REPUBLIC OF TURKEY

ALTINBAS UNIVERSITY

Institute of Graduate Studies

Electrical and Computer Engineering

DETECTION OF VEHICLE ON INFRARED IMAGES IN ROAD TRAFFIC

Ashraf Ashour Rajab ABDULQADIR

Master of Science

Supervisor

Prof. Dr. Galip CANSEVER

Istanbul, 2022

DETECTION OF VEHICLE ON INFRARED IMAGES IN ROAD TRAFFIC

Ashraf Ashour Rajab ABDULQADIR

Electrical and Computer Engineering

Master of Science

ALTINBAŞ UNIVERSITY

2022

The thesis titled DETECTION OF VEHICLE ON INFRARED IMAGES IN ROAD TRAFFIC prepared ASHRAF ASHOUR RAJAB ABDULQADIR and submitted on 16/02/2022 has been **accepted unanimously by of votes** for the degree of Master of Science in Information Technologies.

_____
Prof. Dr. Galip CANSEVER

Supervisor

Thesis Defense Jury Members:

Prof. Dr. Galip CANSEVER	Faculty Of Engineering and Architecture, Altinbas University	_____
Prof. Dr. Sefer KURNAZ	Faculty Of Engineering and Architecture, Altinbas University	_____
Prof. Dr. Serdar KARGIN	Faculty Of Engineering and Architecture, Beykent University	_____

I hereby declare that this thesis meets all format and submission requirements of a Master of Science

Accepted by Altınbaş University Institute of Graduate Studies on the following date:

____/____/____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Ashraf Ashour Rajab ABDULQADIR

Signature

DEDICATION

I devote and pledge this research work to my supervisor who is salient for guiding me through whole research work as well as my family for always assisting me in my hard time.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my supervisor Asst. Prof. Dr. Galip CANSEVER for guiding and helping me along the way in writing this dissertation. Discussing my progress, problems, and ideas with my supervisor Asst. Prof. Dr. Galip CANSEVER a couple of times every week helped me tremendously in understanding the logic behind the research. It made me better realize the technical need for this research work.

ABSTRACT

DETECTION OF VEHICLE ON INFRARED IMAGES IN ROAD TRAFFIC

ABDULQADIR, Ashraf Ashour Rajab,

M.Sc., Electrical and Computer Engineering, Altınbaş University,

Supervisor: Prof. Dr. Galip CANSEVER

Date: February /2022

Pages: 74

The aim of this novel master's Thesis is to detect the vehicles in infrared images using deep learning based Convolutional Neural Network (CNN) algorithm. Infrared image data localization serves as one of the most widely used real-world applications in fields like toll control, traffic scene analysis, and suspected vehicle tracking. Along with vehicle data information, to obtain overall comprehension, the information of the vehicle also plays a great role, and contextual information is defined as the relationship between the vehicles using automated deep learning technique in in our work. We proposed a CNN classifier for simultaneously localizing the region of vehicle data and vehicles' position. The classifier, rather than bounding rectangles, gives bounding quadrilaterals, which gives a more precise indication for vehicle data localization. The adjusted regions are preprocessed to remove noise and passed to the Convolutional neural network (CNN). The preprocessing of the CNN extracts connected components, filters them by size, and groups them into lines of infrared image. The employed CNN is the well-known technology. If the CNN reported a low confidence score for at least one infrared image region, the user is signaled to acquire a new image of that part of the data, which is then matched to the existing image and the infrared image processing steps are executed again. The process completes, if all regions have a sufficiently high score or a fixed number of retries are exhausted. The output consists of the detected data type, the list of regions including the extracted infrared

image and the CNN reliability through CNN with an accuracy of 98.69% with 300 epochs of training and testing. The proposed system has been trained on 70% of data while 20% is used for testing and remaining 10% for validation. To show the significance of the designed feature vector, we used an open source dataset consisting of multiple infrared image classes which represented a subset of all existing types of vehicles. The work has performed in MATLAB programming language with several toolboxes of deep learning were being used for this purpose. There are three functional head networks in our design, and thus the end-to-end and simultaneous training leads to a potential training stability with 70% of data was being utilized for training while rest of the 30% data for testing and validation. If the model fails to converge, it is quite hard to trace which head design is raising a problem. Thus, in our design process, we added each the functional head step-by-step, making sure the single head design idea is working correctly and then expanded the model with the rest functional heads with infrared images.

KEYWORDS: Vehicle, CNN, detection, infrared, deep learning.

TABLE OF CONTENTS

Pages

ABSTRACT ..... **vii**

LIST OF TABLES ..... **xi**

LIST OF FIGURES ..... **xii**

LIST OF ABBREVIATIONS ..... **xiv**

1. INTRODUCTION **1**

1.1. OVERVIEW 1

1.2. MOTIVATION 2

1.3. PROBLEM STATEMENT 3

1.4. RESEARCH CONTRIBUTIONS 4

1.5. THESIS ORGANIZATION 5

2. RELATED WORK **7**

2.1. LITERATURE REVIEW 7

2.2. VEHICLE INFORMATION EXTRACTION 11

2.3. HOUGH TRANSFORMATION 13

2.4. SCANNING THERMAL IMAGES 14

2.5. AUTOMATED FORM RECOGNITION 17

2.5.1. Image Reliability 19

2.5.2. Image Quality 19

2.5.3. Vehicle Data Accumulation 20

2.5.4. Noise Removal Approach 21

2.5.5. Image Rotation and Approximation 22

2.6. DESCRIPTION AND MODIFICATION OF SEGMENTATION (DMOS) 22

3. METHODOLOGY **25**

3.1. VEHICLE DATA DETECTION 26

3.1.1. Feature Extraction 28
3.1.2. Data Localization with Local Binary Patterns 29
3.1.3. Normalization with Feature Vectors 30
3.1.4. Geometrical Analysis 30
3.2. CONVOLUTIONAL NEURAL NETWORK (CNN) 31
3.3. VEHICLE DETECTION PIPELINE 33
3.4. DATASET DESCRIPTION 35
3.5. VEHICLE SEGMENTATION AND BINARIZATION 36
3.6. EXTRACTING THE VEHICLE DATA 38
4. RESULTS 41
5. DISCUSSION 46
6. CONCLUSION 50
6.1. FURTHER WORK 51
REFERENCES 53

LIST OF TABLES

Pages

Table 3.1: Grid search results for the feature extraction parameters. For each parameter, the possible values and the final value is listed. 29

Table 5.1: The comparison of accuracy of recognition with existing and proposed techniques for vehicle data recognition. 48

LIST OF FIGURES

Pages

Figure 1.1: Convolutional Neural Network Schema [10]. 3

Figure 2.1: Robust vehicle detection under various weather condition using the extremal regions in infrared images [15]. 8

Figure 2.2: Vehicle data detection using acquired image is detected, the letters on the country of origin sticker are found as well [21]. 9

Figure 2.3: Vehicle data recognition using visual attention model for vehicles [30]. 12

Figure 2.4: Hough transformation being applied on the two lane road [39]. 14

Figure 2.5: The segmentation of heat areas on the vehicles with individual circles marking periodically [40]. 15

Figure 2.6: The market segmentation for potential targeting of vehicle data detection system [60]. 24

Figure 3.1: The processing on unprocessed raw image. 27

Figure 3.2: The proposed automatic vehicle data recognition system pipeline. 28

Figure 3.3: The employment of CNN with proposed architecture for robust vehicle data detection. 31

Figure 3.4: The novel use of convolutional neural network based pipeline for vehicle data detection. 32

Figure 3.5: The CNN model training and validation with data prediction for labels. 33

Figure 3.6: The CNN simplified scheme of the framework for tracking. 34

Figure 3.7: A sample of thermal infrared image from dataset. 36

Figure 3.8: The graph illustrates the vehicle detection and non-detection with false positive..... 37

Figure 3.9: A graph illustrates the undetected vehicles with false positive..... 38

Figure 3.10: A breakdown cases for extracting the recognition for number of data at the same time. 39

Figure 4.1: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 1..... 42

Figure 4.2: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 2..... 42

Figure 4.3: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 3..... 43

Figure 4.4: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 4..... 43

Figure 4.5: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 5..... 44

Figure 4.6: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 6..... 44

ABBREVIATIONS

AUC	:	Area Under Consideration
CAD	:	Computer Aided Design
CNN	:	Convolutional Neural Network
DCNN	:	Deep Convolutional Neural Network
CSER		Category-Specific Extremal Regions
LBP	:	Local binary patterns
MSER	:	Maximally Stable Extremal Regions
SVM	:	Support Vector Machine
CNN	:	Convolutional neural network
RNN	:	Recurrent Neural Network
ROI	:	Region of Interest

1. INTRODUCTION

1.1. OVERVIEW

Our modern society highly depends on electric power. Therefore, a huge network of electric power supplying devices, such as power pylons, circuit breakers, and transformers, is spread all over the world. To ensure the constant availability of electric power, the underlying infrastructure has to undergo regular servicing as mentioned in [1]. Therefore, attached vehicles provide important information for all kinds of maintenance routines, as well as documentation purposes. The content, which is present on these vehicles, usually consists of textual information, such as serial numbers, manufacturing year, or vendor names, as well as schematic drawings, logos, or even barcodes as mentioned in [2]. In most cases, vehicles are of rectangular shape, and made of metal or synthetic material. These data differ in size, layout, the used font and material, as they originate from different vendors. The first ex- ample vehicle, consists of aluminum, has a tarnished surface, and contains a barcode amongst the human readable infrared image. The data are made of synthetic material, has a polished and reflective surface, contains schematic drawings, and is significantly larger than the first data as mentioned in [3]. Up to now, the information on the vehicle is transcribed manually by the operator during maintenance tasks. This transcription is prone to errors, such as typing errors or omission of important information, which might result in severe damage of the respective device, or even trigger blackouts as mentioned in [4]. Furthermore, this process is quite time consuming because of the complexity of most vehicles. The examples of vehicles are recent production samples. As such, they are new, were never mounted to a device, and therefore are in pristine condition. In contrast, metallic vehicles that have spent a few decades outdoors, mounted on a device. These infrared images illustrate some of the problems that complicate automated content extraction, as well as manual transcriptions as mentioned in [5]. The vehicles are exposed to environmental conditions for decades, and therefore become weathered and stained. On very old vehicles it is even possible that the paint chips off, rendering the infrared image virtually unreadable. Further issues arise from the outdoor location, and therefore are likely to include strong shadows, reflections, illumination changes and partial occlusions, that may become problematic when capturing infrared images for automatic processing. An

example of this can be seen in the data, which contains a specular reflection at the top as mentioned in [6].

In order to extract the semantic information, which is present on the vehicle, one could convert infrared images of the patterns into machine readable patterns. There already exist a number of approaches that aim at solving this task, as well as readily available *CNN* software implementations as mentioned in [7]. However, most of the available systems are designed and trained to process scanned sheets of paper. Considering this, the four presented vehicles show another difficulty, namely each of the four data has a different font that differs from the ones used in text processing as mentioned in [8]. Moreover, one data contains text that is embossed and therefore has a smaller stroke width and lower contrast than painted or printed text. Hence, the existing *CNN* solutions will most likely generate unsatisfactory results.

They use Maximally Stable Extremal Regions (MSER) to detect the vehicle data in the input image. The proposed algorithm uses bright regions surrounded by a dark boundary (MSER+) to detect the vehicle data, while dark regions with a bright boundary (MSER-) are used to detect the characters on the data. The *MSER* output of three sample vehicle data is displayed in this work. A region is detected as a vehicle data if a number of MSER- regions are found inside a larger MSER+ region. Furthermore, the dark regions must be of approximately the same size and their center points must lie on a straight line. The average height of the dark regions must also be approximately the height of the surrounding bright region as mentioned in [9]. The research use Convolutional neural network (CNN) with the one-vs-one strategy to perform vehicle detection, where the inputs of the classifier are the individual characters represented by the MSER- regions. Since they also track the vehicle data over a number of frames, multiple infrared images of the data are available, which are used to improve the classification rate. The final detection result of the vehicle data text are the best individual character results from a majority vote over all frames.

1.2. MOTIVATION

IT provides us provides an overview of several related areas of research. Since no other algorithms have been proposed that exactly match our goal, a number of similar tasks are examined, especially the detection and text extraction from vehicle data and vehicle data,

as well as the detection of objects in scanned infrared images and the classification of forms. In general, these algorithms consist of the detection of data-like objects, followed by the extraction of information contained in the form of human readable text, similar to what we are aiming at. The new image, which may only contain a part of the data, is matched to the existing one and the *CNN* is executed again. This multi-view approach ensures robust text extraction when occlusions or reflections are present. The final output of the algorithm consists of the detected data type, the recognized text, its position on the data, as well as its confidence as mentioned in [10]. The goal of the practical task is to create a prototype of a deep learning application that allows the user to acquire infrared images, on which the previously listed steps to extract the content are performed.

The main focus of the evaluation is placed on the data classification, where we test the features, extracted from the data, individually and combined. Further, we show the impact of training the *CNN* specifically for the fonts used on the data, as well as the applicability of the CNN application on three different data types. Finally, we show the influence of different lighting conditions on the data classification and text extraction stages. This includes a number of extreme cases that lead to failure.

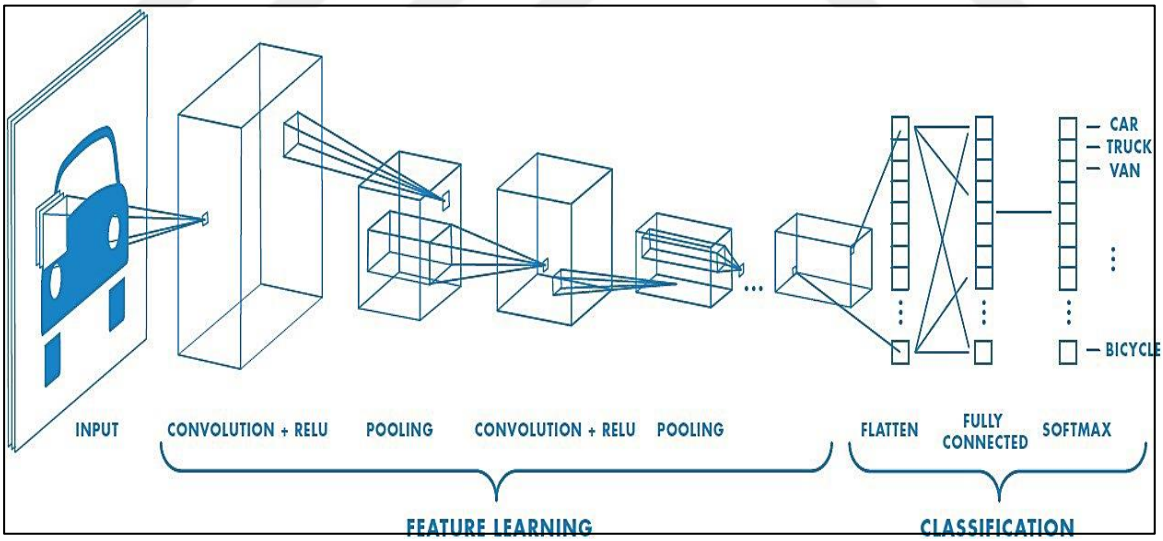

Figure 1.1: Convolutional Neural Network Schema [10]

1.3. PROBLEM STATEMENT

The task of detecting and extracting content from vehicle data exhibits a number of

similarities to our task of vehicle detection and classification. Vehicle data have a rectangular shape and contain human readable text. Therefore, to extract their contents it is necessary to locate them in the image and an Convolutional neural network (CNN) is applied to the individual characters. We propose a system to detect and track vehicle data for traffic management as mentioned in [11]. This research proposes a novel algorithm to detect, classify, and extract text from vehicles. More specifically, vehicles that are attached to electric power supply equipment, with the requirement that the vehicles are rectangular and contain human readable text. Schematic drawings and barcodes are not processed. Our approach consists of the following steps: First, the vehicle is located in the input image, extracted, and warped into an up-right position. Then, we extract features and determine the type using a machine learning approach. For each type, a list that contains the designations and positions as mentioned in [12]. Each region is preprocessed to remove noise and clutter, caused by dirt, parts of neighboring text, or frames around the text. Finally, the filtered regions are input into an CNN engine. If the recognition score of one or more regions is below a threshold, the user is instructed to acquire a new image of the data region where the problem occurred.

