstruction framework for its ability to derive meaningful data representations. During training, the reconstructive model is exposed to numerous instances of normal occurrences. Subsequently, during testing, anomalous events are identified by high reconstructive errors. This approach operates on the assumption that anomalous events cannot be accurately reconstructed using features extracted from normal events.

Typically, AutoEncoders (AE) models are used for data reconstruction. An autoencoder comprises two main components: an encoder and a decoder. The encoder compresses the input into a latent vector, which is then expanded by the decoder to reconstruct the image. The latent vector, being the smallest part of the architecture, forces AE to focus on reconstructing only the most salient features, guided by defined loss functions [47]. Although originally designed for dimensionality reduction, AEs have recently found applications in anomaly detection.

Hassan et al. [48] initially trained a fully connected AE with hand-crafted motion features based on Histogram of Optical Flow (HOF) and Histogram of Optical Gradient (HOG) to learn the typical scene motions. However, they observed that these hand-crafted features might not fully capture complex motion patterns. Consequently, they introduced a fully Convolutional AutoEncoders (ConvAE), which accepts short video clips as input. The architecture of the ConvAE, as seen in Figure 2.3, incorporates max-pooling filters to induce spatial invariance in the outputs, thereby aiding image classification. Fully connected layers are avoided in ConvAE to preserve spatial information. Video data is inputted as temporal cuboids extracted via a sliding window approach. A temporal cuboid

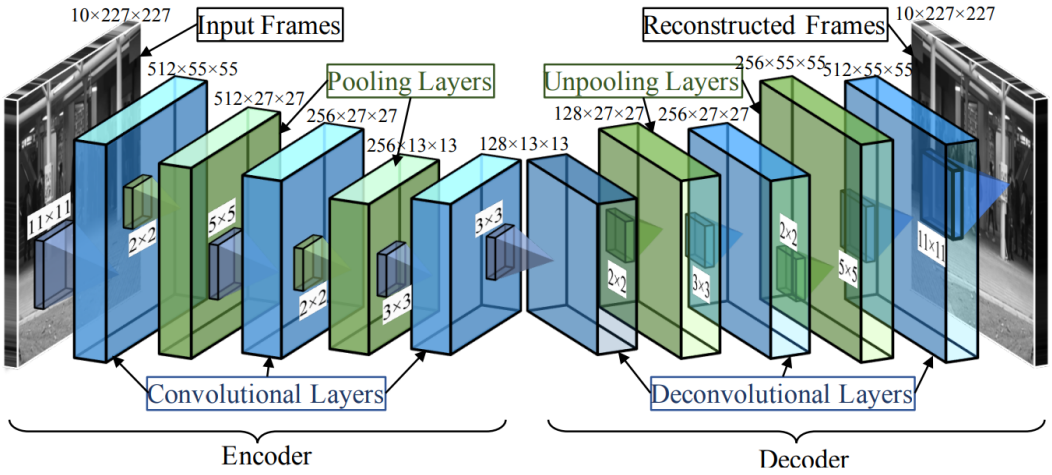


Figure 2.3 Figure showing the detailed architecture of the ConvAE model from [48].

consists of a stack of frame.

The euclidean loss and L_2 regularization are used as training loss in [48]. After training, anomalies are detected through the regulatory score, $s(t)$. First, the reconstruction error per pixel at location (x, y) is computed as the difference between the intensity value I of the original pixel and the pixel reconstructed by the model f_W in frame t as given by:

$$e(x, y, t) = ||I(x, y, t) - f_W(I(x, y, t))||_2 \tag{2.1}$$

The summation of all pixels per frame is used as a frame-wise error, defined as $e(t) = \sum_{x,y,t} e(x, y, t)$. The regulatory score, is then defined as:

$$s(t) = 1 - \frac{e(t) - min_t e(t)}{max_t e(t)} \tag{2.2}$$

The lower the regulatory score, the more likely the frame stack is anomalous during inference. From the regulatory score for each frame stack in the test set the Receiver Operator Curve (ROC) curve can be plotted. From this curve, the optimal threshold can be extracted as well as the Area Under Curve (AUC) which is a good indicator of the performance of the deep learning anomaly detector.

The same concept can be used for the HOG + HOF based model, where instead of the pixel value the feature descriptor is used. Hassan et al. [48] also augmented the video data temporally by altering the skipping strides that sample the videos to increase the size of the dataset for training.

Similarly, Luo et al. [49] and Chong et al. [50] proposed approaches with the Convolutional Long Short-Term Memory autoencoder (ConvLSTM-AE) and Spatiotemporal Autoencoder (SPAE) for video anomaly detection, combining CNNs for spatial information and Convolutional Long Short-Term Memory (ConvLSTM)

units for temporal information. The architecture employs 2D-convolutions for appearance extraction and 2D-deconvolutions for reconstruction. Reconstruction loss is utilized for training and anomaly detection during testing, assuming that the model's inability to accurately reconstruct abnormal videos indicates anomalous events.

Deepak et al. [51] proposed the Residual Spatio-Temporal AutoEncoders (R-STAE) architecture, leveraging Residual Network (ResNet) [52] blocks and ConvLSTM units for anomaly detection. ResNet blocks incorporate skip connections to counter the vanishing gradient problem, while ConvLSTM units facilitate spatial information propagation. The ConvLSTM unit combines convolutions with the fully connected Long Short-Term Memory (LSTM)s through replacing the weights with convolutional filters. The connections in the LSTM allow the network to access information from previous frames. Hence, temporal information is learned by the model. The encoder-decoder architecture employs 3D-convolutional layers for spatial feature extraction, 2D-ConvLSTM for temporal-spatial information processing, and 3D-deconvolutional layers for image reconstruction. A hyperbolic tangent activation is used to ensure the decoder and encoder are symmetrical and Batch Normalization (BN) is used to improve training efficiency. After training the regulatory score is calculated as in 2.2 but in this case, $e(t)$ is the Mean Squared Error (MSE) between the reconstructed frame and the actual frame.

Luo et al. [53] proposed a model which is intended to learn the normality of multiple scenes after only being trained once with a dataset composed of examples from multiple scenes. A sparse-coding-based approach is taken, where a dictionary is trained to learn temporal and appearance features from normal events. Then, the features can be used to reconstruct the events. The Temporally-coherent Sparse Coding (TSC) can be thought of as a stacked Recurrent Neural Network (sRNN), which allows all parameters to be optimized concurrently using an AE and the last layer of the sRNN is used to reconstruct the image. The AE is trained with reconstruction error using a min-batch based Stochastic Gradient Descent (SGD) algorithm. The sRNN architecture allows the model to be shallower, thus training and detecting anomalies is faster. Anomalies are detected based on reconstruction errors, with video frames segmented into patches at various scales for feature extraction via convolutional networks.

Memory Module

Memory modules offer an alternative approach for handling temporal dependencies in video data. While traditional LSTM models aim to extract temporal information through small memory modules, Park et al. [54] argue that such modules

may compress data excessively, leading to inaccurate representation of temporal features. In response, memory modules with global memory, capable of read and write operations, have been proposed to address this issue. However, these models are more challenging to train as they require per-layer supervision.

Gong et al. [55] observed that the assumption that an AE trained with normal data cannot accurately reconstruct anomalous patterns does not always hold true. To tackle this, they introduced the Memory-augmented Autoencoder (MemAE) architecture, which separates the encoder and decoder using a memory block. The memory block is updated by the encoder with the most representative items, allowing the decoder to reconstruct frames using only these selected features. This is enforced by only allowing the memory module to occupy a limited number of space features. During training, a differentiable hard-shrinkage operation is applied to limit the memory module's capacity, ensuring that the memory module does not contain a dense sample of small weight which can potentially reconstruct an anomaly. The memory model is made to save more representative large memory items, which better represent the normality of the scene. To further promote scarcity the training loss function consists of a summation of entropy minimization of the memory items and the L_{2-norm} as a reconstruction loss.

Additionally, Park et al. [54] proposed using memory modules where each item represents a feature, similar to MemAE, but with the inclusion of feature compactness and feature separateness losses. These losses encourage similar features to cluster together in memory space while maintaining separation between different feature clusters to account for the diversity of normal patterns. The use of a feature compactness loss, ensures similar features are grouped close to each other in memory space, penalizing intra-class variation and feature separateness loss encourages the second closest features to be as far as possible from the closest features. Thus, similar items in memory form their own cluster of items which are separate from the other items clusters. With all the clusters representing a normal activity. The encoder and decoder are modelled as the widely used U-net architecture [56] with the last BN and Rectified Linear Unit (ReLU) and replaced with a L_2 normalization layer to scale the features to the same size as the input.

Two versions of the models were implemented, one which reconstructs the input and the other which constructs the future frame. To prevent the model from learning to copy the input frames for reconstruction model, skip connections are only used for the prediction task. An abnormality score is computed based on the Peak Signal to Noise Ratio (PSNR) and L_2 between reconstructed and input frames, which is further normalized. Park et al. [54] also introduced an update strategy based on reconstruction error as an indicator during testing to prevent

abnormal patterns from affecting memory items, ensuring continued model refinement. Notably, their method stores fewer features compared to MemAE and provides more explicit feature separation.

Latent Space Classification

As mentioned earlier, AEs can compress data into a latent vector composed of autonomously selected features, which can then be utilized alongside the reconstruction loss to detect anomalies.

Singh et al. [57] focused on detecting traffic collisions, which encompass various stages such as pre-collision, collision, and post-collision, each with unique characteristics. To address this, they proposed a Stacked Denoising Autoencoder (SDAE) to automatically extract features from normal data. SDAEs are composed of stacked Denoising AE which compresses corrupted samples of data, thus reducing the amount of corruption in the samples. This model is trained on normal data and anomalies are detected using the reconstruction error. In addition, a one-class SVM is trained on the latent features of the SDAE to classify anomalies. To further consider all types of collisions indicators, trajectories from a detector and tracker are used to find overlapping trajectories which indicate that a vehicle has collided.

Similarly, Abati et al. [58] suggested using reconstruction loss and an autoregressive model trained on the latent space as anomaly indicators. Taking advantage of the success of autoregressive models in the modelling of sequential data to increase the accuracy with which anomalies are detected.

2.2.2 Prediction Based Methods

Prediction-based anomaly detection methods are closely related to reconstruction-based methods and are often used together. While reconstruction-based methods are trained to reconstruct the current frame, prediction-based anomaly detectors are trained to predict future frames. Reconstruction models typically learn simple representations and aim to replicate the input frame, whereas prediction models focus on recreating future frames based on the input. Encouragingly, predicting object positions in the future necessitates extracting relevant temporal features, prompting the encoder to capture relevant temporal information for accurate predictions [1, 59].

The fundamental assumption underlying prediction-based models is that normal events are predictable, whereas abnormal events are not. For instance, Park et al. [54] proposed two models: one for predicting future frames and another for reconstructing present frames. Although the architectural difference be-

tween the reconstruction and prediction models was minor, the prediction model exhibited superior performance.

Medel et al. [59] proposed a ConvLSTM architecture which uses an encoder made entirely of ConvLSTM units and decoder with two branches each made of three ConvLSTM and a convolutional layer. One of the branches reconstructs the past frames and another predicts the future frames. The future decoder is conditioned to constrain future variation by incorporating the output from the decoder and the summed output from the previous frame as inputs to the first layer, providing information about the previous frame, which aids in the prediction task. The utilization of ConvLSTM layers is critical due to their ability to learn temporal relationships between frames, essential for both reconstruction and prediction tasks. The transitional kernels within ConvLSTM layers capture transition states between frames, with kernel size impacting the type of motion captured; larger kernels are effective for faster movement, while smaller ones capture slower movement. Reconstruction error is computed as the MSE error between output and corresponding input frames, and the regulatory score is defined similarly, where $e(t)$ represents the MSE score for each frame.

Zhao et al. [1] highlighted the importance of 3D convolutions for propagating temporal features throughout the network. They argued that temporal information collapses after the first layer when using fully connected convolutional layers or 2D convolutions, such as the method proposed in [48], despite the input being a 3D video segment. Therefore, they introduced the Spatio-Temporal AutoEncoders (STAE) to extract spatial and temporal information using 3D convolutions. To further force the model to learn temporal patterns the decoder of the STAE has two branches, one performing reconstruction and the other performing predictions. A diagram of the architecture can be seen in Figure 2.4. The STAE architecture is flexible regarding the number of channels, requiring three channels for RGB frames, one for grayscale frames, and two if optical flow is used with a grayscale image.

The training loss is a summation of the reconstruction loss, prediction loss and a regularization parameter which reduces the complexity of the model. The reconstruction loss is defined as the Euclidean loss as shown in Equation 2.3.

$$L_{rec} = \frac{1}{N} \sum_{i=1}^{N} ||X_i - f_{rec}(X_i)||_2^2 \tag{2.3}$$

where Xi is the i^{th} hyper-cuboid input batch of size N and $f_{rec}(X_i)$ is the output from the reconstruction branch.

The prediction branch aims to recreate the future frames from the input, thus encouraging the encoder to extract relevant temporal features. The au-

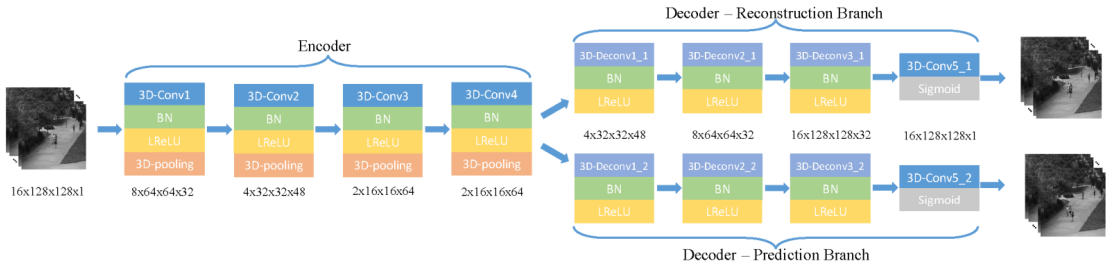


Figure 2.4 Figure showing the detailed architecture of the STAE model from [1]

thors [1] proposed a prediction loss that encourages the detection of future movements, not the prediction of new objects that enter the field of view. Thus, the loss function for prediction gives the highest weight to the correct prediction of frames in the immediate future. The prediction loss is described in Equation 2.4

$$L_{pred} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T^2}\sum_{t=1}^{T}(T-t)||X_{i+T}^{t} - f_{pred}(X_i)||_2^2 \tag{2.4}$$

where $f_{pred}(X_i)$ is the output from the prediction branch, X_{i+T} is the future T frames and the super script t in X^t denotes that it is the t frame from the video clip X. In essence, the weighting of the prediction loss decreases as the t increases.

After the model is trained on normal data, the normalized reconstruction error of the test videos can be used as the regulatory score, as expressed in Equation 2.2 but with $e(t) = L_{rec}$. Taking advantage of the assumption that the model will only be able to accurately reconstruct normal frames to detect anomalies.

Zhao et al. [1] also proposed that video data is augmented spatially using typical methods such as random cropping, changes in brightness and blurring since Deep Neural Network (DNN) require large amounts of data to train. However, temporal augmentation, by changing the strides of the video, is avoided since that would alter the speed of the objects and speed is an important attribute to determine anomalies.

2.2.3 Deep Generative Methods

Recent advances in generative models have introduced deep-learning generative models that can be trained using back propagation. Generative models have been leveraged for anomaly detection tasks, typically trained to learn the distribution of normal samples. Consequently, these models can only generate normal data samples. When presented with anomalous samples during testing, the model fails to generate the anomalous parts of scenes, enabling anomaly detection. Amongst these new generative networks are the Generative Adversarial Networks (GAN)s,

Variational Autoencoder (VAE) and Adversarial Autoencoder (AAE) all of which will be explored in this section. Each of the models uses a slightly different assumption to detect anomalies which will be explained in the following sections.

Generative Adversarial Networks

GANs [60] represent a class of deep generative models comprising a generator network tasked with producing image samples and a discriminator responsible for distinguishing between real training samples and generated fake samples. These two components engage in an adversarial training process, constituting a zero-sum game wherein the generator aims to deceive the discriminator by producing realistic samples, while the discriminator endeavors to accurately discriminate between real and fake samples [47]. In the context of anomaly detection, GANs are trained exclusively on normal data, learning to represent normal scenes. Consequently, the model cannot generate anomalous events, enabling the detection of anomalies through significant discrepancies between the input and generated scenes. Moreover, the discriminator itself can serve as an anomaly detector, having been trained to identify anomalous data, effectively framing anomaly detection as a fake data detection problem. Despite being challenging to train, GANs demonstrate promising performance when trained effectively [46].

Chong et al [50], proposed the Adversarially Learned One-Class Classifier (ALOCC) GAN which is compromised of a reconstructor and deconstructor. The networks compete with each other while training to understand the underlying concept in the target class and classify testing samples. The architecture splits the data into patches as part of the processing pipeline. Specifically, the deconstructor network takes noisy input samples augmented with Gaussian noise and reconstructs them, effectively refining the samples belonging to the target class while distorting outlier samples. This process is applied patch-wise across the input data, allowing for local refinement and distortion. Similarly, the discriminator network operates on patches of data, aiming to distinguish between original (positive) samples and reconstructed ones. Although the model does not explicitly capture temporal information the patch-wise processing enables the model to capture spatial information effectively and make localized decisions about the presence of anomalies or outliers within the input data, which allowed the model to generalize well and detect anomalies in the A Day on Campus (ADOC) dataset [61]. Although the model does not explicitly capture temporal information the patch-wise processing enables the model to capture spatial information effectively and make localized decisions about the presence of anomalies or outliers within the input data, which allowed the model to generalize well and detect anomalies in

the ADOC dataset [61].

Yan et al. [62] introduced 3D convolutional generative adversarial network (3D-GAN), utilizing the third dimension to account for temporal irregularities in videos. Compared to conventional 2D-GANs, 3D-GAN exhibits superior performance in detecting temporal anomalies. The authors approach anomaly detection as a fake data detection task, training the model solely on normal data such that any anomalous event is identified by the model as fake data. During testing, the discriminator, having been trained with normal data, flags normal data as fake. The loss function for a typical GAN can be described as follows:

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}}[log(D(X))] + E_{z\sim p(z)}[log(1-D(G(z)))] \tag{2.5}$$

where the discriminator D tries to maximize the loss function by classifying all the real data from the fake data and the generator G tries to minimize the loss function and create fake convincing data to fool the discriminator. p_{data} is the distribution of the real data whilst $p(z)$ is the distribution of the noise.

In their loss function, Yan et al. [62] treat regular data as the real data with which to train the generator and discriminator. The equation for training the 3D-GAN with regular data becomes:

$$\min_D L(D) = -E_{X\sim p_{regular}}[log(D(X))] - E_{z\sim p(z)}[log(1-D(G(z)))] \tag{2.6}$$

$$\min_G L(G) = E_{z\sim p(z)}[log(1-D(G(z)))] \tag{2.7}$$

where $p_{regular}$ is the distribution of the regular data. Since it was proven to reduce the vanishing gradient problem, the loss function for the generator was replaced with the equation:

$$\min_G L(G) = E_{z\sim p(z)}[log(-D(G(z)))] \tag{2.8}$$

The generated data z is extracted from a Gaussian distribution and X is the video data.

Dong et al. [63] proposed a method which explicitly targets both appearance and motion information from video by using two discriminators. The Dual-GAN model introduced utilizes one discriminator to identify fake appearance within video frames and another discriminator to detect fake motion using optical flow derived from real or generated videos. The generator is tasked with generating future frames.

The generator uses a U-net [56] architecture with skip connections and the discriminators use a PatchGAN [64] architecture which focuses on patches of a frame and their average is taken to determine if the frame is real or fake. To enhance the prediction performance the generator and discriminator are optimized alternately with normal samples only. The generator is trained to minimize the intensity difference between the predicted future frame and the ground truth. A gradient penalty is also added to preserve sharpness in the frames. Adversarial penalties are also set, where one penalty is generated for the unrealistic appearance of frames and another for unrealistic optical flows. Unrealistic optical flows are produced using the pre-trained FlowNet [65] calculated between the previous frame and generated frame, then, compared to the optical flow from the actual frame and ground truth. The discriminator loss functions produce penalties when the discriminators are unable to determine if a frame is fake. Finally, during testing, the PSNR is used to measure the difference between the predicted frame and its ground truth to detect anomalies.

Similarly, Liu et al. [66] also recognised the flaw in the assumption of reconstruction based anomaly detection and proposed an anomaly detection method where the future frames are predicted using a GAN. The GAN is trained to predict normal future frames. Thus, when presented with the task to predict an abnormal future frame, the prediction is not close to the actual frame, which indicates an anomaly has occurred. The generator, or predictor, is implemented as a slightly modified U-Net [56] network to mitigate the vanishing gradient problem, while the discriminator adopts a patch discriminator. The loss function to train the generator imposes constraints on the appearance but also on motion. To ensure the predicted frame matches the ground truth in the RGB space, an intensity loss is used as a motion constraint. Additionally, a gradient loss is imposed to promote sharper images. To ensure motion coherence, FlowNet [65] is employed to generate the optical flow estimation of both the ground truth and the predicted frame. A temporal loss is then calculated based on the differences between these optical flows. Through ablation studies, it was found that the PSNR function is the best measure of the difference between the predicted frame and the actual frame when detecting anomalies.

Variational Autoencoders

VAE are AE which have been trained to encourage regularization of the latent space such that new images can be generated. For a typical AE, the latent space would be a vector of unknown nature, hence the input required to generate new content would be unknown. By adding a regularization term to the loss function,

the latent space can be forced to follow a distribution. The loss function is derived using the statistical technique of variational inference [67]. VAEs are praised as simple to implement and providing state-of-the-art results among the generative models.

Xu et al. [68] proposed a two-stage anomaly detection pipeline leveraging features learned automatically from a VAE. These features are preferred due to their higher independence compared to those extracted from traditional AEs and ConvAEs, making them easier to use in the second stage of anomaly detection. Anomalies are detected by estimating the probability of feature occurrence. In the first stage, a convolutional VAE is trained with frames from normal data to learn the spatio-temporal features of normality. The loss function comprises the reconstruction loss and the Kullback-Leibler divergence, encouraging the latent vector to approach a Gaussian distribution. After training, a Multivariate Gaussian model is fitted to the feature map of normal data for the second stage. During testing, any new point with a low probability when fitted to the Gaussian model is considered anomalous.

To better capture the temporal aspects of video data, Lu et al. [69] proposed combining a VAE with a ConvLSTM. Anomaly detection is framed as a future frame generation task. The authors argue that the video data is sequential data which cannot be recreated using a typical VAE. Hence, the authors introduce the Convolutional Variational Recurrent Neural Network (Conv-VRNN) model by using Recurent Neural Network (RNN)s to learn the sequential video data. The architecture comprises four modules: the first generates the distribution of the latent variable based on the hidden state of the previous frame; the second is an encoder extracting the latent variable from the previous frame's hidden state and the current frame; the third captures temporal information using a ConvLSTM; and the final module uses the hidden state to reconstruct the frame. The model is trained using the L_1 loss, Multi-Scale Structural Similarity Measure (MSSSIM), and gradient difference, with the same loss serving as an indicator of abnormality.

Adversarial Networks

Adversarial Autoencoders (AAEs) are Autoencoders (AEs) transformed into generative models through adversarial training. Similar to Variational Autoencoders (VAEs), AAEs still use regularization terms [70].

Lee et al. [71] proposed the Spatio-Temporal Adversarial Network (STAN) to handle the spatio-temporal characteristics of anomalies by learning patterns in normal data samples. The proposed STAN model consists of a generator and a discriminator. The generator comprises a spatial feature encoder, a bidirec-

tional Convolutional LSTM (ConvLSTM) to encode temporal features, and a spatial decoder to generate a compressed frame. The authors suggest that spatio-temporal features should be encoded in both the forward and backward directions, with previous and future frames fed as input through the bidirectional configuration. The ConvLSTM's inputs consist of 5 concurrent frames, enabling the capture of spatio-temporal features. Figure 2.5 illustrates this architecture. To classify frames as real or fake, the discriminator is designed as a 3D Convolutional Neural Network (3D-CNN).

Lee et al. [71] proposed training their generator with a loss function composed of two components: the realism loss and the pixel-wise loss. The realism loss is determined by the discriminator's score, which ranks the authenticity of a sequence of frames, while the pixel-wise loss measures the pixel-wise difference between the input frame and the generated frame. The abnormality score is defined as the combination of the generator's adversarial score and its pixel-wise score. During training, the generator aims to minimize the pixel-wise score first, followed by adversarial training of the discriminator and generator.

Georgescu et al. [72] aimed to enhance the performance of deep learning-based anomaly detectors by introducing some form of supervision. One approach involved integrating a YOLOv3 object detector [26] to obtain bounding boxes and classes for processing in the deep pipeline. Another method was to include a diverse set of unrelated images to substitute the need for anomaly samples, which were then used for adversarial training. This approach allowed the model to be used across various related scenes while being trained only once, making it nearly scene-agnostic. Optical flows were calculated using Selflow [73] between the previous and next frames to account for motion in the scenes.

Separate AEs were employed to learn motion and appearance, each fed with optical flows from the previous and next frame, as well as object detections. The AEs were trained not only to reconstruct the input but also to unlearn examples from the set of adversarial images through an adversarial decoder branch. This adversarial unlearning procedure compelled the AEs to forget generic object patterns they had learned. Following the AEs, a binary classifier was used to differentiate between normal and adversarial samples. During testing, a sample's normality was assessed based on the binary classifier's results following the three AEs.

2.2.4 Cascaded Models

End-to-end trainable models are sometimes criticised for their inefficiency. For that reason, many proposed the use of cascaded models where small models feed

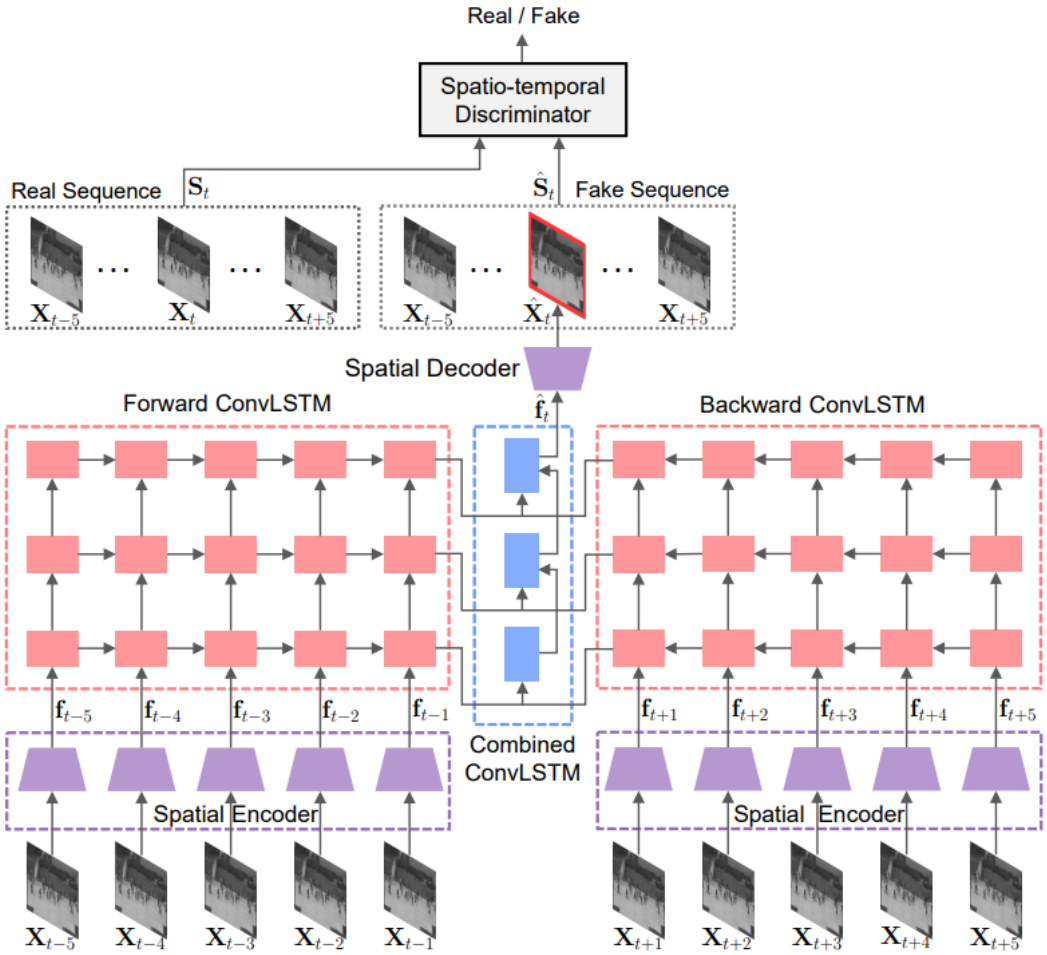


Figure 2.5 Figure showing architecture of the Spatio-Temporal Adversarial network (STAN) model from [71].

their output to the next model. Using a cascaded approach each shallow model can be trained independently, which could simplify the training process and reduce the inference delay.

Sabokrou et al. [74] noted that using the same resolution for detecting anomalies across the entire frame in a video is not computationally efficient. Some locations in the frame, such as the background, are obviously not anomalous and do not require processing at a high computation cost. This unnecessary processing leads to proposed end-to-end models requiring more complex hardware than necessary and using long processing times. To address these issues, they proposed a Cascaded Deep Neural Network (CDN). In the first stage of detection, the video is split into large cubic patches and processed using a cascaded stacked AE. The AE has two substages to classify two resolutions of the cubic patches. A Gaussian classifier is used after each layer to classify each patch as anomalous or not. If the two substages of the AE classify the patch as normal, the patches

are discarded; otherwise, they are sent to the second stage. The second stage uses a cascaded CNN with Gaussian classifiers at the sub-layers. The cascaded CNN consists of four sub-stages to classify a cubic patch as anomalous or not. These classifiers are fit to a Gaussian model, and a threshold is set based on the Mahalanobis distance between a new point and the Gaussian model.

Pawar et al. [75] proposed a convolutional AE to extract the latent space vector per frame in the video and capture spatial features. The frame feature is then fed to a sequence-to-sequence LSTM AE, which captures the temporal information using a sliding window on the video data. To fit the normal data and find a decision boundary to separate the normal data from the abnormal data, a Radial Basis Function (RBF) layer is used prior to the latent space vector for the LSTM AE. This approach allows the normal data to be fit into a Gaussian distribution during training, enabling the identification and removal of anomalous samples through a threshold.

2.2.5 Discussion

The methods reviewed in this section have been summarized in Table 2.5 along with the frame AUC obtained on the most common benchmarking datasets for video anomaly detection which are; UCSD Ped1 and Ped2 [76], CUHK Avenue [77] and ShangaiTech [66]. Some methods reviewed do not make a distinction between pixel level and frame level AUC, in those cases, it was assumed that the paper presented the frame level AUC.

Unfortunately, these benchmark datasets are intended for crowd behaviour anomaly detection, hence there is no guarantee that high detection rates for crowd behaviour anomalies would translate well to the detection of traffic anomalies. Sabokru et al. [74] obtained the highest AUC for the UCSD Ped 1 [76] also report detection latencies of 15 fps on a CPU and 130 fps on a subpar GPU. An impressive result considering that most models tested on state-of-the-art GPUs do not even share their speed results. Overall, the benchmark datasets are good approximate indicators of the quality of an anomaly detection model but give minimal indication of how a model will perform for more complex traffic behaviour.

