UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Viktoria Plemakova

Vehicle Detection Based on Convolutional Neural Networks

Master's Thesis (30 ECTS)

Supervisor: Amnir Hadachi, PhD

Tartu 2018

Vehicle Detection Based on Convolutional Neural Networks

Abstract:

Accurate vehicle detection or classification plays an important role in Intelligent Transportations Systems. Ability to detect vehicles in traffic scenes allows analyzing drivers' behavior as well as detect traffic offenses and accidents. Detection and classification of vehicles is a challenging task due to weather and light conditions and vehicle type diversity. Several solutions use feature extraction algorithms along with support vector machine classifier. However, convolutional neural networks have proved to be potentially more effective. In this thesis, we present a convolutional neural network trained to classify and detect vehicles from multiple angles. Moreover, Fast Fourier Transform is used during data preprocessing. The effect of such preprocessing is examined on the developed vehicle classifier and detector.

Keywords:
Neural network, vehicle detection, vehicle classification, Fast Fourier Transform

CERCS: P170 Computer science, numerical analysis, systems, control

Sõidukite tuvastus kasutades konvolutsioonilisi närvivõrke

Lühikokkuvõte:

Sõidukite täpne tuvastus ja klassifitseerimine mängib suurt rolli intelligentsete transpordi süsteemide valdkonnas. Sõidukite tuvastus liikluses võimaldab analüüsida autojuhtide käitumist ning lisaks tuvastada liikluseeskirja rikkumisi ja liiklusõnnetusi. Sõidukite tuvastus ja klassifitseerimine on keeruline ülesanne erinevate valgustingimuste, ilmastikunähtuste ja sõidukite mitmekesisuse tõttu. Mitmed olemasolevad lahendused kasutavad tunnuste eraldamise algoritme ning tugivektorklassifitseerijat. Hiljuti on konvolutsioonilised närvivõrgud osutunud paremaks lahenduseks. Antud lõputöö esitleb konvolutsioonilist tehisnärvivõrku, mis suudab liigitada ja tuvastada sõidukeid erinevate nurkade alt. Peale selle eeltöödeldakse andmeid kiire Fourier' teisenduse abil. Välja pakutud eeltöötluse mõju uuritakse selle lõputöö käigus valminud sõidukite tuvastus- ja klassifitseerimisprogrammi abil.

Võtmesõnad:
Tehisnärvivõrk, sõidukite tuvastus, sõidukite klassifitseerimine, kiire Fourier' teisendus

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Contents

1 Introduction **6**

1.1 General view 6

1.2 Objectives 7

1.3 Contribution 7

1.4 Road map 8

2 State-of-the-art **9**

2.1 Introduction 9

2.2 Object detection 10

2.3 Object classification 11

2.4 Vehicle detection and classification 12

2.4.1 Using feature-based methods 12

2.4.2 Using convolutional neural networks 14

2.5 Conclusion 15

3 Methodology **16**

3.1 Introduction 16

3.2 Dataset 16

3.3 Preprocessing 17

3.3.1 Resizing images 18

3.3.2 Data normalization 18

3.3.3 Data augmentation 18

3.3.4 Fast Fourier Transform 19

3.4 Convolutional Neural Network 20

3.4.1 Activation function 21

3.4.2 Convolutional layer 21

3.4.3 Pooling layer 22

3.4.4 Fully-connected layer 24

3.4.5 Proposed network architecture 24
3.5 Conclusion 25

4 Results and discussion **26**
4.1 Vehicle classification 26
4.2 Vehicle detection 29
4.3 Conclusion 33

5 Conclusion **35**
5.1 Future perspectives 36

References **42**

Appendix **43**
I. Licence 43

1 Introduction

1.1 General view

Object detection, tracking and classification can be used for various purposes. In the Intelligent Transportation Systems (ITS) field object detection is utilized for vehicle and pedestrian detection, traffic sign and lane detection or vehicle make detection. Ability to detect or classify traffic related objects makes it possible to further improve the state of the roads and traffic flow, prevent serious traffic accidents and even register traffic violations and crimes, such as stolen vehicles or speeding [1]. This is especially important since the number of passenger car users is constantly rising. Besides, lately, the topic of autonomous vehicles has been gaining popularity.

It is easy for humans to recognize vehicles in images or videos or distinguish between different car types. For a computer program, the difficulty of vehicle detection and classification depends a lot on the type of the data. Lighting conditions and weather are one of the main challenges not to mention the overall quality of either images or video. Vehicles come in different shapes and colors, and some models might be even slightly similar. Moreover, it is definitely more challenging to detect a lot of moving objects in real time.

Nevertheless, there are many techniques for vehicle detection and classification. A lot of works approach these tasks by first extracting relevant features and then using a classifier like Support Vector Machine (SVM) to make further decisions. Since computer vision field is constantly evolving some new methods are starting to stand out. In this thesis, we investigate convolutional neural networks and how they can be applied to computer vision problems.