1.4. RESEARCH CONTRIBUTIONS

The research contributions of this dissertation are given by:

i. The research aims to propose a novel approach to detect and recognize vehicles in natural infrared images. The proposed deep learning based CNN algorithm enumerates all extremal regions by thresholding the image at each gray level, starting at level one.

ii. For each intensity level in infrared images, connected components are extracted that represent the extremal regions and descriptors are calculated using convolutional neural network.

iii. The research note that there are only three possible ways that existing extremal regions change in infrared images: they grow larger, two previously separated one's merge, or a new region appears.

iv. This research is preceded by incremental update behavior is exploited by choosing CNN that are also incrementally computable for vehicle detection, which keeps the runtime complexity low on infrared images.

v. The research proposes the use of normalized central algebraic moments, compactness, Euler number (the number of holes in the objects subtracted from the number of objects), entropy of the cumulative histogram, number of convexities, and the area of the convex hull (although the last two descriptors are not incrementally computable).

vi. The deep learning based CNN descriptors takes the input image at each intensity level into a classifier whose output indicates whether the region is of interest or not. This means that rotated or slanted characters must be incorporated into the training samples, to be detected.

vii. It should be noted, that this approach detects text in natural infrared images. Therefore, to only obtain vehicle data, a separate grouping and filtering step must be employed.

viii. The CNN detection pipeline and an example result are processed on infrared images. In addition to the vehicle data, the characters of the country of origin sticker above the vehicle data are detected as well. Furthermore, this approach is not only limited to text, but may also be used to detect other vehicles.

1.5. THESIS ORGANIZATION

The chapter 1 presents the introduction to the deep learning approaches have in common is that they detect approximately rectangular objects and, with the exception of the scanned objects section, extract information printed on them in the form of human readable text, which is then stored or handed on for further processing. In chapter 2, the literature review is presented with detection step of these methods either detects the contents and exploits the simple structure to group them together to locate the objects, or further constrains such as uniform background or foreground color must be fulfilled. In chapter 3, the research methodology is presented with the case of processing forms, CNN detection is needed at all, but only the type of the form must be classified. Furthermore, the CNN format of the

content is usually known and a limited set of characters and fonts are used, which simplifies the content extraction and convolutional neural network. In chapter 4, the experimental results are presented with a notable exception are presented for vehicle data recognition where a large number of fonts may be used. This however also causes issues in the examined approaches. Forms are also often filled by hand, but vehicle data recognition is a field of its own and such cases are not covered in the examined research. In chapter 5, the comparison with existing literature is presented since these approaches exploit a number of specific properties of the objects for which they are designed, which are not guaranteed to be valid in our case, none of these complete approaches can be used as it is. Finally, in chapter 6, the conclusion of proposed research method to detect and classify vehicles, utilizing individual steps of some of the examined methods were presented along with future recommendation

2. RELATED WORK

This thesis work is motivated by the goal of overcoming the challenges and difficulties that vehicle data detection units in CNN with the help deep learning techniques in real time with their structure and the environmental hostility surrounds them, that is by creating a better environment solution, a solution that will avoid the shortcomings of the traditional networks for vehicle data detection and controlling operations. For the sake of explaining the motivation of this thesis work, a detailed review of all these difficulties and constraints is discussed in depth in the sections below; furthermore, a comprehensive review of CNN system and their advantages that this thesis work aims to overcome.

2.1. LITERATURE REVIEW

The task of detecting and extracting content from vehicle data exhibits a number of similarities to our task of vehicle detection and classification. Vehicle data have a rectangular shape and contain human readable text. Therefore, to extract their contents it is necessary to locate them in the image and a Convolutional Neural Network (CNN) is applied to the individual characters. Researchers in [13] propose a system to detect and track vehicle data for traffic management. They use Maximally Stable Extremal Regions (MSER) to detect the vehicle data in the input image. The proposed algorithm uses bright regions surrounded by a dark boundary (MSER+) to detect the vehicle data, while dark regions with a bright boundary (MSER-) are used to detect the characters on the data. The MSER output of three sample vehicle data is displayed. A region is detected as a vehicle data if a number of MSER- regions are found inside a larger MSER+ region. Furthermore, the dark regions must be of approximately the same size and their center points must lie on a straight line as mentioned in [14]. The average height of the dark regions must also be approximately the height of the surrounding bright region. The authors use Support Vector Machines (SVM) with the one-vs-one strategy to perform vehicle detection, where the inputs of the classifier are the individual characters represented by the MSER-regions as mentioned in [15]. Since they also track the vehicle data over a number of frames, multiple infrared images of the data are available, which are used to improve the classification rate. The final detection result of the vehicle data text are the best individual

character results from a majority vote over all frames.

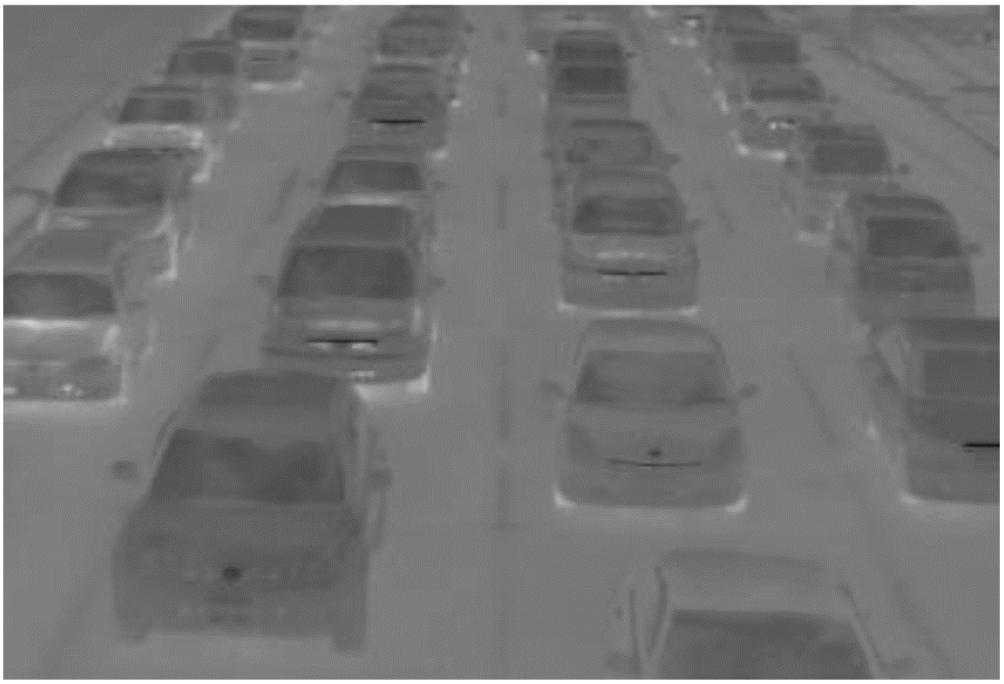

Figure 2.1: Robust vehicle detection under various weather condition using the extremal regions in infrared images [15].

Researchers in [16] propose a similar approach to detect text in natural images. However, they use extremal regions, a superset of MSER, to define Category-Specific Extremal Regions (CSER). The proposed algorithm enumerates all extremal regions by thresholding the image at each gray level, starting at level one as mentioned in [17]. For each intensity level, connected components are extracted that represent the extremal regions and descriptors are calculated. The research note that there are only three possible ways that existing extremal regions change: they grow larger, two previously separated one's merge, or a new region appears as mentioned in [18]. This incremental update behavior is exploited by choosing descriptors that are also incrementally computable, which keeps the runtime complexity low. They propose the use of normalized central algebraic moments, compactness, Euler number (the number of holes in the objects subtracted from the number of objects), entropy of the cumulative histogram, number of convexities, and the area of the convex hull (although the last two descriptors are not incrementally computable) as mentioned in [19]. These descriptors are then input at each intensity level into a classifier whose output indicates whether the region is of interest or not. This means that rotated or

slanted characters must be incorporated into the training samples, to be detected as mentioned in [20]. It should be noted, that this approach detects text in natural infrared images. Therefore, to only obtain vehicle data, a separate grouping and filtering step must be employed as mentioned in [21]. The *CSER* detection pipeline and an example result. In addition to the vehicle data, the characters of the country of origin sticker above the vehicle data are detected as well. Furthermore, this approach is not only limited to text, but may also be used to detect other objects.

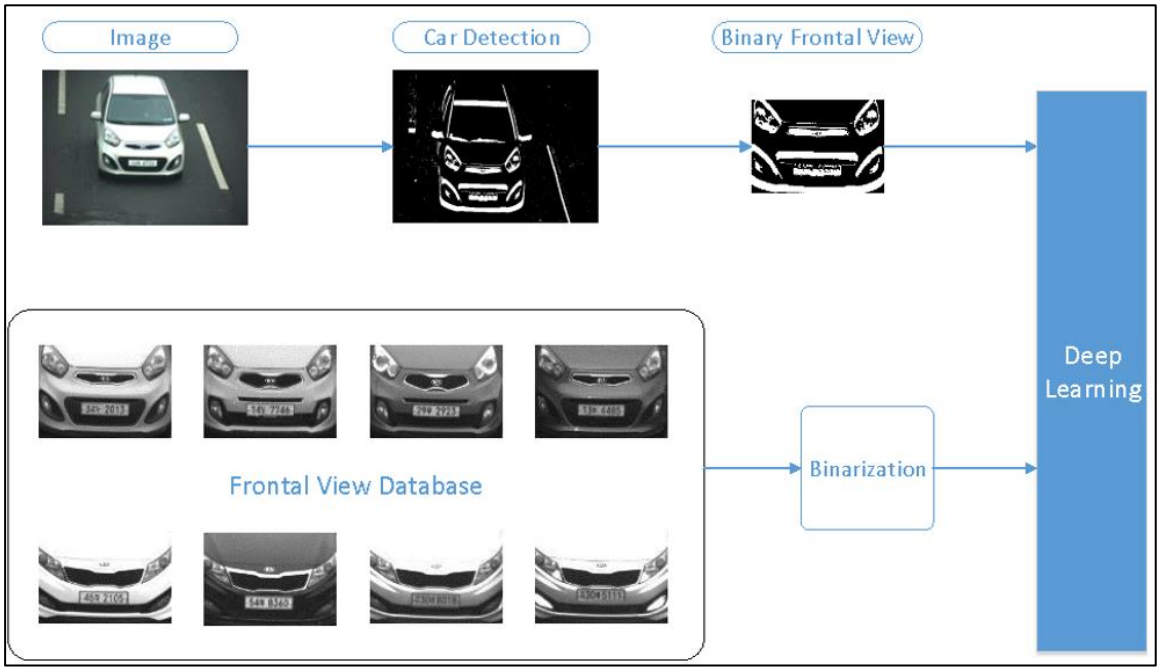

Figure 2.2: Vehicle data detection using acquired image is detected, the letters on the country-of-origin sticker are found as well [21].

For example, in [22], the authors also explore how to detect traffic signs. In contrast to the previous methods, researchers in [23] propose an approach that uses the colors of the data. This consists of two steps, vehicle data location and vehicle number identification. The authors define that only the four colors white, black, red, and green are used on the vehicle data that will be detected. Hence, they require a color image as input. They propose a color edge detector that responds only to edges caused by specific color combinations, which results in very few edges in sections of the image that are not part of a vehicle data. First, the image is converted into the Hue, Saturation and Intensity (HSI) color space and the edge map is calculated. Fuzzy maps that model specific color transitions and encode the probability that a region contains a vehicle data are generated from each plane of the image,

as well as the edge map. Given that vehicle data contain a number of repetitive edges with high gradient magnitude, the maps are combined into a single map, which is used to perform the detection step. To extract the vehicle text, adaptive thresholding is performed and connected components are extracted and filtered by their aspect ratio as mentioned in [24]. The centers of the connected components must form a straight line, connected components that deviate are removed. Since the number of characters on supported vehicle data is exactly eight, connected components are removed, starting from the smallest one, until that number is reached or a significant jump in size occurs. Should the number be lower, the algorithm tries to find new characters starting from the outermost ones in direction of the known characters. The set of detected characters is also checked against the format of the vehicle text as mentioned in [25]. To perform the vehicle detection, they first separate numerical and alphanumerical characters using the known format of the vehicle text. Next, they perform topological sorting using the number of holes and the node types as features. This is done to decrease the number of features to which the candidate must be compared to in the next and final step. Each character is compared to the remaining features and the best one is determined via a self-organizing recognition model as mentioned in [26]. A neural network is used, where the weights for the nodes are derived from the input character. The features are then input into the network and after it stabilizes, the sum of weight changes in the network is used to derive a measure of similarity.

In summary, the examined approaches to vehicle data detection first detect the data in the input image, extract it and employ convolutional neural network to extract the vehicle text. In contrast to the expected input infrared images for our task, these infrared images are very cluttered, with only a small portion of the image being occupied by the actual vehicle data. Furthermore, the layout of a vehicle data is much simpler as mentioned in [27]. It consists of at most two lines of text and the set and diversity of characters as well as the number of fonts is limited. Since the examined approaches rely on the known layout and (in part) on the colors and vehicle text format, they do not fit our task of vehicle detection, although individual steps can be adapted to extract text from the vehicles.

2.2.VEHICLE INFORMATION EXTRACTION

Another related topic is the detection and text extraction from data vehicle data as they are also rectangular and contain information in the form of human readable text. In contrast to vehicle data there is no common layout and a variety of fonts are used, but the content usually consists of a name and address information. There exist a number of papers that cover the detection and text extraction specifically tailored to mobile devices. Researchers in [28] propose a system to extract the contact information from vehicle data on mobile devices with limited processing power. The image from the built-in camera is first converted to grayscale and scaled down to one-fourth of the width and height, which reduces the computational- and memory requirements. Next, the edge map is extracted using the Canny edge detector and the skew angle is calculated on the scaled down image. Since vehicle data usually only contain non overlapping rectangles of text, everything that does not fulfill this criterion is discarded as mentioned in [29]. This yields the location of the business card, by thresholding. To perform the text recognition, the authors propose a CNN based algorithm that uses a two stage classifier, where the first stage is used to limit the number of possible data features in infrared images. The features used for classification of each character are the width/height ratio, as well as the number of foreground and background pixels of the input divided into a 4×4 grid. The second stage additionally incorporates the average distance of the outermost foreground pixels to the borders, and the number of foreground pixels in a 4×4 grid in four directions as mentioned in [30]. If the classification score is low, the algorithm tries to split the input into multiple characters, assuming they were not properly separated during preprocessing. The points where the characters are divided, are determined by analyzing the positions of the extreme points of the character contour. Finally, possible recognition errors are corrected during post-processing.

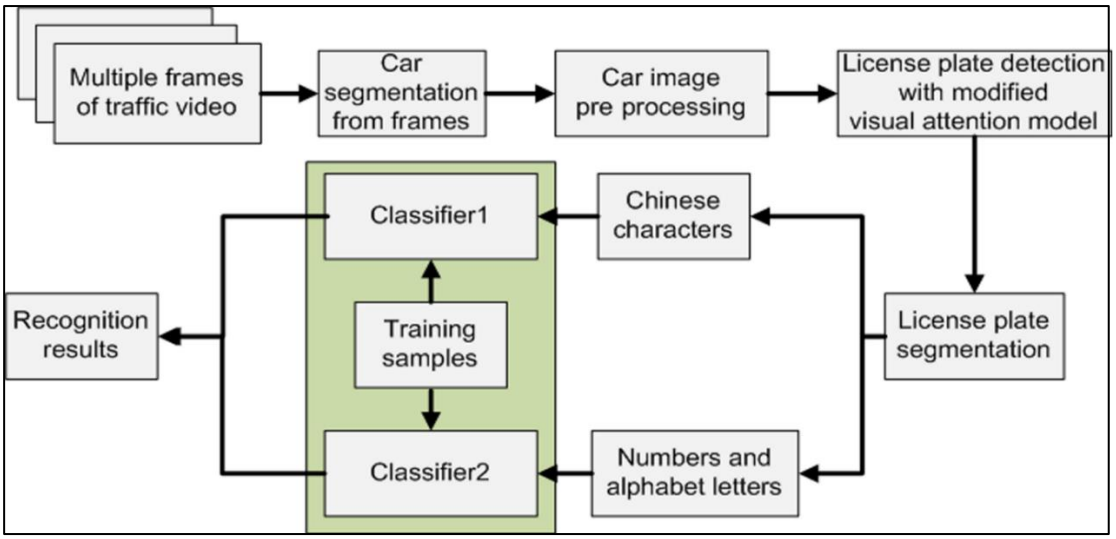

Figure 2.3: Vehicle data recognition using visual attention model for vehicles [30].