Zhao et al. [1] who proposed the STAE models are among those who criticize the simplicity of the commonly used benchmark datasets. They note how none of the anomalies in the UCSD Pedestrian datasets [76] contain strictly temporal anomalies and all anomalies can be detected from a single frame. Which leaves doubts to whether the methods tested on the datasets UCSD Pedestrian datasets [76] can actually detect temporal anomalies well. To address this issue the authors of [1] proposed a temporal anomaly detection dataset of challenging

Table 2.5 Table showing a summary of the methods presented in this literature review with the percentage frame level AUC result for the most used datasets for benchmarking.

Model	Description	Peds1	Peds2	Avenue	Shangai
ConvAE [48]	3D Convolutional AE to reconstruct normal frames	81.00	90.00	71.30	
STAE [1]	3D-Convolutional AE with a prediction and reconstruction decoder	92.30	91.20	77.10	
STAE- optflow [1]	3D-Convolutional AE with a prediction and reconstruction decoder with optical flow input	87.10	88.60	80.90	
AE + CDN [74]	3D Conv AE to identify normal patches followed by a deep CDN to classify the patches.	**99.60**	99.60		
GAN [66]	GAN with two discriminators; motion and apperance	83.10	95.40	84.90	72.80
ConvLSTM-AE [49]	Convolutional AE with ConvLSTMs to learn the temporal information	75.50	88.10	77.00	
sRNN [53]	Stacked RNN to learn the parameters for sparse encoding and detect anomalies		92.21	81.71	68.00
STAE [50]	Convolutional AE with ConvLSTMs to learn the temporal information		89.9	87.4	80.3
STAN [71]	Bidirectional ConvLSTMS and adversarial training	82.10	96.50	87.20	
3D-GANs [62]	3D Convolutuional GAN to genrate normal data			79.60	
SDAE [57]	A branch with detector/tracker to find intersections of tracks. A branch with SDAE to compress frames into their features.		95.60	84.90	73.70
Conv-VAE [68]	VAE to extract features and fit to a Gaussian model	95.70	92.30		86.00
MemAE [55]	Convolutional AE with Memory module to store the most relevant items and reconstruct normal frames		94.10	83.30	71.20
AE + Autoregressive [58]	Deep AE to extract the latent vector form frames with an autoregressive network operating on the latent space to learn spatiotemporal information.		95.40		72.50
ConvVRNN [69]	VAE with ConvLSTM to capture temporal information	86.27	96.06	85.78	
Memory-Guided AE [54]	Reconstructive AE with memory module that stores different normal activities separately		90.20	82.80	69.80
Memory-Guided AE [54]	Predictive AE with memory module that stores different normal activities separately		97.00	88.50	72.00
R-STAE [51]	AE with residual blocks and ConvLSTM units to avoid the vanishing gradient problem and extract temporal features		88.30	82.00	
Dual-GAN [63]	A GAN with two discriminators; motion and appearance		96.00	85.00	74.00
AAE [72]	Introduce an element of supervision in AEs by using object detectors and a set of unrelated images for adverserial training		**99.70**	**90.40**	**89.30**
seq-2-seq LSTM [75]	Convolutional AE to extract features per frame fed to an LSTM to extract temporal relations	88.60		79.22	70.50

traffic datasets from real-world traffic accidents to test their temporal anomaly

detection method and report a high AUC of 79.60%. Sing et al. [57] and Pawar et al. [75] are other methods which have been benchmarked in a traffic scenario with the IITH real world accident dataset [57] and obtained an AUC of 81.06% and 79.00% respectively. The vehicular collision datasets are complicated datasets and detecting traffic accidents as anomalies is a complex task. However, vehicular collisions are not the only traffic anomaly that exists. Another dataset with strong, although staged, temporal components is the UMN dataset, which has crowds of pedestrians walking randomly then suddenly running away. Parwar et al. [75] with the seq-2-seq LSTM and Yan et al. [62] both benchmarked on the UMN dataset, scoring an AUC of 87% and 95% respectively.

The Street Scene Dataset [2] offers comprehensive annotations for various anomalies occurring on roads, including bicycles on pavements and u-turns. However, it also includes irrelevant annotations, such as people conversing on the pavement and people walking dogs, which are not directly related to traffic anomaly detection. The authors compare their work to Hassan et al.'s ConvAE [48]. They note that the autoencoder approach endeavors to model entire frames simultaneously, rather than creating smaller models for distinct spatial regions. While this approach has demonstrated success with previous datasets, it proved less effective in addressing the wide range of normal variations present in the Street Scene dataset.

Another notable observation from the benchmarking results shown are the results from Hassan et al. [48] who found that the ConvAE performed the best when the input are the raw video frames and not the handcrafted features. Indicating that deep learning models are better at extracting features from raw video data than the commonly used handcrafted features. Park et al. [54] who proposed the Memory-Guided AE also show that their model detects more anomalies when using the proxy task of predicting future frames and not reconstructing them.

Pravan et al. [61] also compared various available methods on their novel dataset ADOC. This dataset is extensive and intricate, reflecting real-world scenarios of pedestrian activity spanning a 24-hour period. As a result, the dataset exhibits a variety of illumination changes. Their study focused on assessing the performance of models under different illumination conditions. They observed that the SPAE model [50], which captures temporal features, performs best among all tested models when trained exclusively with daytime photos or solely with nighttime frames. However, the ALOCC model [78] outperform SPAE when trained using data from both day and night time periods. The authors indicated that conducting patch-level analysis appears to be advantageous in learning representations across different illumination conditions, as the approach is capable of capturing local features more effectively compared to other methods. They

concluded that, both temporal analysis and patch-level analysis are essential for building robust algorithms for anomaly detection.

2.3 Datasets

The effectiveness of anomaly detection methods is closely tied to the quality and relevance of the datasets. Therefore, selecting appropriate datasets that accurately reflect real-world scenarios is essential. The following section explores various datasets used in the literature for road traffic anomaly detection, categorizing them into trajectory and video datasets. These datasets are critical for training and evaluating the performance of anomaly detection models under diverse conditions.

Trajectory datasets, ideal for classical anomaly detection, are advantageous due to their small size, facilitating easy download and use. However, their compact nature often limits available features, such as speed and occurrence time, potentially restricting algorithm development.

To be relevant the data needs to be collected from a video camera. The trajectories can then be extracted manually by a team of experts tracking the vehicle or through a detection and tracking algorithm. Since the trajectories are already extracted the user of the data would have no control over error handling and features extracted from the video. Thus, the development of the anomaly detection algorithm would not be tested on a real life scenario. Overall trajectory datasets were more difficult to identify since they tend to be older and no longer available. In fact only three traffic trajectory datasets were found to be available for download as shown in Table 2.6. The Piciarelli Trajectory Dataset [33] is a set of trajectories created through a generator with a few examples of trajectories which are anomalous. The normal trajectories are grouped as clusters. The MIT Trajectory Dataset [39] contain trajectories extracted based on background subtraction using a Adaptive Gaussian Mixture model from a single camera over a parking lot. The dataset has no anomaly labels and is intended manly for semantic region modeling. Next Generation Simulation [79] is a dataset of trajectories and other information from a set of highways tracked by the author's own tracker/detector. This dataset also does not have any anomaly labels. Both datasets with no labels could still be useful in certain settings for the evaluation of anomaly detectors.

Recent data releases focus on video data, the second type of datasets considered in this section. Video data can be used to extract trajectories using a detector and tracker, if the camera angle is suitable. Hence, they can be used

Table 2.6 Table of Trajectory Datasets

Year	Name	Type	Subject	Comments
2008	Piciarelli Trajectory [33]	Simulated Trajectories	Trajectories	Trajectories are simulated
2008	MIT Trajectory [39]	Real trajectories from an object detector/tracker [3]	Traffic and Pedestrian Trajectories	No anomaly labels
2018	Next Generation Simulation [79]	Real trajectories from an object detector/tracker	Traffic Trajectories	No anomaly labels

for evaluation and training of classical anomaly detection algorithms. Video data can also be used to train the deep learning algorithms for anomaly detection. This makes video data more suitable if a direct comparison is needed between the two main types of methods reviewed. Video data also contains richer information than can be shared with a trajectory dataset, giving more freedom to the researcher to extract the data required. Only a few datasets exist which deal exclusively with vehicle traffic videos. The IITH accident dataset [57], CADP dataset [80] and the NVIDIA AI CITY 2021 [81] show various traffic accidents in different settings collected from CCTV cameras in different locations and collected from Youtube, respectively. Similarly, the UCF-Crime dataset [82] also has some examples of vehicle collisions. The Street Scene Dataset [2] is a recent traffic dataset intended for the training of deep learning models. This dataset was labeled for a wide variety of relevant anomalies such as jaywalking, biker on side walk and U-Turns, which is a more realistic depiction of road anomalies. In addition, the angle used to record the video has minimal obstructions, which is ideal if the vehicles are to be detected by an object detector. The MAVAD dataset [83] has a wide variety of relevant anomalies which could be precursors to traffic accidents such as U-turns and pedestrians crossing the road taken from three scenes in Malta.

The crowd datasets must also be mentioned since the deep learning methods which are found in literature are often benchmarked on crowd scene dataset such as the CUHK Avenue Dataset [77] and UCSD Peds1 and Ped2 datasets [76]. The CUHK and the UCSD datasets consider anomalous events to be non pedestrian entities on walkways and anomalous pedestrian paths. The characteristics of these anomalies are similar to the anomalies that would need to be detected in a traffic scene which could be anomalous U-turns and exits and foreign objects on the road. The anomalies detected in the UMN [84] crowd dataset are also

[3] pedestrian and vehicle trajectories

relevant. They stage scenes of a crowds running away which is a type of collective anomaly. In the traffic setting, a traffic jam would be considered a collective anomaly, hence would need to be detected by the deep learning model. Hence, methods which perform well on the [84] crowd dataset would be likely to detect collective traffic anomalies. A more recent crowd dataset is the ADOC [61] dataset, which monitors a pedestrian area for 24 consecutive hours, in different lighting conditions. Detecting anomalies in different lighting conditions is important in a road traffic making this dataset also very relevant. Table 2.7 shows details of the datasets used in literature.

Table 2.7 Table of Video Datasets

Year	Name	Anomaly	Subject	Comments
2008	UCSD Peds1 [76]	skaters, runners, bikers, cart, cross walking, wheelchair	Crowded Scenes	grayscale
2008	UCSD Peds2 [76]	cart, bicycle	Crowded Scenes	grayscale
2008	Subway Enter and Exit [85]	walking in the opposite direction, person cleaning the walls, jumping over turnstiles	Crowded Scenes	
2009	UMN [84]	crowds running away	Crowded Scenes	staged
2013	CUHK Avenue [77]	strange action, wrong direction, abnormal object	Crowded Scenes	slight camera shake
2017	Live Video [86]	car accidents, robberies, kidnappings,	Crowded Scenes	various scenes from IP cameras
2018	ShangaiTech [66]	bicycle, push chair, fights, falls.	Crowded Scenes	13 different scenes
2018	UCF-Crime [82]	arson, assault, road accident, burglary, explosion	Dangerous behaviour	labels also suitable for activity recognition
2018	IITH accident [57]	accidents	Traffic Accident	various scenes
2018	CADP [80]	accidents	Traffic Accident	collected from youtube
2020	Street Scene [2]	jaywalking, biker on side walk, U-Turn	Traffic Scene	angle can be used with a tracker detector
2021	NVIDIA AI CITY 2021 [81]	car crashes, stalled vehicles.	Traffic Scene	various scenes
2022	UBnormal [87]	jay walking, stealing, car accident	Crowd, Traffic scene	virtual scenes
2022	ADOC[61]	bag left unattended, truck on walkway, scooter, camera overexposure, smoker	Crowded Scenes	different lighting conditions through 24 hours
2023	MAVAD Dataset[83]	pedestrians crossing, bicycle, bus, u-turn, scooter	Traffic Scenes	three different scenes in Malta

3 Anomalies

By definition, anomalies are data points or patterns that do not conform to the rest of the normal data [5] and are often uncommon. In the context of road traffic, many previously proposed methods and datasets defined anomalies as car crashes and accidents [57, 80]. This type of anomaly detection can serve to reduce response times for medical personnel and traffic authorities. The anomaly detection methods investigated in this work were, however, intended for a different purpose. That is, to investigate the anomalous events that correspond to questionable judgement of drivers on the road, which are precursors to accidents. The anomalies detected could then be used to decide on infrastructure updates and enforcement that can prevent accidents.

Anomalies are fundamentally unpredictable and rare events, making it ambiguous to define which anomalies should be investigated to assess driver judgment errors. To make this research applicable to a wide variety of scenarios, a set of anomalies was chosen and presented in Table 3.1. These anomalies are divided into generalizable categories that other researchers can use. While the list is not exhaustive, categorizing anomalies allows for broader application in analysis and detection. Table 3.1 also lists the categories of each anomaly along with the predicted challenges these categories may present to an anomaly detector.

Five categories of anomalies were identified. The first category is *Non-typical vehicle paths*. This category encompasses U-turns, vehicles crossing centre lines or exiting/entering at unexpected points in the scene. The second category is *Non-typical location of object*. This category detects vehicles or pedestrians that go outside their dedicated area in a scene. This includes pedestrians who cross the road, bicycles not using their dedicated lane and vehicles in the wrong lane. The third category is *Non-typical slow vehicles*. This category aims to detect vehicles which are stuck in traffic, vehicles slowing down to perform strange manoeuvres and vehicles stopping and causing obstructions. The final category is *Non-typical vehicle*. This type of anomaly includes the detection of vehicles that are anomalous in certain locations for example heavy goods vehicles in residential areas and bicycles in Maltese datasets. The fifth category is *Unforeseen Occurrences*. These are anomalies that have not been observed in the data before, but would still be anomalous if they are spotted in the future data.

Each anomaly could belong to multiple classes, for example *overtaking* could belong to the *Non-typical location of object* as well as *Non-typical vehicle paths*. However, Table 3.1 only shows the main class an anomaly belongs to. Note that, heavy goods vehicles are defined as vehicles which are over 3,500kg in weight

Table 3.1 List of anomalies that will be considered with their associated category and the predicted challenge each category will

Anomaly Class	Anomaly	Predicted Challenges
1. Non-typical vehicle paths	Crossing center line U-turn Overtaking	The detector has to learn that a certain sequence of manoeuvres is anomalous.
2. Non-typical location of object	Pedestrian on the road or jaywalking Bicycle on wrong lane Vehicle on wrong lane	The detector has to learn that an object is anomalous only when in a certain area
3. Non-typical slow vehicles	Traffic Obstructions Parking	The detector has to learn that certain speeds of vehicles are anomalous
4. Non-typical vehicle	Heavy goods vehicles Bicycles Scooters	The detector has to learn certain vehicles are anomalous whenever they occur. Bicycles are only anomalous in the Malta datasets.
5. Unforeseen Occurrences	Unforeseen Anomalies	This section encompasses any anomaly that was not observed in the data previously thus could not be foreseen. Examples of this type of data but might be observed by annotators or found in the data in the future. An example of this is the horse cart on a road.

when carrying a load [1]. Weight is difficult to assess visually from a video, hence heavy goods vehicles are defined as vehicles with more than four wheels for the purpose of this work which corresponds to the definition used by the Greenroads Ltd. developers for the tracker-detector design.

[1] https://road-safety.transport.ec.europa.eu/eu-road-safety-policy/priorities/safe-vehicles/archive/heavy-goods-vehicles_en

4 Data

The diverse nature of the anomaly detection methods examined in this dissertation necessitates the collection of different types of data for training and testing of the algorithms. For the adopted structured data-based method, videos must undergo preprocessing using the detector-tracker provided by Greenroads Ltd. This detector-tracker processes video data as input and generates structured data, such as the trajectories of traffic entities within the video. Further elaboration on the structured data acquisition procedure will be provided in Section 4.1, including the necessary data annotation for evaluating the structured data anomaly detection method, as well as the datasets created specifically for this purpose. In contrast, the deep learning-based method relies on videos for both training and testing the algorithm. Specific requirements for this method will be outlined in Section 4.2, along with details regarding the datasets deemed suitable to fulfill these requirements.

4.1 Structured Data for Classical Anomaly Detection

The structured traffic data is extracted from video data using the detector-tracker provided by Greenroads Ltd. Broadly speaking the detector-tracker was not designed for anomaly detection but intended to collect long term data on the traffic behaviour from a video feed. Post-processing steps to reduce the size of the data for long-term storage and clean up the data inadvertently removed anomalous events from the outputs. The various settings, limitations and post processing of the detector-tracker that limit the performance of the structured data anomaly detector are discussed in Subsection 4.1.1. The limitations of the detector-tracker require the video input to be recorded with cameras positioned at specific angles for optimal performance. This is discussed in Subsection 4.1.2. Furthermore, the Mgarr, Zejtun and Street Scene datasets were annotated to better work with the detector-tracker output as discussed in Subsection 4.1.3. Structured data acquisition also needs to be evaluated with respect to detected anomalies and the method adopted is described in Subsection 4.1.4.

4.1.1 Structured Data Acquisition

The design of the classical anomaly detector method depends greatly on the quality and type of data that can be collected from the videos by the detector-tracker that had been already designed prior to the start of this research. The detec-

tor used was a YOLOv4 [88] trained and evaluated on publicly available datasets such as MIO-TCD [89], MS-COCO [90], and some dedicated datasets collected on Maltese roads labelled for the most common road traffic entity types. These labels are: pedestrian, car, bus, light-goods-vehicle, heavy-goods-vehicle, motorcycle, and bicycle. Table 4.1 shows the F1-score obtained for each traffic entity. This table was provided by the developers of the detector-tracker. The tracker used is a MOSSE [91] tracker, for which no evaluation metrics were provided.

Any multi-object detector-tracker method inevitably incurs a certain level of errors. In the case of the detector-tracker utilized, errors typically manifest as truncated, short trajectories. These occurrences arise when an entity is lost by the tracker due to obstructions or the entity moving at high speeds. These tracklets do not always correspond to vehicles in motion but may be artifacts, such as the bounding box jumping from one parked car to another. Such errors are also often referred to as *tracklets* by Gao et al. [17]. Post-processing following the detector-tracker involves filtering out short trajectories by retaining only those trajectories that have entered and exited through valid entry and exit lines. Using entry-exit processing, the detector tracker filtered out 57% of trajectories for the Mgarr dataset, 32% for the Zejtun dataset, and 49% for the Street scene dataset.

Another type of error, known as ID switches, occurs when part of a track belongs to one object and the remainder of its track to another. Unlike the tracklets, this type of error is not rectified by the post-processing step.

Figure 4.1 illustrates a frame from the Street Scene dataset [2], with entry and exit lines overlaid. These lines are established by the user during the initial setup process. While this post-processing step significantly cleans up the data, it also discards data from vehicles that neither entered nor exited at the designated lines. For instance, if a vehicle executed a U-turn just outside the entry line, this data would be discarded. Similarly, if a vehicle entered the lines and parked without exiting, its data would also be discarded. The entry and exit pair rule applies to all traffic entities except pedestrians, as the paths taken by pedestrians are less predictable, often involving exits from various points in the scene. Hence, pedestrian trajectories are more likely to be compossed of tracklets.

For every vehicle that successfully enters and exits the designated lines, the corresponding label, trajectory points, average speeds, entry and exit frames,

Table 4.1 F1-Score obtained for each traffic label entity provided by Greenroads Ltd.

Traffic Entity	car	bus	light goods vehicle	heavy goods vehicle	motorcycle	bicycle	pedestrian
f1-score	0.827	0.845	0.799	0.774	0.758	0.701	0.654

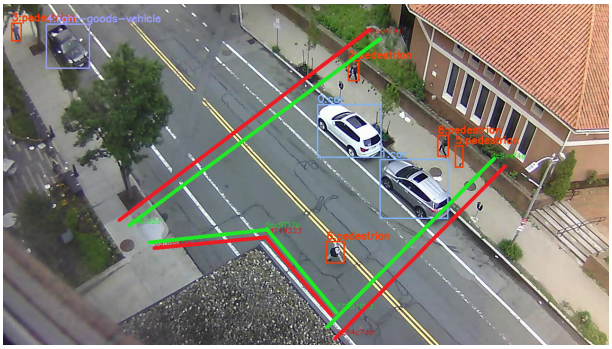


Figure 4.1 Frame from the Street Scene dataset [2] with exit lines marked in red and entry lines marked in green. Entities detected by the detector tracker and marked with bounding boxes.

and entry-exit lines are inferred and saved. For pedestrians, only trajectories are inferred.

It is important to note that the average speed calculated for vehicles serves as a rough estimate. This calculation is based on the assumption that each vehicle follows the same path between entry and exit lines. Consequently, operators are prompted to input the distance travelled over the most likely path between these lines, and speed is determined based on the time taken to traverse this distance. Additionally, trajectory points are extracted whenever a vehicle covers a minimum distance. The minimum distance threshold was set as low as possible for all tests conducted in this study. The operator is also asked to manually draw a mask on the areas in the frame not used by the vehicles. Masking out the useless areas improves the performance of the detector-tracker. It is worth mentioning that for each dataset location, the lines, background mask and other settings were established by the Greenroads Ltd. developer.

4.1.2 Data Requirements

For the classical anomaly detection method, a video stream labeled for relevant anomalies with a suitable camera angle (minimal perspective, but not overhead, and without visual obstructions) for the provided detector-tracker is necessary. To optimize the performance of the detector-tracker, the videos should ideally be a few minutes long to capture entire trajectories. Furthermore, the method must be evaluated on data typical of urban and suburban Maltese roads and junctions, where expected behavior is more complex than on a highway. The datasets should also include anomaly labels corresponding to the anomalies of interest as outlined in Table 3.1.

Despite the number of datasets available, none were found to be suitable for the classical anomaly detection method evaluation. Traffic datasets such as

Singh et al. [57] and Shah et al. [80] were collected from different scenes thus no varied data from a single location is available. Pedestrian datasets such as the UCSD dataset [76] did not have any relevant traffic anomalies and parking lot datasets such as the MIT trajectory dataset [39] and highway data from the Next Generation Simulation [79] did not have any available anomalous labels. Trajectory datasets such as Picarelli et al. [33] are also not suitable since it is simulated data which does not directly map to a real anomalous event on a road. The MAVAD dataset [83] is composed of trimmed videos that are a few seconds long showing only the short period during which an anomalous event has occurred. The whole trajectory of an anomalous event is thus not available, which makes it an unsuitable dataset, since the tracker-detector requires the trajectory from entry to exit, else the trajectory is discarded.

The Street Scene Dataset [2] was by far the most suitable since it recorded a busy road with some interesting events at an angle that is typically suitable for the detector-tracker, even though one exit point is obstructed. The data can also be considered to be similar to what could be collected on Maltese urban roads. The dataset collects video from a road junction with a variety of entities such as cyclists, various types of vehicles and pedestrians. The only limitation with this dataset is that it is not labelled for anomalies that could be indicative of errors of judgement by the drivers. The authors focus on relevant anomalies such as bicycles on sidewalks and U-turns, but then also include pedestrians loitering and dogs on sidewalks which are not relevant to the anomaly detection problem tackled in this project. The authors also only label as anomaly and do not include a descriptor of the type of anomaly. Thus, the anomalies of interest could not be extracted. In addition, no anomaly such as an unusual vehicle was taken into consideration. For these reasons the dataset needed to be re-annotated for the anomalies of interest in this work. The anomaly detection method developed required testing not only on one dataset but also on datasets with different camera angles. As a result, Greenroads Ltd. provided data for the annotation of two additional video streams. Annotators for these two datasets, as well as for annotating the Street Scene Dataset [2], were employees of the same company. To maintain privacy and confidentiality, the data is masked to obscure most of the background. Furthermore, recordings are made at a sufficient distance from pedestrians and vehicles to ensure that no identifiable information, such as number plates or faces, is visible to the annotators. The author of this work supported the annotation process by providing detailed instructions and templates for annotation, along with continuous assistance to annotators whenever questions arose.

4.1.3 Data Annotation

Annotations for the classical anomaly detection method must be labeled with respect to the video to enable evaluation of the method based on anomalies within the video rather than just on the extracted structured data. This approach is crucial because the detector-tracker may occasionally produce false detections or lose tracks, resulting in the creation of anomalous-looking trajectories that do not correspond to actual video anomalies. Ideally, an anomaly detector should be capable of distinguishing between real anomalies in the video and false anomalies caused by the detector. Such evaluation is only possible if the data is labeled with reference to the video stream.

Moreover, anomalies need to be labeled to facilitate the measurement of the anomaly detection performance per label. Event-level annotation indicates only whether an anomaly has occurred and the time of occurrence, without specifying the location of the anomaly within the frame. While this annotation method accelerates the annotation process, enabling evaluation of the anomaly detector across a broader range of data, it presents a limitation: if two anomalies occur simultaneously, it becomes impossible to discern which one the anomaly detector is identifying.

The annotation process involved annotators watching a series of videos. Upon identifying one of the anomalies listed in Table 3.1 in a video, annotators recorded the video name, start and end times of the anomaly, and the type of traffic entity performing the anomaly in a spreadsheet. The start and end times of the anomaly are defined as the times when an entity executing an anomalous maneuver enters the frame. These times were recorded with an accuracy of one second.

It is important to note that not all anomalies listed in Table 3.1 are relevant to each location. For instance, bicycles are only considered anomalous for datasets collected in Malta, as they are rare occurrences in Maltese datasets. Similarly, heavy goods vehicles are only considered anomalous in the Street Scene datasets, as they are not common vehicles in that dataset. Additionally, the *vehicle crossing the centerline anomaly* is not relevant for the Zejtun Dataset, as no centerline is drawn on the Zejtun roadway. For a more comprehensive understanding of the data annotation procedure, detailed instructions provided to annotators can be found in Appendix A.

Datasets Specifications

The datasets annotated all had distinct and different characteristics such that the anomaly detection method can be tested in a widest possible variety of realistic

and practical settings. Table 4.2 shows the technical specifications for the three collected datasets. Table 4.3 lists the number of anomalies per label annotated per dataset.

The Street Scene dataset consists of 81 video sequences collected at various times of the day with an overhead USB camera overlooking the road, attached to the side of a high building. A frame from this dataset can be seen in Figure 4.1. The overhead angle prevents vehicles from obstructing each other, however the detector-tracker performs worse in overhead settings. This is due to the detector-tracker not being trained on any overhead data. This dataset is also challenging for the detector-tracker due to a large tree which obstructs an entire bicycle lane and prevents the entry and exit lines from being placed optimally. In addition the

Table 4.2 Specification of Each Structured Data Dataset

	Street Scene	**Mgarr**	**Zejtun**
Number of Videos	81	124	122
Total time annotated (mins)	226	620	610
Total trajectories annotated	3447	3426	4463
Resolution	1280×720	1920×1080	1920×1080
Frames per second	15	25	25

Table 4.3 Anomalies in ground truth per Structured Data datasets

Anomaly	**Street Scene Dataset**	**Mgarr Dataset**	**Zejtun Dataset**
Crossing Center Line	47	74	Not relevant
U-turn	5	31	8
Overtaking	0	16	8
Pedestrian on the road or Jaywalking	87	86	23
Bicycle on wrong lane	33	Not relevant	Not relevant
Vehicle on wrong lane	11	23	6
Traffic	54	2	0
Obstructions	7	33	19
Parking	36	23	1
Heavy Goods Vehicles	59	Not relevant	Not relevant
Bicycles	Not relevant	6	6
Scooters	0	13	2
Unforeseen Anomalies	0	5	0
Total	**339**	**312**	**73**

dataset has an obstructed entry point by the roof which also results in sub optimal entry/exit line placements. The sub optimal lines near the roof ledge especially the entry/exit line which overlaps the bicycle lane and the exit/entry line which is parallel to the car lane can be seen in Figure 4.1. Also, no information about the distance between exit and entry lines is available for this dataset, hence the distance between the entry/exit lines for each case was set to be ones such that a normalized sped measurement is obtained. The dataset offers a wide variety of anomalous events. Most interestingly it allows the analysis of anomalies performed by bicycles and an opportunity to investigate the accuracy that could be obtained in a scene with heavy visual obstructions.

The Mgarr Dataset consists of 124, 5 minute video sequences collected at various times of the day. The Mgarr location is a residential area which is frequently used by all types of traffic entities. The camera used to record this data is angled in such a way that it captures a busy bus stop and a junction clearly. The angulation however results in a heavy perspective which makes distances and vehicles appear significantly smaller as they travel further away from the camera. In addition the small size of the vehicles results in more ID switch errors and missed detections. The entry and exit point of all vehicles in this location are clearly visible which helps improve the performance of the detector-tracker. However, the entry and exit points are also parking spots. The parked vehicles are sources of many ID switches. This camera allows the investigation of a residential area with a frequently used junction. The area in question has a wide variety of behaviours which are considered normal as well as many other behaviours which are not. The greatest challenge in this camera is due to it's angulation, which causes many visual obstruction with vehicles hiding behind each other and heavy effects of linear perspective. The five unforeseen anomalies detected by the annotator were an instance of a driver driving in reverse, two instances of a horse pulling a cart and two instances of a garbage truck.

Finally, the Zejtun dataset consists of 122, 5 minute videos collected from a camera with a particular uncommon angle. The camera angle results in a heavy linear perspective with part of its view of the road being obstructed by a balcony. This camera angle is particularly challenging but it is also a realistic portrayal of what data can be collected in particular locations which lack high infrastructure for the installation of recording equipment. Some of the data was collected on a rainy day which resulted in blurred videos. A small number of anomalies were collected from this location when compared to the larger number collected in other locations. This was probably due to the fact that the road only had two lanes as opposed to the junctions observed in the other three datasets. This data set allows for the investigation of the anomaly detector for a camera installed an

extremely challenging angle.

Inter-Annotator Agreements

The data was labeled by two Greenroads Ltd. employees following the instructions detailed in Appendix A. The annotations, which involved common traffic events, did not require specialized expertise to recognize. Two annotators, A and B, performed the labeling. Annotator A had over a year of experience in annotating traffic events, while Annotator B was a student in a related field. Annotator A worked on the Street Scene and Zejtun datasets, and Annotator B worked on the Mgarr dataset. Additionally, it is important to note that the videos selected for the Mgarr dataset had been previously flagged by another annotator as potentially containing interesting activities.

To assess the reliability of the data and the clarity of the guidelines shared with the annotators, each annotator was asked to label a sample of videos from the datasets they were not the primary annotating. The inter-annotator agreement was then calculated for each sample. The raw agreement percentage, which counts the number of items with identical labels, does not account for accidental agreements, especially significant for sparse labels such as anomalous events [92]. Therefore, the Cohen's kappa metric, known for its robustness to chance agreements, was used to calculate the inter-annotator agreement.

For the Cohen's kappa calculation, each video annotation was represented as an array with one element per second of video. Each element was marked as one if an anomaly was found in that second, otherwise, it was marked as zero. These arrays were concatenated to form one array per dataset per annotator. The sample included 10 videos from the Mgarr dataset, 12 videos from the Street Scene dataset, and 49 videos from the Zejtun dataset. A larger sample from the Zejtun dataset was used due to the majority of its videos containing only normal events.