An approach similar to the previous one is proposed by researchers in [31], which also targets computationally limited mobile devices, although they only cover the segmentation of the individual characters. The input image is separated into non overlapping cells, for each one the variance is calculated and used to remove the background. Next, everything that is not recognized as text is discarded. This includes graphics, lines, and noise. From the remaining structures generated by text, connected components are extracted. The research note that multiple distorted or narrowly placed lines of text may merge and result in a single connected component, whereas one well-spaced line of text is represented by one connected component. The skew angle for the text lines is determined by calculating the distance of the first pixel at the bottom of the connected component, to an axis parallel line as mentioned in [32]. Skew angles are calculated at the borders and the middle of the connected component. Should they differ too much, the procedure is repeated, although the pixels from the top of the connected component are used. The final skew angle is the mean of the three values. To separate the up-right connected components into individual lines, the region is thresholded and a horizontal histogram is calculated as mentioned in [33]. The region is split at approximately same sized intervals, as indicated by the histogram. To separate a line into characters, the same approach is employed by calculating and analyzing a vertical histogram, however this means that italic or cursive text cannot be segmented properly as mentioned in [34]. The result consists of the separated characters of the individual lines, which may then be input into an CNN.

2.3. HOUGH TRANSFORMATION

Researchers in [35] propose a Hough transform based vehicle license plate reader that runs on CNN devices. In contrast to the other solutions, they do not develop their own CNN, but use Tesseract instead, which is also our approach. First, they apply adaptive threshold to the image and then a number of morphological operations, which results in white regions inside the business card and black regions outside. To detect the boundaries of the card, another morphological operation is applied and the Hough transform [36] of the threshold difference of the eroded and dilated infrared images is computed. They use the strongest peaks in the Hough accumulator to detect the four boundary lines, with the constraint that all corners, calculated by intersecting the lines, must be inside the image. Should all detected lines violate the constraint, the input image is used as it is, and the perspective transform later on is omitted. The authors note that often vehicle data do not have a uniform background but instead contain graphics or color gradients, which can lead to wrong outline detection results as mentioned in [37]. To detect and correct such cases, they further analyze the text flow. The bounding box that contains all black pixels of the card is split into six bars and in each one, the angle of the text flow is calculated. The angles of the outermost bars should be very similar to the angle of the corresponding boundary lines of the card. Should they differ too much, the text flow angle takes precedence. The authors note that this is only possible for the horizontal lines because vehicle data usually do not contain vertical text, which could be used to calculate the angle for the vertical lines. The preprocessing is completed by warping the image into an up-right position, which removes the background. The preprocessing steps for a sample vehicle data as mentioned in [38]. The extracted image is adaptively threshold again and used if the number of black pixels' changes significantly compared to first threshold result. Otherwise the previous result is used. To improve *CNN* performance, different text blocks are separated by splitting the image into two parts at corridors that are at least twenty pixels wide and contain black pixels at each side. This is done by sweeping the image at different angles starting from different points originating from the top line of the road as mentioned in [39].

Figure 2.4: Hough transformation being applied on the two-lane road [39].

The considered approaches for business card readers primarily rely on the uniform color of the vehicle data to segment them. Furthermore, the amount as well as the content of the text, present on vehicle data, is limited to names and address information, which allows the usage of dictionaries or text correction to limit CNN errors. The size of vehicle data is also restricted as they must fit into wallets, which also keeps them in mint condition. However, a large number of fonts and font faces may be used. Unfortunately, some of the examined approaches just fail with cursive and italic input.

2.4.SCANNING THERMAL IMAGES

The approaches considered in this section aim at scanning rectangular objects in preview scans, so that multiple vehicles may be put onto the scanner, but each one is automatically stored into an individual file as mentioned in [40]. This topic only relates to our first step, namely the data detection and extraction. Since these approaches consider overlapping and approximately rectangular objects, they are relevant because the data we need to detect are usually not perfectly rectangular or not well separated from the background.

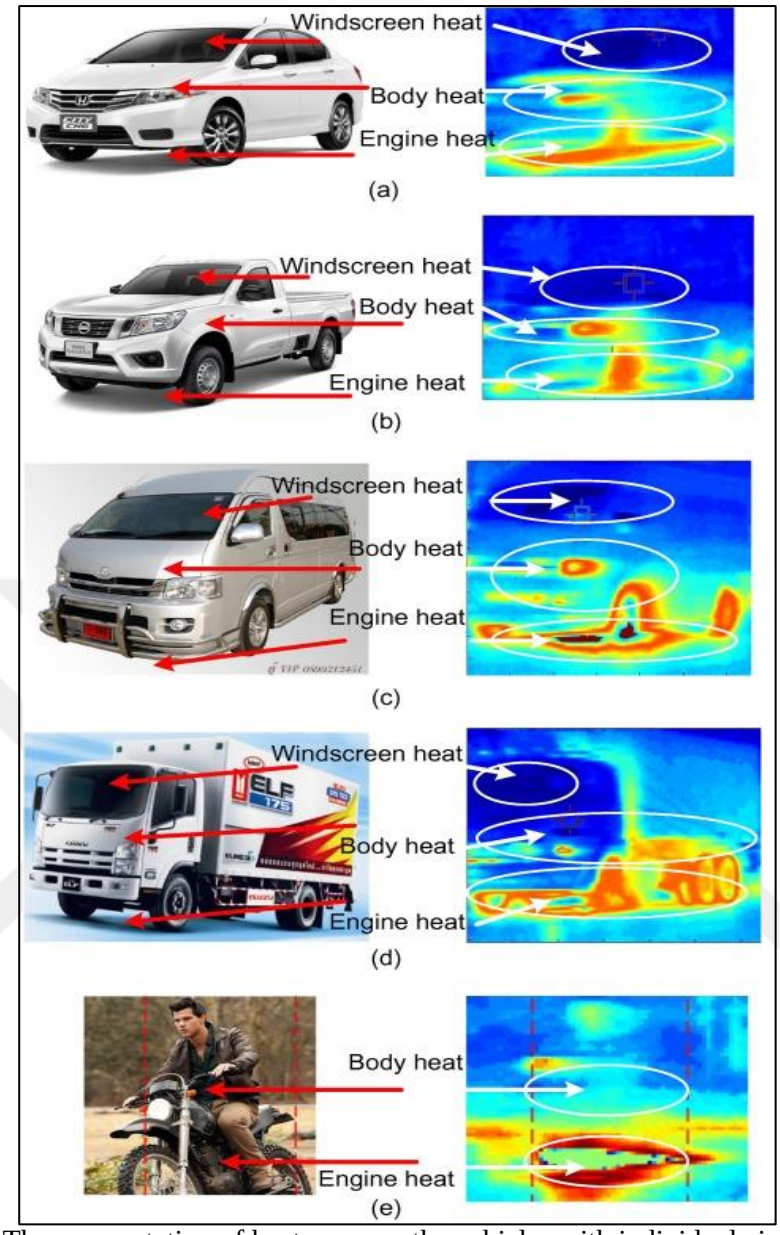

Figure 2.5: The segmentation of heat areas on the vehicles with individual circles marking periodically [40].

Researchers in [41] proposes a computationally efficient algorithm to segment rectangles where corners may be missing, or edges may be ragged. First, the input image is split recursively into smaller infrared images, such that the resulting infrared images only contain one object and little background. Next, they calculate the number of foreground and background pixels in the split infrared images and the center of mass of the object, which is only an approximation of the real center because the objects are not perfectly rectangular. The approximate angles of the corners of the object are estimated by calculating the

distance from the center to the borders. The authors note that the corners of rectangular objects appear periodically as mentioned in [42]. Therefore, their angle can be estimated from the calculated distance values and refined by applying the precondition that the objects must contain at least two parallel edges. Finally, to obtain the bounding rectangle, the authors grow a rectangle located at the already known center with the previously calculated angle as mentioned in [43]. The growing rectangle is split into four quadrants. Each of them is validated after every growing iteration with respect to the number of pixels not belonging to the object. They terminate if three quadrants are above a threshold.

Researchers in [44] propose an algorithm to segment rectangular objects that may overlap and are placed on a lightly textured background with unknown color, which is the usual output of scanner preview infrared images. The authors in [45] aims at detecting the background color, which is challenging when the image consists primarily of differently colored foreground objects, but it allows them to segment the objects more easily. To retrieve the background color, they first separate the image into line segments which have the same tint, by calculating the neighboring color differences. Under the assumption, that the background color segments are long, a voting scheme is used to extract possible background colors as mentioned in [46]. For each of the background color candidates, the edge strength at the boundaries to non-background colors are calculated. The authors observed that the gradient between background and object colors is larger than it is between foreground colors, and that the number of edge pixels correlates with the number of foreground objects. Therefore, they determine the true background color by only using line segments that exceed a certain length, are of the most frequent color, and have a large number of edge points with high gradient values as mentioned in [47]. From these edge pixels, connected components are extracted and used to fit lines, with weights derived from the edge strength. Neighboring pixels, located in the direction of the line are added until a stopping criterion is reached. Orthogonal lines are used to calculate the corners of the objects, as well as the line segments which form the rectangle. To eliminate wrong line segments, the median and mean color differences between foreground and background along the line segments are calculated and used to derive the parameters of a score function from a number of automatically generated infrared images. Next, for each rectangle candidate, the same values used for line fitting and a score that relates to the number of

background pixels in the rectangle are input into an SVM classifier, which determines if the rectangle candidate is a proper object or not. To find better matches, the rectangle candidates are also shifted in the local neighborhood. A candidate is accepted when at most 10% of the pixels inside the rectangle are classified as background.

In summary, all approaches are specifically designed to be fast and detect potentially imperfect rectangular objects with possible overlap or missing corners. However, they rely on uniform background color which also must be distinct from the majority of the content found in the foreground objects. Both conditions are usually not satisfied in our application, as data are mounted on arbitrary background, and unknown lighting conditions may cause significant brightness changes. Also, the color of the data is not necessarily different from the background. Hence, the only way of distinguishing them from the background is by the border of the data.

2.5. AUTOMATED FORM RECOGNITION

Classification and content extraction from preprinted forms is also a closely related research topic. Instead of manual sorting and transcription of filled forms, a number of automated approaches that process scanned documents have been presented. Forms are data, that consist of text describing its purpose, captions for the blank regions, and tables or cells that must be filled in or are already preprinted. Examples include administrative forms, bank checks, payment slips, and many others. Since the use of forms is widespread, a number of different approaches have been developed over the years to automatically process them. In [48], an overview of the topic is given, including infrared images of sample forms and descriptions of a number of popular approaches. The general layout and the small variability within the form types, where the only differences are the filled-in regions, are almost identical to our problem.

In a large number of forms, the text must be entered in rectangular regions. Therefore, a number of approaches exist that exploit this fact, such as [49]. Their approach is designed to work with low quality scans, that produce broken lines and noise. First, lines are detected and skew correction is performed. To correct errors introduced by broken lines and noise, the extracted cells are checked for inconsistencies under the assumption that the lines of

cells and tables start and end in other lines of cells and tables. The consistency is checked using three rules in the following order: a line must end on another line. If this is not the case, the line is extended until it crosses another line. The outermost lines of tables must end in a line, otherwise they are also extended until they do. And finally, crossing points are only allowed in cell corners. The current structure is modified using these rules until no more rules are violated, where matching rules with higher precedence are preferred. The final result consists of the table and cell structure from which the values can be extracted.

Researchers in [50] propose an algorithm that uses the line structure of the document to classify it. Their first step involves removing all characters, since their information is not used. This is done by thinning out all structures, followed by extraction and clustering of features. Finally, all clusters that belong to characters are removed, leaving an image that consists solely of lines. Next, three matrices are generated. The first contains the line intersections, where each vertical line is represented as a column, and each horizontal one as a row. When two lines intersect, the first matrix contains a number at the appropriate location, representing the intersection type. The other two matrices encode the distances of the lines to each other, one matrix for horizontal and one for vertical lines. To remain scale invariant, the authors assign numbers to distances relative to the document size as mentioned in [51]. Matching a new document against a known one starts by calculating per element and matrix differences that must be above a threshold, which eliminates completely unfitting forms. Should the matrix sizes differ, a sub matrix of the larger one is used? A matching score that is derived from the similarity of the line crossing types and the distances, is calculated for the remaining candidates as mentioned in [52]. This also requires that the matrices are of the same size. Should this not be the case, the authors extract a sub matrix with the proper size, that has the best match with the full version. The document type with the best matching score is the determined and returned document type.

An approach that performs the form classification using only keywords and their location on the form. They extract all characters from the form and perform *CNN*. Then a previously created keyword database is used to classify it. The database contains keywords and their locations, which are unique to each form type. A matching score is calculated by

comparing the positions and string similarity as mentioned in [53]. Finally, the form type with the best match determines the result type. To counteract *CNN* inaccuracies, the authors match the strings by counting the number of insertions and deletions needed to transform the first string into the second. To retrieve the form contents, a list of regions is available for each form type. Those are the contents of interest which should be extracted. Each of these regions is thresholded separately to improve the *CNN* precision, and additional constrains, such as string composition suffixes, are used to extract the part of the region (excluding descriptive text) that is of interest for further processing.

2.5.1. Image Reliability

In each iteration, the PPHT selects a point randomly, performs the Hough voting, and removes it from the list of unprocessed points. If a cell in the accumulator exceeds the detection threshold, a line has been found. The pixels voting for the line are removed from the input, as well as the accumulator and the line is stored if it exceeds the minimum line length. The authors state that the line detection results are not satisfying, because of distortions from the camera, resulting in lines that are not perfectly straight. To detect such distorted lines, they propose to replace each pixel with a cross-shaped object. Furthermore, they define a mapping from a pixel to its corresponding cross structure and vice versa. This mapping is used to check line candidates only at their cross centers, as well as to remove the entire cross structure from the input list if it voted for an accepted line as mentioned in [54]. This is done because the surplus pixels would result in additional detections of the same line. The detected lines are grouped into rectangles by extracting parallel lines of the same length, which are then combined with orthogonal pairs to complete the rectangle.

2.5.2. Image Quality

An efficient rectangle detection system, intended to be used on mobile devices. First, they extract the edge map using the Canny edge detector as mentioned in [55]. The threshold parameters are automatically selected. To improve the robustness and increase speed, all edges produced by text are removed. Consequently, a lower number of lines need to be processed. They do this by extracting connected components and calculate the bounding box for each one. The aspect ratio, height of the bounding box, and number of pixels in

relation to the bounding area are used as filtering criterions. Next, the filtered image is used to extract lines by applying the Hough transform. Two lines form a line bundle if their angle does not differ more than fourteen degrees, which accounts for some amount of perspective distortion. The authors then group line bundles to rectangle hypotheses, if all line intersections are inside the image. Finally, they compute the edge support on the dilated edge image, where the most dominant rectangle hypothesis with the highest edge support is the resulting rectangle. The edge map is dilated to increase robustness against camera distortions, causing lines that are not completely straight.

A more generic approach to extract rectangles using a windowed Hough transform is proposed. They use a ring shaped sliding window, where the outer diameter approximately equals the size of the largest and the inner diameter the size of the smallest rectangle that can be detected. Next, the Hough transform of the region under the sliding window is calculated, where the discretization depends on the outer ring diameter. To detect the peaks in a robust manner, a modified butterfly evaluator is applied to the accumulator. To retrieve the lines, the accumulator is thresholded.

2.5.3. Vehicle Data Accumulation

The accumulation of vehicle data selects the threshold to be half of the inner ring diameter, which only detects lines of at least half the required length. This yields a number of peaks, which are grouped to extended peak pairs if the following conditions hold: The two lines are approximately parallel, have the same length, and are symmetrically placed in the accumulator space. An extended peak pair forms a rectangle hypothesis if the orthogonal angle difference is within a threshold. Finally, the authors perform non-maximum suppression by calculating an error measure that incorporates the values used for grouping the peaks into extended peaks. They note that a shifted rectangle is visually more distinctive than a slightly wrong rotation angle, therefore they assign a larger weight to the distance measures. All these approaches apply a Hough transform to extract lines which are grouped into rectangles if a number of conditions are satisfied as mentioned in [56]. As noted in a number of these papers, the Hough transform is quite slow and requires a lot of memory. Therefore, to use it on a mobile device, the input image must be downscaled and

the resolution of the Hough accumulator reduced. Furthermore, the content of vehicles is often contained in a grid structure or cells. A number of schematic drawings are also frequently present, which results in a large number of lines, that cannot be easily removed beforehand. With the examined approaches, grouping and checking the hypotheses is time consuming. The printed lines inside the data may also exhibit stronger edge characteristics than the actual separation between the data and the background, thus leading to wrong detection results.

2.5.4. Noise Removal Approach

Under our assumption that the image consists mostly of the data, this window size ensures that the individual pixel thresholds are adjusted to possible brightness changes, resulting from the illumination conditions with removing the noise. Yet, the size is large enough that regions in the image that contain no or very little structure are not larger than the averaging window. In such regions, bright pixels caused by noise or dirt would produce a very noisy thresholding result. The next step consists of morphological opening with a square 3×3 structuring element, which ensures that the data is one connected object. The size of the structuring element is chosen to bridge small gaps caused by dirt or noise. Testing has shown, that a larger size would merge the data with the background. The preprocessing output for the first sample input image, and the output for the second image. From the preprocessed image, connected components are extracted using the approach proposed in [57]. The approach operates very efficient, as each row in the image is scanned from top to bottom. Background pixels are visited once, and contour pixels a constant number of times. Therefore, the time required is linear in the number of pixels in the image. The authors state that their algorithm consists of the following four principal steps: When an unlabeled foreground pixel is encountered, the entire outline of the connected component is assigned the same label until the starting point is reached again. If an already labeled pixel is encountered, the current row is followed until a background pixel is reached and all pixels are assigned the label of the initial pixel. Should the next background pixel belong to an unlabeled internal contour, the entire contour is labeled until the starting point is reached. Finally, if a labeled internal contour is encountered, the pixels in the row are followed and assigned the same label as the first pixel until a background pixel is found.

2.5.5. Image Rotation and Approximation

Since the angle of the data is not known, the rotation of image is inverted and the entire procedure repeated, which allows detection of data with bright and dark back- ground. Should the combined contour list contain more than one data candidates, the object where the distance between its centroid and the image center is smallest is chosen. This removes cases where parts of the background form large connected components next to the data, that have either a larger width or larger height than the centered object. This means, that a selection purely based on size or area cannot be used. Finally, if the absolute value of the object angle does not exceed ten degrees, the rotated object corners are returned as approximate data location. Otherwise, the up-right bounding rectangle is returned. The angle is limited, because it is calculated using moments and therefore, concavities on the connected component may cause large and entirely wrong rotation angles. This constraint can be applied as large angles are unlikely to occur since the user will likely keep the imaging device leveled when capturing the data.

2.6. DESCRIPTION AND MODIFICATION OF SEGMENTATION (DMOS)

Researchers in [57] employ a method named Description and Modification of Segmentation (DMOS) that uses Enhanced Position Formalism (EPF) to describe the structure of the input, and was previously used to extract musical notes, mathematical formulas, and military application forms from the 19th century. Lines and symbols are extracted from the input image and processed according to the rules described in EPF, where the parser may change the structure during parsing. The authors note that there are two sources of noise: The image itself, as well as parts of the document structure that are not formalized in the grammar. Also, other grammar based approaches are not robust against noise because the next element that is parsed is the next token, whereas using their approach, the parser selects the next token to parse from the entire image by itself. EPF includes a number of operators such as the position operator (which specifies that an object is located relative to another object), the factorization operator (used to factor common parts of rules),

save operators (that allow to store and later load a nonterminal), the declaration operator (that allows to refer to a number of rules by name), and the space reduction operator (that limits the space where rules may be applied). To cope with noise, they propose two new operators that work on terminals and have pre- and post-conditions to skip noise, and a new find operator, that applies the given rule to each token until it matches or a stopping condition is satisfied. To prove the robustness against noise, the authors present a complete grammar to detect the seven lines that make up an image court. They note that it is not necessary that the complete tennis court is visible and that the grammar is scale invariant. The implementation of the parser that can execute a given grammar is described in [58]. An extension of Prolog named Prolog is used to translate the grammar into a Prolog program, which then performs the recursive descend parsing and bindings to are used to segment and extract the tokens from the input image. An approach that employs line and text positions and is applicable to forms without tables or tabular structure is proposed. They use fourteen different primitives that occur on forms, such as text, lines, vertical distances between adjacent rows, intents (positive and negative), and nine types of corners that are encoded into a string by iterating over the objects that are sorted by their vertical position. To match a new document, the number of common tokens of the learned blank and the new incoming form are counted with and without the text blocks and combined into a score. A measure without text is also incorporated because learning is performed with empty forms.

Contrary to the considered approaches that are trained on empty forms, there exist a number of methods where the structure of the document is specified using various flavors of grammars which are then used to parse a new form. One such approach is proposed which uses a graph grammar to describe forms. The authors first define an XML based description for grids, where the rows and the indention of the cells are described, that can also be used as the result of the form analysis. Grids can be described using graphs, where the cells are the vertices and the edges represent the connectivity between them. The authors in [59] note that this grammar can be processed by a conventional graph grammar parser, although since it is context sensitive, parsing is not efficient. To speed up processing and make it suitable for real world application, the authors assign priorities to the rules. An exception is for example DMOS , but writing the grammars for each type is time consuming and cannot

be done by a regular user. Finally, each grammar must be translated to Prolog and executed to check if the form matches its type as mentioned in [60]. Overall, scanned forms usually contain a very low amount of noise and are well segmented, the problematic cases are merged objects and non continues lines, where the amount of noise found on weathered data is much higher. When matching a new form against the known form database, approaches that use string or graphs, consider the new form a noisier version than the initially learned one, which might also lead to misclassifications on very similar forms. Most grammars used to describe forms are either limited in expressiveness or very complex to write, and very sensitive to noise.

Figure 2.6: The market segmentation for potential targeting of vehicle data detection system [60].

Conclusively, a number of approaches have been proposed over the years to classify and process forms which solve problems similar to our task. However, methods that extract cells and grids from the image are not applicable, since not all data contain them, while approaches that extract a grid like structure from the text blocks encounter issues with complex layouts. Approaches that further incorporate text blocks and keywords, by performing CNN at the beginning, are also problematic for data types that are weathered or contain embossed text, where CNN performance without preprocessing is not satisfying and slow to execute on a mobile device.

3. METHODOLOGY

Approach to vehicle detection and classification of rectangular vehicles, that are attached to electric power supply equipment. All necessary steps, which are needed to extract the desired information from a known set of vehicles are explained, with the restriction that only human readable text is processed. Schematic drawings and barcodes are not in the scope of this work. Figure 3.1 shows a scheme of the proposed approach. First, the vehicle is detected, extracted, and warped. The result is an image that contains only the up-right vehicle, which is then used to determine the type of the data. Since there is no standardized layout for such vehicles, each manufacturer produces different layouts for different product lines. Therefore, the number of existing data types is large, which makes it desirable that the training process of the classifier is fast, so that new types can be added quickly. This is achieved by using Convolutional neural networks, which are also fast during classification and therefore of advantage when used on a mobile device. To generate the features for classification, the image is split into cells by overlaying a grid. For each cell of the grid, we compute a feature vector that consists of the median color value and histograms of Local Binary Patterns (LBP). We concatenate the values of all cells and add the size ratio of the data as final feature. For each type, a list containing the designations and positions of the text regions, that should be extracted, is usually available, because we deal with a set of known vehicles. As all data of a specific type share the same layout, knowing the positions of the text regions simplifies the content extraction, since the boundaries of the fields do not need to be determined. Further, the mapping of the content from a specific region to a label is available, which is needed to meaningfully use the values later on. The extracted regions are preprocessed, which removes artifacts caused by dirt and other structures on the data. Preprocessing consists of size filtering, morphological opening and closing, and character grouping. Finally, the preprocessed text regions are fed to the Tesseract Convolutional neural network (CNN) engine one and the *CNN* steps are repeated. When the scores of all regions are sufficiently high or the maximum number of retries has been reached, the results are stored for further processing. The final output consists of the detected data type, the recognized text, as well as its confidence and position on the data. After the presence of a vehicle and its approximate location have been determined, the data

should be extracted as accurately as possible to simplify further processing. The boundary returned from the previous step is unsuitable, because of inaccuracies introduced by thresholding the image and the morphological opening. Furthermore, the rotation of the returned rectangle is only an estimate based on the extracted connected components and therefore unlikely to be accurate.

3.1. VEHICLE DATA DETECTION

The data has been located and extracted, its type must be determined. The research conquers the issue of numerous different types using a machine learning approach. In the literature, several related methods for the task of image classification have been presented. The goal of these approaches is to assign a category to the entire image. Therefore, no a priori assumptions about the content of the input image can be made, which differs from our situation. Nevertheless, the extraction and selection of image features is of interest. An approach which classifies an image based on the categories of the objects it contains. The image is represented at different levels by dividing it into cells, arranged in a grid, where a higher level increases the number of rows and columns in the grid. In each cell, the research compute features. Scale-Invariant Feature Transform (SIFT) features with four different radii, extracted at equally spaced points, represent the appearance of the objects in the image. These points are clustered into visual words. The shape of the objects is represented by histograms of oriented gradients. These values are then calculated at each level over the entire image, and used to detect regions of interest. A region of interest is a visually similar region that appears in multiple test infrared images. Only these regions are used to train a convolutional neural network.

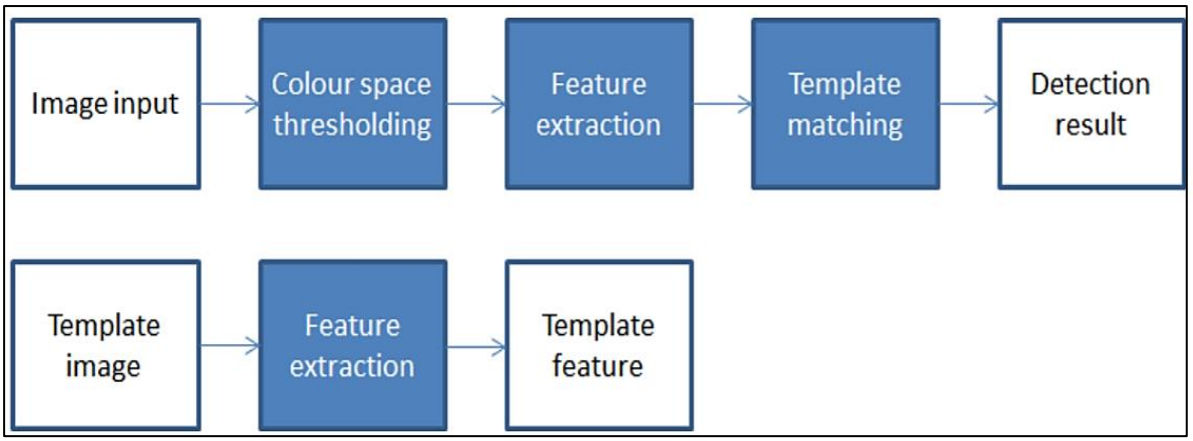

Figure 3.1: The processing on unprocessed raw image.

The research proposes a classification framework that, like the previous one, operates with features extracted from cells in a grid. The first step of their approach consists of transforming the calculated descriptors, extracted at regular intervals, into a feature vector. The research state that their coding allows the original and nonlinear function, as described by the feature vector, to be approximated by a linear function. They further explain that any descriptor, such as *SIFT*, that takes the local neighborhood into account.

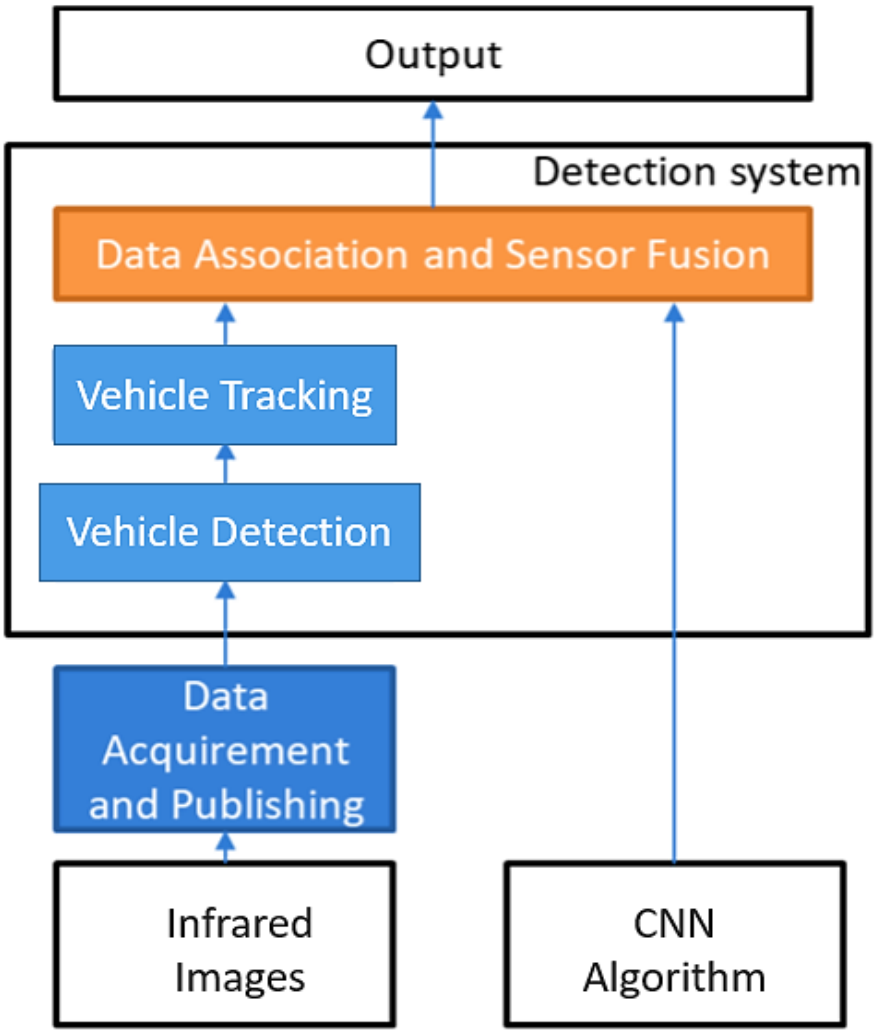

Figure 3.2: The proposed automatic vehicle data recognition system pipeline.

3.1.1. Feature Extraction

Our feature vector contains information about the data color, which makes it possible to differentiate vehicles with the same structure but different color, and histograms of local binary patterns, that encode the texture of the data. The final feature is the size ratio of the vehicle, which helps to distinguish data with similar content but different size proportions. For classification, we use a convolutional neural network. Therefore, preprocessing to

remove them is applied. The listed steps are applied to each extracted text region, which are provided by the previous stage. The text regions are extracted from the unscaled data image. As such, the size of a region depends on the image size. An input region may also contain multiple lines of text.

3.1.2. Data Localization with Local Binary Patterns

The next feature, added to the feature vector, consists of histograms of Local Binary Patterns (*LBP*). This feature encodes the texture of the data. The *LBP* codes are computed by comparing the values of a fixed number of neighboring pixels to the value of the current center pixel. The neighboring pixels are evenly spaced around the center pixel, in a circle with a fixed radius. For each neighbor, the corresponding bit in the code is set to zero if its value is larger than the value of center pixel. If the value of the neighboring pixel is smaller, the bit is set to one. The output image has the same dimensions as the input image and contains a *LBP* code for each pixel.

We extract the L-channel from the data image that was converted to the CIE L*a*b color space, as it contains the intensity values. Next, we apply Gaussian blurring with a kernel that is 7×7 pixels large, which removes noise that would otherwise be incorporated into the *LBP* features. The blurred image is then used to calculate the *LBP* codes. In our implementation, we use *numLBPNeighPixels* neighboring pixels and a radius of *LBPRadius*. Next, the same grid structure that was used to extract the color features is applied to the image that contains the *LBP* codes. For each cell, we calculate a histogram of the codes, which contains *LBPHistSize* bins. The cell histograms encode the number of occurrences for each the direction, that are present in the corresponding area of the data.

Table 3.1: Grid search results for the feature extraction parameters. For each parameter, the possible values and the final value is listed.

Parameter	Possible values	Final value
numColsRows	10, 20, 30, 40, 50, 60	20
numLBPNeighPixels	4, 8, 16	4
LBPRadius	2, 4, 8, 16	16
LBPHistSize	30, 40, 50, 60	60
numLBPLayers	100, 200, 300	300

3.1.3. Normalization With Feature Vectors

The final feature vector that is calculated for each image, used for training and classification, contains median color values, histograms of *LBP* codes, and the aspect ratio of the data image. The selection of the parameters used to extract the different features, referenced only with symbolic names (*numColsRows*, *numLBPNeigh-Pixels*, *LBPRadius*, and *LBPHistSize*). The deep learning algorithm, a convolutional neural network is used alongside with LBP. This is important, because the camera resolution varies with different capturing devices. We should note that the size ratio is the least important of the three features. Nevertheless, it represents a characteristic of the vehicle that helps to distinguish types with similar content, but a different size ratio. The infrared images of the text regions frequently contain structures that are not part of the text, caused by borders around the text or neighboring text regions, as well as dirt on the vehicle. Since most *CNN* systems are developed to process scanned text, which mostly consists of well segmented characters, these additional structures would deteriorate the output significantly.