The scores, as shown in Table 4.4, indicate that the Zejtun and Mgarr datasets achieved almost perfect agreement, based on the commonly cited scale in [93]. In contrast, the Street Scene dataset annotations only reached a substantial agreement level, which is still satisfactory for the anomaly detection evaluation. The higher accuracy in the Zejtun dataset is likely due to its simpler, less busy street behavior, as it records a two-way street with fewer possible anomalies. In contrast, the other two datasets capture a complex junction, making it easier for annotators to miss certain events.

Table 4.4 Cohen Kappa Scores per Structured data dataset

	Street Scene	Mgarr	Zejtun
Cohen Kappa Score	72.27%	80.61%	90.12%

4.1.4 Structured Data Acquisition Evaluation

As previously mentioned, the detector-tracker utilized for acquiring structured data was not originally designed to detect anomalous events. The methods employed to clean up tracks and reduce the data size inadvertently removed some indicators of certain anomalous events. Additionally, there are instances where the detector-tracker fails to detect the objects performing these activities. Therefore, it becomes necessary to determine which anomalies in the ground truth, identified from a video, are also present in the structured data. This ensures that the anomaly detector can be evaluated based on anomalies that are indeed present in the structured data. By focusing on these anomalies, the evaluation process becomes more reliable and reflective of real-world scenarios.

To facilitate this process, additional labeling of the ground truth data was necessary. For every anomalous event identified from the video, the structured trajectory data for that specific time period needed to be analyzed and marked as either *detected* by the tracker-detector or *not detected*. For anomalies labeled as *not detected*, accompanying reasons were provided to explain why the tracker-detector either discarded or failed to perceive this data.

To achieve this, an existing tool developed by Greenroads Ltd. was adapted to allow for the visualization of detections made by the detector-tracker on the video. This adaptation provided a user-friendly interface for annotators to review and assess the trajectory data, aiding in the accurate labeling of detected and undetected anomalies.

The traffic entities detected are marked with a bounding box, a label which identifies the object and a number corresponding to the temporal order in which the object was detected. The bounding boxes change colour when the detector-tracker considers the object to have successfully entered through a line and changes colour again when the object successfully exited the line. Pedestrians are the only traffic entity which does not need to successfully enter or exit a line to be recorded by the detector-tracker. If an object experiences an ID switch, the number displayed next to the object bounding box will change as it gets re-detected. It is important to note that this number is solely generated by the visual annotation tool for ease of visual tracking and does not correspond to the data used during anomaly detection.

With this knowledge a set of rules were identified to consider an object to be detected by observing a video:

1. A vehicle entity needed to have a bounding box surrounding it with the same number identifier from entrance to exit.
2. The label of a traffic entity needs to be accurate except in cases where the traffic entity in question is a vehicle such as a scooter which has no proper label by the detector.
3. A vehicle needed to successfully enter and exit the lines to be considered detected.
4. A pedestrian only needed to be detected for at least part of its trajectory.

The annotation process involved labeling each anomaly in the ground truth as either detected or not detected. It is important to note that while the detector-tracker may have also had detection errors for non-anomalous traffic entities in the video, labeling all tracks in a video is impractical due to the time-consuming nature of the task. Additionally, such labeling requires a deep understanding of how the underlying algorithm works, which can be challenging to convey to non-experts. Therefore, the focus was on annotating anomalies specifically, rather than all tracks in the video.

4.2 Video Data for Deep Learning

The semi-supervised nature of deep learning video anomaly detection models necessitates a substantial amount of normal data for training, along with a set of normal and anomalous data for testing. Normal data refers to data from which any defined anomalous events have been removed. As discussed in Section 2.3, a wide range of publicly available datasets is available.

One of the primary datasets of interest was the MAVAD dataset [83], which includes data from three locations in Malta. The Mgarr and Zejtun Field scenes were captured using the same cameras used for evaluating the structured data in Mgarr and Zejtun for the structured anomaly detection method. For the video data anomaly detection section, the third location of the MAVAD dataset [83], the Zejtun Scrapyard scene, was omitted from evaluation. This decision was made because a direct comparison could not be drawn with the structured data, and it did not offer any additional benefits due to its similarity to the Zejtun Field scene. Another dataset used was the Street Scene Dataset [2], featuring the same labels as those used for evaluating the classical anomaly detection. Both datasets encompassed a wide variety of anomalies of interest, as shown in Table 4.5 for the MAVAD dataset and Table 4.3 for the Street Scene dataset.

To achieve the highest possible performance, both datasets had to undergo augmentation. Section 4.2.1 describes the data augmentation procedures required for the MAVAD dataset [83] and the Street Scene dataset [2].

4.2.1 Data Augmentation

In the MAVAD dataset [83], each video was initially masked by the authors to conceal private property. As the mask contains no relevant information, each frame was cropped to remove most of the mask, as depicted in Figures 4.2. On the other hand, the Street Scene dataset [2] did not originally have a mask and no large area of frame contained irrelevant information, so no cropping was deemed necessary.

The MAVAD dataset [83] was originally sampled at 25 frames per second (fps), while the Street Scene dataset [2] was sampled at 15 fps. However, these high sampling rates were deemed unnecessary. At such a high sampling rate, consecutive frames are nearly identical, and subsampling does not remove any significant information. Moreover, it reduces the data size, facilitating quicker training and testing processes. Therefore, both datasets were subsampled to 3 fps.

An intriguing aspect of the MAVAD dataset [83] that posed a challenge during training was the significant variation in lighting conditions throughout the dataset. Scenes were captured at different times of the day, often on extremely sunny days, which are typical in Malta. This resulted in strong shadows in the videos. This effect was the most severe for the Zejtun Dataset.

To address this issue, each video in the test set was labeled based on its lighting conditions, categorized as either *Long shadow*, *shadows mostly overhead*, *nighttime*, or *dusk*. The distinction between *dusk* and *nighttime* was necessary to account for the peculiar orange lighting of the camera in low-light conditions. Figure 4.2 provides an example of the Zejtun field scene under each of these different lighting conditions. On the other hand, the Street Scene dataset [2] also experienced changes in lighting, although they were much less severe compared to the MAVAD dataset.

From Table 4.6, it is evident that shadowed videos formed a minority of the data. Consequently, during training, the model would tend to bias towards learning the *sun overhead* condition as normal. To address this imbalance, videos without a *sun overhead* label were tripled during training. This approach helped ensure that the training dataset was more evenly distributed among the various lighting conditions, enabling the model to learn effectively across all scenarios.

During training, an inconsistency with the Mgarr anomaly labeling was identified. A defined anomaly was *pedestrians*, implying that all video snippets

Table 4.5 Labels per scene of interest for the MAVAD dataset [83]

Label	Mgarr	Zejtun Field
normal	161	125
pedestrians	65	19
pedestrians crossing	21	33
bicycle	10	17
bus	20	1
exit side street	44	0
heavy goods vehicle	21	3
obstruction	3	8
u-turn	1	6
scooter	0	2
Total	346	**214**

Table 4.6 Illumination labels per scene in the MAVAD Dataset [83]

Label	Mgarr	Zejtun Field
Sun Overhead	124	66
Night	16	15
Long Shadow	21	36
Dusk	0	8
Total	161	125

(a) Dusk (b) Night (c) Long Shadow (d) Sun Overhead

Figure 4.2 Cropped frame from the Zejtun Field Scene from the MAVAD Dataset [83] as an example of the illumination labels

featuring any pedestrian on the pavement should have been labeled as an anomalous event. However, several pedestrians were observed in the normal videos, as depicted in Figure 4.3. Consequently, all videos labeled as *pedestrians* were relabeled as normal data. However, videos labeled as *pedestrian crossings* were still considered anomalous, as no instances of pedestrian crossings were observed in normal videos. This adjustment ensured that the anomaly labeling remained consistent and accurate for the Mgarr dataset and anomalous features related to pedestrians were still tested. For the Zejtun Dataset, all pedestrian anomalies where relabeled as pedestrian crossing, since there is no side walk in that dataset, and the boundary between a pedestrian crossing and not is ambigous.

The normal data was approximately split into 80/10/10 train/test/valida-

(a) video 'normal_147', frame 10

(b) video 'normal_155', frame 30

Figure 4.3 Examples of Pedestrians in the normal data of MAVAD Dataset [83]

tion sets. Emphasis is on "approximately" because care was taken to ensure that a video could only belong to one set, preventing an exact 80/10/10 split. This approach was chosen to avoid similar frames depicting the same events from being spread across different sets, ensuring fair testing. Additionally, the data was stratified according to illumination type for the Zejtun and Mgarr datasets, ensuring that each split contained the same proportion of videos with different lighting conditions. Stratified sampling by illumination type for the Mgarr camera was more challenging due to the lower number of samples in different lighting conditions. All anomalous events in the dataset were included as part of the test data.

Furthermore, all frames were converted to grayscale for training. Visual augmentation techniques, such as changing brightness and blurring the image, were applied to enhance the dataset's robustness to illumination changes and effectively double its size. This augmentation strategy aimed to improve the model's ability to generalize across different lighting conditions and scenarios. An analysis of how augmentation affects the training procedure was then performed, and will be highlighted in the following chapter.

5 Methodology

The methodology chapter is divided into two sections. Section 5.1 outlines the classical anomaly detection method developed to identify anomalies from structured data, categorized by the type of anomaly detected. Section 5.2 describes the deep learning-based method, which detects anomalies directly from the traffic videos.

5.1 Classical Anomaly Detection

The classical anomaly detection method was designed to complement the existing traffic entity detector-tracker provided by Greenroads Ltd. This detector-tracker was trained to identify and track traffic entities within a camera view. Section 4.1.1 comprehensively outlines the usage and limitations of this detector-tracker in relation to anomaly detection.

Furthermore, the anomaly detection method was tailored for application in the context of Maltese roads. These narrow suburban roads with high traffic volumes exhibit a wide spectrum of acceptable behaviors and a diverse array of potential anomalies. This differs from highways or less busy roads, where acceptable behaviors are more limited, making deviations more apparent. Additionally, all collected datasets featured parked vehicles within the field of view, often leading to numerous ID switches in the data.

The literature review revealed that various methods focused on distinct categories of anomalies. For instance, Wang et al. [8] concentrated solely on speed anomalies, while Jiang et al. [34] emphasized trajectories with unusual shapes. Moreover, most methods did not account for detector-tracker errors, with exceptions such as Gao et al. [17].

To address this issue, an ID switch detection method was implemented, as detailed in Section 5.1.1, aimed at reducing the number of trajectories flagged as anomalous due to detector-tracker errors. Subsequently, a per-category anomaly detection method is described in Section 5.1.2. Figure 5.1 illustrates the block diagram of the data flow for the classical anomaly detection method developed.

5.1.1 ID Switch Detection

Upon reviewing the data acquired from the detector-tracker, it became evident that alongside the anomaly detector, a method to detect detector-tracker errors was necessary due to the significant number of ID switches detected. This un-

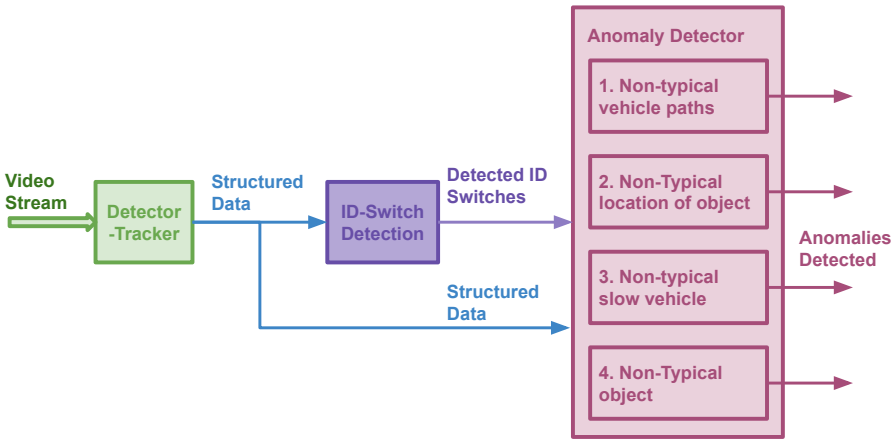


Figure 5.1 Block diagram of the data flow for the classical anomaly detection method

derscores the importance of distinguishing genuine anomalous events from those caused by detector-tracker errors. In their study, Gao et al. [17] classified the anomalous events of interest as detector-tracker errors. They identified two main types of detector-tracker errors: short tracklets, which are short tracks resulting from rapid movements of objects, causing the tracker to lose track of the object; and ID switches, which occur when the detector-tracker misidentifies one object as another, leading to part of the track belonging to one object and part to another. Effectively addressing these detector-tracker errors is essential for ensuring the accuracy of anomaly detection methods, as it helps distinguish genuine anomalies from false alarms caused by tracker inaccuracies.

Short tracklets were already filtered out through post-processing during data acquisition, where a track is only considered valid if it enters and exits through predefined lines. However, the ID switches were not filtered out by the detector-tracker post-processing. Consequently, a method to identify the ID switches to accompany anomaly detection was necessary. Unlike the approach taken in Gao et al. [17], where ID switches were filtered out as anomalies, here they would not be considered anomalies but marked for the user and removed for the calculation of some of the thresholds. The anomalies of interest are not the ID switches, but the other anomalies listed in Table 3.1. Thus, this accompanying ID switch detection serves as a tool for the user to filter out potential anomalous detections that do not correspond to actual anomalous events.

One clear effect of ID switches is the commonly observed phenomenon of very high speeds outputted by the detector. These anomalous and often impossibly high speeds are most caused by the detector or tracker mistaking one vehicle for another. Consequently, the detected vehicle appears to have jumped from one location to another in a short period, resulting in a high-speed detection. For in-

stance, in the Zejtun Dataset, the maximum speed of a vehicle recorded was 2250 km/h, with eight other vehicles detected with speeds upwards of 200 km/hr. To prevent these impossible speeds from affecting the rest of the anomalous vehicle detection, any speed above the 99th percentile is automatically marked as an ID switch error. The 99th percentile corresponds to approximately 80 km/hr for the Zejtun dataset, which aligns with the national speed limit and is unlikely to be surpassed in residential areas.

Utilizing a percentile threshold is preferred over setting a hard speed value threshold, as the speed outputs from the detector tracker are estimates and not calibrated to account for the effects of perspective. Additionally, percentiles are effective for datasets such as the Street Scene dataset, which have normalized speed calculations due to the absence of distance information. This approach ensures robustness in identifying ID switch errors across different datasets.

More commonly, ID switches do not exhibit suspiciously high speeds but rather show a sharp change in the direction of the trajectory. This effect is often noticeable in locations where there are parked vehicles, as the detector-tracker is more likely to mistake a moving car for one next to it. Figure 5.2 provides an example of this phenomenon, with the orange trajectory representing a regular trajectory point without ID switches, and the blue trajectory indicating an ID switch. The ID switch can be clearly identified, as the vehicle is marked to be passing through a building. Unlike the normal orange trajectory, which has multiple data points generated close to each other, the blue trajectory with the ID switch exhibits an unrealistically large distance between two points. This sharp deviation in trajectory direction serves as a key indicator of ID switches and aids in their identification during anomaly detection.

Whilst, trajectory points in a normal trajectory are typically spaced close

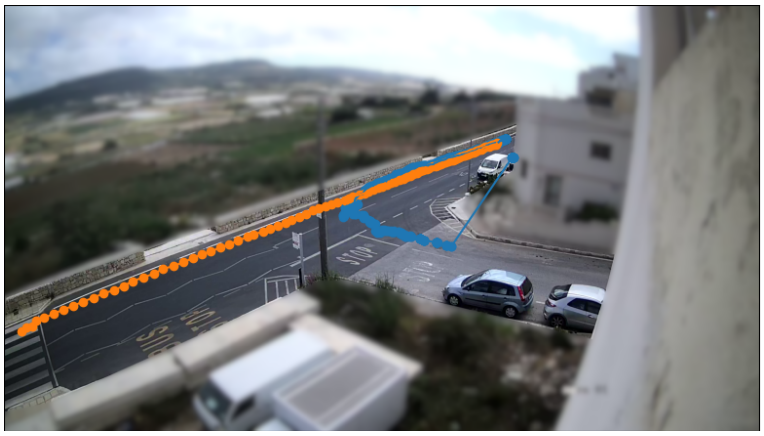
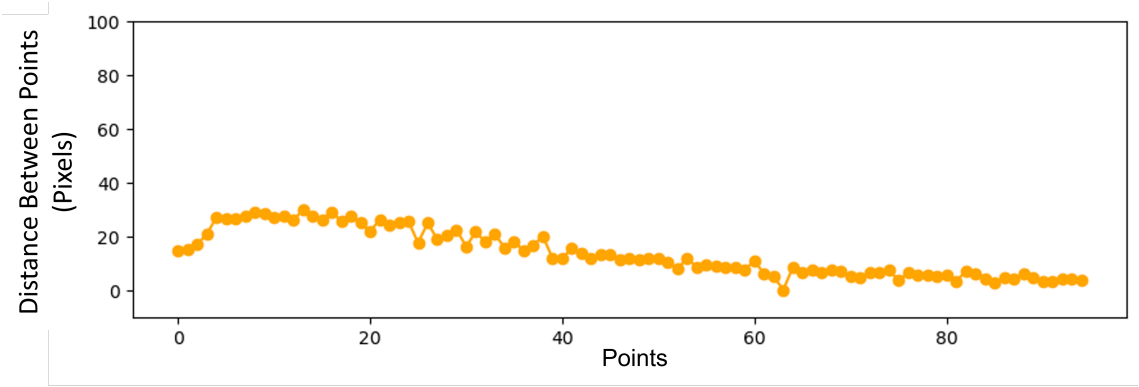
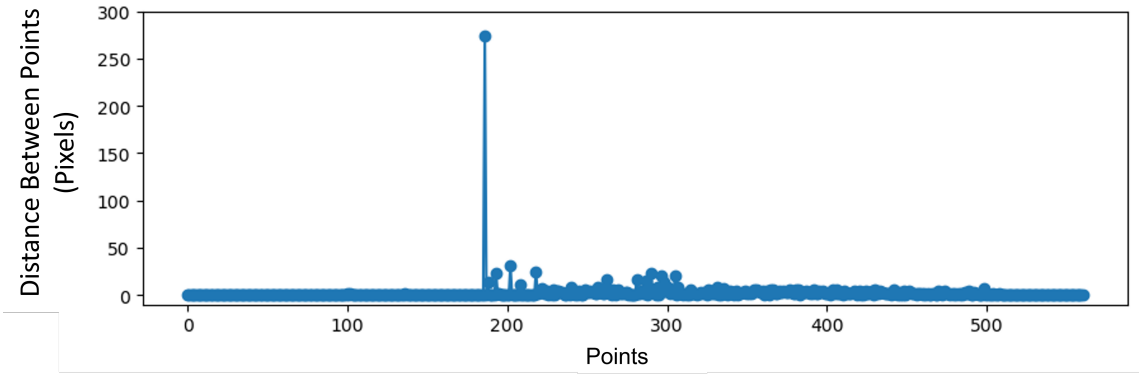


Figure 5.2 Trajectories from the Mgarr Dataset plotted on a frame from the Mgarr dataset. The trajectory points are produced whenever an object moves more than the minimum distance.



(a) Normal Trajectory



(b) Trajectory with ID switch

Figure 5.3 Showing plots of distances between trajectory points for the two trajectories shown in Figure‘5.2. Trajectory in Figure 5.3a is the trajectory with no ID switch and the trajectory Figure 5.3b is the trajectory with an ID switch. The distance between points for the blue trajectories are smaller than those for the orange track. These differences are expected since the vehicles are travelling at different speeds.

to each other, trajectory points during an ID switch error show large change in distance between one point and another. Figure 5.3 is a plot of the Euclidean distance between each point for the trajectories shown in Figure 5.2 . Figure 5.3a shows the plot for the normal trajectory and Figure 5.3b shows the trajectory with the ID switch.

The sharp peak in Figure 5.3b serves as a clear indicator of the large distance between points exhibited by the ID switches, enabling the establishment of a cutoff threshold to identify them. However, relying solely on the distance between points is prone to failure due to the effects of linear perspective. In scenarios where the camera angle is not overhead (unlike the Street Scene Datasets, which feature an overhead angle) but rather at an angle with the road (as in the Zejtun and Mgarr datasets), the distance between points is heavily influenced by perspective. This means that points farther away from the camera appear closer to each other than points closer to the camera, leading to distortions in the perceived distances.

Figure 5.4 A trajectory plotted on a frame from the Zejtun dataset. Due to an ID switch a vehicle exiting the scene is mistaken for a vehicle entering the scene. This creates a trajectory with a sharp U-shape. The steep angle of the trajectory indicates that this is an ID switch and not a turn made by the vehicle.

Figure 5.3a illustrates the effect of perspective on trajectory data, where the initial part of the trajectory shows a larger Euclidean distance between points. This discrepancy arises because the vehicle appears to cover a smaller distance as it moves further from the camera. Figure 5.4 further demonstrates this effect with a trajectory showing a vehicle making a U-turn at the upper right corner of the frame. This trajectory is actually the combination of two vehicle trajectories joined at this corner due to an ID switch. The sharp angle of the U-turn indicates an ID switch rather than an actual U-turn. In Figure 5.5, the Euclidean distances between each point on the joint trajectory are plotted. A peak in the center of the plot indicates an ID switch. However, this peak is not significantly higher than the point at the end of the trajectory. The plot forms a U-shape, lower in the center where the vehicle is furthest from the camera. Thus, the ID switch, although representing a large distance, is comparable to the normal distances between points when the vehicle is close to the camera.

The impact of perspective on the distance between points in each trajectory can be likened to low-frequency signal noise that needs to be filtered out, given that the indicator of ID switches represents a high-frequency peak. To mitigate the influence of this low-frequency effect caused by perspective, a moving average filter with a window size of 10 is employed for each trajectory to isolate the low-frequency signal. The equation for the moving average filter is outlined in Equation 5.1.

$$y[n] = \frac{1}{N} \sum_{i=0}^{N-1} x[n-i] \tag{5.1}$$

Where $x[n]$ is the input signal which is the distance between each trajec-

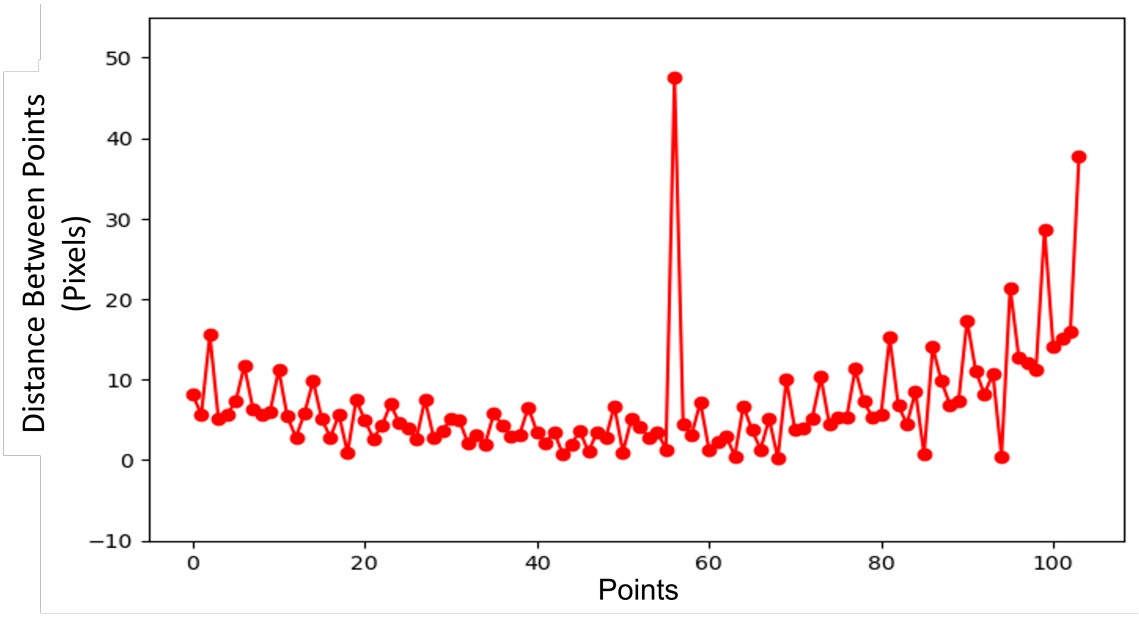


Figure 5.5 Plots of Euclidean distances between trajectory points the joint trajectory shown in Figure 5.4

tory point, N is the window size and $y[n]$ is the output low frequency signal.

After subtracting the low-frequency signal from the distance between trajectory signals, as given in Equation 5.2,

$$z[n] = x[n] - y[n] \tag{5.2}$$

only high-frequency artifacts remain, where $z[n]$ represents the signal filtered to eliminate low frequency components. Figure 5.6 illustrates the extracted low-frequency signal, while Figure 5.7 displays the plot of the distance between trajectory points after removing the low frequency component.

The high-pass filtered signal now exhibits a clearly defined peak, which can be detected using a simple threshold. However, the filtering process primarily aids in determining the threshold for peak detection. Without filtering, the low-frequency noise would have elevated the threshold unnecessarily, potentially leading to missed ID switch detections.

A trajectory is deemed to have an ID switch if its filtered signal $z[n]$ surpasses a threshold. This threshold is calculated as the median of the filtered signals, plus four times the standard deviation of all the filtered signals for that scene. In Figure 5.8, this threshold is represented by the red line.

The threshold selection is based on the robustness of the median to outliers, ensuring that datasets with a large number of ID switches do not unduly influence the threshold value. Additionally, four times the standard deviation is chosen, since it represents a threshold for the most extreme peaks. To mitigate

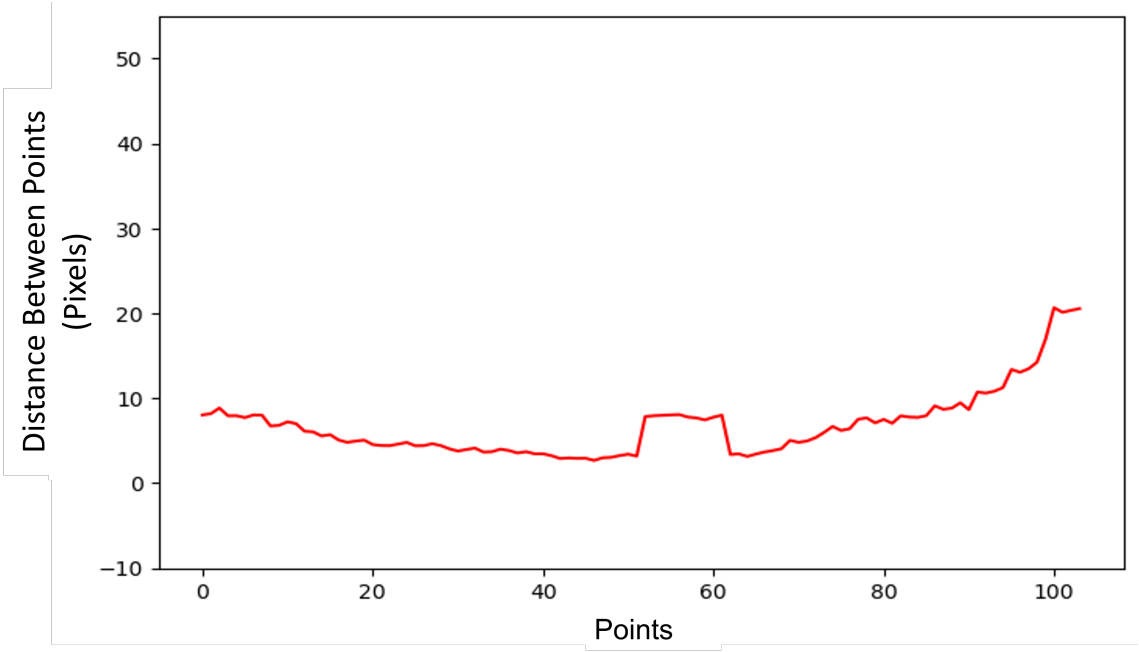


Figure 5.6 Low frequency extracted from distance between trajectory points signal on Figure 5.5

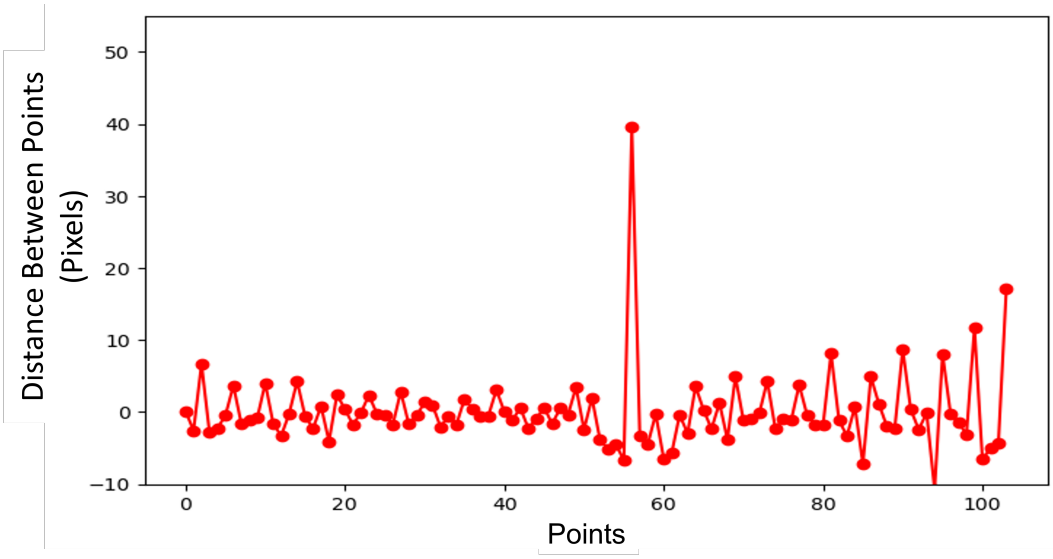


Figure 5.7 Distance between trajectory point signal with low frequencies removed.

the impact of irregular trajectory points caused by vehicles partially within the frame, the first and last ten percent of each signal are excluded from the threshold calculation.