3.1.4. Geometrical Analysis

The absolute geometrical analysis with the vehicle data gradient magnitude of the input image should be computed. Using the gradient instead of adaptive thresholding yields better results if the image contains a gray value gradient, which is frequently caused by rust stains. In such cases, simple thresholding would create large black regions unsuitable for further processing. The minimum and maximum gradient values are determined and used to calculate an adaptive threshold for filtering, where all values smaller than the threshold are set to zero. To retrieve the threshold, a histogram with 256 bins is calculated over the magnitude values and the bin with the largest amount of values is located. This is done under the assumption that the largest number of gradients is produced by text. A fixed thresholding value is unsuitable because embossed text has very low contrast and would be removed by blurring the image before calculating the gradient. For further processing, the filtered image is normalized to the 0 to 255 range and adaptively thresholded, where the block size equals half the image height, resulting in a binary image. A sample binary

image for printed text, where the rust stain from the input image is not visible anymore. Further, another example for embossed text, that contains a lot of noise.

3.2. CONVOLUTIONAL NEURAL NETWORK (CNN)

A convolutional neural network consists of a number of layers, where each layer is trained on a different subset of the training set. The elements in the subsets are generated by bagging, which chooses elements randomly and with restitution. Therefore, a sample may appear multiple times. When training the layers, a random subset of the features in the feature vector is used to split the samples for each node. In our case, the number of features used for the splits is half the square root of the number of all features. From all calculated splits, the one that causes the largest decrease of entropy and therefore the largest information gain of the label histogram is used. In our case, the learning is completed if CNN-Layers are generated, or the estimated classification error is smaller than 1%. The error is estimated during training by classifying samples that were not chosen by the bagging procedure. The layers in the convolutional neural network are not pruned. For classification, each tree generates a result from the input data and a majority vote is used to derive the final class label.

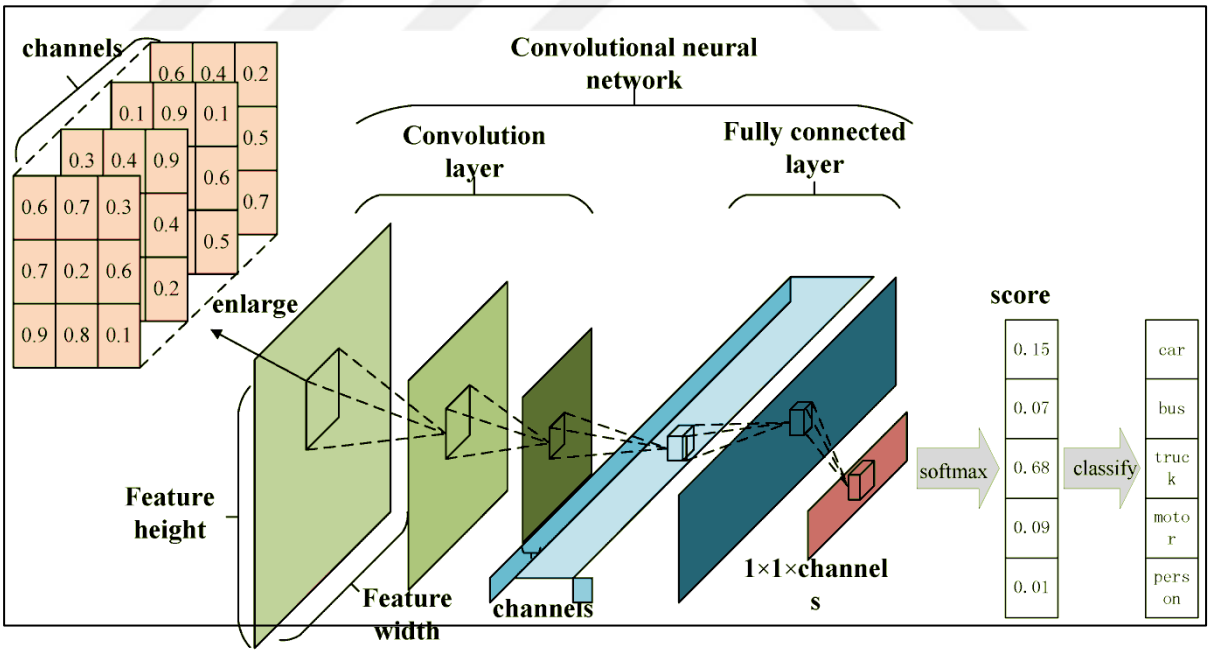

Figure 3.3: The employment of CNN with proposed architecture for robust vehicle data detection.

We use convolutional neural networks because they have a number of advantages. We state that increasing the number of layers does not cause over fitting. Therefore, a large number can be used safely, as it increases accuracy. However, with an increasing number of layers, the improvement in classification accuracy declines at a certain point. Furthermore, a larger number of layers increases the time needed for training, and linearly increases the time for classification. Training and classification are also very fast, since each tree can be processed in parallel. Finally, a very important factor is the classification performance. The supervised deep learning compares a number of popular approaches, including support vector machines, boosted layers, convolutional neural networks and others. The dataset contain binary classification problems are used for testing. The research concludes that there exists no universally superior algorithm, as no one excelled at each problem. Their test results show that calibrated boosted layers, convolutional neural networks and bagged layers generally deliver the best performance. The CNN algorithms is vastly used for sign recognition. Their test set consists of 43 classes and over 5000 test infrared images. The results list convolutional neural networks on the third place of the machine learning algorithms. We evaluate our use of convolutional neural networks. The research employs maximally stable extremal regions to locate vehicle data. The vehicle data are located by searching for a number of smaller regions placed within a larger one.

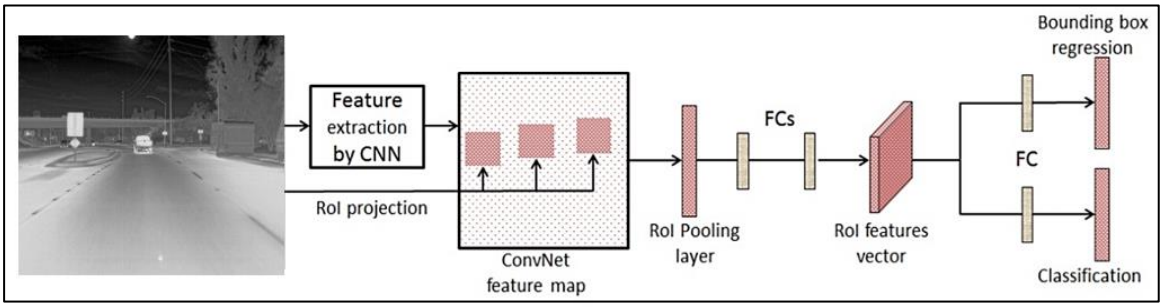

Figure 3.4: The novel use of convolutional neural network-based pipeline for vehicle data detection.

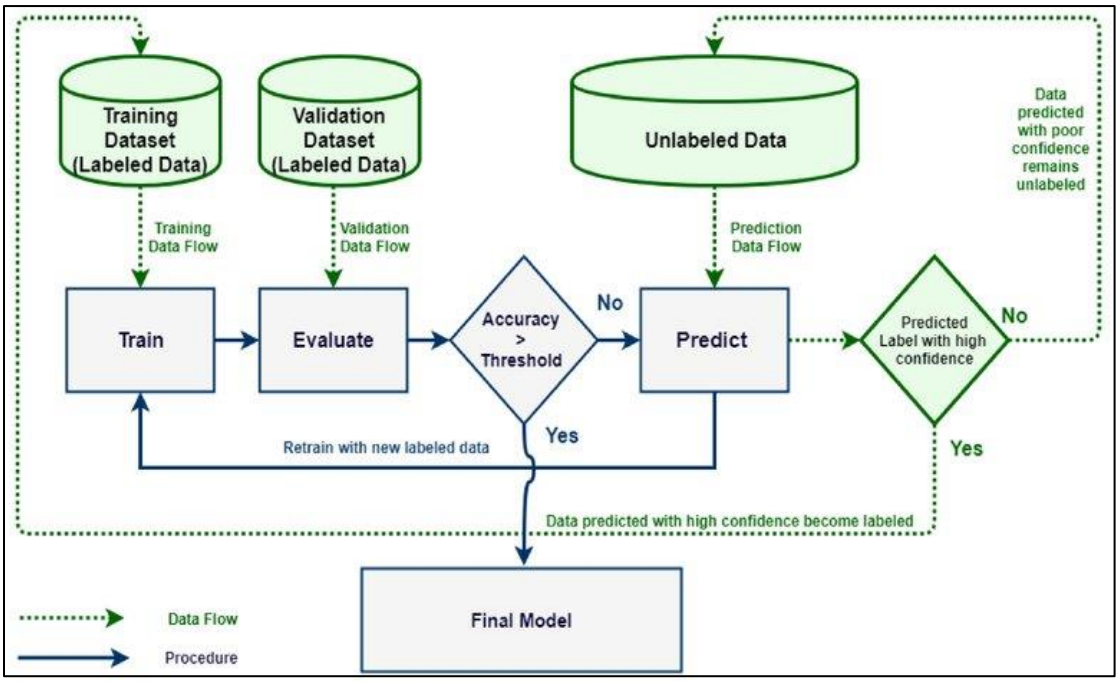

Figure 3.5: The CNN model training and validation with data prediction for labels.

In CNN, most of these systems are designed to detect text in natural infrared images. Since our data infrared images do not contain nearly as much clutter as found in natural infrared images, such sophisticated approaches like the *SWT* are not needed. Instead, we employ a similar *MSER* and grouping based approach to extract characters and words from vehicles, for vehicle data detection. We have the extracted up-right data image, as well as a list that contains the position, content type, and name of the regions that should be extracted. Since the data is not always extracted perfectly, we need to adjust the given regions, so that they align with the content of the data.

3.3. VEHICLE DETECTION PIPELINE

The research proposes a number of steps for vehicle detection and these steps consists of locating the text regions on the data and matching them to a predefined list. For each data type, a number of regions that hold information which should be extracted are defined beforehand. The located text regions are mapped to the user defined ones, and the image regions are passed to the *CNN* to convert them to machine readable text. This section starts with an examination of popular text detection methods and subsequently our approach is

explained in detail. There exist a number of methods, that are specifically designed to extract text from natural infrared images. The output of the first stage is an image of the same dimensions as the input image, that contains the stroke width for each pixel. First, they compute the edge map of the input image using the Canny edge detector and initialize all the stroke width result values to infinity. For each returned edge pixel, the gradient direction is followed until another edge pixel is found. If the gradient direction of the second edge pixel roughly points back to the starting pixel, all stroke width values on the line connecting them are set to the distance between the starting and the end pixel, but only if they do not already have a lower value. When all edge pixels are processed, all start pixels where an end pixel was found are revisited again. The stroke width values on the lines connecting them are set to the median value of all pixels on the line. The research note that this is necessary to get correct results in corners. Next, they extract connected components, where neighboring pixels are assigned to the same contour if their stroke widths are similar. These contours are filtered to retrieve text regions. A contour is discarded if the stroke width variance of its enclosed pixels is too large, or when its aspect ratio or diameter to stroke width is not within a threshold. Furthermore, the bounding box of a contour must be properly sized and not contain more than two inner contours. The resulting letters are grouped into pairs if they have a similar stroke width, height, and color and are not located too far apart. Finally, the character pairs are merged into text lines and word boundaries are detected using a histogram of the character distances.

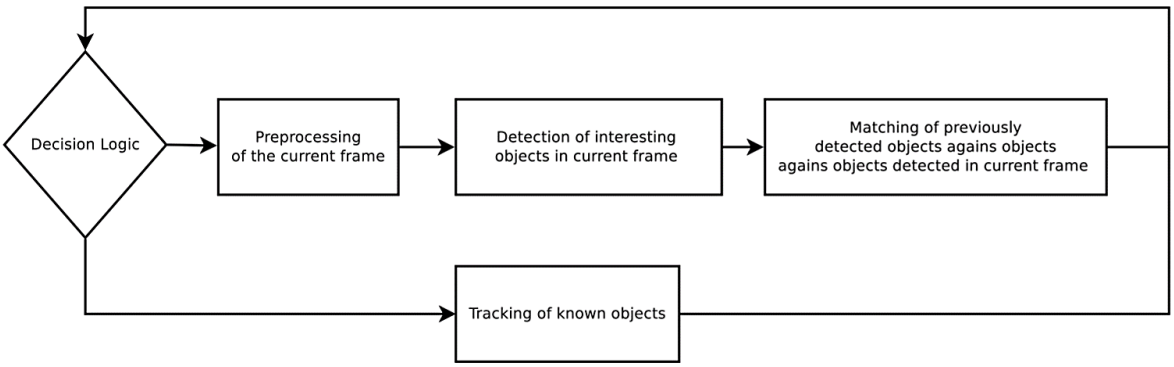

Figure 3.6: The CNN simplified scheme of the framework for tracking.

They first detect character candidates using *MSER*, which are filtered by size, aspect ratio, stroke width, and number of holes. For the remaining regions, the following features are

calculated: Number of pixels, bounding box area, diameter of the bounding circle, mean gray and color value in the L*a*b color space for the region itself and its boundary, the stroke width as described above, and the mean gradient value of the outer boundary. Next, they calculate a dendrogram for each feature. For each node of the dendrogram, the probability that a number of regions share the same feature is calculated using the binomial distribution and used for clustering. Then, a co-occurrence matrix over all clusters is calculated. The final grouped regions are obtained by applying the previous clustering steps again to the co-occurrence matrix. A CNN classifier is used to prune regions that do not contain a character-shaped object. Finally, a CNN classifier is used to remove clustered regions that do not contain text. The category specific regions and a classifier that determines if the input region is a character or not. The detected text regions are grouped into vehicle data by incorporating knowledge of the layout.

The input image is thresholded at each intensity level, starting at zero which results in a completely white image. For each level, connected components are extracted and stored in a tree structure. As the threshold increases, black regions appear, increase in size, and finally merge, until the entire image is black. Each node in the tree represents an extremal region and its subtree contains the nested components. Finally, regions that are maximally stable are located and returned. A region is maximally stable, if its size remains mostly unchanged over a number of thresholding levels. First, the gradient magnitude of the input image is calculated and normalized to the 0 to 255 range. To remove the majority of noise from this image, caused by rust stains and dirt, we set all pixels that have a value larger than 50 to the new value of 255. The threshold has been determined experimentally, so that it removes noise and the characters remain intact. The resulting image is used to extract *MSER* features.

3.4. DATASET DESCRIPTION

The dataset is named as “FLIR Starter Thermal Dataset for Autonomous Vehicle Testing” being acquired from open source repository. The grouped infrared images are mapped to the user provided list, which contains their position in a reference data, the content type, and the names of the regions that are used to store the results. A group of infrared images

in dataset provides the detected region which is assigned to the defined region, where the Euclidian distance between the two bounding box centers is the smallest. Since this simple mapping method is likely to produce incorrect matches, usually caused by missing text regions on the data or wrong character grouping, the correspondences are used to estimate a rigid transform. The coordinates of all user defined regions are warped using the calculated transform, resulting in text regions with the user specified size and positions adapted to the detected regions. At this point, we have a list of text regions and their positions on the full size data image. The regions contain infrared images of the desired content, which are transformed by the *CNN* into machine readable text in the next step. The dataset can be acquired from the link: https://www.flir.co.uk/oem/adas/adas-dataset-form/

Figure 3.7: A sample of thermal infrared image from dataset.

3.5. VEHICLE SEGMENTATION AND BINARIZATION

The opened image is subtracted pixel-wise from the original image. Since the vehicle width is small in relation to the structuring trucks, they remain intact. The graph illustrates this for the first sample region. As the lines are not leveled, some parts of them remain. However, using the same approach to remove vertical lines would also remove most of the text. To prevent

this, the presence and location of borders is determined first, and only those regions are filtered. At the first 20% of the left and right side of the image, peaks are detected. A peak is reported if 70% of all pixels in a column are black. Since we assume that the text is roughly centered, the distance of the first detected peak on either side to the image border should also be approximately the same. Again, the opening result is subtracted pixel-wise from the original image. Since the removed lines usually are not perfectly horizontal or vertical, some debris of them remains. The filtering result for the second region. To remove the noise, we extract and filter connected components. A connected component is removed if its bounding box touches the borders of the image, the bounding box width and height are smaller than three pixels, or the area of the connected component is smaller.

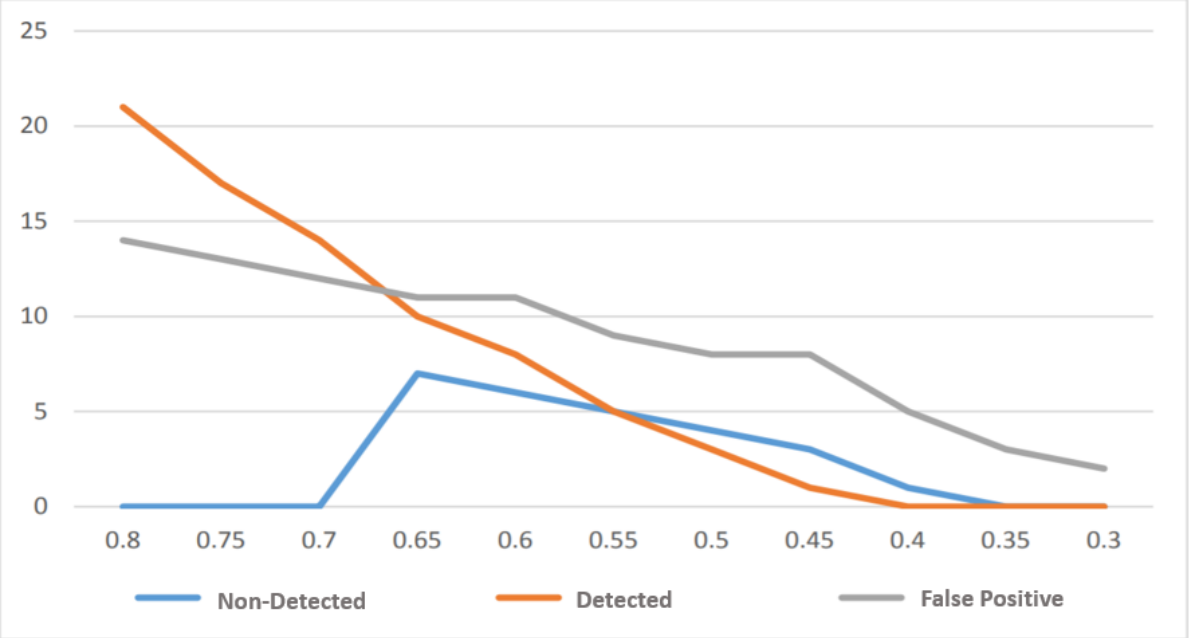

Figure 3.8: The graph illustrates the vehicle detection and non-detection with false positive.

Next, the individual words are grouped into lines of text. Therefore, we first sort all detected words by the y-coordinate of the centroid of their bounding box in ascending order. A new line is started if the y-coordinate differs from the previous one by more than 20% of the text region height. To assign commas and accents to the correct line, the overlap with the position of the starting contour is checked. After all characters have been assigned to lines of text, each line that contains more than five characters is checked for remaining border fragments that were not discarded earlier, because they are skewed. The bounding box of the first and last characters of the line are checked against the median

bounding box contour height. If the height difference to the median contour height is larger than 25%, the bounding box does not overlap with the bounding box of the neighboring contour, and the bounding box width is smaller than 33% of the text region width, the contour is removed. At this point, the preprocessing is complete. However, the characters only consist of borders, as all processing was done on an image created by thresholding the gradient magnitude. Such infrared images cannot be processed by the Tesseract *CNN*. To retrieve filled characters, the original unmodified input image is adaptively thresholded with a block size and the filtered image is morphologically opened with a 20×15 ellipsoidal structuring element. The large black region is caused by the rust stain. The thresholding result for the second regions, also contains some clutter. The opened image is used as a mask, when copying from the adaptively thresholded image. The results for both regions that are passed to Tesseract are everything that does not belong to a character has been removed.

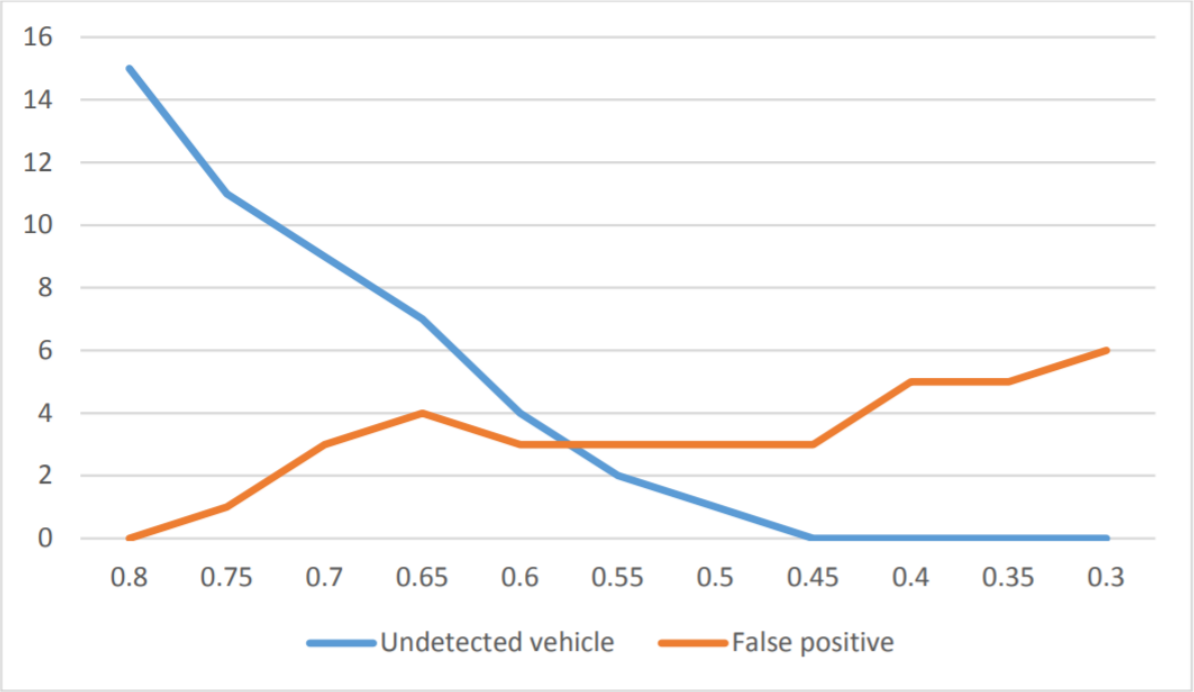

Figure 3.9: A graph illustrates the undetected vehicles with false positive.

3.6. EXTRACTING THE VEHICLE DATA

The final step of our approach consists of requesting and processing further infrared images

of the vehicle, if the CNN confidence score for at least one text region is low. The user is prompted with a request to acquire a new image of the approximate region of the data, where the most problems occurred. The extracted data and the initial CNN results are stored as reference. Should the averaged recognition score of at least one region be lower than 80%, a new image is requested? The image is separated into four disjunctive parts (top, bottom, left, and right) and a new image is requested for the region, in which the most errors occurred. Key-points are extracted from the new image and the stored data using the Oriented FAST and Rotated BRIEF (ORB) detector, and Binary Robust Invariant Scalable Key-points descriptors are calculated. ORB and BRISK are used because of their lower computational and memory requirements as well as comparable performance to CNN. A sample of an existing up-right data image, and a new input image that contains the right portion of the data at a larger magnification. Further, the extracted key-points, the calculated matches between the infrared images, and the location of the new image in relation to the reference data, are displayed. Each completely visible region is warped into its up-right position, where the size is approximately the same as in the new input image. Since each region is processed and warped separately, the process can be executed in parallel and less resources are needed since not the entire image must be processed. The preprocessing and CNN steps, as described earlier, are executed for each new region.

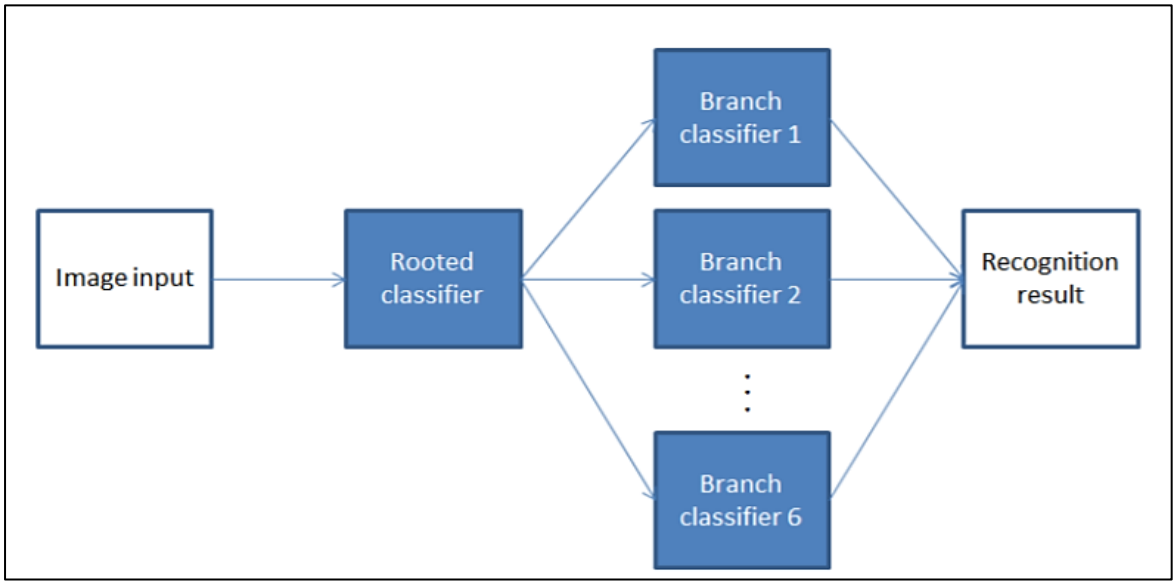

Figure 3.10: A breakdown cases for extracting the recognition for number of data at the same time.

The results are merged with the existing ones, where for each region only the one with the highest CNN confidence is kept. If there still exist regions with a low recognition score, and the maximum number of retires is not exceeded, another image is requested and processed as already described. Should the number of retries be exceeded, the process stops and the user is informed about the low score regions. The application experiments, show the input infrared images and the CNN output for three data, which also includes the additionally requested image. Our approach to locate, segment, and classify vehicles, as well as text localization and convolutional neural network (CNN). This research contains an evaluation of all these steps, where we primarily focus on the classification of vehicles, and show experimental results obtained using an CNN device. First, we evaluate the individual features that are extracted from the data. They are tested separately and combined, with different parameters for the machine learning algorithm. Next, the output of the CNN is compared to the actual data contents. Since the used CNN is off-the-shelf software with no runtime tuning parameters (just training), only the effect of training with the actual fonts that appear on the data is evaluated. Finally, we test three available data types on an CNN device and show the results. This includes different lighting conditions, such as shadows and reflections, and their impact on the processing steps. Furthermore, we show a number of breakdown cases, such as extreme angles when capturing the image, or blurriness caused by insufficient lighting.

The dataset used for the experiments consists of several infrared images of vehicles from 15 classes. The number of samples is relatively low, because it is not easy to acquire infrared images of real world aged data for all the different types available. Eleven classes are composed of infrared images of aged data, that are mounted on deployed devices. These data are for the most part thirty or forty years old. The final four classes (k, l, m and n) are composed of new data and were photographed using an CNN device. Due to the low number of samples, all tests are conducted using leave-one-out cross-validation, where each sample (from first to last) is classified while the rest is used to train the classifier.

4. RESULTS

This chapter is structured as steps of our approach to detect, classify and extract text from vehicle-data from the vehicle. Contours are used to roughly locate the data and to create a mask for its extraction using CNN. The feature vector used for classification of the extracted data consists of local binary patterns, median color values, and the size ratio, and is used in conjunction with a convolutional neural network. The positions of the text regions that should be extracted for each type are mapped to the text regions of the extracted vehicle. Each region is preprocessed before it is passed to the *CNN*, which improves the detection result. Finally, should the *CNN* recognition score be low, a new image of the problematic regions is requested from the user and processed again. In the next section, the proposed steps are evaluated, with special attention placed on the vehicle classification part. Additionally, some results under different lighting conditions, as well as breakdown cases are shown. First, the features are combined pairwise and then the complete feature vector for the classifier is created by concatenating all three. For each combination, we show the classification results.

The feature vector, calculated for each data and passed to the convolutional neural network for classification, consists of N individual features. During training, the convolutional neural network chooses m features from the feature vector at each node and uses the best split to split the data. The number of features m, which remains constant for the entire forest, influences the correlation between the layers and the classification strength of each tree. The classification performance of the forest increases with decreasing correlation between the layers and an increasing number of strong classifiers. Further, increasing m increases the strength of the individual classifiers and the correlation between them, while decreasing it has the opposite effect. The research lists the precision and recall values for three typical values of active variables when drawing the splits, where the square root of the number of variables is a frequently used value. Therefore, we use this value from now on. The research shows the pairwise combinations of the three features. It can be seen that the *CNN* features contribute the most to the result, while the size ratio contributes the least. The precision difference between the final feature vector and only *CNN* features is 0.26%, while the recall remains the same. Since the dataset contains only a small amount

of infrared images, which are slightly blurry at the most, the color information contributes very little to the classification result. To simulate the effect of blurred data with distorted content, we apply median blurring to the infrared images. The research lists the classification results for the entire dataset with the complete feature vector, as well as *LBP* features only, for different median blur kernel sizes. For this test, we do not use leave-one-out cross validation. Instead, all infrared images from the dataset are used to initially train the classifier. Then, all infrared images are blurred before the feature vector used for classification is extracted. As such, every image is classified correctly with both tested feature vectors if no blurring is applied.

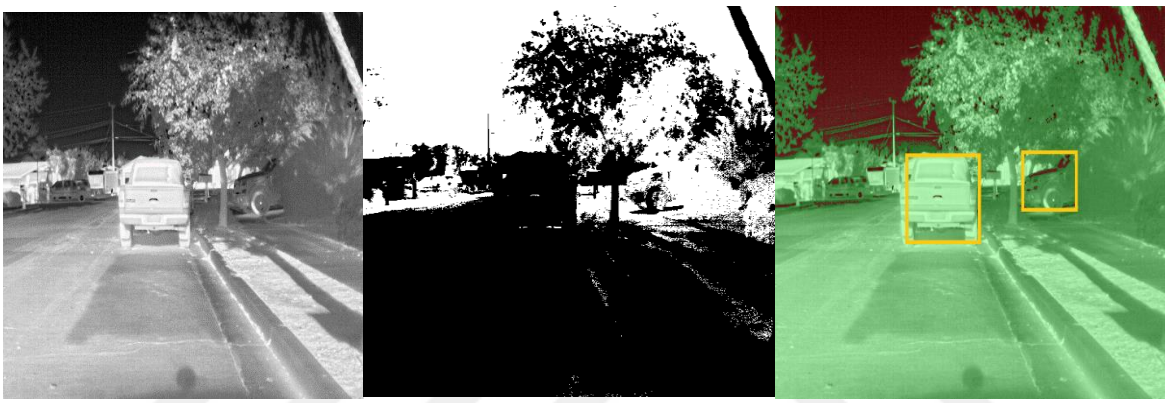

Figure 4.1: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 1.

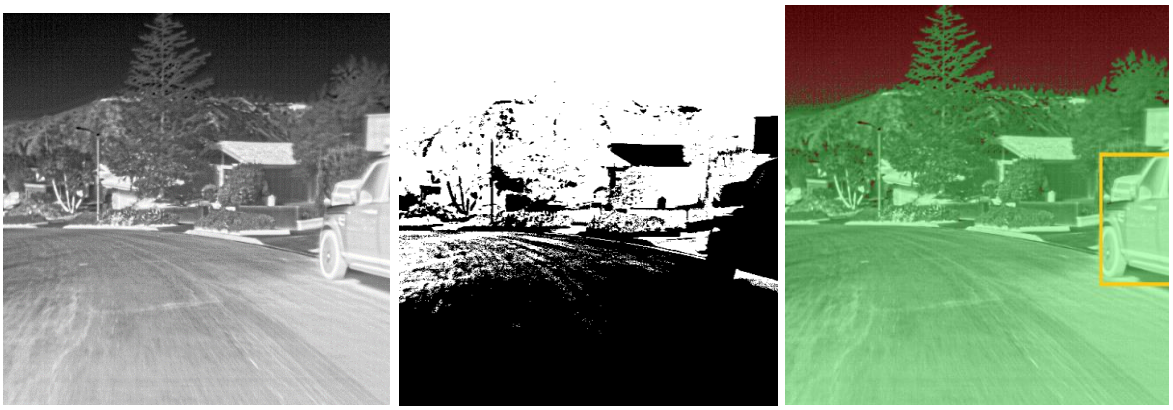

Figure 4.2: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 2.