5.1.2 Per Category Anomaly Detection

The anomalies listed in Table 3.1 exhibit diverse features across different categories. Consequently, a distinct detection method was developed for each cate-

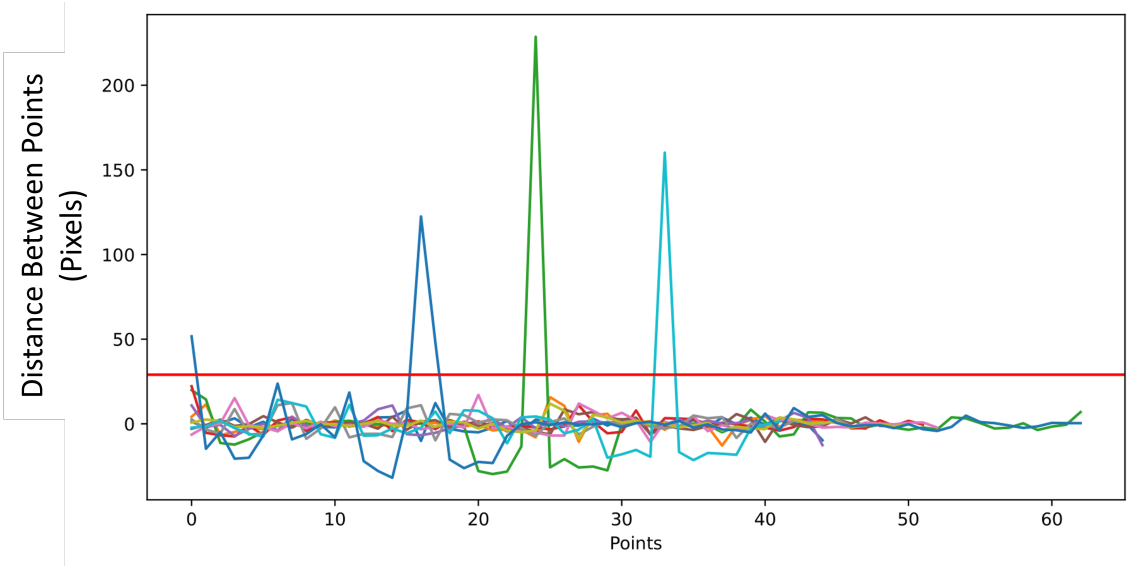


Figure 5.8 Distance between trajectory point signal with low frequencies subtracted for ten trajectories from the Mgarr dataset as shown with the coloured lines. The red horizontal line marks the ID switch threshold.

gory, detailed in the following subsections. These methods aim to identify nearly all the anomalies outlined in Table 3.1, with some anomalies being detectable across multiple categories. However, it is worth noting that these methods do not explicitly target *Unforeseen anomalies*. These anomalies pose a challenge for feature-based detection methods due to their unknown characteristics

Before settling on the per-category method, an exploration of HMM-based approaches was conducted. Drawing from the work of Morris et al. [35] and Cai et al. [38], trajectories were initially clustered based on their directional movement and then filtered before fitting an HMM. However, despite successfully grouping trajectories with similar directions, the noise inherent in the trajectory dataset hindered the development of representative HMMs from the clusters.

Additionally, experiments inspired by LDA, as suggested by Jeong et al. [31], were conducted. In this approach, features such as the total distance traveled, speed, vehicle type, angle of travel, and stopping area were quantized and transformed into a bag of words. Although this method effectively clustered trajectories based on their general movement direction, it struggled to detect the complex anomalies of interest.

In general, both HMM-based and LDA-inspired methods may be more suitable for identifying trajectories vastly different from the norm, such as those with entirely different directional headings. They might also be applicable in scenarios where prior knowledge about expected anomalies is limited, as the features used are more general. However, due to the specific nature and variety of anomalies, as well as the noise in the data, the per-category method was preferred. More-

over, the per-category method benefits from the availability of a priori knowledge about each traffic scene, as user input is required to set up the detector-tracker.

Category 1: Non-typical vehicle path

The non-typical vehicle path category method aims to detect anomalies characterized by abnormal entry and exit points. Leveraging the output of the tracker-detector, which lists the entry and exit lines for each detected vehicle, this method records all entry and exit pairs instead of just the most common ones, as typically done. By specifying which entry and exit pairs should be considered anomalous, unusual paths can be identified.

This approach resembles clustering based approaches on the location of entry and exit points, as demonstrated by Cai et al. [38] and Jiang et al. [34], but with the advantage of having all possible entry and exit points predefined through entry and exit lines. The method of specifically asking the user to define anomalous paths was chosen because prior knowledge of typical routes taken on streets is easy to obtain. Simple assumptions, such as the least-used entry-exit pairs being anomalous, do not hold in this setting. Detector-tracker errors often generate fake trajectories, making such anomalous paths appear more frequently used than they actually are. Therefore, defining anomalous paths based on prior knowledge enhances the method's robustness. Figure 5.9 illustrates trajectories from the Street Scene dataset, with the lower-right exit and lower-right entry pair specified as anomalous.

Figure 5.9 Example of Anomalous paths detected using the Category 1 Method. The orange, blue and brown trajectories are likely caused by vehicle parking or making strange maneuvers. The green, purple and red lines seem to be caused by ID switches due to the obstruction caused by the roof. The trajectories and frames are from the Street Scene Dataset.

Category 2: Non-typical location of object

To detect traffic entities in the wrong area an anomaly detection method was designed based on the assumption that the majority of traffic entities use the correct areas on the road to commute, and anything outside that area is anomalous. The mask creation process was loosely inspired by Zhao et al. [20] enhanced trajectory-based mask solution, where they use the vehicle trajectories to create a mask to separate the static vehicles (which are not likely to show any anomalous event) from the moving vehicles (which is where anomalous events are likely to occur). They cluster vehicles based on the direction of movement and the bounding box of a trajectory is used to expand a mask if the trajectory is considered to have normal movement. They suggest this solution as an alternative to image segmentation, which is difficult due to scene complexity. For the method implemented in this work, the trajectory-based mask solution was expanded such that a mask of the allowable paths per type of object in a field of view can be created.

Different traffic entities have different areas in the scene where they would belong to. Pedestrians would only be allowed on the pavement and bicycles are constrained to their designated lanes. Other vehicles would then also have the most likely path taken per entry and exit pair.

The initial step involves filtering out trajectories prone to ID switches, reducing detector errors. Despite this refinement, anomalous trajectories may persist in the data, rendering them unsuitable for mask estimation. Hence, a clustering technique is employed to isolate the predominant trajectory groups for further analysis.

The DBSCAN clustering technique [42] was employed for this task due to its robustness to noise, as it can identify outliers that do not fit into any cluster. This ensures that only trajectories common to a specific group of traffic entities are considered for mask creation. Additionally, the DBSCAN algorithm is capable of detecting clusters of varying sizes and shapes, which is essential for capturing all typical paths needed to construct the mask.

The end points, start points, midpoint and speeds of a traffic entity are the features used for clustering. The start and end points are common features used for clustering as used by Cai et al. [38] and Morris et al. [35] for their initial clustering steps. The midpoint was added as a feature since traffic entities behaving normally and travelling at a constant speed would all have a similar midpoint in their trajectory. The speed was also used as a feature since normal traffic entities travelling through a path would be expected to have similar speeds. If the traffic entity is a pedestrian the speed is first calculated per pedestrian by first measuring the distance between each trajectory point with respect to the pixels in the im-

age. Then the relative distance is divided by the difference in time between entry and exit frame. Overall the aim of the clustering was not to cluster all trajectories into some cluster but to extract trajectory clusters which surely belong to normal behaviour. Thus, the anomalous cluster is expected to have a large number of trajectories. After clustering all trajectories belonging to a cluster (except for the outlier cluster) are considered to be valid for creating a mask.

The DBSCAN clustering requires the manual selection of two parameters: the epsilon value, which is the maximum distance between two points for one to be considered in the neighborhood of the other, and the minimum samples, which is the minimum number of data points required to form a distinct cluster. Since the clustering is used for pedestrians, vehicles, and bicycles in locations with distinct characteristics, a common minimum value for epsilon and minimum samples could not be set. However, general guidelines were found which were able to facilitate deployment for the three very diverse datasets tested.

For the epsilon value, a starting point of 50 is recommended. This value should be increased gradually if the trajectories of interest exhibit erratic behaviors, such as the pedestrians in Zejtun and Mgarr, and decreased if the trajectories are regular, like the bicycle paths in the Street Scene dataset. For the minimum samples, the value should be set to $\frac{1}{30}$ of the total number of entities to be clustered if there are many samples of that type of entity, such as vehicles in a lane. If there are fewer samples, a good rule of thumb is to set the minimum samples to 5 trajectories.

For the mask creation process, an initially empty mask, the size of a frame, is generated with all values set to 0. Each point within the valid trajectories is then translated to a pixel value, and a 5×5 area around it is designated with a 1 on the mask. The 1 value indicates the valid area for that trajectory, while the 0s represent the invalid areas. Subsequently, a series of filtering and morphological operations are applied to refine the mask. These operations are aimed at smoothing out irregularities. Specific details of these operations are outlined in Table 5.1. Figure 5.10 shows the unsmoothed and smoothed masks created for the bicycle paths in the Street Scene dataset. The figures demonstrate that the operation slightly enlarges the allowable area and smooths the edges. Consequently, bicycles at the boundary of the allowable area do not trigger false positives due to noisy edges. The operations also suppress any point noise in non-allowable areas.

The mask creation method was employed to identify various types of anomalies, tailored to specific traffic entities. For pedestrians, the focus was on detecting instances of jaywalking, while for bicycles, it aimed to identify bicycles outside designated lanes. Similarly, for vehicles, the method targeted anomalies such as overtaking and being in the wrong lane, based on predefined entry and exit pairs.

Table 5.1 Mask smoothing morphological operations

Operation	Kernel size	Number of iterations
Gaussian blur	21,21	1
Morphological opening	7,7	1
Morphological dilation	7,7	2
Morphological close	7,7	2

(a) Unsmoothened bicycle mask (b) Smoothened bicycle mask

Figure 5.10 Mask created for bicycle path for Street Scene dataset

Using the mask created from typical trajectory behaviors offers several advantages over directly applying the DBSCAN algorithm to identify outlier trajectories in anomalous parts of the frame. Firstly, the mask-based approach is easily deployable in production settings. The mask can be efficiently stored and updated as needed, requiring only the comparison of current trajectory points to the mask to determine anomalies. In contrast, utilizing DBSCAN directly would involve calculating relevant features for each trajectory and comparing them to previously clustered data, adding complexity and computational overhead. Secondly, the mask provides a straightforward visual representation of typical trajectory patterns, making it easier for engineers to assess its effectiveness. In contrast, visualizing high-dimensional data and understanding potential issues with DBSCAN clustering can be challenging, making it harder to identify and address problems in the anomaly detection process.

Category 3: Non-typical slow vehicles

Anomalous slow vehicles, indicative of parking, obstructions, or traffic queueing, were identified based on their speed being below a specified threshold. For both the Mgarr Dataset and the Street Scene dataset, this threshold was set at the 5th percentile of speed values not associated with ID switch errors. In the Mgarr Dataset, this corresponded to 20 km/h, as observations showed that vehicles engaged in activities such as parking or causing obstructions typically moved at speeds below this threshold. For the Street Scene dataset, while absolute speed calculations were not possible due to the lack of distance information, the per-

centile was still applied based on relative speed calculations. The Category 3 method was not applied to the Zejtun Dataset, as no anomalies in this category were identified as of interest.

Category 4: Non-typical vehicle

Non-typical vehicles can also be easily detected using the provided data acquisition method, as the vehicles are labeled by the detector. However, a limitation is that the unusual vehicle must belong to one of the classes listed in Table 4.1. Additionally, rarer vehicles tend to have lower F1 scores. For example, bicycles, which are rare on Maltese roads, have the lowest F1 score among all vehicle classes, making them the least likely to be detected. If an unusual vehicle of interest does not belong to one of the detector's labels, the detector would require retraining with examples of this vehicle. This task is challenging because finding enough examples for training would be difficult. Some unusual vehicles are detected under other labels; for instance, scooters are often detected as bicycles or pedestrians due to their similarity at the recording distance.

For a detector-tracker, the detector confidence is a metric output for every detection that indicates the certainty level of the detector for each object detected. One might suspect that detector confidence could be used as an indicator for non-typical vehicles that the detector was not trained to recognize. However, during training, object detectors such as YOLOv4 [88] explicitly learn what is not an object. This means the model is trained to differentiate between labeled objects and background regions. As a result, vehicles outside the trained classes are not detected as other objects with uncertainty; they are simply not detected at all. The model's training process involves learning to ignore regions that do not resemble any trained object class, thereby reducing false positives and improving detection accuracy. Consequently, non-typical vehicles not included in the training classes will likely be missed entirely rather than detected with low confidence.

5.2 Deep Learning Based Detection Method

The classical data anomaly detection method presented in the previous section exhibits inherent biases. Firstly, it relies on the object-detector and tracker to extract data, consequently limiting the detectable features. Secondly, the features for anomaly detection were manually chosen, with no guarantee that they are the most representative of the anomalies of interest. To address these biases, this section proposes the use of a deep learning method that directly counters them.

By employing a spatio-temporal encoder, features can be extracted directly from a video, thus freeing the method from the constraints of hand-picked features.

From the deep learning methods reviewed, an AE based method was preferred due to its stability when training unlike the GAN based methods. As discussed in the literature review Section2.2, AEs can learn to capture spatial and temporal relationships within data, rendering them suitable for learning the complex normality in traffic videos.

When selecting the AE architecture for anomaly detection, two primary criteria were deemed essential. First, priority was given to temporal representation due to the temporal nature of traffic anomalies. Second, it was crucial that the architecture had been validated with real-world traffic data. Many models reviewed primarily relied on pedestrian datasets, which may not ensure the same level of accuracy to the specific anomalies of interest in this study. The STAE model proposed by Zhao et al. [1] meets both criteria, emphasizing a temporal representation while being developed and tested with traffic data. Consequently, their model architecture was adopted.

5.2.1 Model Architecture

The AE architecture proposed by Zhao et al. [1] follows a familiar concept utilized by many previous models for video anomaly detection [48, 51, 53] . The architecture comprises of two main parts: an encoder and a decoder. The encoder spatially compresses the input images through the utilization of pooling layers, thereby extracting important features. Subsequently, the decoder reconstructs the image to its original resolution using transpose convolutions.

The input to the encoder is a stack of T consecutive frames. The encoder consists of four 3D convolutional layers, each employing a $3 \times 3 \times 3$ convolution kernel with a stride of $1 \times 1 \times 1$ to extract spatio-temporal features. Batch normalization and Leaky ReLU activation are applied to each convolutional layer to expedite convergence. Additionally, 3D max-pooling layers with a kernel size of $2 \times 2 \times 2$ and a stride of $2 \times 2 \times 2$ are utilized after each convolutional layer to downsample the feature maps. The increase in the number of kernels in deeper layers aids in capturing complex semantic features.

The decoder mirrors the structure of the encoder, comprising three 3D transpose convolutional layers followed by one 3D convolutional layer. The transpose convolutional layers reverse the operation of convolution to reconstruct the input signal from the hidden layer. Batch normalization and the Leaky Relu activation are also used in all layers except the last. In the last layer a Sigmoid activation is applied to the last layer to normalize the output data.

The model includes two decoders: a reconstruction branch and a prediction branch, both identical but with different training objectives. The reconstruction branch is responsible for rebuilding the input frame stack from the hidden layer, while the prediction branch forecasts the future T frames following the input video clip. Both branches are connected to the same encoder and are trained concurrently. Figure 5.11a shows a diagram of the connected layers with the two branches. Table 5.2 gives a detailed description of all the layers in the architecture used. While Figure 5.11 describes the blocks mentioned in Figure 5.11a and Table 5.2.

The prediction branch serves as a method in this design to force the model to learn the temporal aspects of videos. It accomplishes this by capturing the trajectory of moving objects through predicting future frames, thereby anticipating the movement of objects in the video sequence and enhancing temporal feature extraction.

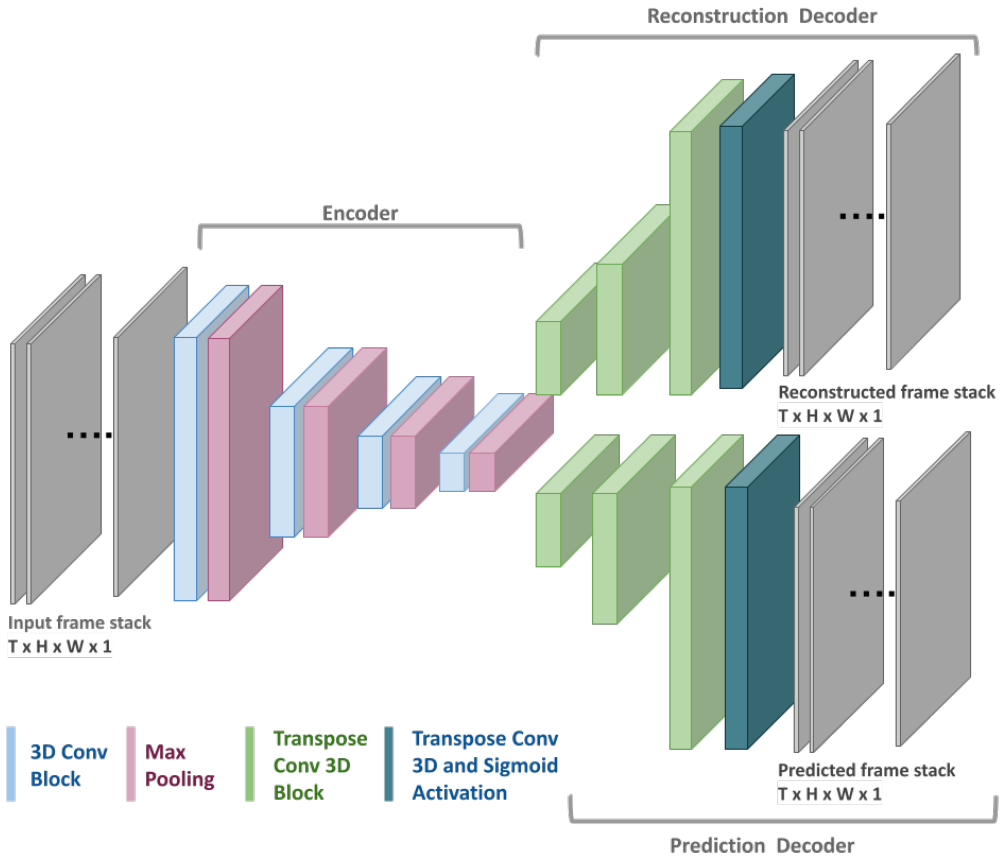
Another tool employed in the model to emphasize temporal aspects is 3D convolutions, inspired by the works of Tran et al. and Varol et al. [94, 95]. The authors Zhao et al. [1] argue that standard 2D convolutional neural networks excel in extracting spatial features from images but fall short in capturing temporal information. So in the cases such as as the fully convolution network used by Hasan et al. [48], although the input to the network is a stack of frames the temporal aspect of the input would collapse instantly, since the convolutions are only being performed spatially.

In 3D convolutions, a 3D kernel is applied to a 3D input volume to produce a 3D output volume. The kernel slides across the input volume in three dimensions (width, height, and depth), computing the dot product with the overlapped region at each position. This operation aggregates spatial and temporal information from the input, facilitating the extraction of spatio-temporal features.

The value of a cell in the output volume is calculated by convolving the 3D kernel with the corresponding region in the input volume. Each cell's value is determined by summing the products of kernel weights and input values within the receptive field, followed by applying a bias term and an activation function. The equation for a 3D convolution [1] at the value of a cell in the i_{th} channel position at (x, y, z) in the l_{th} layer denoted by a_l^{ixyz} and calculated as follows:

$$a_l^{ixyz} = f(\sum_{k=1}^{K_l}\sum_{h=1}^{H_l}\sum_{w=1}^{W_l}\sum_{d=1}^{D_l}\theta_{l,i}^{khwd}\alpha_{l-i}^{k(x+h)(y+w)(z+d)} + \beta_l^i) \tag{5.3}$$

where $\theta_{l,i}^{khwd}$ is the value of cell at the position (h, w, d) of k_{th} channel in the i_{th} kernel connected to the l_{th} layer. H_l, W_l, D_l and K_l are the height, width, depth and channel number of the kernel. β_l^i is the bias for the i_{th} in the l_{th} layer. $f(.)$ is



(a) Diagram of the Architecture of the model employed as proposed by by Zhao et al. [1]

(b) conv 3D block

(c) transpose conv 3D block

Figure 5.11 Diagram of the Architecture of the model employed as proposed by by Zhao et al. [1]. Details of the block shown are described in Figure 5.11b and Figure 5.11c

the activation function.

To effectively capture temporal information, the input is constructed as a hyper-cuboid by stacking frames along the temporal dimension. T frames are stacked and the frames are resized. All datasets were converted to grayscale, thus only one colour channel is required. This results in a hyper-cuboid input shape of $T \times W \times H \times 1$. Multiple tests were performed with different sized input images, which will be further described in Section 6.

Table 5.2 Architecture of the model used as proposed by by Zhao et al. [1]. Details about the conv 3d block and the transpose conv 3D block can be found in Figure 5.11c

type	kernel size	stride	output size
encoder			
input			$T \times H \times W \times 1$
conv 3D block	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$T \times H \times W \times 32$
3D maxpooling	$2 \times 2 \times 2$	$2 \times 2 \times 2$	$\frac{T}{2} \times \frac{H}{2} \times \frac{W}{2} \times 32$
conv 3D block	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$\frac{T}{2} \times \frac{H}{2} \times \frac{W}{2} \times 48$
3D maxpooling	$2 \times 2 \times 3$	$2 \times 2 \times 2$	$\frac{T}{4} \times \frac{H}{4} \times \frac{W}{4} \times 48$
conv 3D block	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$\frac{T}{4} \times \frac{H}{4} \times \frac{W}{4} \times 64$
3D maxpooling	$2 \times 2 \times 4$	$2 \times 2 \times 2$	$\frac{T}{8} \times \frac{H}{8} \times \frac{W}{8} \times 64$
conv 3D block	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$\frac{T}{8} \times \frac{H}{8} \times \frac{W}{8} \times 64$
3D maxpooling	$2 \times 2 \times 2$	$1 \times 1 \times 1$	$\frac{T}{8} \times \frac{H}{8} \times \frac{W}{8} \times 64$
decoder reconstruction branch			
transpose conv 3D block	$3 \times 3 \times 3$	$2 \times 2 \times 2$	$\frac{T}{4} \times \frac{H}{4} \times \frac{W}{4} \times 48$
transpose conv 3D block	$3 \times 3 \times 3$	$2 \times 2 \times 2$	$\frac{T}{2} \times \frac{H}{2} \times \frac{W}{2} \times 32$
transpose conv 3D block	$3 \times 3 \times 3$	$2 \times 2 \times 2$	$T \times H \times W \times 32$
transpose conv 3D	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$T \times H \times W \times 1$
sigmoid activation out 1			$T \times H \times W \times 1$
decoder prediction branch			
transpose conv 3D block	$3 \times 3 \times 3$	$2 \times 2 \times 2$	$\frac{T}{4} \times \frac{H}{4} \times \frac{W}{4} \times 48$
transpose conv 3D block	$3 \times 3 \times 3$	$2 \times 2 \times 2$	$\frac{T}{2} \times \frac{H}{2} \times \frac{W}{2} \times 32$
transpose conv 3D block	$3 \times 3 \times 3$	$2 \times 2 \times 2$	$T \times H \times W \times 32$
transpose conv 3D	$3 \times 3 \times 3$	$1 \times 1 \times 1$	$T \times H \times W \times 1$
sigmoid activation out 2			$T \times H \times W \times 1$

The model proposed by Zhao et al. was then augmented with techniques inspired by the work of Pravan et al. [61]. Pravan and colleagues noted that the model which performed best on the ADOC dataset tiled the input before processing it. They inferred that tiling contributed significantly to the model's superior performance, particularly in handling the dataset's varying illumination conditions. Consequently, tests were conducted in which the input frame was divided into 4 and then 16 tiles prior to input, in order to evaluate the effects of tiling on model performance.

5.2.2 Training

With the completion of the model architecture design, the subsequent step involves training. The reconstruction branch is tasked with learning to reconstruct the current cuboid of frames. Consequently, the reconstruction error is measured using the Euclidean loss:

$$L_{rec} = \frac{1}{N}\sum_{i=l}^{N} ||X_i - f_{rec}(X_i)||_2^2 \tag{5.4}$$

where X_i is the i_{th} hyper-cuboid of the input batch of size N, and $f_{rec}(X_i)$ is the output of the reconstruction branch. The aim is for the reconstruction branch to effectively reconstruct the input cuboid post-compresion, with the ground truth being identical to the input frames.

However, the objective differs for the prediction branch. Here, the goal is to learn to predict future frames, with the ground truth being frames shifted by T. Although the same Euclidean function used for calculating reconstruction loss could be applied, Zhao et al. [1] noted that this might not be the most suitable method. The prediction branch is intended to encourage the model to learn patterns of movement rather than predict sudden changes in frames, such as the appearance of new objects. Thus, the model should ideally focus on learning the most probable movements without being distracted by objects appearing on the screen.

As time progresses, the likelihood of new objects appearing gradually increases. Therefore, Zhao et al. [1] introduce a diminishing weight on each frame of the predicted video clip. The prediction loss is formulated as follows:

$$L_{pred} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{T^2}\sum_{t=1}^{T}(T-t)||X_{i+T}^t - f_{pred}(X_i)||_2^2 \tag{5.5}$$

where $f_{pred}(X_i)$ is the output from the prediction branch, X_{i+T} is the future T frames and the super script t in X^t denotes that it is the t frame from the video clip

X. In essence, the weighting of the prediction loss decreases as the t increases.

This formulation encourages the model to accurately predict the motion of existing objects while considering the temporal context of the video sequence. The decreasing weight assigned to each frame reflects the decreasing relevance of distant frames to predict future movements.

These equations are combined to form the optimisation objective:

$$\min_{W} L_{rec} + L_{pred} + \lambda||W||_2^2 \tag{5.6}$$

Here, W represents the parameters of the model, and λ is a regularization parameter controlling the model's complexity. The model was trained for a hundred epochs for each dataset using AdaGrad [96] with a learning rate of 0.001.

5.2.3 Anomaly Detection

After training the model, the reconstruction error of the test video sequence x can be given by:

$$e(x) = L_{rec}(X) \tag{5.7}$$

$$s(x) = 1 - \frac{e(x) - min_x e(x)}{max_x e(x)} \tag{5.8}$$

where $min_x e(x)$ and $max_x e(x)$ are the minimum and maximum reconstruction scores found for that dataset. In real-time anomaly detection systems, it is impractical to compute $min_x e(x)$ and $max_x e(x)$since future data is unobservable. Therefore, these values are typically set experimentally based on historical data.

Thus video sequences containing regular events are expected to yield higher regularity scores as they closely resemble the normal training data in the feature space. Conversely, anomalous sequences exhibit lower regularity scores.

Anomalies can also be visualized by computing the Euclidean loss per frame. The reconstruction error is calculated for every pixel in every frame, enabling the error to be segmented into individual frames and facilitating the visual localization of anomalous regions in the images.

6 Results and Evaluation

This chapter presents the results and evaluations for the anomaly detection methods. The first section, Section 6.1, outlines the evaluation metrics used for the various results presented. Next, two sections evaluate the two methods presented based on their performance as anomaly detectors. Section 6.2 presents the results from the classical anomaly detection method, and Section 6.3 presents the results from the deep learning method. For each method, an error analysis is also performed. Then, both methods' performance is evaluated and compared in Section 6.4. Finally, the computational expense for both methods is discussed in Section 6.5.

The results are presented for three datasets collected at distinct urban locations, aiming to evaluate the models under varied scenarios reflecting realistic urban road conditions. The first location, Mgarr, features heavily angled cameras causing perspective distortion, numerous parked cars, and a junction with multiple vehicle entry and exit points. Videos from this location are subject to masking and are recorded at different times of the day, occasionally exhibiting significant shadows.

Similarly, the Zejtun location presents challenges with numerous obstructions and angled cameras impacting perspective. Like Mgarr, videos from Zejtun undergo masking and are recorded at various times, potentially introducing various lighting conditions and strong shadows.

The Street Scene dataset offers a different perspective, captured from an overhead angle. It showcases a diverse range of vehicles and includes features such as bicycle lanes, numerous obstructions, side entrances and exits, as well as street parking observations.

6.1 Evaluation Metrics

In video anomaly detection, the most direct method of analysis is frame-level evaluation. If any region in a frame is detected as an anomaly and aligns with the ground truth, it is considered a correct detection, regardless of the region's location or size [97]. Frame-level analysis often includes qualitative evaluation using the True Positive Rate (TPR), a metric that measures the rate of correctly detected frames to all abnormal frames in the ground truth.

$$TPR = \frac{\#\ True\ anomaly\ frame\ detections}{\#\ Anomalous\ Frames} \tag{6.1}$$

The False Positive Rate (FPR) is the rate of incorrectly detected frames to

all normal frames in ground truth.

$$FPR = \frac{\#\ False\ anomalous\ frame\ detections}{\#\ Normal\ Frames} \tag{6.2}$$

The ROC curve is the plot of the TPR against the FPR, plotted for various classification thresholds and model parameters. From the ROC the Equal Error Rate (EER) can be obtained from the point on the ROC curve where $FPR = (1 - TPR)$ [97]. The AUC metric can also be obtained as the area under the ROC. A small EER and a larger AUC indicate better performance [97]. The AUC is the typical metric used to evaluate deep learning based methods as seen in Table 2.5.

The threshold that yields the best performance is subsequently employed to conduct event-level analysis for the deep learning-based model. Given the diverse range of anomalous events with varying durations, solely relying on frame-level analysis, while a commonly used metric, could introduce bias towards detecting longer anomalies. Therefore, for event-based analysis, an anomalous event is considered detected if it was flagged as anomalous for at least α of the duration it was annotated for.

Frame-level evaluation is unsuitable for the classical anomaly detection method because it relies on structured data of the detected entities, rather than on individual frame-level data. Ramachandra et al. [2] proposed using a track-based detection criterion when tracking information is accessible. According to their suggestion, a ground truth region in a frame is deemed detected if the Intersection Over Union (IOU) between the ground truth region and a detected region is greater than or equal to a predefined fraction. Furthermore, for a track to be considered a true detection, only a fraction α of the track needs to be detected [2].

The metric established by Ramachandra et al. [2] was modified for evaluating event level detection for the classical anomaly detection method. Annotations were made based on time stamps rather than on each detected track. This approach was chosen to assess the method's performance in detecting real anomalies in the video, rather than evaluating tracks identified by the object detector and tracker. Consequently, no track ground truth was available. Therefore, the detection criteria were adjusted, and a track would be considered detected if a fraction α of the ground truth was detected and can be defined as follows:

$$\frac{min(exitTime_{gt}, exitTime_{detected}) - max(entryTime_{gt}, entryTime_{detected})}{exitTime_{gt} - entryTime_{gt}} \geq \alpha \tag{6.3}$$

where α is the minimum fraction of the anomalous event that needs to be detected, $exitTime_{gt}$ and $entryTime_{gt}$ are the ground truth for the entry and exit

time of an anomalous entity and $exitTime_{detected}$ and $entryTime_{detected}$ are the detected entry and exit time of an anomalous entity.

Then the TPR can be calculated as:

$$TPR = \frac{\#anomalies\ detected}{\#anomlies\ in\ ground\ truth} \tag{6.4}$$

and the FPR as:

$$FPR = \frac{\#false\ detections}{\#trajectories - \#of\ anomalous\ trajectories} \tag{6.5}$$

In both methods, the fraction α was set to 10%, following the approach of Ramachandra et al. [2]. This choice of 10% was considered appropriate, as even a small percentage of the anomaly would be sufficient to trigger an alert for an operator to confirm the occurrence of an anomaly.