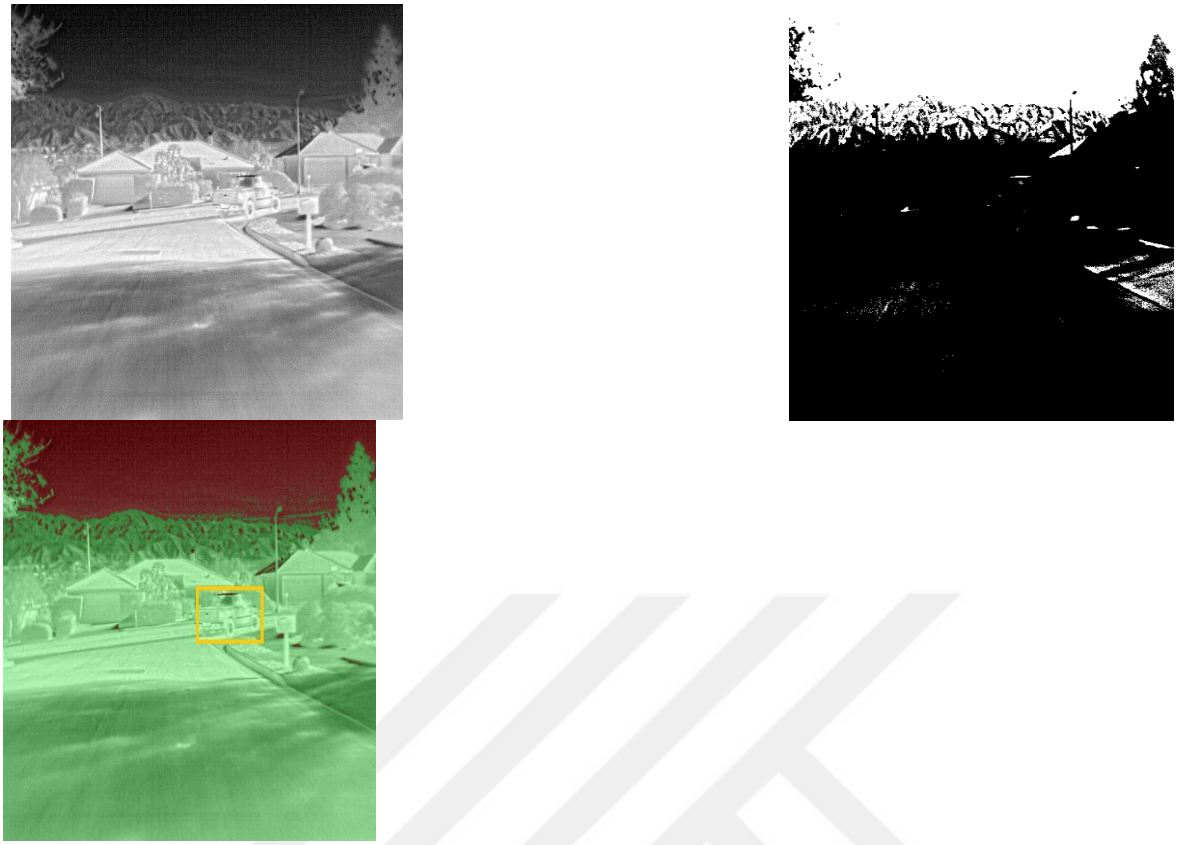

Figure 4.3: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 3.

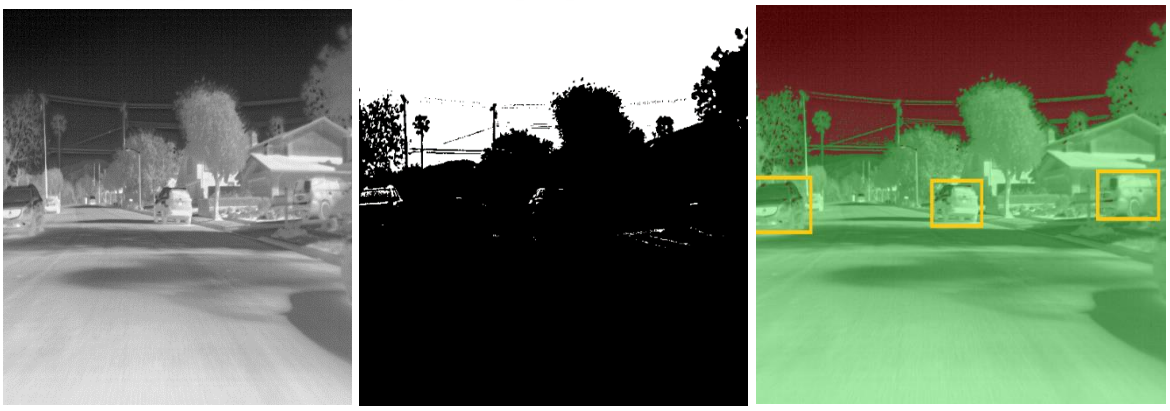

Figure 4.4: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 4.

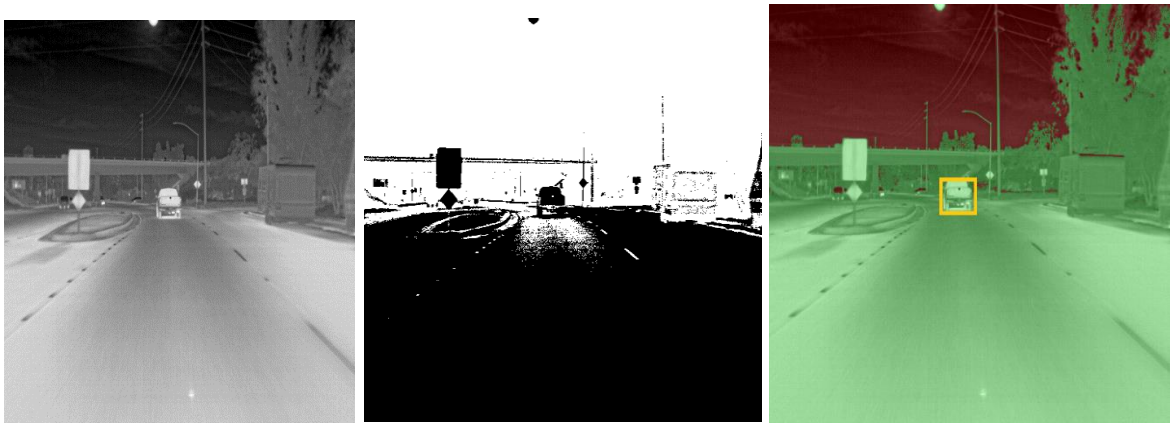

Figure 4.5: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 5.

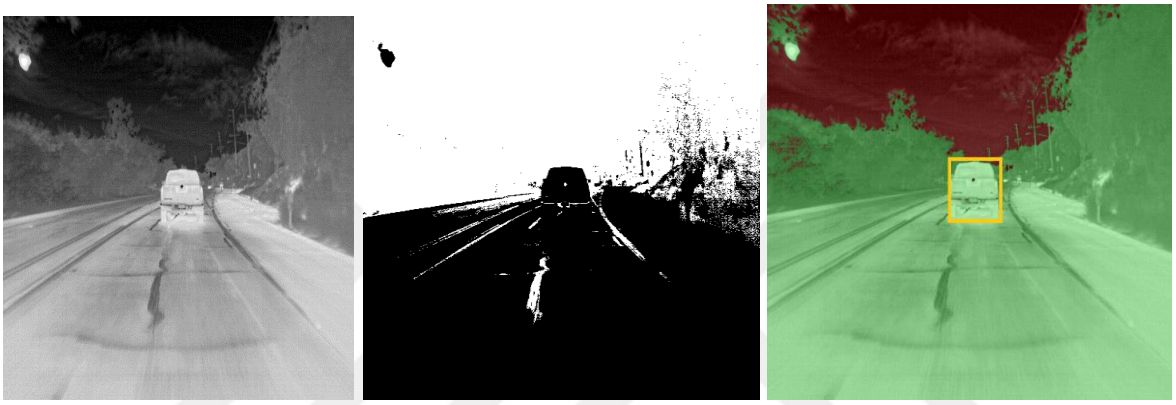

Figure 4.6: The input image of vehicle, segmented image of vehicle and detected vehicle being cropped with region of interest (ROI) with enhanced kernel size on sample 6.

The process of assigning the vehicle class to an image that contains a rendering of the vehicle, is called Convolutional neural network (CNN). This research work examined contain their own CNN approaches. We use the format of vehicle data to separate digits from characters and topological sorting to reduce the number of vehicle, to which the input is compared. A convolutional neural network is used to classify the image. The research employs a two stage classifier that determines the result by comparing a number of extracted features from the input to a set of data.

Furthermore, while the text is processed, a second classifier is trained with the output of the first one. After the first classifier has processed the entire page, the second classifier is used to refine the result. Finally, use a CNN to classify the extracted *MSER* regions of the vehicle data. As previously mentioned, these approaches are unlikely to provide good results in our setting, because the research exploit a number of constrains that do not hold for

vehicles. Examples are a known format of the string, only digits or no digits in words, and a limited number of fonts and character classes. For our approach, we decided to use the CNN, designed to process scanned sheets of infrared image.

In general, the output for the original infrared images is always incorrect or even missing. In contrast, the output for each preprocessed region is correct. However, there also exist cases where the preprocessing modified the actual content. In two examples, parts of the characters are removed, while in the other two examples, the commas are removed. It should be noted, that only in one case the output of the unmodified image is correct, while the preprocessed one is not. It should be noted that we disable the internal dictionary, as most of the strings we need to process are arbitrary combinations of digits and letters. Therefore, using the dictionary would worsen the result by altering the correctly detected characters. The Tesseract project provides a number of language files, trained with fonts that are common in scanned documents. Unfortunately, the shape of the characters of the fonts used for training are different from the ones used on most vehicles, which results in wrong output. Furthermore, the thresholding algorithm, applied internally by Tesseract to the input image, is not designed to handle uneven illumination and noise, which results in mostly unusable output. However, first problem can be tackled by training a new language with character samples from the unknown font. To reduce the issues caused by noise and problematic illumination, we apply our preprocessing step.

5. DISCUSSION

This research work produces good results, and could be widely used. It initially started as a research project, but it is now available under an open source vehicle. First, connected components are extracted and grouped into lines. The line slope is tracked over the entire image region, therefore de-skewing of the input image with its associated loss of resolution is not needed. For each line, the initial baseline is calculated using characters that are continuously placed. The CNN is able to process curved baselines which occur frequently in scanned documents. Next, the lines are separated into characters. In case of fonts with a fixed character width, the gaps are analyzed and split. Otherwise, a limited region above the baseline is analyzed, separated, and classified. The words with the lowest recognition score are split into further characters, until the result improves. If this does not improve the recognition score, contours are merged and classified again to check the change in the recognition score. The features for classification are extracted from small fixed size patches over neighboring contours and matched against the trained character representations. The research note that this approach enables Tesseract to easily handle characters broken into multiple parts, which would not be possible if each contour is processed separately. The following classification is done in two steps, the first selects possible candidates which are compared to each other in the second stage to retrieve the result. The detection result is also improved by incorporating a dictionary. The infrared images are first resized to a width of 800 pixels, as described previously, and then the median blurring is applied to the scaled infrared images. This is necessary to obtain comparable results, as the sizes of the infrared images in the dataset vary greatly. The results show that the complete feature vector always outperforms the one that consists only of CNN features. With a blur kernel size of 9 ×9, a single image is classified with CNN features, while with the complete feature vector, all infrared images are recognized correctly. With increasing kernel size, the number of correctly classified infrared images decreases for both feature vectors. However, for the three largest blur kernel sizes in the table, the difference of correctly classified infrared images remains about the same. It is largest with a kernel size of 29 × 29, where the classification result obtained with the complete feature vector includes 13 additional correctly classified infrared images. The other two cases, with kernel sizes of 19 × 19 and 39 × 39, show a difference of 10 infrared images. An

unmodified and a blurred image for each kernel size, where all blurred infrared images are classified correctly with the complete feature vector, but are wrong with the feature vector that contains only CNN values. It should be noted, that with infrared images blurred to such an extent, processing will fail in the later stages, as no infrared image is readable anymore. Nevertheless, we included these results to show the robustness of the classifier. Furthermore, it is possible that such infrared images are present during regular usage, e.g., if the camera is out of focus, or the operator moves while capturing the image, which results in motion blur. In these cases, the CNN will fail for all regions, but the user will be prompted to capture additional infrared images, which can then be used to generate usable output. Deep learning associated with odd indexes hops are increasing chirps while chirps associated with even indexes hops are decreasing chirps. The first is an increasing chirp, the second is decreasing, third is increasing and the fourth is decreasing. This makes any two consecutive chirps to be orthogonal. It has lower power consumption compared to the methods that depend on using cameras. Using ultrasonic waves for series recognition has a lower computational complexity compared to camera-based algorithms. Also, it is insensitive to the illumination of the environment and ensures privacy. Deep learning classification is required in many devices that have low processing powers, and for that reason having a ranging system that requires low computational cost is needed. There are methods to estimate the impulse response of a wireless channel with low computational cost. In CNNs, there are layers based on convolution. Instead of having layers of neurons, like in fully connected networks, one layer has multiple kernels and all the values in a kernel are trainable parameters. A forward pass consists of applying convolution with each of the kernels in a layer, in addition to adding a bias for each kernel. Convolution with one kernel produces one channel in the output image. This is because the kernel covers all channels in the input image, but only output one value at a time resulting in one channel. The output channels are stacked depth-wise, resulting in a 2D image with a channel for each kernel; assuming the input data is a 2D image with channels

The final classification results with the complete feature vector and the parameters determined using grid search. Errors occur with the type, where some of the samples contain dark regions, that cover large portions of the data, as well as bright reflections, which also cover a large area. As these conditions are present in most infrared images, they

cause large variations in the samples of this class. These issues are caused by the surface of the data, which is highly reflective. Therefore, the quality of the captured image changes with the lighting conditions, as well as the region on the data on which the camera focuses. Four infrared images from the dataset of this data class, where the first two show reflections, and the last two dark regions at the borders. We should note that these infrared images are less problematic for infrared image extraction, as our gradient based approach is robust against brightness variations. Nevertheless, should the reflections or dark regions cause issues, more infrared images will be requested where the operator can move to a different position, such that the infrared image becomes readable.

Table 5.1: The comparison of accuracy of recognition with existing and proposed techniques for vehicle data recognition.

ARTICLE	TECHNIQUE	ACCURACY
[61]	Support Vector Machine (SVM)	80.00%
[62]	Recurrent Neural Network (RNN)	95.70%
Proposed	Convolutional Neural Network (CNN)	98.69%

A convolutional neural network can estimate the importance of the individual features in the feature vector after training. To calculate the variable importance, research proposes four different measures, which are defined in the manual for the reference convolutional neural network implementation. The values for the tested feature, of the samples that were not chosen by the bagging procedure, are randomly manipulated and classified again by the CNN. For each tree, the number of votes for the correct class of the original and the modified samples are stored. The difference (i.e., the decrease in accuracy) is averaged over all layers and indicates the importance of the tested feature. However, the approaches proposed are problematic because they can produce misleading results. The research performs a number of simulation studies, that show that the variable importance is influenced by the number of layers in the neural network, which is selectable by the user, but not the sample size. Nevertheless, we chose to show the results of the estimated variable importance for informational purposes. Even if the values are not completely reliable, they still give an indication which of the three main features extracted from the data, are ranked

as most important by the forest. Examining the type of the first 100 highest ranked features shows that they consist of more than 100 CNN values and different color values. The size ratio is located at the end with a very low score, which matches our observations when combining the features pairwise. Furthermore, including the CNN features in the feature vector results in smaller neural networks, because the neural network accuracy is reached before the maximum number of layers are generated. The time (average over five runs) needed to train a neural network with all samples amount seconds. During classification, first the feature vector must be extracted. The time needed depends on the size ratio of the data, as the image is resized to a fixed width, but the height is scaled to preserve the aspect ratio. The proposed CNN network uses 12 convolutional layers as well as 3 fully connected layers. The classification of brain tumor in brain tumor sample images is quite different from classification in ImageNet. ImageNet requires classification of everyday objects such as animals, household items, etc., while brain tumor sample images are medical data where the objective is detecting and classifying the vehicles. Therefore, it is safe to assume the fully connected classification layers were trained from scratch and it is highly likely the deeper convolutional layers were replaced as well. This is where looking back at the training evaluation can be useful and indeed; the observation that smaller training losses still correlated with improvement in testing performance do not permit any conclusion towards an overfitting hypothesis. This makes these devices expensive. Active devices have high accuracy and low complexity, but they can be uncomfortable since it needs physical contact between the user and the device. While passive devices are computationally demanding, they don’t require physical contact, and they are friendly compared to active devices. It can be observed that the general tendency is for imaging networks to better fit the data than feedforward ones. Secondly, the plot also shows that networks that had two layers retrained could reach lower training losses than the model in which a single layer was retrained.