Another anomaly detection metric worth considering is pixel-level evaluation, which emphasizes the precise localization of anomalies within a frame. However, for traffic anomalies, pinpointing the exact location of the anomaly within the frame does not provide significant additional information. Consequently, pixel-level annotation was not included in the evaluation.

6.2 Classical Anomaly Detection Evaluation

This section presents the evaluation of the classical anomaly detection method. Section 6.2.1 presents the overall results per dataset. Sections 6.2.2, 6.2.3, 6.2.4 and 6.2.5 discuss the results obtained per category and perform and error analysis. Then, an analysis of the ID switch detection is presented in a Subsection 6.2.6. Subsection 6.2.7 summarizes all the results.

In this section, it is crucial to emphasize that the evaluation of the classical anomaly detection method is solely based on anomalies perceived by the object detector-tracker in the video. Any vehicle performing an anomaly not recorded by the object detector-tracker cannot be identified by the classical anomaly detection method. Therefore, filtering out all undetected anomalies by the object detector-tracker ensures a fair assessment of the classical anomaly detection's performance on its own. A comprehensive analysis, including anomalies not recorded by the object detector, will be provided in Section 6.4.

6.2.1 Overall Results

Tables 6.1, 6.2, and 6.3 present the results for the Mgarr, Zejtun, and Street Scene datasets, respectively. For each dataset, the analysis includes results per anomaly category and overall performance, utilizing $\alpha = 0.1$.

In the Mgarr dataset, both anomaly detection scenarios, with and without removing trajectories with ID switches, achieved reasonably high TPR and low FPR. The Zejtun dataset achieved comparable performance with satisfactory TPR and FPR. The nature of the video proved to be a challenge for the anomaly detector, due to the obstructions and perspective, which meant a large number of anomalies in the video were not perceived by the object detector and tracker, resulting in a low number of anomalies in the structured data ground truth.

The Street Scene dataset, on the other hand, demonstrated robust performance with a diverse range of anomalies. Like the other datasets, it also achieved a high TPR and low FPR in both anomaly detection scenarios, emphasizing the effectiveness of the approach across varied urban scenarios.

From these results, it can be concluded that the classical anomaly detector performs well in detecting anomalies from structured data across a variety of datasets with diverse conditions. Achieving a reasonably high TPR and a low FPR in all cases underscores its effectiveness.

Notably, upon examination of Tables 6.1, 6.2, and 6.3, it becomes evident that not all anomalies are detected at the same rate. In-depth analysis to uncover the reasons behind these limitations is presented in subsequent sections.

The low FPR observed across all cases indicates a high reliability of the method, as it produces few false alerts. Furthermore, the FPR is further improved with the ID switch detection method. However, it is important to note that while this improvement is notable, the ID switch detection also leads to a reduction in detected anomalies. This does not necessarily imply that the ID switch detection is inaccurate. Instead, it suggests that the track likely retained features indicative of an ID switch, such as temporarily losing the bounding box and subsequently reappearing in the correct position of the vehicle. An analysis of this aspect is also provided in Section 6.2.6.

Various anomalies were detected by multiple categories which could be due to anomalies overlapping in time. Subsequent discussions will primarily concentrate on anomalies that were intended to be detected within their respective categories.

Table 6.1 Anomaly detection results for the Mgarr dataset

Anomaly	Number of anomalies	TPR With ID Switch Detection	TPR Without ID Switch Detection
Crossing Center Line	59	86.44%	94.92%
Overtaking	9	55.56%	77.78%
Jaywalking	84	90.48%	91.67%
Traffic	1	100.00%	100.00%
Bicycle	4	50.00%	100.00%
Parking	14	64.29%	85.71%
Obstructions	19	73.68%	100.00%
Vehicle on wrong Lane	19	63.16%	78.95%
U-Turn	21	95.24%	100.00%
Scooter	11	63.64%	63.64%
Unforeseen Anomalies	2	100.00%	100.00%
All	**243**	**81.89%**	**90.95%**
Overall FPR		**3.39%**	**7.35%**

Table 6.2 Anomaly detection results for the Zejtun dataset

Anomaly	Number of anomalies	TPR With ID Switch Detection	TPR Without ID Switch Detection
Jaywalking	21	85.71%	85.71%
Bicycle	5	60.00%	80.00%
Vehicle on wrong Lane	5	60.00%	80.00%
U-Turn	3	0.00%	66.67%
Scooter	1	100.00%	100.00%
Overtaking	8	87.50%	100.00%
All	**43**	**74.42%**	**86.05%**
Overall FPR		**0.23%**	**0.59%**

Table 6.3 Anomaly detection results for the Street Scene dataset

Anomaly	Number of anomalies	TPR With ID Switch Detection	TPR Without ID Switch Detection
Crossing Center Line	31	90.32%	93.55%
Jaywalking	70	90.00%	90.00%
Traffic	51	74.51%	80.39%
Bicycle on wrong lane	20	85.00%	100.00%
Heavy goods vehicles	3	0.00%	100.00%
Parking	7	85.71%	100.00%
Obstructions	2	50.00%	50.00%
Vehicle on wrong Lane	9	77.78%	88.89%
U-Turn	1	0.00%	100.00%
All	**194**	**82.47%**	**89.18%**
Overall FPR		**3.69%**	**8.36%**

6.2.2 Category 1: Non-Typical Vehicle Path

The method employed in this section was straightforward, involving flagging paths that vehicles were not expected to traverse. While the simplicity of the approach suggested a high TPR, it was found that the assumption that vehicles performing unusual maneuvers would necessarily exit from uncommon points did not consistently hold true.

The following lists provide an error analysis organized by dataset, highlighting detected anomalies, missed detections, and the impact of the ID switch detection method. Any error that was not mentioned was due to the analyzed labeled video being ambiguous and the source of the error is unclear.

Error Discussion

- Mgarr Dataset:
  - Vehicles crossing the center line: 54 out of 59 instances detected.
    - Missed detections: 4 instances where vehicles crossed the center line to park were not detected. One other vehicle did not exit the side road as assumed and hence not detected.
  - U-turns: 19 out of 21 detected.
    - Missed detections: 2 instances were vehicles executed three-point turns, and exited from a normal point .
- Zejtun Dataset:
  - U-turns: 2 out of 3 detected.
    - Missed detection: 1 instance off-screen, causing re-entry detection as a new car.
  - Additionally detected: 1 wrong lane anomaly, 1 overtaking anomaly.
- Street Scene Dataset:
  - Vehicles crossing the center lines: High accuracy with 3 misses.
    - Missed detections: Vehicles stopping partially outside the frame. The detector marks a vehicle as exited when the vehicle crosses the line whilst the annotator marked it when it completely exited the frame. Thus there was a discrepancy in the time taken .
  - Parking anomalies: 4 out of 7 detected when ID switches were removed.

Impact of ID Switch Detection:

- Mgarr Dataset:
  - False detections reduced from 24 to 4.
    - The remaining false detections were undetected ID switches
  - True positives reduced to 47 from 54 (vehicles crossing the center line) and 17 from 19 (U-turns).
- Zejtun Dataset:
  - False detections reduced from 17 to 1.
    - Remaining false detection caused by a lighting fixture mistaken for a vehicle in low visibility.
  - True positives reduced to 0 from 2 (u-turns)
- Street Scene Dataset:
  - False detections reduced from 8 to 0 with ID switch detection.
  - True positives reduced to 28 from 29 (vehicles crossing the center line) and 6 from 7 (parking).

The Category 1 anomaly detection method demonstrated success in urban road scenarios, particularly in identifying *U-turns* and *vehicles crossing center lines* across various datasets. Its simplicity makes it suitable for real-world deployment, although improvements to the object detector assumptions would increase the TPR.

The ID switch detection method affected detections, with some true positives being marked as ID switches. This does not indicate incorrect behavior of the ID switch detection but suggests that the track exhibited characteristics of an ID switch, such as temporarily losing the bounding box and then correcting it. Integrating the ID switch detection mechanism improved accuracy by reducing false detections, though it slightly decreased the TPR. The assumptions provided a balanced approach to reducing the FPR while maximizing the TPR.

6.2.3 Category 2: Non-Typical Location of Object

This category involves automatically creating a mask defining the allowable area for certain types of vehicles with very minimal manual input. It is an explainable method, as the created mask clearly shows the entire region of interest. The mask is designed for detecting three types of location rules: *Jaywalking Anomalies*, *Wrong Lane Anomalies*, *and Bicycle Outside Lane Anomalies*. Note that *Bicycle Outside Lane Anomalies* is only relevant for the Street Scene dataset, as the other datasets do not include bicycle paths.

Jaywalking Anomalies

The object detector and tracker encountered challenges in accurately detecting pedestrians throughout their entire walk across the frame. Consequently, their trajectories were often fragmented into short pieces. The masking method was developed to mitigate this flaw in the data acquisition. To create the masks of allowable paths the assumption that pedestrians will not jaywalk was made.

This assumption did not hold true for the Mgarr dataset. Most pedestrians either jaywalked in the road, crossed on the stop sign or walked on the side where there is no pavement. The irregular patterns pedestrians took resulted in clusters of allowable paths on only parts of the pavement. However, with DBSCAN clustering parameters set to an epsilon of 75 and minimum sample set to five, 3 clusters centred around the bus stop waiting area were collected. The clusters were used to create the mask, with the clusters outlining the allowable area to be in, as shown in Figure 6.1. The clusters around the bus stop correctly identified the locations where pedestrians who did not jaywalk stood in the frame. Pedestrians using the rest of pavement were likely to jaywalk.

For the Zejtun dataset, all detected pedestrians are considered anomalous due to the absence of pavements. Hence, one could argue that jaywalking anomalies should be categorized as Category 4 anomalies in this context. However, the mask creation method unexpectedly provided a benefit for this dataset. Most pedestrians detected in this dataset passed underneath balconies, resulting in even more fragmented trajectories than usual. Consequently, these trajectories were difficult to cluster. Interestingly, a group of 62 pedestrian trajectories resulted from the detection of parts of the balcony as pedestrians, from which a mask was created. The created mask effectively prevented these 62 trajectories from being falsely identified as anomalous jaywalkers, a scenario undetected by the ID switch detection or covered by the mask during data acquisition. In this case the masking method was able to filter out reoccurring errors in the detection. A DBSCAN clustering with an epsilon of 75 and a minimum sample size of five was utilized. Figure 6.2a displays the trajectories that were deemed to be outliers with the DBSCAN clustering. These clusters are not used to create the mask of allowable paths. Figure 6.2b displays the trajectories clustered in one group using DBSCAN clustering, which in this case were errors in detection. The resulting mask, created from this cluster, is depicted in Figure 6.3a.

To detect jaywalkers, in the Street Scene dataset, the DBSCAN clustering was performed with an epsilon of 25 and a minimum sample of five. This resulted in 16 clusters which were used to create the mask. The mask as shown in Figure 6.3b, correctly classified the pavement as the allowable area for pedestrians.

Figure 6.1 Generated mask of allowable paths for pedestrians for Mgarr Dataset

(a) Anomalous Cluster

(b) Cluster used for mask creation

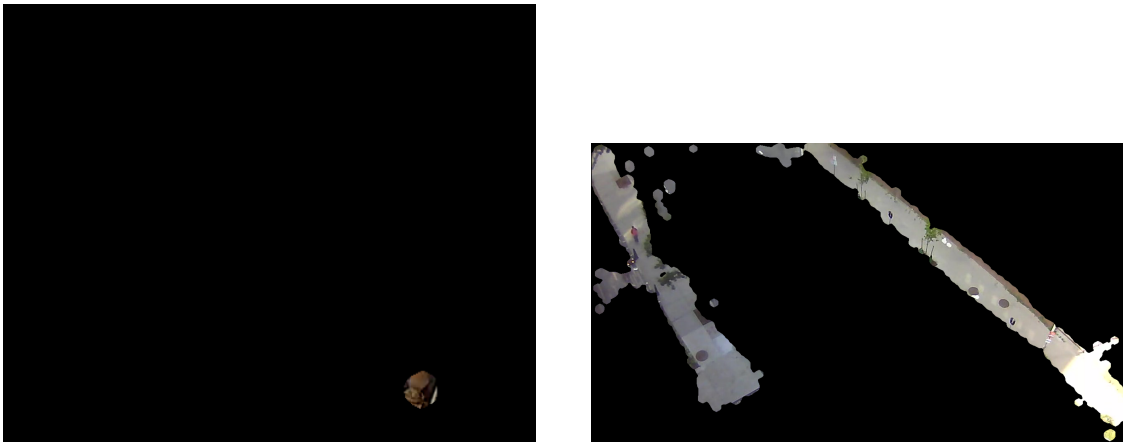
Figure 6.2 Trajectories from the Zejtun dataset plotted on a cropped frame from the Zejtun dataset.

Areas like the stairs leading up buildings were also marked as allowable paths, since pedestrians in the dataset used them often. Some points of the mask are seen to not be completely covered by the mask. This noise does not heavily affect the efficacy of the mask since a jaywalker would still have to cross across the masked areas to pass through the unmasked spots, thus would still be detected.

The lists below offer an error analysis categorized by dataset, emphasizing identified anomalies, missed detections, and the influence of the ID switch detection technique. Errors that were not addressed were a result of ambiguities in the analyzed labeled video.

Error Discussion

- Mgarr Dataset:
  - Jaywalkers: 75 out of 84 detected.
  - Additional detections: 7 out of 11 scooters and 1 out of 4 bicycles.
    - Missed detections: ground truth entry/exit time being not close to the detected entry/exit time since the lost tracks for pedestrians would start and end at random times.
- Zejtun Dataset:



(a) Cropped generated mask of allowable paths for pedestrians for Zejtun Dataset

(b) Generated mask of allowable paths for pedestrians for Street Scene Dataset

Figure 6.3 Trajectories from the Zejtun dataset and Street Scene Dataset

  - Jaywalkers: 18 out of 21 detected.
  - Additional detections: 1 out of 11 scooters detected as a jaywalker.
    * Missed detections: 2 pedestrians detected as bicycles and 1 pedestrian track not saved in the log.

- Street Scene Dataset:
  - Jaywalkers: 63 out of 70 detected.
  - Additional 3 parking anomalies were detected since there were cases of vehicles parking whilst people were also jaywalking.
    * Missed detections: Likely caused by the detector-tracker discarding very short pedestrian trajectories which is difficult to asses visually from videos.

The ID switch detection method was solely used to exclude trajectories during the mask creation process and was not applied in the detection metrics. Due to the irregularity of pedestrian trajectories, no reliable method for identifying false detections was found, resulting in a high number of false positives.

False Detections

- Mgarr Dataset:
  - 54 False detections:
    * 13 due to missed detection in the ground truth.
    * 13 caused by ID switches cutting anomalous trajectories in parts or making pedestrians appear to be stepping on the road.
    * 8 due to people boarding the bus.
    * 1 caused by a poster on a bus being detected as a pedestrian.

        - The rest were due to the mask not covering all allowable areas for pedestrians.
- Zejtun Dataset:
    - 1 False detections:
        - Caused by a flying bird detected as a pedestrian.
- Street Scene Dataset:
    - 43 False detections:
        - 21 due to ID switches and detector errors.
        - 8 jaywalking cases likely missed by the annotator.
        - The rest were pedestrians close to the edge of the pavement detected as outside the mask.

The Category 2 method for detecting jaywalking achieved a high TPR. However, it also resulted in a significant number of false detections, primarily due to fragmented pedestrian trajectories and erroneous detections. For better reliability in detecting pedestrian anomalies, further improvement of the object detector-tracker is required.

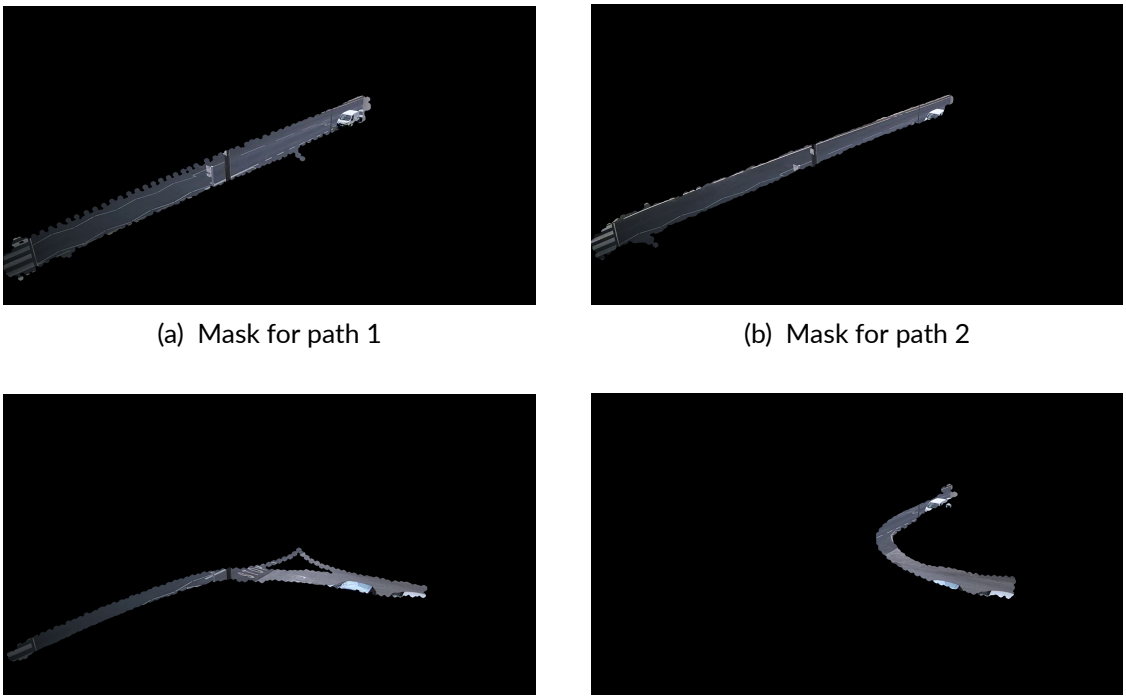
Wrong Lane Anomalies

Given that vehicles tend to follow the most probable path, creating a mask outlining this expected route enables the detection of anomalies such as *overtaking*, *parking*, *obstruction*, and *driving in the wrong lane*. The masks were generated by initially clustering vehicles based on their entry and exit pairs, excluding those deemed anomalous in Category 1. For each dataset, DBSCAN clustering was used with a minimum sample size set to $\frac{1}{30}^{th}$ of the total number of trajectories for that entry-exit pair. An epsilon value of 50 was utilized for the Mgarr dataset, 75 for the Zejtun dataset, and 50 and 150 for different lanes in the Street Scene dataset. The masks generated for the Mgarr dataset are depicted in Figure 6.4.

An error analysis, organized by dataset, is provided in the following lists, which highlight detected anomalies, missed detections, and the effects of the ID switch detection method. Errors not mentioned were due to ambiguities in the analyzed video.

Error Discussion

- Mgarr Dataset:
    - Overtaking: 7 out of 9 instances detected.
    - Parking: 9 out of 14 instances detected.



(a) Mask for path 1

(b) Mask for path 2

(c) Mask for path 3

(d) Mask for path 4

Figure 6.4 The masks of allowable regions for the normal entry and exit pairs in the Mgarr Datasets

  - Obstruction: 17 out of 19 instances detected.
  - Wrong lane: 14 out of 19 instances detected.
  - Unforeseen anomalies: 1 out of 2 instances detected.
    * Missed detections: 5 instances (overtaking and wrong lane) due to detector errors and inaccuracies in mask. Taller vehicles such as buses, would be in the correct area however their centers would be higher than the rest of the vehicles, which results in the buses being detected outside the mask

- Zejtun Dataset:
  - Wrong lane: 4 out of 5 instances detected.
  - Overtaking: 8 out of 8 instances detected.
    * Missed detections: The road had no clearly defined lanes, hence the allowable area for both directions of traffic was almost completely overlapping. Occurred when the overtaking procedure remained within the allowable mask area.

- Street Scene Dataset:
  - Wrong lane: 5 out of 9 instances detected.
  - Parking: 3 out of 7 instances detected
    * Missed detections: Occurred when anomalous behaviors were per-

formed within the allowable mask areas.

Impact of ID Switch Detection

- Mgarr Dataset:
  - False detections reduced from 110 to 9.
    - One false detection was due to an undetected ID switch
    - Remaining false detections due to slight deviations from the allowable area.
  - True positive detections reduced to:
    - Overtaking: 7 to 5 instances detected.
    - Parking: 9 to 7 instances detected.
    - Obstruction: 17 to 12 instances detected.
    - Wrong lane: 14 to 11 instances detected.
    - Unforeseen anomalies: 1 instance, no change.
- Zejtun Dataset:
  - False detections reduced from 24 to 8.
  - Remaining False Detections:
    - 5 were attributed to the mask not entirely covering the required area
    - 2 were due to missing annotations in the ground truth
    - 1 was a false detection of a bus when there was no bus in the video
  - True positive detections reduced to:
    - Wrong lane: 4 to 3 instances detected.
    - Overtaking: 8 to 7 instances.
- Street Scene Dataset:
  - False detections reduced from 122 to 24.
  - Remaining false detections:
    - 12 were attributed to non-detected ID switches occurring towards the end or beginning of the path.
    - 8 cases were vehicles were out of the lane but not annotated.
    - 1 instance of a car exiting parking space was falsely detected.
    - 1 case involved a bicycle being mistakenly detected as a motorcycle.
  - True positive detections reduced to:
    - Wrong lane: 5 to 4 detected instances.
    - Parking: 3 instances, no changes.

The Category 2 method demonstrated its effectiveness in detecting anomalies related to vehicles being in the wrong lane, despite encountering challenges such as ID switches in the data. The method proved adaptable across different datasets, demonstrating applicability in varied scenarios, from unorganized roads like Zejtun to highly organized urban roads like the Street Scene location. The implementation of the ID switch detection method notably reduced false detections caused by ID switches and contributed to the creation of more accurate masks.

Bicycle Outside Lane Anomalies

The Street Scene dataset provided a clear opportunity to examine the Category 2 method's performance in bicycle lanes. With two well-defined bicycle paths on the observed roads, it offered a focused investigation into the method's efficacy in detecting anomalies within these lanes, expanding its applicability to diverse road infrastructures. The DBSCAN clustering of bicycles was executed with an epsilon of 25 and a minimum sample size of five.

The following lists provide an error analysis, highlighting detected anomalies, missed detections, and the impact of the ID switch detection method.

Error Analysis

- Street Scene Dataset:
  - Bicycles outside their lane: 20 out of 20 detected.

Impact of ID Switch Detection

- Street Scene Dataset:
  - False detections reduced from 91 to 47.
    - 26 false detections due to ambiguous trajectories not labeled as anomalous in the ground truth often due to bicycles being exactly at the edge of the lane.
    - 15 false detections were due to the mask not extending to the edge of the frame since the detector did not detect bicycles in that location most of the time
    - 6 false detections due to detector errors, such as pedestrians mistakenly identified as bicycles.
  - True positives reduced from 20 to 17.

Despite facing significant obstructions, the method effectively outlined bicycle travel areas using masks from DBSCAN clustering. Challenges arose due to

significant obstructions in the lower left portion of the image, with the left lane obstructed by a roof ledge and the top of the lane partially obscured by a tree, leading to many bicycles being lost from view. Coupled with the detector-tracker's limited accuracy for bicycles, this resulted in a high number of false detections. However, applying the ID switch detection method notably reduced these errors, improving anomaly detection accuracy within bicycle lanes

6.2.4 Category 3: Non-Typical Slow Vehicle

It was observed that several anomalies exhibited by vehicles on urban roads involved them slowing down before performing the said anomaly. Hence, the Category 3 method aimed to identify anomalies characterized by slow-moving vehicles. The primary anomalies of interest were *vehicles crossing the center line* due to their tendency to slow down before exiting a road, *traffic jams*, *parking anomalies*, and *u-turns*, although other anomalies were also detected using this approach. However, the Zejtun dataset did not contain anomalies with slow-moving characteristics, rendering this method inapplicable for that dataset.

The performance of this method varied due to data being collected from diverse locations. Errors not mentioned were caused by ambiguities in the analyzed video. An error analysis is provided, organized by dataset, highlighting detected anomalies, missed detections, and the impact of the ID switch detection method.

Error Discussion

- Mgarr Dataset:
  - Vehicles crossing the center line: 30 out of 59 instances.
  - Parking anomalies: 4 out of 14 instances.
  - All instances of obstructions, traffic, u-turns, and unforeseen anomalies.
  - Missed detections:
    - Five parking anomalies: Vehicles did not slow down to park in easily accessible parking spots. These same vehicles were also not detected by the Category 2 method since the area where cars were parked was susceptible to ID switches, which resulted in inaccurate masks.
    - Vehicles crossing the centerline: Did not slow down sufficiently but were detected by Category 2 method.
- Street Scene Dataset:
  - Parking anomalies: 1 out of 7 instances detected.

  - Obstructions: 1 out of 2 instances detected.
  - Traffic anomalies: 36 out of 51 instances detected.
  - Vehicles crossing the center line: 6 out of 31 instances detected. This was caused since the vehicles tended to slow down before exiting from the side road
  - Missed detections: Speeds of anomalous events not slow enough to be flagged by the method.

Impact of ID Switch Detection

- Mgarr Dataset:
  - False detections reduced from 61 to 40.
  - Detections reduced to:
    * Vehicles crossing the center line: 26 instances from 30.
    * Traffic: not reduced.
    * Parking: 3 instances from 4.
    * Obstructions: 14 instances from 19.
    * U-turns: 19 instances from 21.
    * Unforeseen anomalies: not reduced.
  - The false detections which were not detected to have ID switches were caused by:
    * Two video artifacts detecting vehicles at low speeds.
    * 38 vehicles driving slowly without performing an anomaly.
- Street Scene Dataset:
  - False detections reduced from 4 to 2.
  - Traffic anomalies detections reduced from 36 to 33.
  - False detections which were not detected to have ID switches were caused by:
    * Vehicle slowing down in traffic not annotated as an anomaly due to the ambiguity in defining slow vehicles.

The Category 3 method for detecting slow-moving vehicles effectively identified the anomalies of interest, such as vehicles crossing the center line, traffic jams, parking anomalies, and u-turns. The ID switch detection mechanism significantly reduced false detections.

6.2.5 Category 4: Non-Typical Vehicles

This method was intended to identify unusual vehicles in a location. Bicycles need to be detected for the Mgarr and Zejtun locations, and heavy goods vehicles for

the Street Scene dataset.

An error analysis by dataset highlights identified anomalies, missed detections, and the impact of the ID switch detection method is presented below. Any unreported errors were caused by ambiguities in the analyzed videos.

Error Description

- Mgarr Dataset:
  - Bicycles: 4 out of 4 detected.
- Zejtun Dataset:
  - Bicycles: 4 out of 5 detected.
  - Missed detection: 1 undetected bicycle due to an ambiguous ID switch in the middle of its route.
- Street Scene Dataset:
  - Heavy goods vehicles: 3 out of 3 detected.

Impact of ID Switch Detection

- Mgarr Dataset:
  - no false detection caused by this method which needed to be filtered out
  - Detections reduced from 4 to 2 (bicycles) due to ID switches.
- Zejtun Dataset:
  - Detections reduced from 4 to 3 (bicycles) due to ID switches.
  - 1 False detection reduced to 0 .
- Street Scene Dataset:
  - All three heavy goods vehicles exhibited detector errors during part of the detection. Large size and overhead angle caused more ID switches.
  - no false detection caused by this method which needed to be filtered out

A high TPR rate was expected due to the simplicity of the rule for anomaly detection, which was achieved for bicycle anomalies. However, the large size of the heavy goods vehicles and the overhead angle caused several ID switches throughout the trajectory, resulting in them being filtered out. The investigation on the data acquisition in Section 6.4 further explains the limitations with the object-detector tracker recording heavy goods vehicles in the Street Scene dataset.

6.2.6 ID Switch Detection Analysis

Although the impact of ID switches has been briefly discussed per category in previous sections, this section aims to delve deeper into their effects on the results. The ID switch detection method is designed to identify instances where there is an unusually large Euclidean distance between consecutive trajectory points, signaling a potential ID switch occurrence. As depicted in Table 6.4, the ID switch detection method significantly reduced the number of false detections by more than half across all datasets.

Indeed, instances of significant gaps between trajectory points can occasionally occur even in the absence of an ID switch. This phenomenon often coincides with the bounding box *jumping*, where the tracking system momentarily loses track of a vehicle before subsequently correcting itself. In the event of an ID switch, such a jump would result in the misidentification of the vehicle. Moreover, ID switches can occur towards the end of a trajectory, where the majority of the trajectory is accurate except for the concluding segment. Eliminating these instances could inadvertently exclude trajectories that represent anomalies, thereby potentially reducing the TPR.

Table 6.5 demonstrates that the reduction in true positives due to ID switch detection is less than 14% for all datasets. The significant reduction in false detections, combined with the minimal decrease in true detections when ID switches are removed, indicates that ID switches are responsible for many false detections. This also suggests that the ID switch detection method is effective across various camera angles and locations. The occurrence of true positive detections with an associated ID switch does not imply a failure of the ID switch detection method. In fact, there are instances where an ID switch occurs, yet the remaining trajectory still accurately represents an anomalous event, as previously explained.

A qualitative analysis of ID switch detection is also conducted to further investigate the detected ID switches, as detailed in the subsections below. This analysis encompasses all three datasets and focuses on instances where ID switch detection significantly impacted the output, either by substantially reducing the number of false detections or decreasing the number of true detections. The investigation involved visually analyzing the trajectories of interest by examining the trajectory plots and the filtered plots showing the distance between each trajectory point.

Mgarr Dataset

For the Mgarr dataset, the ID switch detection significantly reduced false detections for Category 1 anomaly detection, decreasing from 24 to 4. An even more

Table 6.4 Table showing false positives per dataset with and without ID switches.

False Positives	Without ID switch detections	With ID switch detection	Percentage reduction
Street Scene	272	120	55.88%
Mgarr	234	108	53.85%
Zejtun	26	9	65.38%

Table 6.5 Table showing true positives per dataset with and without ID switches.

True Positives	Without ID switch detections	With ID switch detection	Percentage reduction
Street Scene	173	160	7.51%
Mgarr	221	199	9.95%
Zejtun	37	32	13.51%

substantial reduction in false detections was observed for Category 2 (vehicles in the wrong lane), with a decrease from 110 to 9. Among the 20 trajectories with ID switches in Category 1 false detections, all plots indicate that an ID switch occurred. Two plots exhibited short ID switches with very high speeds, while the rest showed large distances between certain trajectory points, forming fake U-turns and turns, as depicted in Figure 6.5b. The plot of the filtered distance between points in Figure 6.5a clearly shows the peaks detected by the threshold.

For the false detections with ID switches in Category 2 (vehicles in the wrong lane), all observed trajectories exhibited characteristics indicative of ID switches. An example can be seen in Figure 6.5c. In these trajectories, the green trajectory forms a fake U-turn due to an ID switch, and in the blue and orange trajectories, it is likely that a vehicle was momentarily misidentified as one of the pedestrians on the pavement.

For the real anomalous trajectories that were also detected to have an ID switch, it was observed that in most cases, an ID switch occurred towards the end of the trajectory involving a parked car, even though the rest of the trajectory exhibited genuine anomalous behavior. Figure 6.6 provides an example of this. The orange and green trajectories are correctly detected as anomalous events but include an ID switch with a parked car. The blue trajectory has missing points along its path, which could be due to an ID switch or the detector briefly losing track of the vehicle due to an obstruction.