6. CONCLUSION

In this Master's Thesis, a method to detect, classify, and extract infrared image from a known set of vehicles using deep leaning technique was presented. The intended use case involved an operator that captures data attached to vehicle data components, such as cropping the data or generating the region of interest (ROI), using an image processing techniques where the extracted content is processing with deep learning based CNN technique. The vehicle data is localized in the input image, segmented, and classified using machine learning. The classification stage uses color information, histograms of Local Binary Patterns (LBP), and the data size ratio as features to determine using CNN. For each type, the layout of the data and so the positions of the labeled infrared image regions were given. The list of defined fields is matched against the detected fields on the data to account for possible inaccuracies introduced by the data extraction. The adjusted regions are preprocessed to remove noise, and passed to the Convolutional neural network (CNN). The preprocessing of the CNN extracts connected components, filters them by size, and groups them into lines of infrared image. The employed CNN is the well-known CNN technique. If the CNN reported a low confidence score for at least one infrared image region, the user is signaled to acquire a new image of that part of the data, which is then matched to the existing image and the infrared image processing steps are executed again. The process completes, if all regions have a sufficiently high score or a fixed number of retries are exhausted. The output consists of the detected data type, the list of regions including the extracted infrared image and the CNN reliability through CNN with an accuracy of 98.69% with 300 epochs of training and testing. The proposed system has been trained on 70% of data while 20% is used for testing and remaining 10% for validation. To show the significance of the designed feature vector, we used open source dataset consisting of multiple classes which represented a subset of all existing types of vehicles. It contains aged, as well as new types. The results showed, that the local binary pattern histograms contribute most to the classification result, followed by the color features, and finally the data size ratio. Evaluation of the combined feature vector showed the robustness of the classification under different lighting conditions, such as the presence of shadows and specular reflections, as well as blurriness and incorrect colors, caused by insufficient illumination. Furthermore, we demonstrated that training the CNN using the specific font of

the test data could reduce the number of recognition errors. The remaining errors were caused by the infrared image region preprocessing, which removes hyphens, commas, and dots, in case they are small, compared to the infrared image region, or merged with the borders around the infrared image. This is caused by the filtering parameters, which are tuned for aged data. The very restrictive infrared image region preprocessing also removes letters that were partly chipped off or faded. Further, the data localization may be inaccurate or fail if there is no clear boundary between the data and the background. Such cases occur if the borders of the data are overpainted, or dirt and moss accumulate. Testing on an CNN with three available data showed good results, if the CNN was trained for the fonts on the vehicles. Highly reflective data are difficult to capture and require manual focus control, which is not implemented in the prototype. However, it is unlikely that many such data are mounted to devices, because if a data is exposed to environmental conditions for a time, the surface will tarnish quickly. Overall, the developed prototype is able to extract the requested information from data and execution time is suitable for actual usage.

6.1. FURTHER WORK

We have shown that the proposed CNN algorithm is usable in real world conditions. However, we have some ideas how to further improve it. The segmentation could be enhanced by incorporating information acquired from the data infrared images that are used to train the deep convolutional neural network. Examples are the positions of infrared image or logos, which can be used to estimate the data boundary more precisely. Furthermore, the evaluation of the classifier shows that two classes are sometimes misclassified. The first class consists only of three visually distinctive infrared images, which makes it very hard to classify them correctly. The second one also contains a lot of variance in the form of reflections. To improve the accuracy, no changes to the classification approach are necessary, but a larger number of samples are needed for training. Finally, some improvements are possible for the last stages, preprocessing and CNN. It is highly unlikely to always achieve perfect output, because some infrared image (especially embossed images or content of very old data) is difficult to recognize even by a human. A possibility would be to individually preprocess different types of marking. For example,

embossed infrared image is much fainter, but the metal around the letters contains a lot of noise that is caused by the imprinting. Further, knowledge about the format of the field contents could be used to automatically verify and correct the detected content, such as decimal separators of digits or hyphens in serial numbers. While the employed CNN is generally regarded as the best open-source solution,

REFERENCES

[1] Rizvi, S.; Patti, D.; Björklund, T.; Cabodi, G.; Francini, G. Deep Classifiers-Based Vehicle Data Detection, Localization and Recognition on GPU-Powered Mobile Platform. Future Internet 2017, 9, 66.

[2] Rafique, M.A.; Pedrycz, W.; Jeon, M. Vehicle vehicle data detection using region-based convolutional neural networks. Soft Comput. 2018, 22, 6429–6440.

[3] Salau, A.O.; Yesufu, T.K.; Ogundare, B.S. Vehicle data number localization using a modified GrabCut algorithm. J. King Saud Univ. Comput. Inf. Sci. 2019.

[4] Kakani, B.V.; Gandhi, D.; Jani, S. Improved CNN based automatic vehicle number data recognition using features trained neural network. In Proceedings of the 2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Delhi, India, 3–5 July 2017.

[5] Arafat, M.Y.; Khairuddin, A.S.M.; Paramesran, R. A Vehicular Vehicle Data Recognition Framework For Skewed Infrared images. KSII Trans. Internet Inf. Syst. 2018, 12.

[6] Available Online: https://melabglobal.com/blogs/news/why-use-vehicle-data-recognition-parking-system

[7] Ansari, N.N.; Singh, A.K.; Student, M.T. Vehicle Number Data Recognition using Temdata Matching. Int. J. Comput. Trends Technol. 2016, 35, 175–178.

[8] 12. Samma, H.; Lim, C.P.; Saleh, J.M.; Suandi, S.A. A memetic-based fuzzy support vector machine model and its application to vehicle data recognition. Memetic Comput. 2016, 8, 235–251.

[9] Tabrizi, S.S.; Cavus, N. A Hybrid KNN-SVM Model for Iranian Vehicle Data

Recognition. Procedia Comput. Sci. 2016, 102, 588–594.

[10] Available Online: https://medium.datadriveninvestor.com/convolutional-neural-networks-explained-7fafea4de9c9

[11] Niu, B.; Huang, L.; Hu, J. Hybrid Method for Vehicle Data Detection from Natural Scene Infrared images. In Proceedings of the First International Conference on Information Science and Electronic Technology, Wuhan, China, 21–22 March 2015; Atlantis Press: Paris, France, 2015.

[12] Arafat, M.Y.; Khairuddin, A.S.M.; Khairuddin, U.; Paramesran, R. Systematic review on vehicular licence data recognition framework in intelligent transport systems. IET Intell. Transp. Syst. 2019.

[13] Chai, D.; Zuo, Y. Extraction, Segmentation and Recognition of Vehicle's Vehicle Data Numbers. In Advances in Information and Communication Networks; Arai, K., Kapoor, S., Bhatia, R., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 724–732.

[14] Dhar, P.; Guha, S.; Biswas, T.; Abedin, M.Z. A System Design for Vehicle Data Recognition by Using Edge Detection and Convolution Neural Network. In Proceedings of the 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2), Rajshahi, Bangladesh, 8–9 February 2018; pp. 1–4.

[15] Wang, W. et al. "Car vehicle data detection based on MSER." 2011 International Conference on Consumer Electronics, Communications and Networks (CECNet) (2011): 3973-3976.

[16] Brillantes, A.K.M.; Bandala, A.A.; Dadios, E.P.; Jose, J.A. Detection of Fonts and Characters with Hybrid Graphic-Text Data Numbers. In Proceedings of the

TENCON 2018—2018 IEEE Region 10 Conference, Jeju, South Korea, 28–31 October 2018; pp. 629–633.

[17] S. Du, M. Ibrahim, M. Shehata and W. Badawy, “Automatic Vehicle Data Recognition (ALPR): a state-of-the-art review,” IEEE Transactions on Circuits and Systems for Video Technology 23(2), pp. 311-325, 2013.

[18] Sayanan Sivaraman and Mohan M. Trivedi, "A General Active Learning Framework for On-road Vehicle Recognition and Tracking," IEEE Transactions on Intelligent Transportation Systems, 2010.

[19] S. Silva and C. Jung, “Real-time Brazilian vehicle data detection and recognition using deep convolutional neural networks,” in 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), 2017.

[20] Matas, Jiri & Zimmermann, Karel. (2005). Unconstrained licence data and text localization and recognition. 225 - 230. 10.1109/ITSC.2005.1520111.

[21] R. Girshick, “Fast R-CNN,” in IEEE International Conference on Computer Vision (ICCV), 2015.

[22] . Ren, K. He, R. Girshick and J. Sun, “Faster R-CNN: towards real-time vehicle detection with region proposal networks,” in Conference on Neural Information Processing Systems (NIPS), 2015.

[23] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu and A. Berg, “SSD: single shot multibox detector,” in The European Conference on Computer Vision (ECCV), 2016.

[24] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi and A. C. Berg, “Dssd: deconvolutional single shot detector,” arXiv preprint arXiv:1701.06659, 2017.

[25] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan and C. L. Z. P.

Dollár, “Microsoft COCO: Common Vehicles in Context,” in The European Conference on Computer Vision (ECCV), 2014.

[26] J. R. Uijlings, K. E. v. d. Sande, T. Gevers and A. W. Smeulders, “Selective search for vehicle recognition,” International Journal of Computer Vision (IJCV), 2013.

[27] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus and Y. LeCun, “OverFeat: integrated recognition, localization and detection using convolutional networks,” arXiv preprint arXiv:1312.6229, 2013.

[28] J. Redmon and A. Farhadi, “YOLOv3: an incremental improvement,” arXiv preprint arXiv:1804.02767, 2018.

[29] T.-Y. Lin, P. Goyal, R. Girshick, K. He and P. Doll´ar, “Focal loss for dense vehicle detection,” in IEEE International Conference on Computer Vision (ICCV), 2017.

[30] H. Law and J. Deng, “CornerNet: detecting vehicles as paired keypoints,” in The European Conference on Computer Vision (ECCV), 2018.

[31] C. Zhu, Y. He and M. Savvides, “Feature selective anchor-free module for single-shot vehicle detection,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[32] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in International Conference on Learning Representations (ICLR), 2015.

[33] K. He, X. Zhang, S. Ren and J. Sun, “Deep residual learning for image recognition,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[34] A. Newell, K. Yang and J. Deng, “Stacked hourglass networks for human pose

estimation," in The European Conference on Computer Vision (ECCV), 2016.

[35] O. Ronneberger, P. Fischer and T. Brox, "U-net: convolutional networks for biomedical image segmentation," in International Conference on Medical image computing and computer-assisted intervention, 2015.

[36] Y. Lin, P. Doll´ar, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature pyramid networks for vehicle detection," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[37] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[38] S.-H. Kuo, J.-E. Ok and E.-Y. Cha, "Vehicle Front-Rear detection for vehicle data detection enhancement," in 15th International Conference on Multimedia Information Technology and Applications(MITA), 2019.

[39] Available Online: https://www.chegg.com/homework-help/questions-and-answers/hello-m-trying-detect-driving-lanes-using-hough-transform-matlab-think-ve-successful-excep-q28180023

[40] Available Online: https://gitmemory.com/issue/AlexeyAB/darknet/1492/480806996

[41] A. Shrivastava, A. Gupta and R. Girshick, "Training region-based vehicle detectors with online hard example mining," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[42] Z. Xu, W. Yang, A. Meng, N. Lu and H. Huang, "Towards end-to-end vehicle data detection and recognition: A Large dataset and baseline," in The European Conference on Computer Vision (ECCV), 2018.

[43] 18. Zhu, Z.; Ren, X.; Chen, Z. Integrated detection and tracking of workforce and

equipment from construction jobsite videos. Autom. Constr. 2017, 81, 161–171.

[44] Rezazadeh Azar, E.; McCabe, B. Part based model and spatial–temporal reasoning to recognize hydraulic excavators in construction infrared images and videos. Autom. Constr. 2012, 24, 194–202.

[45] Kim, J.; Chi, S. Action recognition of earthmoving excavators based on sequential pattern analysis of visual features and operation cycles. Autom. Constr. 2019, 104, 255–264.

[46] Akhavian, R.; Behzadan, A.H. Construction equipment activity recognition for simulation input modeling using mobile sensors and machine learning classifiers. Adv. Eng. Inform. 2015, 29, 867–877.

[47] Kim, J.; Chi, S.; Seo, J. Interaction analysis for vision-based activity identification of earthmoving excavators and dump trucks. Autom. Constr. 2018, 87, 297–308.

[48] Gong, J.; Caldas Carlos, H. Computer Vision-Based Video Interpretation Model for Automated Productivity Analysis of Construction Operations. J. Comput. Civ. Eng. 2010, 24, 252–263.

[49] Silva, S.M.; Jung, C.R. Real-Time Vehicle Data Detection and Recognition Using Deep Convolutional Neural Networks. J. Vis. Commun. Image Represent. 2020, 71, 102773.

[50] Omar, N.; Sengur, A.; Al-Ali, S.G.S. Cascaded deep learning-based efficient approach for vehicle data detection and recognition. Expert Syst. Appl. 2020, 149, 113280.

[51] Wang, D.; Tian, Y.; Geng, W.; Zhao, L.; Gong, C. LPR-Net: Recognizing Chinese vehicle data in complex environments. Pattern Recognit. Lett. 2020, 130, 148–156.

[52] Hsu, G.; Chen, J.; Chung, Y. Application-Oriented Vehicle Data Recognition.

IEEE Trans. Veh. Technol. 2013, 62, 552–561.

[53] Masood, S.Z.; Shu, G.; Dehghan, A.; Ortiz, E.G. Vehicle data detection and recognition using deeply learned convolutional neural networks. arXiv 2017, arXiv:1703.07330.

[54] Saraswathi, S.; Subban, R.; Shanmugasundari, T.; Manogari, S. Research on Vehicle Data Recognition Using Digital Image Processing. In Proceedings of the 2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Coimbatore, India, 14–16 December 2017; pp. 1–6.

[55] Khan, A.M.; Awan, S.M.; Arif, M.; Mahmood, Z.; Khan, G.Z. A Robust Segmentation Free Vehicle Data Recognition Method. In Proceedings of the 2019 International Conference on Electrical, Communication, and Computer Engineering (ICECCE), Swat, Pakistan, 24–25 July 2019; pp. 1–6.

[56] Liu, L.; Ouyang, W.; Wang, X.; Fieguth, P.; Chen, J.; Liu, X.; Pietikäinen, M. Deep Learning for Generic Object Detection: A Survey. Int. J. Comput. Vis. 2020, 128, 261–318.

[57] Puarungroj, W.; Boonsirisumpun, N. Thai Vehicle Data Recognition Based on Deep Learning. Procedia Comput. Sci. 2018, 135, 214–221.

[58] Cao, Y.; Fu, H.; Ma, H. An End-to-End Neural Network for Multi-Line Vehicle Data Recognition. In Proceedings of the 2018 24th International Conference on Pattern Recognition (ICPR), Beijing, China, 20–24 August 2018; pp. 3698–3703.

[59] Chen, R.C. Automatic Vehicle Data Recognition via sliding-window darknet-YOLO deep learning. Image Vis. Comput. 2019, 87, 47 – 56.

[60] Available Online: https://medium.com/@anand.mishra/using-segmentation-analysis-to-improve-marketing-10b4a95bf62c

[61] Abdullah, Saman & Omar, Khairuddin & Sahran, Shahnorbanun & Khalid, Marzuki. (2009). Vehicle data recognition based on Support Vector Machine. 78 - 82. 10.1109/ICEEI.2009.5254811.

[62] Usmankhujaev, Saidrasul & Lee, Seonwoo & Kwon, Jangwoo. (2020). Korean Vehicle Data Recognition System Using Combined Neural Networks. 10.1007/978-3-030-23887-2_2.