Zejtun Dataset

For the Zejtun dataset, the Category 1 method reduced false detections from 17 to 1 by removing trajectories with ID switches. The plots of the trajectories identified with ID switches showed features indicative of an ID switch by the

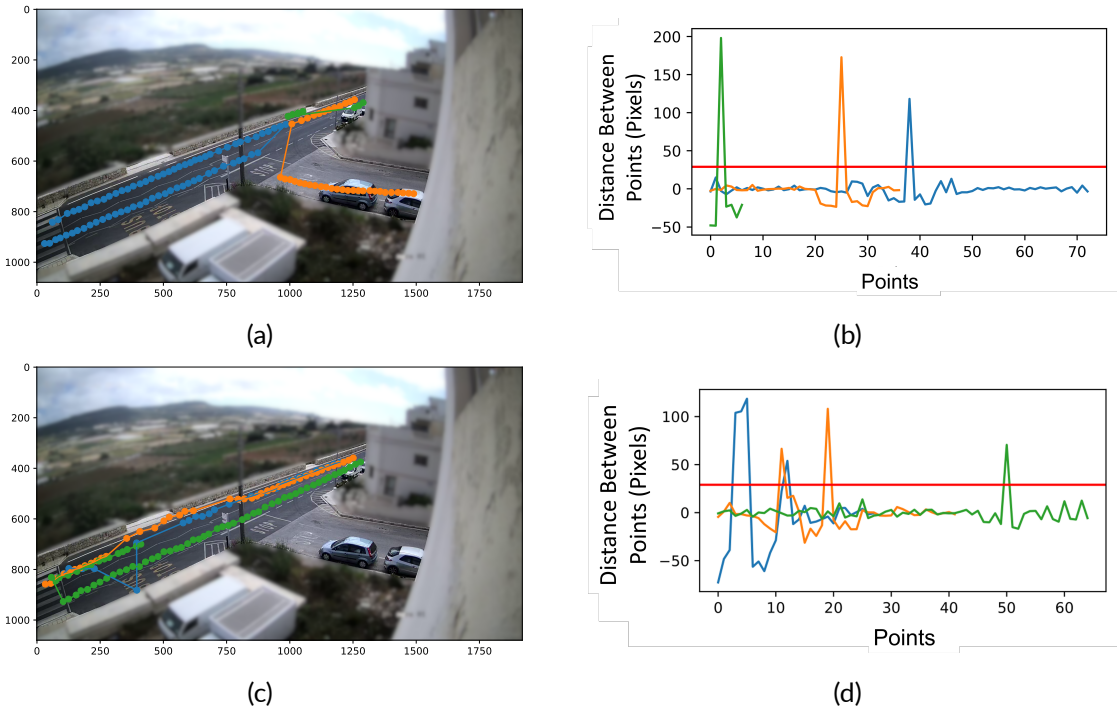


Figure 6.5 False detections for the Mgarr dataset detected by the Category 1 and Category 2 method which had an ID switch. Figure 6.5b and Figure 6.5a are three trajectories detected in the Category 1 method which were then detected to have a an ID switch, Figure 6.5c and Figure 6.5d are three false detections by the Category 2 methods which show examples of ID switches. The plot of the trajectories on a frame from the Mgarr location is shown for each category along with the corresponding plots of the filtered distances to their right

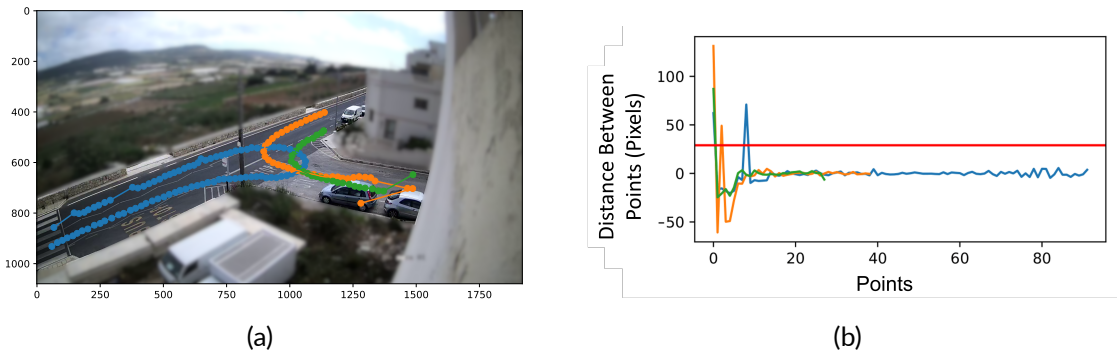


Figure 6.6 True detections for the Mgarr dataset detected by the Category 1 which had an ID switch. Figure 6.6a shows the plot of the trajectories and Figure 6.6b shows the plots of the filtered euclidean distances between each trajectory point.

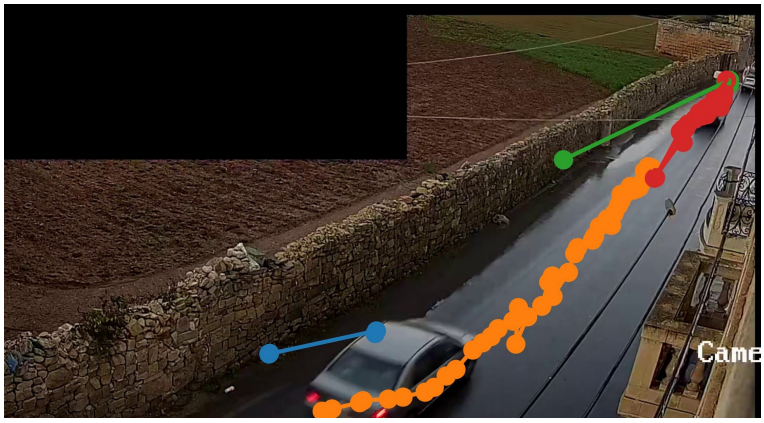


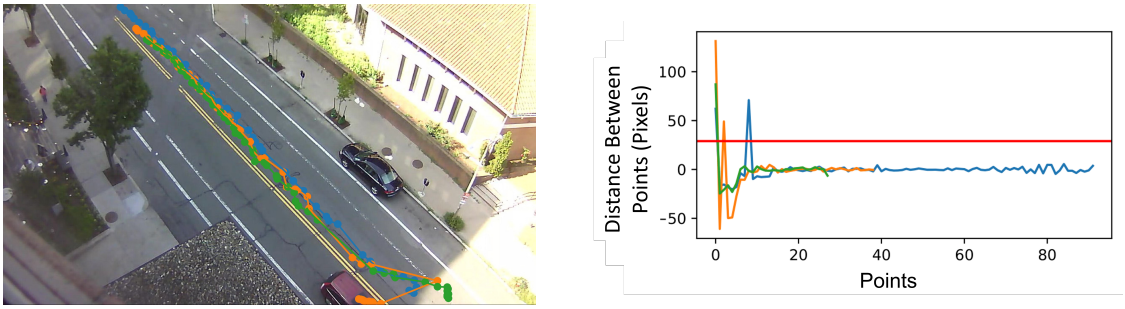
Figure 6.7 Plots of 4 false detections from the Zejtun datset detected to have ID switches. The speeds of the following trajectories are withing the range of 150 to 2250 km/h

detector-tracker. Among the 16 ID switch detections, 4 were identified due to their abnormally high speed. These trajectories were also often characterized by their extreme brevity, as seen in the green, blue, and red trajectories in Figure 6.7. The other trajectories with ID switch detections displayed characteristics such as forming fake U-turns and unlikely paths, confirming that the reduction of false detections through ID switch removal is valid.

Street Scene Dataset

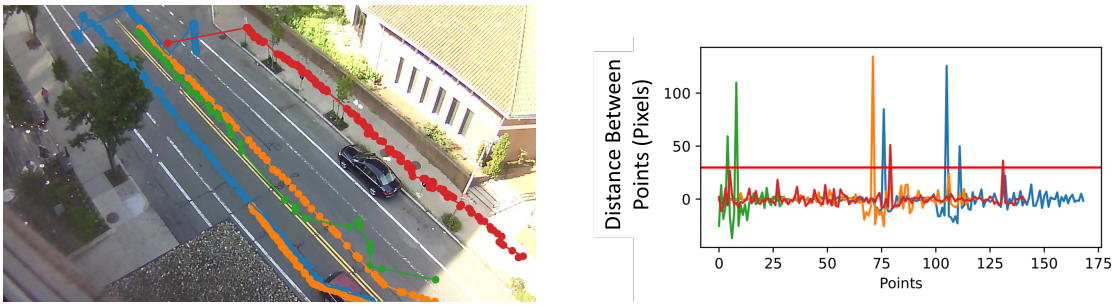
For the Street Scene dataset, one of the most significant effects of ID switch detection was that all three heavy goods vehicles were detected as having ID switches, even though these vehicles were previously identified during the data acquisition evaluation process. However, as shown in Figure 6.8a, all three heavy goods vehicles exhibited characteristics compatible with ID switches at the beginning of their trajectories, along with peaks in their filtered Euclidean distance between trajectory points, as seen in Figure 6.8b. This behavior was likely caused by the large size of the vehicles, which may have led to the detector not detecting the vehicles smoothly or an ID switch occurring with the vehicle behind them in traffic. Additionally, the orange trajectory indicates that the heavy goods vehicle was briefly mistaken for a vehicle in another lane at the beginning of the track.

The false detections for Category 2 vehicles on the wrong lane were also substantially reduced when ID switches were removed, decreasing from 122 to 24. The detected ID switches all exhibited characteristics of impossible trajectories, an example of which can be seen in Figure 6.9a. The orange and blue trajectories appear to be combinations of two vehicle trajectories forming a fake U-turn. The green trajectory seems to suddenly skip from one place on the lane to another, while the red trajectory identifies a vehicle that instantly moves from



(a) (b)

Figure 6.8 Heavy goods vehicles detections for the Street Scene dataset detected as a Category 4 anomaly which had an ID switch. Figure 6.6a shows the plot of the trajectories and Figure 6.6b shows the plots of the filtered euclidean distances between each trajectory point.



(a) (b)

Figure 6.9 False detections for the Street Scene dataset detected by the Category 3 which had an ID switch. Figure 6.9b shows the plot of the trajectories on one frame of the Street Scene Dataset and Figure 6.9b shows the plots of the filtered euclidean distances between each trajectory point.

the lane to the pavement. The blue and orange labels correspond to cars, and the green and red trajectories are labeled as buses. The ID switches all show peaks in the Euclidean distances between points, as seen in Figure 6.9b.

6.2.7 Summary

The classical anomaly detection method demonstrates robust performance across diverse datasets, exhibiting resilience to location variability and challenges such as obstructions, perspective shifts, and data acquisition errors. The accompanying ID switch detection method effectively mitigates false positives resulting from detector-tracker artifacts, thereby enhancing the overall reliability. Furthermore, the method's transparency—wherein the model is fully interpretable and all decisions and processes are explicitly justified—coupled with its minimal requirement for human intervention, significantly contributes to its user-friendliness.

To further enhance anomaly detection rates, significant improvements in object detection and tracking are warranted. Specifically, advancements in object tracking algorithms could mitigate the occurrence of ID switches, potentially rendering the ID switch detection method redundant. Incorporating visual features such as appearance descriptors, in addition to speed and location data, could be pivotal in reducing ID switches. Utilizing the developed ID switch detection method to identify instances of ID switches can facilitate the testing and refinement of tracking algorithms in challenging scenarios.

Furthermore, refining the object detector, particularly for pedestrian tracking and overhead angles, is essential for accurately capturing more complex anomalies. Enhancing the detector's capability to handle overhead views will minimize the errors observed in scenarios involving heavy goods vehicles in the Street Scene dataset. This fine-tuning will contribute to a more robust and reliable anomaly detection system.

Optimizing camera angles to minimize obstructions and perspective distortions would further bolster anomaly detection accuracy. Clearer views, especially in the presence of parked cars, can mitigate ID switch occurrences.

The entry and exit lines method employed by the object-detector tracker proves valuable for preliminary anomaly filtering. While other clustering approaches were considered, this method emerged as the most effective and explainable option.

The developed method necessitates minimal human intervention, primarily for determining suitable epsilon values for Category 2. Depending on the priorities of anomaly detection, additional rules can be incorporated. For instance, with access to long-term data, it is possible to create masks of allowable paths for each vehicle type, thereby enhancing detection specificity.

6.3 Deep Learning Based Anomaly Detection Method Evaluation

This section presents and discusses the results obtained for the deep learning-based anomaly detection method. The evaluation is conducted at the frame level, which is the typical method of evaluation for deep learning-based anomaly detectors. Section 6.3.1 presents all results for the trained datasets and the different variations of models used. The effects of masking, different illuminations, tiling of input images and data augmentation on the results are discussed in Subsections and 6.3.2, 6.3.3, 6.3.4 and 6.3.5, respectively. Subsequently 6.3.6 gives a summary of the results. A per event analysis of the deep learning-based anomaly

Table 6.6 Frame level AUC and EER performance metrics per dataset

Dataset	Method	AUC	EER
CUCK Avenue	**whole frame input**	**0.81**	**0.27**
Mgarr	whole frame input	0.81	0.26
	4 tiles per frame	0.81	0.28
	4 tiles per frame, augmented	**0.83**	**0.25**
Zejtun	whole frame input	0.67	0.44
	4 tiles per frame	0.69	0.36
	16 tiles per frame	0.68	0.39
	4 tiles per frame, augmented	**0.71**	**0.35**
Zejtun, sun overhead only	**4 tiles per frame, augmented**	**0.75**	**0.33**
Street Scene	whole frame input	0.77	0.29
	4 tiles per frame	0.70	0.35
	4 tiles per frame, augmented	0.72	0.35
Street Scene with mask	**whole frame as input**	**0.81**	**0.24**

detection method is then presented in Section 6.4.

6.3.1 Overall Results

Table 6.6 presents the frame-level AUC and EER for all datasets and data preparation methods assessed. All datasets were processed in grayscale in every test. The primary datasets of interest were the Mgarr, Zejtun, and Street Scene datasets. However, the initial step before training on the datasets of interest was training and evaluating on the CUCK Avenue Dataset [77]. This step confirmed that the model was set up as described by the authors [1] and achieved a performance close to theirs. Zhao et al. [1] achieved an AUC of 77.1% and an EER of 33.8% for grayscale input. In this work, training on the CUCK Avenue Dataset [77] yielded a higher AUC and a lower EER, indicating that the training procedure was sound.

In addition, Zhao et al. [1] trained and tested their model on various traffic datasets and achieved an average AUC of 79.6% and an EER of 26.2%. Therefore, it was expected that the traffic datasets here would achieve similar performance metrics. All datasets achieved results within a similar range, although certain datasets exhibited a higher EER. The elevated EER can be attributed to changes in illumination and the wide variety of anomalies that need to be detected. This aspect will be further discussed in subsequent sections.

The Zejtun and Mgarr datasets are relatively new, and as such, no other

method has been published that was tested on these datasets except for the work done by Leporowski et al. [98]. They achieved an impressive AUC of 87.91 ± 3.47% for Mgarr and 76.58 ± 13.09% for the Zejtun location when only using the video channel. However, their method relies on training with examples of anomalous events, not just normal events. Given the limited availability of videos containing abnormal events, it is likely that some videos would have parts of them in both the training, testing, and validation sets. Frames do not vary significantly between each other at a high frame rate, which means that almost identical frames would appear in both the test set and the training set. Since the aim of this section was to use a model capable of detecting anomalies with unpredictable features, this method of splitting data would not be considered optimal and a direct comparison cannot be made.

A limited evaluation was also performed on the Street Scene dataset in the original paper by Ramachandra et al. [2]. However, since the dataset has been re-labeled to include anomalies of specific interest for this study, the original results are not directly comparable to the findings presented in this work.

Figure 6.10 is the ROC curve for the Mgarr dataset from which the AUC is calculated, and Figure 6.11 is a plot of the Regularity Score per frame stack for the Mgarr test set. The part of the plot highlighted corresponds to the anomalous events regulatory score. As can be noticed from this plot, some normal events are detected as anomalous, and some events are detected as normal. An analysis of these errors is given in the following subsections.

6.3.2 Masking

The Mgarr and Zejtun datasets were masked to hide any areas except for the road and pavement, using black shapes. This masking likely affects the training process in some way. To investigate this effect, the Street Scene dataset was also masked, and the model was retrained and tested on this masked dataset. Figure 6.12 displays a masked frame from the Street Scene dataset, with the same mask applied consistently to all frames.

In the case of the Street Scene dataset, the masking procedure appeared to enhance the AUC of the model by 4% as seen in Table 6.6. This improvement is likely attributed to the mask forcing the model to focus on accurately reconstructing the road during training. Figure 6.12 illustrates an original frame alongside its reconstructed counterpart, along with a visualization of the Euclidean distances between frames for the model trained without a mask.

The visualization reveals that the unmasked background introduces significant noise into the Euclidean distance representation, indicating that the model

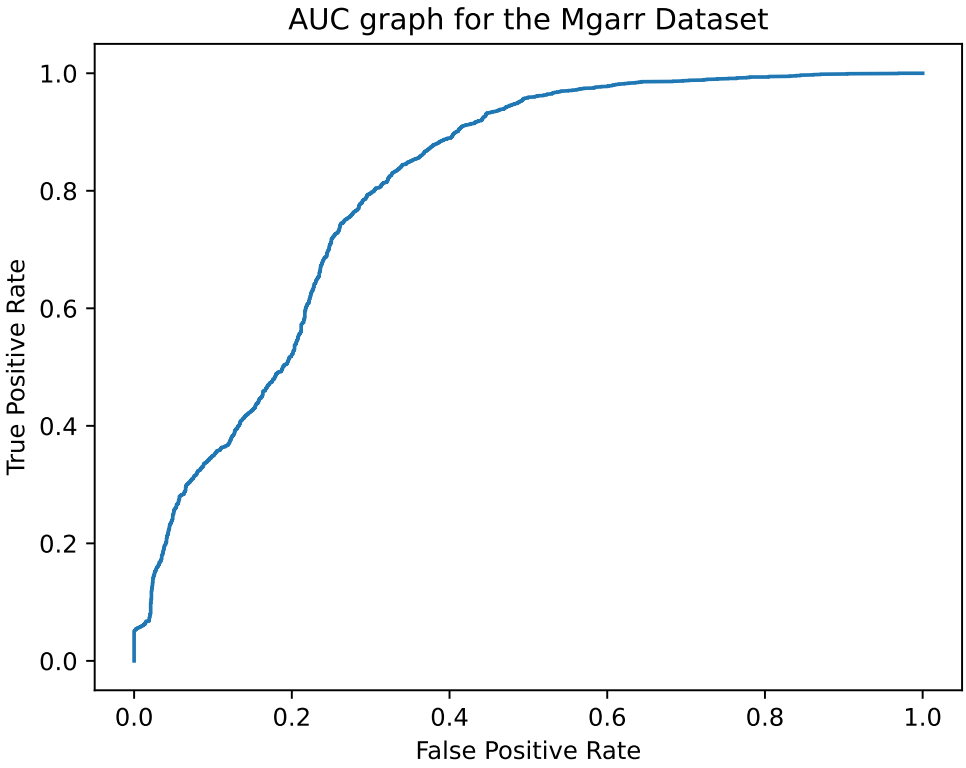


Figure 6.10 ROC Curve for the Mgarr Dataset for the best performing Mgarr model trained with augmented data and 4 tiles per frame

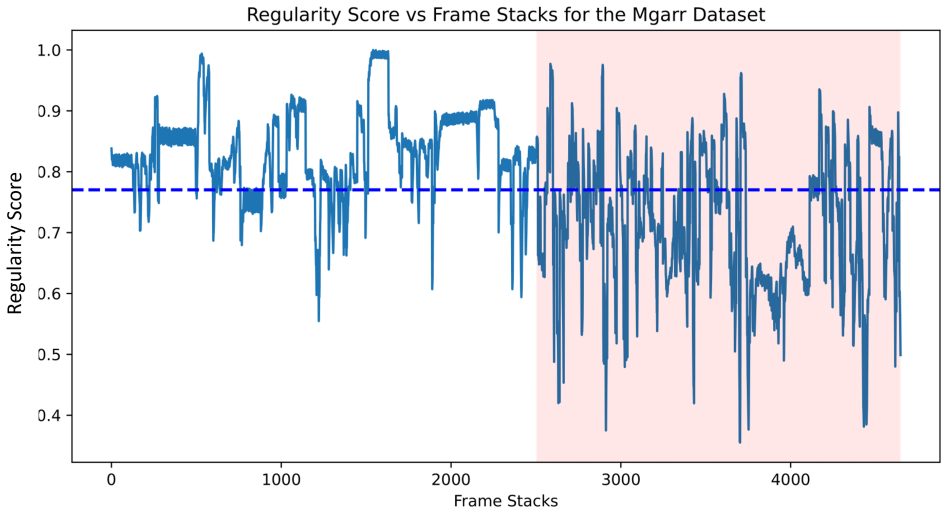


Figure 6.11 Regularity Score vs Frame Stacks for the Mgarr Dataset for the best performing Mgarr model trained with augmented data and 4 tiles per frame. The red shading corresponds to the anomalous events plot. The dotted line corresponds to the best threshold calaculated through the ROC curve

struggles to reconstruct background details accurately. Conversely, when the masking is applied, as depicted in Figure 6.13, the model trained with this masking

(a) (b) (c)

Figure 6.12 Figure 6.12a shows a grayscale frame from the Street Scene dataset. Figure 6.12b presents the reconstructed frame from the model. Figure 6.12c visualizes the Euclidean distance between the original and reconstructed frames, with yellow pixels indicating larger distances. Thus the more yellow pixels in a reconstruction, the more likely the frame belongs to an anomalous event. The frame, taken from the Test005 sample video, shows a traffic anomaly. The model struggles to accurately reconstruct this anomaly, resulting in blurry vehicles.

exhibits minimal additional errors from the background. This enhanced training enables better separation between events and normal videos.

However, it was observed that consistent use of the same mask across training, testing, and validation datasets is crucial. During tests where datasets were augmented prior to training, in one scenario, the mask was also augmented with the rest of the image. This entailed blurring the mask and altering its brightness along with the rest of the frame for the Zejtun Dataset. In this case, the varying values of the augmented masks hindered the model's ability to learn effectively. With augmented masks, the model achieved an AUC of 55% only, indicating that the results were nearly random. Figure 6.14 shows an example of a test frame being reconstructed and its Euclidean distance visualization.

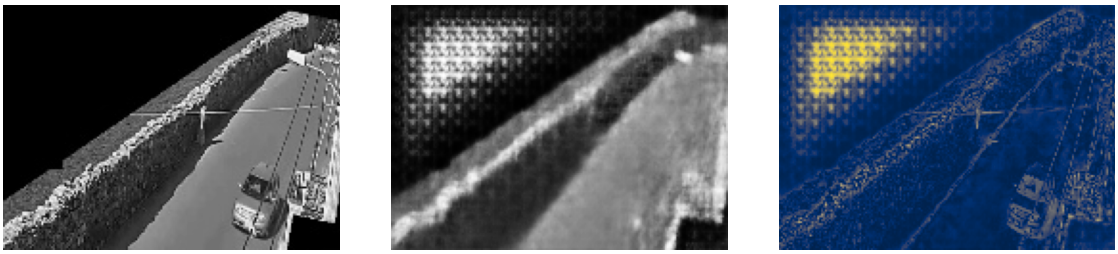
From the visualizations, it is evident that the mask was not accurately reconstructed, leading to very high reconstruction errors. This inaccuracy significantly impacts the training process and results, since the highest errors are not produced by anomalous events but by the masking procedure. This is demonstrated by the inability to reconstruct the normal vehicle. No other trained models incorporate augmenting the mask in the results.

6.3.3 Illumination Changes

The Zejtun Dataset performed the poorest among the three datasets tested for deep learning, achieving an AUC of 71% and a high EER of 35%. Analysis of the dataset revealed that many misclassifications were attributed to normal videos containing long shadows. Notably, all three videos with long shadows in the normal test set were incorrectly classified as anomalous. Figure 6.15 illustrates a tile from a frame with a strong shadow, its reconstruction, and the visualized Eu-

(a) (b) (c)

Figure 6.13 Figure 6.13a is a grayscale frame with mask from the Street Scene dataset resized. Figure 6.13b is the reconstructed frame from the model. Figure 6.13b is the visualized Euclidean distance between the pixels in the original frame and the pixels and the reconstructed frame. The more yellow the pixel the larger the distance between the pixels. The frame is taken from the Test005 sample video and is currently showing a traffic anomaly. The traffic anomaly could not be accurately reconstructed, that is why the vehicles in traffic are reconstructed very blurry.



(a) (b) (c)

Figure 6.14 Figure 6.14a is a grayscale frame with mask from the Zejtun dataset resized. Figure 6.14b is the reconstructed frame from the model. Figure 6.14c is the visualized Euclidean distance between the pixels in the original frame and the pixels and the reconstructed frame. The more yellow the pixel the larger the distance between the pixels. The frame is taken from the normal sample video.

clidean distance, demonstrating that vehicles in shadows are not reconstructed well enough. From observations it could be noted that the vehicles outside the shadows where reconstructed as blurs and vehicles inside shadows were not reconstructed at all.

Interestingly, dusk images characterized by intense color shifts were all correctly classified as normal. This suggests that shifts in illumination or color in the frame are less disruptive to training than strong shadows.

To further investigate the impact of strong shadows, a model was trained and tested using only data marked with the sun directly overhead. This specialized model achieved an AUC of 75%, a 4% improvement over the model trained and tested with all available data.

6.3.4 Tiling

Tiling was one of the changes introduced to minimize the effects of illumination changes, following the methodology proposed in the ADOC paper by Pranav et al. [61]. The authors hypothesized that introducing tiling to the method made it more robust to illumination changes. Figure 6.15 shows a tiled frame from the Zejtun dataset, the frame reconstructed and the Euclidean error.

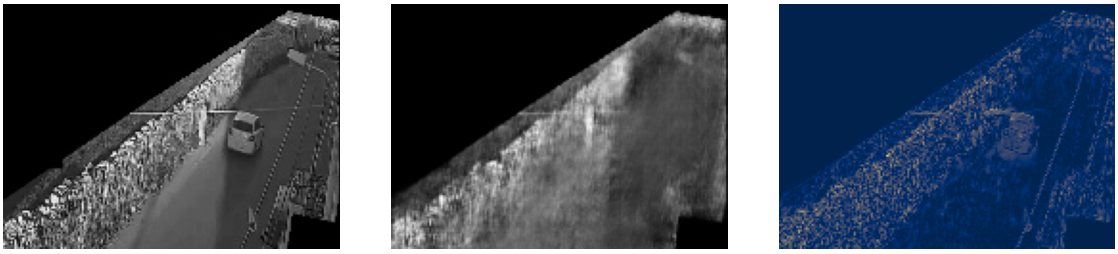
The introduction of tiling did improve the AUC by 2% and reduce the EER by 8% , as seen in Table 6.6 for the Zejtun Dataset. However, further increasing the tiling to have 16 tiles instead of 4 did not result in any significant improvement. It is likely that the tiles were too small to capture the visual features of the vehicles.

The tiling procedure did not improve the results for the Mgarr and Street Scene datasets. In fact, for the Street Scene dataset, tiling reduced the AUC by 7%. The tiling procedure only improved results for datasets with significant shadow and illumination changes, such as the Zejtun dataset and the ADOC dataset [61], as hypothesized by the authors.

6.3.5 Augmentation

Augmentation was another method introduced to increase the robustness of the method to illumination changes and severe strong shadows. The result shows that augmentation improves the result by a further 2% for the Zejtun dataset and the Street Scene dataset. The augmentations were seen to minimally improve the results for the Mgarr dataset as well.

Although the augmentation had a positive effect for all the tested datasets, the augmentation doubles the amount of data used for training. Thus the training



(a) (b) (c)

Figure 6.15 Figure 6.15a is a grayscale tile from a frame with mask from the Zejtun dataset. Figure 6.15b is the reconstructed tile from the model. Figure 6.15c is the visualized Euclidean distance between the pixels in the original frame and the pixels and the reconstructed frame. The more yellow the pixel the larger the distance between the pixels. The frame is taken from the normal sample video. The model used to extract the tiles is from the Zejtun model which obtained 71% AUC

time is doubled. The improvement noted might not be worth the extra computational expense required to train the model with augmented data.

6.3.6 Summary

The study aimed to evaluate anomaly detection methods using frame-level AUC and EER metrics on the Mgarr, Zejtun, and Street Scene datasets. Initial validation on the CUCK Avenue Dataset confirmed improved model performance compared to prior results. Despite varying illumination and anomaly complexities, all datasets demonstrated comparable performance as anticipated from previous traffic dataset experiments.

Masking, significantly enhanced model accuracy by focusing on road reconstruction, resulting in a 4% AUC improvement for the Street Scene dataset. However, maintaining consistent mask usage across datasets is crucial to avoid performance degradation, as observed with mask augmentation in the Zejtun Dataset.

The Zejtun Dataset exhibited the lowest performance due to challenges posed by long shadows in normal videos. Training with overhead sunlight data led to a 4% AUC improvement, highlighting the shadows' impacts on the model performance.

Tiling, following the ADOC methodology, improved the Zejtun Dataset's AUC by 2%, but did not improve the results of the Mgarr and Street Scene datasets, suggesting variable effectiveness across datasets.

Augmentation yielded a 2% improvement in AUC for the Zejtun and Street Scene datasets, with modest gains for Mgarr. However, doubling training data through augmentation raises concerns about cost-effectiveness versus performance benefits.

In conclusion, tailored data preprocessing strategies like masking and tiling are crucial for optimizing anomaly detection models, especially under challenging conditions like heavy shadows. Augmentation further enhances model robustness but requires careful consideration of computational resources.

6.4 Comparing Anomaly detection methods

To perform a direct comparison between the classical and the deep learning anomaly detection methods, a per-event analysis is necessary, focusing on the anomalies directly detected with respect to the video. The previous Section 6.2 was designed to assess the performance of the classical anomaly detection method exclusively, evaluating anomalies detected by vehicles or pedestrians recorded by the object detector and tracker. Now, in Subsection 6.4.1, the results are presented considering all actual anomalies detected in the video, including missed vehicles or pedestrians performing anomalies due to errors in the data acquisition method.

In Section 6.3, the evaluation of the deep learning-based method was presented at the frame level, as it is the most commonly used approach for evaluating the performance of deep learning anomaly detectors. Subsection 6.4.2 subsequently presents the Event level Evaluation.

With these results, a direct comparison is feasible. Subsections 6.4.3 to 6.4.7 discuss the performance of anomaly detectors per anomaly category. Subsection 6.4.8 then examines the ease of implementation of both methods. Finally, Subsection 6.4.9 summarizes all the results.

6.4.1 Classical Anomaly Detection per Event in the Video and Data Acquisition Evaluation

As previously discussed, in order to accurately assess the effectiveness of the classical anomaly detection method, anomalies not captured by the data acquisition process were excluded from the analysis. This validation demonstrated the high performance of the anomaly detector. However, a direct evaluation of the method necessitates that the detected anomalies are compared against those observed in the actual video footage. It is important to emphasize that the data acquisition framework was not developed as part of this study; rather, it was designed to gather data for an existing ITS and provided as such. The focus of this study was to evaluate the data acquisition process specifically in terms of anomaly detection.

For each dataset, anomalies identified in the ground truth from video footage were manually analyzed to determine whether the detector-tracker detected these anomalies. The criteria used for this manual detection process can be found in Section 4.1.1.

This subsection presents the actual anomalies detected with respect to the anomalies in the video clips. For each dataset a discussion on the least detected anomalies is also presented.

Table 6.7, 6.8, and 6.9 display the total anomalies detected in the video by the annotators, the percentage of anomalies recorded by the detector-tracker data acquisition, and the overall percentage of anomalies detected by the classical anomaly detection method. The results are presented for the Mgarr, Zejtun, and Street Scene datasets, respectively.

It can be observed that the data acquisition method loses approximately 32% of anomalies for the Mgarr dataset, 42% for the Zejtun Dataset, and 43% for the Street Scene dataset. Thus, while the anomaly detector performs well in detecting anomalies across all datasets, the actual detection rate concerning anomalies in the video decreases significantly due to the performance of the data acquisition method.

The Mgarr dataset exhibited the best performance for the data acquisition method, with a larger percentage of anomalous events being recorded by the detector-tracker. This superior performance can likely be attributed to the detector being specifically trained for this type of camera angle, as reported by the detector-tracker developers. Notably, no particular anomaly was significantly unrecorded by the detector-tracker, as the entry and exit lines could be positioned close to the edge of the frame, ensuring that all parking vehicles would still intersect the entry and exit lines. The optimal placement of these lines was achievable since there were no significant obstructions in the field of view.

Table 6.7 Anomalies detected with respect to the video for the Mgarr dataset

Anomaly	number of anomalies in the video	percentage recorded by data acquisition	percentage of total anomalies detected with ID switch detection
Crossing Center Line	74	79.73%	68.92%
Overtaking	16	56.25%	31.25%
Jaywalking	86	97.67%	88.37%
Traffic	2	50.00%	50.00%
Bicycle	6	66.67%	33.33%
Parking	23	60.87%	39.13%
Obstruction	33	57.58%	42.42%
Wrong Lane	23	82.61%	52.17%
U-Turn	31	67.74%	64.52%
Scooter	13	84.62%	53.85%
Unforeseen anomalies	5	40.00%	40.00%
All	**312**	**77.88%**	**63.78%**

Table 6.8 Anomalies detected with respect to the video for the Zejtun dataset

Anomaly	number of anomalies in the video	percentage recorded by data aquisition	percentage of total anomalies detected with ID switch detection
Jaywalking	23	91.30%	78.26%
Bicycle	6	83.33%	50.00%
Parking	1	0.00%	0.00%
Obstruction	19	0.00%	0.00%
Wrong Lane	6	83.33%	50.00%
U-Turn	8	37.50%	0.00%
Scooter	2	50.00%	50.00%
Overtaking	8	100.00%	87.50%
All	**73**	**58.90%**	**43.84%**

Table 6.9 Anomalies detected with respect to the video for the Street Scene dataset

Anomaly	number of anomalies in the video	percentage recorded by data aquisition	percentage of total anomalies detected with ID switch detection
Crossing Center Line	47	65.96%	59.57%
Jaywalking	87	80.46%	72.41%
Traffic	54	94.44%	70.37%
Bicycle on wrong lane	33	60.61%	51.52%
Heavy Goods Vehicles	59	5.08%	0.00%
Parking	36	19.44%	16.67%
Obstruction	7	28.57%	14.29%
Wrong Lane	11	81.82%	63.64%
U-Turn	5	20.00%	0.00%
All	**339**	**57.23%**	**47.20%**

The Zejtun dataset exhibited a low percentage of recorded anomalies by the data acquisition method, with particular concern arising from the classes of anomalies that were completely undetected. None of the obstructions or parking maneuvers were perceived by the detector-tracker, mainly because they were visually obscured by the balcony, preventing the detector-tracker from identifying them. Additionally, many U-turns were either interrupted by visual obstructions or failed to enter the correct entry and exit lines, further contributing to their undetection.

The Street Scene dataset recorded the fewest anomalies compared to the other datasets. Parkings, U-turns, and obstructions were poorly recorded by the detector-tracker. This issue often arose from the video ending before vehicles moved out of parking spaces or vehicles failing to enter the valid lines initially. If these anomalies are of interest, the detection-tracking method would require different post-processing techniques. For example, considering a vehicle trajectory as valid if the vehicle stops moving for a period of time could improve detection accuracy.

Furthermore, heavy goods vehicles were another type of anomaly that were poorly recorded by the detector-tracker in the Street Scene dataset tests. Out of the 59 heavy goods vehicles detected by the annotators, only 3 were detected by the detector-tracker. The overhead angle of the Street Scene dataset made it challenging to capture these vehicles accurately. Additionally, heavy goods vehicles were often large and occupied a significant portion of the lane in the frame, resulting in multiple ID switches for the tracker. Ideally, a detector-tracker should be fine-tuned before being applied in scenarios where the viewing angle is severely modified if the initial training was only performed on a particular type of data.

Overall, when an ideal location is utilized with a suitable angle and minimal obstructions, the detector-tracker demonstrates robust performance, as evidenced by the Mgarr dataset. However, for cameras with visual obstructions, the detector-tracker experiences a decrease in performance due to the suboptimal placement of entry and exit lines, as well as the loss of tracks caused by obstructions. So although the data acquisition method was deemed satisfactory in collecting counts and trajectories by its original creators, it did not adequately meet the requirements for anomaly detection.

6.4.2 Deep Learning Anomaly Detection Event level Evaluation

Section 6.3 presented the frame-level metrics for the deep learning evaluation. From these results, it can be noted that the models performed as expected. How-

ever, the commonly used AUC and EER frame-level metrics do not fairly depict the detection of anomalous events. Since the analysis is conducted at the frame level, accurately detecting a longer event would significantly raise the AUC more than a shorter event. Thus, the AUC alone does not indicate whether a specific type of anomaly is being effectively detected. Therefore, event-level analysis was performed for the best-performing model for each dataset in terms of AUC. The optimal threshold for each model was determined by selecting the threshold that yielded the best split between anomalous and normal events from the ROC. Using this threshold, if at least α of the frames were considered anomalous, the frame event was classified as detected. The α value was set to 0.1, consistent with the analysis of the classical anomaly method and inspired by the work of Ramachandra et al. [2]. The results for each dataset are presented in Table 6.10, Table 6.11 and Table 6.12.

The best-performing model for the Mgarr and Zejtun datasets was the model trained by splitting the frame into 4 tiles and utilizing data augmentation. Meanwhile the best-performing models for the Street Scene dataset was using the whole frame as an input and masking the original frame.

For the Mgarr dataset, all events were detected with high accuracy. However, the lowest performing anomaly types were pedestrian crossing and exit anomalies. The exit anomaly likely involves a temporal aspect while the pedestrian anomaly may be affected by its small size or specific location.

In the Zejtun Dataset, the worst-scoring anomalies were bicycles and pedestrian/pedestrian crossing incidents, all of which were relatively small anomalies. It's likely that these anomalies resulted in smaller peaks compared to changes in illumination, making them challenging to detect under the set threshold.

In the Street Scene dataset, the worst-performing anomalies included pedestrian crossing (potentially due to their small size and proximity to vehicles) and obstructions, wrong lane incidents, and parking issues, which involve heavy temporal aspects that could impact detection accuracy. However, further study is needed on these aspects.

Table 6.10 Event level anomaly detection for the Mgarr dataset the deep learning anomaly detection method

anomaly	number of anomalies	percent detected
obstruction	3	100.00%
bus	20	80.00%
bicycle	10	80.00%
u-turns	1	100.00%
exit	44	59.09%
pedestrian crossing	21	57.14%
heavy goods vehicle	21	85.71%
All	**120**	**70.00%**

Table 6.11 Event level anomaly detection for the Zejtun dataset for the deep learning anomaly detection method

anomaly	number of anomalies	percentage detected
obstruction	8	50.00%
scooter	2	0.00%
bus	1	100.00%
bicycle	17	41.18%
u-turns	6	50.00%
pedestrian crossing	52	38.46%
heavy goods vehicle	3	100.00%
All	**89**	**42.70%**

Table 6.12 Event level anomaly detection for the Street Scene Dataset the deep learning anomaly detection method

anomaly	number of anomalies	percentage detected
obstruction	7	85.71%
bicycle on road	33	69.70%
u-turns	5	80.00%
crossing center line	47	68.09%
wrong lane	11	54.55%
pedestrian crossing	87	59.77%
heavy goods vehicle	59	96.61%
traffic	54	94.44%
parking	36	77.78%
All	**339**	**76.40%**

6.4.3 Non-typical vehicle paths

For the classical anomaly detection method, U-turn detections were low for the Zejtun and Street Scene datasets, primarily because the trajectories of U-turns exhibited irregularities in the movement. Additionally, parking detections were often missed because the vehicles did not exit the scene and therefore were not detected. Overall, 60% of the total vehicles not on the expected path were detected.

For the deep learning method, it was expected that this type of anomaly would be challenging to detect since detecting non-typical vehicle paths requires the model to capture and model the temporal features of the vehicle performing the anomaly. Overall, 64% of U-turns, vehicles crossing the center line and exits were detected for deep learning method across the three datasets.

6.4.4 Non-typical location of objects

Overall, this category was well detected by the classical anomaly detector due to the data acquisition detecting vehicles and pedestrians reasonably well in these scenarios and the method designed to detect anomalies also performing well. Overall, 72% of all the anomalies in this category were detected.

For the deep learning category, this section involved the model learning that an object that could be normal in one location is anomalous in another part of the scene. For bicycle on the wrong lane, vehicle on the wrong lane and pedestrians crossing/jaywalking 55% of the events were detected.

6.4.5 Non-typical slow vehicles

For the classical anomaly detection method, the traffic category was consistently well-detected. This success can be attributed to traffic anomalies typically exhibiting low speeds and being fully captured in the data acquisition process. However, parking and obstruction anomalies posed more significant challenges. Parking vehicles often remained stationary and thus were not recorded by the data acquisition method, while obstructions in traffic occurred in areas of the scene that were not fully visible. Additionally, instances of vehicles parking without visibly slowing down were noted. The Zejtun dataset proved to be particularly challenging due to vehicles causing obstructions in parts of the scene that were mostly hidden from the camera view. Overall 44% of the total non typically slow vehicles were detected.

The deep learning anomaly detection method outperformed the classical anomaly detection method specifically for obstructions, traffic, and parking

anomalies, achieving an overall 85% detection rate for this category. The successful detection of anomalies within this category demonstrates that the deep learning model is capable of capturing the slow temporal aspects characteristic of these anomalies.

6.4.6 Non-typical vehicle

For the classical anomaly detection method, this category presented two main challenges. Firstly, it involved detecting less common vehicles such as bicycles and heavy goods vehicles, which the object detector is capable of labeling. However, the object detector had difficulty accurately identifying these vehicles. Secondly, it required detecting vehicles not explicitly labeled buy the object detector but would present themselves as vehicles with unusual behaviors, such as scooters.

Unexpectedly, heavy goods vehicles in the Street Scene dataset were not detected due to the object detector and tracker having issues detecting the large vehicles at overhead angles. Bicycles and scooters were detected more reliably in the Mgarr and Zejtun datasets, though the overall detection performance was unsatisfactory. Interestingly, scooters were detected despite not having specific labels, often being misidentified as pedestrians jaywalking or bicycles. Due to the high number of undetected heavy goods vehicles, this category had a detection rate of only 15%. The deep learning method performed well at detecting non-typical vehicles type anomalies, detecting 83% of all the non typical vehicles.

6.4.7 Unforeseen Occurrences

As previously discussed, anomalous events are rare and unforeseeable, making it impractical to manually select features that can capture all possible anomalies in the context of a classical anomaly detector. Additionally, an anomaly detector may perform well on a dataset collected from past events but fail to capture future anomalies, as they may differ significantly from those observed in the past.

To investigate the classical and deep learning anomalous event detection methods, five unforeseen anomalies were identified by annotators in the Mgarr dataset. These anomalies included two instances of a garbage truck, two instances of a horse cart, and one instance of a vehicle reversing. The classical anomaly detection method detected two out of the five unforeseen anomalies: one instance of the garbage truck and the reversing vehicle. This indicates that the classical anomaly detection method can detect unforeseen anomalies if the data acquisition method captures them and the features of the anomalous event

are addressed. In this case, the detected unforeseen events had unusual entry and exit points, and the vehicles involved drove slowly and caused obstructions.

The five unforeseen occurrences were also processed by the deep learning model trained to detect anomalies in Mgarr. All videos were trimmed so that each video contained only one anomaly. All five anomalies were detected. However, the result was biased because the dataset used for training the Mgarr model was collected about a year before the unforeseen samples were taken. During that time, the road underwent several updates, including additional road markings, a new boundary wall, and the removal of part of the pavement. These new features, which the model had not seen during training, could not be accurately reconstructed, thereby inflating the reconstruction loss and leading to the detection of these anomalies.

To perform a more rigorous test of unforeseen occurrences, the data providers were asked to supply any video of interest that could be considered an anomalous event. One notable occurrence was collected from the Zejtun location, capturing two vehicles in a head-on collision at night. The collision occurred in the corner of the frame. The moment of the collision is shown in Figure 6.16.

To detect the anomaly using the classical data anomaly detection method, the video was first processed with the data acquisition method. The data acquisition did not detect the vehicle entering the entry and exit lines, likely due to blurred vision and suboptimal lighting conditions from the headlights. Consequently, no data was collected from the video, and the classical anomaly detection method could not detect the anomaly.

After preprocessing, the video was input into the best-performing deep learning anomaly detection method trained on the entire Zejtun dataset. Figure 6.17 shows the anomaly score plotted against the frame stacks for the video depicting the collision. The score was normalized relative to all other videos in the dataset. The anomaly score dips when the vehicles collide and come to a stop. However, the score indicates normal behavior when the vehicles were speeding before the crash and after they came to an abrupt stop. No values fell below the threshold, so the entire video was classified as normal. Although the model showed a dip when the anomaly occurred, the calculated threshold was too low to effectively detect the event, likely due to differences in lighting conditions affecting the anomaly scores, pushing the ideal threshold down to lower values.

6.4.8 Ease of Implementation

Overall, the classical and deep learning based methods both demand a thorough comprehension of the traffic scene and the data to effectively design, train, or test

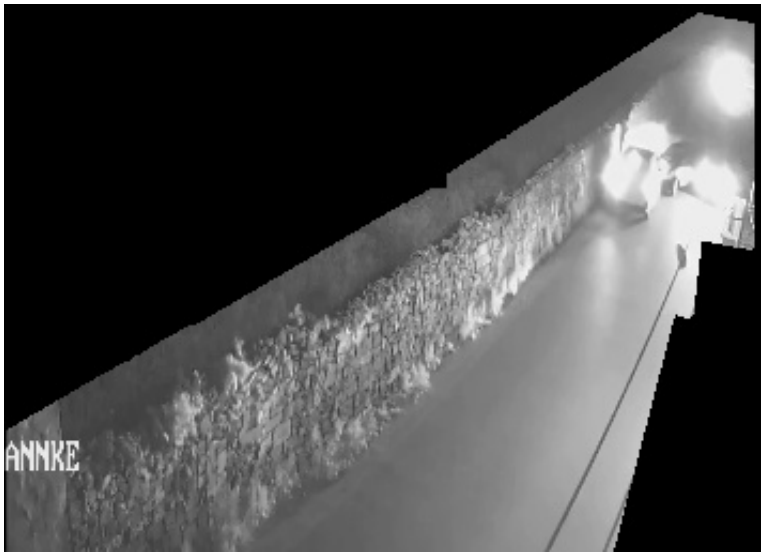


Figure 6.16 Frame taken from a video at the Zejtun location showing ahead on collision between two vehicles.

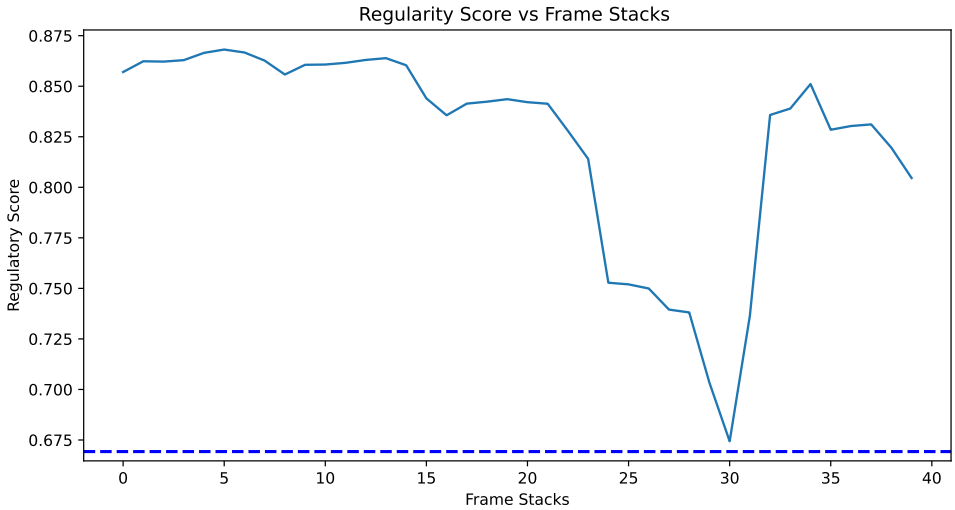


Figure 6.17 Regulatory Score plot for the collision video

anomaly detection methods. This subsection aims to explore factors influencing the preference for one method over another based on the design experience acquired in this study. The ease of implementation was assessed through three criteria: robustness to changes in data acquisition methods and scene dynamics, modularity and debugging capabilities.

The classical data anomaly detection method requires defining two DBSCAN parameters and setting up the data acquisition method, which also necessitates historical data for mask creation. If the data acquisition method is modified for other ITS purposes, these parameters may need adjustment, and new artifacts could be introduced that require analysis and adjustment (e.g., ID switch detection). Consequently, the classical anomaly detection method is closely in-

tertwined with and dependent on the ITS framework. In contrast, the deep learning anomaly detection method operates as a standalone component, requiring only video data as input after training, and will function consistently regardless of changes in the ITS configuration.

However, changes beyond the data acquisition method could also impact the model. For instance, if the scene changes, such as a change in camera angle, the classical anomaly detection method can adapt by resetting the data acquisition process and collecting some historical data before resuming operation. In contrast, a significant scene change for the deep learning anomaly detection method would necessitate retraining the model with new data.

The classical data anomaly detection model also has the advantage of being more modular. If only a couple of categories are interesting in a certain application, the other categories can just be removed. The same cannot be said for the deep learning method; if an anomaly stops being of interest, the model would have to be fine-tuned or retrained with samples of that anomaly in the training set and then redeployed. If the data is available, that would not be a difficult task, albeit less simple than switching off a module.

During deployment, it is likely that some undesirable behavior will be discovered. The classical anomaly detection method is more interpretable and easier to debug because the modules perform specific tasks that can often be visualized. In contrast, the deep learning method operates as a black box, making it difficult to understand its behavior.

In conclusion, both methods offer advantages and challenges, depending on the specific requirements and constraints of the ITS system. A careful consideration of these factors is essential in determining the most suitable approach for anomaly detection in real-world deployments.

6.4.9 Summary

In summary, this study compared two anomaly detection methods using per-event analysis on video data. The analysis considered all anomalies in the video, including those missed due to data errors. Results were categorized by anomaly type and ease of implementation was also examined. Table 6.13 displays a summary of the best performing method per category discussed.

From the evaluation it could be noted that there is no one method that performs best for all types of anomalies detected. Both methods performed relatively similarly for *Non-typical vehicle paths* anomalies. The structured data anomaly detection performed best for *Non-typical location of objects* and the Deep learning method performed best for *Non-typical slow vehicles* and *Non-typical vehicles*. The

Table 6.13 Summary of the best performing anomaly detection method per type of anomaly and ease of implementation. The check mark indicates that that model was deemed to perform the best in that circumstance. A category with no check marks indicates that neither method performed well enough and a category with two check marks indicates that both methods performed comparably.

	Anomaly Detection Method	
Category	**Classical**	**Deep Learning**
Non-typical vehicle paths	✓	✓
Non-typical location of objects	✓	
Non-typical slow vehicles		✓
Non-typical vehicles		✓
Unforeseen Occurrences		
Robustness to changes in data acquisition methods		✓
Robustness to changes in scene dynamics	✓	
Modulrity	✓	
Debugging capabilities	✓	

classical data anomaly detection method's performance was hindered by the limitations of the data acquisition method.

Unforeseen Occurrences anomalies were not detected by either method, or the tests were biased. Overall, this section is challenging to assess because anomalies are, by definition, rare, and unique occurrences are even more so. The evaluation of the deep learning method revealed that background changes could cause unwanted low anomaly scores.

The ease of implementation between the classical anomaly detection method and the deep learning anomaly detection method hinges on understanding the traffic scene and data intricacies. The classical method requires setting DBSCAN parameters and historical data integration, with dependencies on the ITS framework. Changes in data acquisition or scene dynamics may necessitate parameter adjustments. In contrast, the deep learning method operates independently, relying solely on video data post-training but requiring retraining for significant scene changes. The classical approach offers modularity, allowing selective category inclusion, while the deep learning method requires retraining for evolving anomaly interests. During deployment, the classical method's modular nature facilitates understanding and debugging, whereas the deep learning model's poses challenges in interpretability.

Selecting the best anomaly detection method depends on understanding the system's limitations and the specific anomalies and scene dynamics involved. This means recognizing the challenges and constraints within the system setup, such as data acquisition methods and changes in scene dynamics.

6.5 Computational Expense

In this section, the computational expense of the two anomaly detection methods are evaluated by assessing their respective computational time. This analysis aims to provide insights into the practical feasibility and performance considerations of each method within the context of this dissertation. It includes a detailed examination of the processing time required by each method, highlighting key factors that influence computational efficiency and performance. All tests were performed on a 12th Gen Intel(R) Core(TM)i7-12700 with 32.0 GB RAM and NVIDIA GeForce RTX 3080 GPU using Jupyter notebook.

6.5.1 Classical Anomaly Detection

The classical anomaly detection was tested as a batch offline process. For the tests all data was uploaded in a *Pandas* data frame before the start time was recorded. The computational time for each dataset is less than 30 seconds, as shown in Table 6.14. However, the vast majority of this time is consumed by Category 2 steps. Specifically, the Category 2 step accounts for 92.95% of the time for the Street Scene Dataset, 95.89% for the Mgarr dataset, and 96.40% for the Zejtun Dataset.

The Category 2 method involves several complex tasks: clustering trajectories, creating a mask, and identifying trajectories outside the mask. The time taken for these tasks is influenced by the number of trajectories present in the dataset. Moreover, the size of the original frame plays a critical role in the computational effort. The mask is tailored to the video's original resolution to retain all trajectory information, and its creation process involves convolutional filtering and morphological operations, which are more time-consuming for larger images.

Detection of trajectories outside the mask involves summing the pixels within the mask twice and then finding the difference, a process also impacted by image size. Other steps within the method, largely affected by the number of trajectories, typically entail calculating a threshold and identifying trajectories that exceed this threshold.

To explore the relationship between image size and computational time, a plot of computational time versus the number of pixels is presented in Figure 6.18a. The image sizes were scaled to generate the data points on the graph. In the plot, it is observed that the computational time for the Street Scene dataset increases more rapidly compared to the Zejtun and Mgarr datasets. This difference can be attributed to the Street Scene Method employing the Category 2 detection method three times (for jaywalking, bicycles, and vehicles), whereas

Table 6.14 Computational time taken per step per dataset.

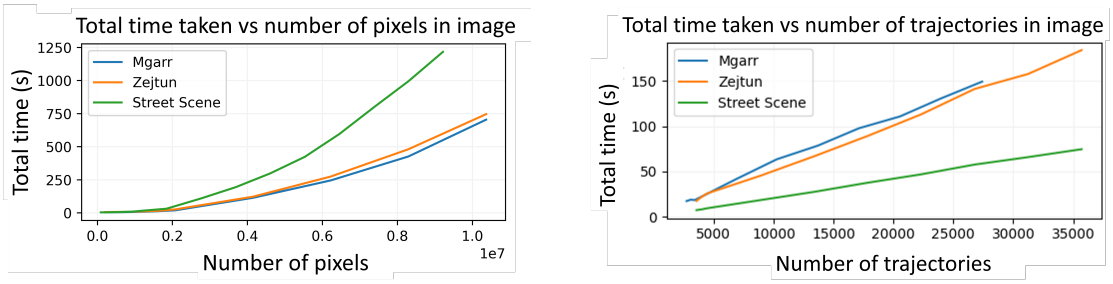
Step	Percentage of total time taken			Description
	Street Scene	Mgarr	Zejtun	
Detector Error	4.94%	2.25%	1.99%	Calculate threshold and mark all trajectories which surpass it for all types of objects
Category 1	2.08%	1.84%	1.61%	Mark entry exit pairs which are anomalous for a vehicle to take
Category 2: Jaywalking - Create mask	6.10%	0.35%	0.19%	Cluster and create mask
Category 2: Jaywalking - Mark Trajectories	33.82%	7.03%	1.60%	Mark pedestrians outside designated area
Category 2: Bicycle not on path - Create mask	1.53%	/	/	Cluster and create mask
Category 2: Bicycle not on path - Mark Trajectories	5.83%	/	/	Mark bicycles outside designated area
Category 2: Vehicle using wrong lane - Create 4 Masks	9.26%	8.87%	11.68%	Cluster and create masks
Category 2: Vehicle using wrong lane - Mark Trajectories	36.42%	79.64%	82.93%	Mark vehicles using wrong lanes
Category 3: Calculate Threshold	0.01%	0.02%	/	Calculate threshold
Category 3: Mark Trajectories	0.01%	0.00%	/	Mark all all trajectories which surpass it for all types of vehicles
Category 4	0.01%	0.00%	0.00%	Mark all all trajectories which are heavy goods vehicles
Total Time taken (s):	**9.6**	**19.3**	**29.7**	

the other two datasets use it only twice. The increased frequency of utilizing the Category 2 method in the Street Scene dataset underscores its impact on computational time.

To investigate the relationship between the number of trajectories and computational time, Figure 6.18b displays a plot of the number of trajectories versus image size. A random sample of trajectories is utilized to test computational time with fewer trajectories. Notably, the smallest feasible sample size is set at 70% of the total trajectories to ensure adequate coverage across potential paths on the road, facilitating comprehensive testing of all processing steps.

Additionally, trajectories are replicated to assess computational time with increased trajectory counts. The plotted data reveals a linear relationship between computational time and the number of trajectories, highlighting the direct impact of trajectory volume on processing demands. This relationship underscores the scalability of computational requirements relative to trajectory quantity, with insights drawn from varying sample sizes and their corresponding computational time metrics. Figure 6.18b offers a visual representation of this relationship, aiding in the analysis of computational efficiency across different trajectory densities and dataset configurations.

The performance of the structured data based method is constrained by



(a) Time taken to process all trajectories vs the number of pixels per dataset

(b) Time taken to process the dataset trajectories vs the number of trajectories

Figure 6.18 Plots of the time taken to process trajectories per dataset

the efficiency of the containerized detector-tracker. To assess the speed of the detector-tracker, all videos from the three datasets were processed.

Table 6.15 presents the total video length per dataset alongside the processing time taken by the detector-tracker, along with resolution, frames per second, detector size, and the number of trajectories per video. The Mgarr dataset exhibited the quickest processing time due to the utilization of a smaller detector size. In this dataset, pedestrians were sufficiently close to the camera, enabling effective detection with a smaller detector size.

Conversely, the Street Scene dataset necessitated a larger detector size due to the overhead angle and greater distance from the pedestrians, rendering them smaller in the frame. Despite the larger detector size, lower frames per second, and image resolution, the detector-tracker efficiently processed the videos in less time than their total length.

In contrast, the Zejtun dataset experienced considerably longer processing times than the video duration, mainly attributed to the larger detector size and the higher number of trajectories requiring detection. The substantial detector size was essential to enhance detection capability amidst significant obstructions within the scene, thereby increasing the likelihood of perceiving partially obstructed figures by the detector-tracker. Thus, this indicates that for complex locations the data acquisition for the classical method could not be used in real time.

This analysis underscores the intricate balance between detector specifications, scene characteristics, and processing efficiency, influencing the overall performance and speed of the detector-tracker within different dataset contexts. The performance of the handcrafted feature-based method is constrained by the efficiency of the containerized detector-tracker.

Table 6.15 Computational time taken to process videos from the dataset with the detector-tracker along with other settings used per dataset

Dataset	Video Length (hh:mm:ss)	Time taken to process (hh:mm:ss)	Resolution (pixels)	Fps	Detector Size (pixels)	Number of Trajectories
Street Scene	03:45:51	03:28:30	720×1280	15	1600	3447
Mgarr	11:15:36	05:17:42	1920×1080	25	800	3426
Zejtun	10:05:40	15:06:41	1920×1080	25	1600	4463

6.5.2 Deep Learning Based Anomaly Detection

During training, the models were trained for a hundred epochs. Two main variants of the model were utilized, one trained with a the whole frame as input and another with the frame tilled into four non overlapping segments. For both models the input and output was a stack of 8 frames (or a stack of 8 tiles). All data was preprocessed prior to model training. The non augmented, versions of the model which used whole frames as input took approximately 3-4 days to train for the datasets. The non augmented versions which took 4 tiles as an input, with a batch size of 4, took approximately 1-2 days for the datasets.

During inference the model requires a frame stack of 8 frames for processing. With loading the 8 frames the model can reach and FPS of 16 fps for the whole frame as input. 1.38GB of memory are required to run inference. If the input is tiled an FPS of 16 fps is also achieved. 1.41GB of memory are required to run inference.

6.5.3 Summary

In summary, the classical anomaly detection method processes entire batches of trajectories in just a few seconds. This rapid processing allows for the swift detection of anomalies once the trajectories are extracted, making it feasible to analyze large datasets of stored trajectory data efficiently. However, the speed of structured data acquisition, which is needed for the classical anomaly detection, is contingent on the quality of the video data; in challenging datasets, such as the Zejtun dataset, real-time processing is not achievable. On the other hand, the deep learning method can achieve a frame rate of 16 FPS, enabling real-time performance during inference. Nonetheless, the model requires 3-4 days of training before inference, which represents a significant computational cost. This analysis provides valuable insights into the trad e-offs between different methods, serving as a guide for selecting the most appropriate approach based on specific dataset characteristics and processing .

7 Conclusion

This chapter serves as the conclusion of the dissertation, reiterating the project objectives, highlighting the achievements made and proposing suggestions for future research directions. Additionally, concluding remarks are provided to summarize the key findings and contributions of the study.

7.1 Achievement of Objectives

In traffic environments similar to that of the Maltese Islands, where large volumes of traffic are constantly experienced, anomalies occurring on narrow, busy urban roads often have ripple effects throughout the entire network. Detecting these anomalies can facilitate improvements in infrastructure design and enhance law enforcement efficiency. However, detecting anomalies in these types of roads is challenging due to the multitude of normal and abnormal events that can occur. This project faced several challenges, including noisy data, imperfect data acquisition, insufficient data coverage, and difficulties in defining all relevant anomalies. Therefore, the aim of this thesis was to investigate anomaly detection methods tailored to the realistic road and data limitations typical of Maltese urban roads, enabling the real-life application of the findings.

The literature review focused on anomaly detection techniques for road traffic data, exploring both classical anomaly detection, which detects anomalies from structured data extracted by a detector-tracker, and deep anomaly detection methods, which detect anomalies directly from videos. Generally, classical anomaly detection methods failed to account for the artifacts created by the data acquisition process and the missed detections, making it difficult to accurately assess their performance in real-life settings. For deep learning methods, the approach involved training on normal data, with anomalies detected due to the high error in reconstructing anomalous events. However, most methods were evaluated on pedestrian datasets, leaving their performance on real traffic streams uncertain.

Both methods were evaluated using two datasets collected from different locations in Malta, as well as a relabeled version of the Street Scene Dataset [2]. For the data in the Malta location a combination of data from the MAVAD [83] and data provided by Greenroads Ltd from the same locations were used. Through discussions with stakeholders, anomalies of interest were defined and categorized into five distinct categories to ensure a comprehensive coverage. This categorization allowed for a thorough evaluation of the methods' effectiveness in detecting

various types of anomalies within the datasets.

The classical method was developed to filter out ID switch artifacts from the output of the object detector and tracker and identify specific anomalies using a combination of filtering, DBSCAN clustering, masking, and rule-based techniques. This approach effectively distinguished between artifacts generated by the object detector and tracker and actual anomalies in the video data. For the deep learning method, an AE model with the STAE [1] architecture was chosen for its ability to capture temporal representation and prior testing on traffic data. This deep learning-based approach involved training models to reconstruct normal events and subsequently fail to reconstruct anomalous events, thereby indicating the presence of anomalies in the road traffic video data.

The classical method demonstrated high reliability in detecting anomalies within structured data, achieving an 82% TPR and a 3% FPR for the Mgarr dataset when using the ID switch detection method. However, the data acquisition method did not accurately capture all anomalies, resulting in a reduced true positive rate for actual video anomalies. When considering anomalies not recorded by the detector-tracker (and thus not present in the structured data), the TPR of anomaly detection was observed to be as low as 63% for the Mgarr dataset.

The deep learning method exhibited strong performance across all datasets, achieving an 83% AUC and a 25% EER for the Mgarr dataset. The lowest performance was observed for the Zejtun dataset, which was most affected by heavy shadows. Experiments concluded that tiling the frame into four tiles and augmenting the dataset prior to training improved performance when dealing with shadows. Masking irrelevant regions also helped improve performance in one of the datasets.

An event-level comparison was performed to directly compare both methods. Both methods performed similarly in detecting *non-typical vehicle paths* based on event-level analysis. The classical method was found to be better at detecting *non-typical locations* of objects and was also more robust against changes to scene dynamics, more modular, and easier to debug. The deep learning method excelled at detecting *non-typical slow vehicles* and *non-typical vehicles* and was more robust against changes in the data acquisition method within the ITS. Neither method proved effective in detecting *unforeseen anomalies*.

In conclusion, this thesis successfully achieved its aims by developing and evaluating effective anomaly detection methods tailored to the realistic road and data limitations typical of Maltese urban roads, providing valuable insights for real-life applications.

7.2 Future Work

Throughout this dissertation, it has become evident that data availability and quality are significant obstacles. To achieve substantial advancements in the field of anomaly detection, it is crucial to improve the datasets available for such studies. One notable limitation identified is the lack of differentiation between anomalies detected in structured data and those originating from video data. The literature review highlighted that many existing methods fail to distinguish between these two types of anomalies. Often, the focus is on anomalies within structured data without adequately addressing the data acquisition process from videos or considering potential artifacts introduced during this process. This oversight makes it challenging to perform analyses similar to those conducted in this dissertation, where the classical method was designed to evaluate actual anomalies caused by vehicles while ignoring artifacts created during data acquisition.

When dealing with datasets for both methods, it was noted that per-event anomaly detection labeling was lacking, resulting in limited information about the specific types of anomalies detectable by each method. To enable this analysis, future datasets should be labeled with event names, similar to the approach taken in the MAVAD dataset [83] and the ADOC dataset [61].

This dissertation also demonstrated that while an object detection and tracking method may be suitable for other elements within an ITS, it should also be evaluated using an anomaly dataset. The data acquisition method might achieve high detection and tracking rates under normal conditions but could fail in anomalous situations, potentially missing micro-traffic events crucial for gaining a deeper understanding of traffic systems. Future research could benefit from using a state-of-the-art detector such as YOLOv8 [99] which offers improved performance. Additionally, employing the DeepSORT [18] algorithm, the most commonly used tracking algorithm for visual traffic applications [10], could enhance tracking accuracy. Utilizing more advanced data acquisition methods is expected to improve performance, but it would also be valuable to evaluate these state-of-the-art methods in challenging anomalous situations. Additionally, camera calibration techniques can also be utilized to obtain precise distance and speed data which should increase anomaly detection performance as shown in Yu et al. [19] and Giannakeris et al. [22].

The deep learning model struggled with cases involving harsh shadows, demonstrating that these methods are not robust under all environmental conditions and weather scenarios. To address this, it is essential to record large datasets throughout the year to account for seasonal and lighting variations for training and evaluation.

Future work should also explore the incorporation of multimodal data into model training to further improve robustness to environmental changes. For example, the MAVAD dataset [83] includes audio, which could be integrated into the model, as shown in Leporowski et al. [98]. Additionally, some methods have used the output of an object detector output and optical flow when training the deep learning models [72]. Since the output from object detectors is less affected by environmental changes, such as shadows, including this data could lead to more robust performance under varying conditions.

7.3 Concluding Remarks

In conclusion, this study aims to bridge the gap between existing anomaly detection methods in literature and their practical applications in the Maltese traffic landscape. Through this dissertation, a data-driven approach was employed to label and acquire relevant data, design and evaluate anomaly detection methods, and identify optimal solutions per type of anomaly of interest. By addressing these challenges, this research contributes to advancing the effectiveness and real-world implementation of anomaly detection technologies, thereby narrowing the gap between theory and practice in traffic surveillance.

References

[1] Y. Zhao, B. Deng, C. Shen, Y. Liu, H. Lu, and X.-S. Hua, "Spatio-temporal autoencoder for video anomaly detection," in *Proceedings of the 25th ACM International Conference on Multimedia*, ser. MM '17, Mountain View, California, USA: Association for Computing Machinery, 2017, pp. 1933–1941, ISBN: 9781450349062. DOI: 10.1145/3123266.3123451. [Online]. Available: https://doi.org/10.1145/3123266.3123451.

[2] B. Ramachandra and M. J. Jones, "Street scene: A new dataset and evaluation protocol for video anomaly detection," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, Feb. 2020, pp. 2569–2578. DOI: 10.1109/WACV45572.2020.9093457. [Online]. Available: https://www.merl.com/publications/TR2020-017.

[3] European Commission - Directorate-General for Mobility and Transport, "Road safety: 20,640 people died in a road crash last year – progress remains too slow," CARE (Community Road Accident) database, Tech. Rep., Oct. 2023, Tech. Rep.

[4] University of Malta, Faculty for Social Wellbeing, "A study about 'the perceived effect of traffic on our wellbeing' amongst the maltese population," *University of Malta Faculty for Social Wellbeing*, 2023.

[5] K. K. Santhosh, D. P. Dogra, and P. P. Roy, "Anomaly detection in road traffic using visual surveillance," *ACM Computing Surveys*, vol. 53, no. 6, pp. 1–26, Nov. 2021, ISSN: 1557-7341. DOI: 10.1145/3417989. [Online]. Available: http://dx.doi.org/10.1145/3417989.

[6] O. Che Puan, N. S. Muhamad Nor, N. Mashros, and M. R. Hainin, "Applicability of an automatic pneumatic–tube–based traffic counting device for collecting data under mixed traffic," *IOP Conference Series: Earth and Environmental Science*, vol. 365, p. 012 032, Nov. 2019. DOI: 10.1088/1755-1315/365/1/012032.

[7] L. Li, R. Jiang, Z. He, X. Chen, and X. Zhou, "Trajectory data-based traffic flow studies: A revisit," *Transportation Research Part C: Emerging Technologies*, vol. 114, pp. 225–240, 2020, ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2020.02.016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X19312987.

[8] C. Wang, A. Musaev, P. Sheini, and T. Atkison, "Towards detection of abnormal vehicle behavior using traffic cameras," in Jun. 2019, pp. 125–136, ISBN: 978-3-030-23550-5. DOI: 10.1007/978-3-030-23551-2_9.

[9] E. P. Ijjina, D. Chand, S. Gupta, and K. Goutham, "Computer vision-based accident detection in traffic surveillance," in *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 2019, pp. 1–6. DOI: 10.1109/ICCCNT45670.2019.8944469.

[10] X. Zhang, Y. Feng, P. Angeloudis, and Y. Demiris, "Monocular visual traffic surveillance: A review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 14 148–14 165, 2022. DOI: 10.1109/TITS.2022.3147770.

[11] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi, "Traffic monitoring and accident detection at intersections," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 2, pp. 108–118, 2000. DOI: 10.1109/6979.880968.

[12] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask r-cnn," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, 2017.

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[14] A. Jazwinski, "Adaptive filtering," *Automatica*, vol. 5, no. 4, pp. 475–485, 1969, ISSN: 0005-1098. DOI: https://doi.org/10.1016/0005-1098(69)90109-5. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0005109869901095.

[15] J. Nascimento, A. Abrantes, and J. Marques, "An algorithm for centroid-based tracking of moving objects," in *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, vol. 6, 1999, 3305–3308 vol.6. DOI: 10.1109/ICASSP.1999.757548.

[16] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, Mar. 2015. DOI: 10.1109/tpami.2014.2345390. [Online]. Available: https://doi.org/10.1109%2Ftpami.2014.2345390.

[17] F. Gao, J. Li, Y. Ge, J. Shao, S. Lu, and L. Weng, "A trajectory evaluator by sub-tracks for detecting vot-based anomalous trajectory," *ACM Trans. Knowl. Discov. Data*, vol. 16, no. 4, Jan. 2022, ISSN: 1556-4681. DOI: 10.1145/3490032. [Online]. Available: https://doi.org/10.1145/3490032.

[18] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” in *2017 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2017, pp. 3645–3649. DOI: 10.1109/ICIP.2017.8296962.

[19] L. Yu, D. Zhang, X. Chen, and A. Hauptmann, “Traffic danger recognition with surveillance cameras without training data,” Nov. 2018, pp. 1–6. DOI: 10.1109/AVSS.2018.8639166.

[20] Y. Zhao, W. Wu, Y. He, Y. Li, X. Tan, and S. Chen, “Practices and a strong baseline for traffic anomaly detection,” Jun. 2021, pp. 3988–3996. DOI: 10.1109/CVPRW53098.2021.00450.

[21] P. Batapati, D. Tran, W. Sheng, M. Liu, and R. Zeng, “Video analysis for traffic anomaly detection using support vector machines,” in *Proceeding of the 11th World Congress on Intelligent Control and Automation*, 2014, pp. 5500–5505. DOI: 10.1109/WCICA.2014.7053655.

[22] P. Giannakeris, V. Kaltsa, K. Avgerinakis, A. Briassouli, S. Vrochidis, and I. Kompatsiaris, “Speed estimation and abnormality detection from surveillance cameras,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018, pp. 93–936. DOI: 10.1109/CVPRW.2018.00020.

[23] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28, Curran Associates, Inc., 2015. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf.

[24] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988. DOI: 10.1109/ICCV.2017.322.

[25] Z. Cai and N. Vasconcelos, “Cascade r-cnn: Delving into high quality object detection,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2018, pp. 6154–6162. DOI: 10.1109/CVPR.2018.00644. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPR.2018.00644.

[26] J. Redmon and A. Farhadi, *Yolov3: An incremental improvement*, 2018. DOI: 10.48550/ARXIV.1804.02767. [Online]. Available: https://arxiv.org/abs/1804.02767.

[27] J. Chen *et al.*, "Dual-modality vehicle anomaly detection via bilateral trajectory tracing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, Jun. 2021, pp. 4016–4025.

[28] G. Jocher *et al.*, *ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation*, version v7.0, Nov. 2022. DOI: 10.5281/zenodo.7347926. [Online]. Available: https://doi.org/10.5281/zenodo.7347926.

[29] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, Mar. 1960, ISSN: 0021-9223. DOI: 10.1115/1.3662552. [Online]. Available: https://doi.org/10.1115/1.3662552.

[30] L. Song, F. Jiang, Z. Shi, R. Molina, and A. K. Katsaggelos, "Toward dynamic scene understanding by hierarchical motion pattern mining," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 3, pp. 1273–1285, 2014. DOI: 10.1109/TITS.2014.2299403.

[31] H. Jeong, Y. Yoo, K. M. Yi, and J. Y. Choi, "Two-stage online inference model for traffic pattern analysis and anomaly detection," *Mach. Vision Appl.*, vol. 25, no. 6, pp. 1501–1517, Aug. 2014, ISSN: 0932-8092. DOI: 10.1007/s00138-014-0629-y. [Online]. Available: https://doi.org/10.1007/s00138-014-0629-y.

[32] N. Anjum and A. Cavallaro, "Multifeature object trajectory clustering for video analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1555–1564, 2008. DOI: 10.1109/TCSVT.2008.2005603.

[33] C. Piciarelli, C. Micheloni, and G. L. Foresti, "Trajectory-based anomalous event detection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 11, pp. 1544–1554, 2008. DOI: 10.1109/TCSVT.2008.2005599.

[34] F. Jiang, Y. Wu, and A. K. Katsaggelos, "A dynamic hierarchical clustering method for trajectory-based unusual video event detection," *IEEE Transactions on Image Processing*, vol. 18, no. 4, pp. 907–913, 2009. DOI: 10.1109/TIP.2008.2012070.

[35] B. T. Morris and M. M. Trivedi, "Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 11, pp. 2287–2301, 2011. DOI: 10.1109/TPAMI.2011.64.

[36] F. Jiang, J. Yuan, S. A. Tsaftaris, and A. K. Katsaggelos, "Anomalous video event detection using spatiotemporal context," *Computer Vision and Image Understanding*, vol. 115, no. 3, pp. 323–333, 2011, Special issue on Feature-Oriented Image and Video Computing for Extracting Contexts and Semantics, ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2010.10.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1077314210002390.

[37] A. Saggese, L. Brun, and M. Vento, "Dynamic scene understanding for behavior analysis based on string kernels," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, Feb. 2014. DOI: 10.1109/TCSVT.2014.2302521.

[38] Y. Cai, H. Wang, X. Chen, and H.-b. Jiang, "Trajectory-based anomalous behaviour detection for intelligent traffic surveillance," *Iet Intelligent Transport Systems*, vol. 9, pp. 810–816, 2015.

[39] X. Wang, K. T. Ma, G.-W. Ng, and W. E. L. Grimson, "Trajectory analysis and semantic region modeling using a nonparametric bayesian model," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8. DOI: 10.1109/CVPR.2008.4587718.

[40] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, Jul. 2009. DOI: 10.1145/1541880.1541882.

[41] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982. DOI: 10.1109/TIT.1982.1056489.

[42] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96, Portland, Oregon: AAAI Press, 1996, pp. 226–231.

[43] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: A partition-and-group framework," in *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '07, Beijing, China: Association for Computing Machinery, 2007, pp. 593–604. DOI: 10.1145/1247480.1247546. [Online]. Available: https://doi.org/10.1145/1247480.1247546.

[44] L. Rabiner and B. Juang, "An introduction to hidden markov models," *IEEE ASSP Magazine*, vol. 3, no. 1, pp. 4–16, 1986. DOI: 10.1109/MASSP.1986.1165342.

[45] D. Blei, L. Carin, and D. Dunson, "Probabilistic topic models: A focus on graphical model design and applications to document and image analysis," *IEEE signal processing magazine*, vol. 27, pp. 55–65, Nov. 2010. DOI: 10.1109/MSP.2010.938079.

[46] S. Chandrakala, K. Deepak, and G. Revathy, "Anomaly detection in surveillance videos: A thematic taxonomy of deep models, review and performance analysis," *Artificial Intelligence Review*, Aug. 2022. DOI: 10.1007/s10462-022-10258-6.

[47] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[48] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, "Learning temporal regularity in video sequences," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 733–742. DOI: 10.1109/CVPR.2016.86.

[49] W. Luo, W. Liu, and S. Gao, "Remembering history with convolutional lstm for anomaly detection," in *2017 IEEE International Conference on Multimedia and Expo (ICME)*, 2017, pp. 439–444. DOI: 10.1109/ICME.2017.8019325.

[50] Y. S. Chong and Y. H. Tay, "Abnormal event detection in videos using spatiotemporal autoencoder," in *Advances in Neural Networks - ISNN 2017*, F. Cong, A. Leung, and Q. Wei, Eds., Cham: Springer International Publishing, 2017, pp. 189–196, ISBN: 978-3-319-59081-3.

[51] K. Deepak, S. Chandrakala, and C. Mohan, "Residual spatiotemporal autoencoder for unsupervised video anomaly detection," *Signal, Image and Video Processing*, vol. 15, Feb. 2021. DOI: 10.1007/s11760-020-01740-1.

[52] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[53] W. Luo, W. Liu, and S. Gao, "A revisit of sparse coding based anomaly detection in stacked rnn framework," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 341–349. DOI: 10.1109/ICCV.2017.45.

[54] H. Park, J. Noh, and B. Ham, "Learning memory-guided normality for anomaly detection," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA: IEEE Computer Society, Jun. 2020, pp. 14 360–14 369. DOI: 10.1109/CVPR42600.2020.01438. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/CVPR42600.2020.01438.

[55] D. Gong, L. Liu, V. Le, B. Saha, M. R. Mansour, S. Venkatesh, and A. Van Den Hengel, "Memorizing normality to detect anomaly: Memory- augmented deep autoencoder for unsupervised anomaly detection," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 1705–1714. DOI: 10.1109/ICCV.2019.00179.

[56] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds., Cham: Springer International Publishing, 2015, pp. 234–241, ISBN: 978-3-319-24574-4.

[57] D. Singh and C. K. Mohan, "Deep Spatio-Temporal Representation for Detection of Road Accident using Stacked Autoencoder," *IEEE Transactions on Intelligent Transportation Systems (T-ITS)*, 2018. DOI: 10.1109/TITS.2018.2835308.

[58] D. Abati, A. Porrello, S. Calderara, and R. Cucchiara, "Latent space autoregression for novelty detection," Jun. 2019, pp. 481–490. DOI: 10.1109/CVPR.2019.00057.

[59] J. R. Medel and A. Savakis, *Anomaly detection in video using predictive convolutional long short-term memory networks*, 2016. DOI: 10.48550/ARXIV.1612.00390. [Online]. Available: https://arxiv.org/abs/1612.00390.

[60] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial networks*, 2014. DOI: 10.48550/ARXIV.1406.2661. [Online]. Available: https://arxiv.org/abs/1406.2661.

[61] M. Pranav, L. Zhenggang, and S. S. K, "A day on campus - an anomaly detection dataset for events in a single camera," in *Proceedings of the Asian Conference on Computer Vision (ACCV)*, Nov. 2020.

[62] M. Yan, X. Jiang, and J. Yuan, "3d convolutional generative adversarial networks for detecting temporal irregularities in videos," in *2018 24th International Conference on Pattern Recognition (ICPR)*, 2018, pp. 2522–2527. DOI: 10.1109/ICPR.2018.8546039.

[63] F. Dong, Y. Zhang, and X. Nie, "Dual discriminator generative adversarial network for video anomaly detection," *IEEE Access*, vol. PP, pp. 1–1, May 2020. DOI: 10.1109/ACCESS.2020.2993373.

[64] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5967–5976. DOI: 10.1109/CVPR.2017.632.

[65] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. v. d. Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2758–2766. DOI: 10.1109/ICCV.2015.316.

[66] W. Liu, D. L. W. Luo, and S. Gao, "Future frame prediction for anomaly detection – a new baseline," in *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[67] J. Rocca and B. Rocca, *Understanding variational autoencoders (vaes)*, Published in Towards Data Science, Accessed on 2023-01, Sep. 2019. [Online]. Available: https://towardsdatascience.com/understanding-variation%20al-autoencoders-vaes-f70510919f73.

[68] M. Xu, X. Yu, D. Chen, C. Wu, and Y. Jiang, "An efficient anomaly detection system for crowded scenes using variational autoencoders," *Applied Sciences*, vol. 9, no. 16, pp. 3337–, 2019. DOI: 10.3390/APP9163337.

[69] Y. Lu, K. M. Kumar, S. s. Nabavi, and Y. Wang, "Future frame prediction using convolutional vrnn for anomaly detection," in *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2019, pp. 1–8. DOI: 10.1109/AVSS.2019.8909850.

[70] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, *Adversarial autoencoders*, 2015. DOI: 10.48550/ARXIV.1511.05644. [Online]. Available: https://arxiv.org/abs/1511.05644.

[71] S. Lee, H. G. Kim, and Y. M. Ro, *Stan: Spatio-temporal adversarial networks for abnormal event detection*, 2018. DOI: 10.48550/ARXIV.1804.08381. [Online]. Available: https://arxiv.org/abs/1804.08381.

[72] M. I. Georgescu, R. Ionescu, F. S. Khan, M. Popescu, and M. Shah, "A backgro und-agnostic framework with adversarial training for abnormal event detection in video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021. DOI: 10.1109/tpami.2021.3074805. [Online]. Available: https://doi.org/10.1109%2Ftpami.2021.3074805.

[73] P. Liu, M. Lyu, I. King, and J. Xu, "Selflow: Self-supervised learning of optical flow," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4566–4575. DOI: 10.1109/CVPR.2019.00470.

[74] M. Sabokrou, M. Fayyaz, M. Fathy, and R. Klette, "Deep-cascade: Cascading 3d deep neural networks for fast anomaly detection and localization in crowded scenes," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1992–2004, 2017. DOI: 10.1109/TIP.2017.2670780.

[75] K. Pawar and V. Attar, "Deep learning model based on cascaded autoencoders and one-class learning for detection and localization of anomalies from surveillance videos," *IET Biometrics*, Jan. 2022. DOI: https://doi.org/10.1049/bme2.12064.

[76] A. B. Chan and N. Vasconcelos, "Modeling, clustering, and segmenting video with mixtures of dynamic textures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 5, pp. 909–926, May 2008, ISSN: 0162-8828. DOI: 10.1109/TPAMI.2007.70738. [Online]. Available: https://doi.org/10.1109/TPAMI.2007.70738.

[77] C. Lu, J. Shi, and J. Jia, "Abnormal event detection at 150 fps in matlab," 2013.

[78] M. Sabokrou, M. Khalooei, M. Fathy, and E. Adeli, "Adversarially learned one-class classifier for novelty detection," Jun. 2018, pp. 3379–3388. DOI: 10.1109/CVPR.2018.00356.

[79] I. Tyler Technologies, *Next generation simulation (ngsim) vehicle trajectories and supporting data*, Aug. 2018. [Online]. Available: https://www.opendata%20network.com/dataset/datahub.transportation.gov/8ect-6jqj.

[80] A. P. Shah, J. Lamare, T. Nguyen-Anh, and A. Hauptmann, "Cadp: A novel dataset for cctv traffic camera based accident analysis," in *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Nov. 2018, pp. 1–9. DOI: 10.1109/AVSS.2018.8639160.

[81] M. Naphade *et al.*, "The 5th ai city challenge," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, Jun. 2021.

[82] W. Sultani, C. Chen, and M. Shah, *Real-world anomaly detection in surveillance videos*, 2018. DOI: 10.48550/ARXIV.1801.04264. [Online]. Available: https://arxiv.org/abs/1801.04264.

[83] N. Bonnici, A. Muscat, B. Leporowski, A. Bakhtiarnia, L. Zanella, Y. Wang, and A. Iosifidis, *MARVEL - Malta Audio Visual Anomaly Dataset (MAVAD)*, Zenodo, Jun. 2023. DOI: 10.5281/zenodo.7950008. [Online]. Available: https://doi.org/10.5281/zenodo.7950008.

[84] R. Mehran, A. Oyama, and M. Shah, "Abnormal crowd behavior detection using social force model," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 935–942. DOI: 10.1109/CVPR.2009.5206641.

[85] A. Adam, E. Rivlin, I. Shimshoni, and D. Reinitz, "Robust real-time unusual event detection using multiple fixed-location monitors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 3, pp. 555–560, 2008. DOI: 10.1109/TPAMI.2007.70825.

[86] R. Leyva, V. Sanchez, and C.-T. Li, "The lv dataset: A realistic surveillance video dataset for abnormal event detection," in *2017 5th International Workshop on Biometrics and Forensics (IWBF)*, 2017, pp. 1–6. DOI: 10.1109/IWBF.2017.7935096.

[87] A. Acsintoae, A. Florescu, M.-I. Georgescu, T. Mare, P. Sumedrea, R. T. Ionescu, F. S. Khan, and M. Shah, "Ubnormal: New benchmark for supervised open-set video anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 20 143–20 153.

[88] A. Bochkovskiy, C.-Y. Wang, and H.-y. Liao, "Yolov4: Optimal speed and accuracy of object detection," Apr. 2020.

[89] Z. Luo, F.B.Charron, C.Lemaire, J.Konrad, A. S.Li, A. Achkar, J. Eichel, and P.-M. Jodoin, "Mio-tcd: A new benchmark dataset for vehicle classification and localization," *IEEE Transactions on Image Processing*, 2018.

[90] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Cham: Springer International Publishing, 2014, pp. 740–755, ISBN: 978-3-319-10602-1.

[91] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2544–2550. DOI: 10.1109/CVPR.2010.5539960.

[92] R. Artstein, "Inter-annotator agreement," 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:64918638.

[93] A. Viera and J. Garrett, "Understanding interobserver agreement: The kappa statistic," *Family medicine*, vol. 37, pp. 360–3, Jun. 2005.

[94] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Los Alamitos, CA, USA: IEEE Computer Society, Dec. 2015, pp. 4489–4497. DOI: 10.1109/ICCV.2015.510. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/ICCV.2015.510.

[95] G. Varol, I. Laptev, and C. Schmid, "Long-term temporal convolutions for action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 06, pp. 1510–1517, Jun. 2018, ISSN: 1939-3539. DOI: 10.1109/TPAMI.2017.2712608.

[96] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. 61, pp. 2121–2159, 2011. [Online]. Available: http://jmlr.org/papers/v12/duchi11a.html.

[97] X. Zhang, S. Yang, X. Zhang, W. Zhang, and J. Zhang, *Anomaly detection and localization in crowded scenes by motion-field shape description and similarity-based statistical learning*, 2018. DOI: 10.48550/ARXIV.1805.10620. [Online]. Available: https://arxiv.org/abs/1805.10620.

[98] B. Leporowski, A. Bakhtiarnia, N. Bonnici, A. Muscat, L. Zanella, Y. Wang, and A. Iosifidis, "Audio-visual dataset and method for anomaly detection in traffic videos," 2023. arXiv: 2305.15084 [cs.CV].

[99] G. Jocher, A. Chaurasia, and J. Qiu, *Ultralytics yolov8*, version 8.0.0, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics.

Appendix A Annotator Instructions for Anomalous Trajectory Labelling

This annotation task is not complex and requires no special software installations. You should have access to a folder of videos. Some of the the videos are Matroska Video files (MKV) files hence *VLC media player* will be the best tool to watch the videos. If the media player or an equivalent is not already installed you can find download instructions here [1]. You should also have access to a sample Google Sheet file with a template to write your annotations.

The task entails finding anomalous events in videos and writing the time of occurrence in the provided Google Sheet file. Since this is an anomalous trajectory detection problem the start and end time of an anomalous event are defined as the point where a vehicle or pedestrian entered the video field of view. The anomalous events that you should look out for are seen below in Table A.1. For annotations follow the detailed step by step instructions.

A.1 Detailed step by step instructions

1. Open the first video
2. Write its filename in the first column on the provided google sheet. Copy the EXACT name of the video file

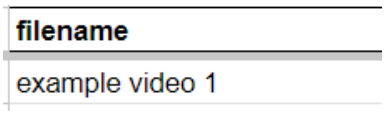


Figure A.1 Example of filename

3. Start watching the video and pause anytime you spot an anomalous event listed in the table
4. Write the anomalous label in the next column

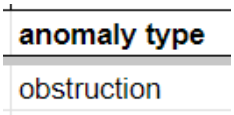


Figure A.2 Example of anomaly labeling

5. Rewind the video to find the exact entry time of the vehicle performing the anomalous event and write it in the next column

[1] https://www.videolan.org/vlc/

start timestamp	e
02:32	

Figure A.3 Example of start time labeling

6. Continue watching the video till the end of the anomalous event.
7. Pause the video and write the exact time the vehicle who performed the anomalous event exited the frame in the corresponding column

end timestamp	v
05:40	c

Figure A.4 Example of end time labeling

8. Finally write the type of vehicle that performed this anomalous event

vehicle type
car

Figure A.5 Example of vehicle type labeling

9. Continue watching the video
10. If another anomalous event is spotted in the same video, go to the next line in the sheet write the file name in the first column, and repeat steps 2-8 for this anomalous event
11. In no other anomalous event is found in the video you can move on to the next video in the folder and repeat steps 1-8 for that video
12. If you come across any video with no anomalous event, still write its name as in the excel sheet, and fill in the rows as follows

3	filename	anomaly type	start timestamp	end timestamp	vehicle type
6	example video 2	normal	/	/	/

Figure A.6 Example of annotation for normal video

13. If the video has any noise that make it impossible to view the video still write its name as in the excel sheet, and fill in the rows as follows

filename	anomaly type	start timestamp	end timestamp	vehicle type
example video 3	video has artifact	/	/	/

Figure A.7 Example of annotation for noisy video

Any uncertainties whilst annotating should be reported immediately to the task leader.

Table A.1 List of anomalies annotators should lookout for

Anomaly	Description
Crossing Center Line	Any vehicle crossing the center line between lanes, if there is one
U-turn	Vehicle performing a U turn on the road (A three point turn or any other weird turn is included in this section)
Overtaking	Any examples of vehicles overtaking other vehicles
Pedestrian on the road or Jaywalking	Any pedestrian on the road
Bicycle on wrong lane	Bicycle not using dedicated lane, if a dedicated lane exists
Vehicle on wrong lane	Any example of a vehicle driving in the wrong lane, even if partially
Traffic	Any que of cars at low speed or stationary
Obstructions	Any vehicle, pedestrian or object causing an obstruction on the road or parked in the incorrect space. Obstructions typically cause vehicles to stear around them
Parking	Any vehicle parking should be marked
Heavy Goods Vehicles Bicycles Scooters	Any occurrences of these types of vehicles on screen. Bicycles are only anomalous if the dataset is recorded in Malta.
Unforeseen Anomalies	This section encompases any major anomaly that could not be forseen and might be observed by annotators. An example of this is the horse cart example in certain datasets.