1.2 Objectives

The goal of this thesis is to develop a convolutional neural network (CNN) to perform vehicle detection and classification on vehicle and background images. More precisely the objectives are as follows:

- Implement a classifier that is able to predict correct image classes: vehicles or non-vehicles.
- Implement a vehicle detector that has to predict bounding box coordinates of vehicles.
- Use Fast Fourier Transform (FFT) during data preprocessing.
- Investigate whether FFT improves or reduces the accuracy of the developed solution.

In this thesis, the detection is limited to finding one vehicle per each input image.

1.3 Contribution

The contributions include a CNN for vehicle detection and classification. In addition to that, investigating the effect of FFT on input data. To achieve that the following steps have to be done:

- Finding the vehicle and background datasets.
- Applying preprocessing techniques to the collected data. This step includes the experimental preprocesing with FFT.
- Developing a CNN architecture suitable for the dataset and the goal of this thesis. Experimenting with different hyperparameters is essential to get a good model.
- Training the CNN to do vehicle classification and detection.
- Testing the final solution.

1.4 Road map

This thesis consists of five chapters in total. The rest of the thesis is structured as follows:

- Chapter 2 introduces existing solutions that use CNNs for object detection as well as vehicle detection and classification. Additionally, feature based vehicle detection and classification methods are described.
- In Chapter 3 the classification and detection pipeline of this thesis is presented. This chapter describes the dataset used for the experiments as well as the preprocessing steps applied to the data. More importantly, the building blocks of CNNs are explained along with the proposed network architecture.
- In Chapter 4 the results of the experiments are presented. This chapter includes the outcomes of using a basic CNN and a CNN with FFT preprocessing for vehicle classification and detection.
- Lastly, Chapter 5 concludes the work of this thesis and future work perspectives are presented.

2 State-of-the-art

This chapter provides an overview of how CNNs are applied to object detection tasks. Additionally, this chapter presents some solutions that use CNNs for vehicle detection and classification. Moreover, feature-based methods for vehicle detection and classification are described.

2.1 Introduction

Object detection and classification play important role in computer vision field. To achieve good results both object detection and classification need, however, a lot of preliminary work. This includes image pre-processing such as removing noise, adjusting contrast, re-sizing, and background subtraction. Next, feasible features have to be detected and extracted. For instance, either Scale-Invariant Feature Transform (SIFT) [2], Speeded-Up Robust Features (SURF) [3] or Histogram of Oriented Gradients (HOG) [4] can be used to generate points of interest. For a long time, object detection and classification depended on these hand-crafted features. Consequently, selecting good features is crucial to achieve high accuracy [5].

However, lately, interest in deep learning techniques, such as CNNs, has increased. Although the first CNN dates back to the 1990s [6], it gained popularity largely after AlexNet [7] won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). Overall, CNNs have achieved outstanding results not only within computer vision field but also within speech recognition and natural language processing fields. CNNs' advantages come from needing no extensive pre-processing [8] and hand-crafted features.

Intelligent transportation systems (ITS) rely a lot on object detection and classification. For instance, object detection is helpful in analyzing traffic flow and behavior or identifying traffic accidents [1]. There are various factors that may make the recognition process more difficult, such as varying lighting and weather conditions, the number of different

vehicle types in one image or the distance of captured vehicles.

This chapter introduces existing literature on object detection combined with CNNs. Moreover, this chapter describes various vehicle detection and classification methods. It includes older methods that are based on feature extractors and support vector machine classification as well as methods that use CNNs.

2.2 Object detection

In 2013 Girshick et al. [9] introduced R-CNN, a CNN with region proposals. In their paper, the authors wish to show that CNN is capable of achieving better results than methods using low-level features such as HOG. Their object detection pipeline consists of three parts. First of all, R-CNN uses selective search to create region proposals, i.e. various regions that might include some object. Next, 4096-dimensional feature vectors are extracted from each detected region using AlexNet. The only change in the architecture is the number of units in the classification layer. Afterward, each feature vector is classified using SVMs where each SVM has been trained for a specific class. On the ILSVRC2013 detection dataset, R-CNN scored better than Overfeat.

Sermanet et al. [10] come up with a neural network that can be used for three computer vision tasks: object classification, localization and detection. In addition to that, a feature extractor, called Overfeat, is released. The authors aim to prove that CNNs are capable of doing more than classification. AlexNet is used as the base network for classifications. The modifications done to AlexNet include non-overlapping pooling, smaller stride size in the first two layers and omitting contrast normalization. For the localization task, the fully-connected layers of the classification model are changed to a regression network. The regression network is trained to output four bounding box coordinates. The classifier and the regression network are run simultaneously. Then, their results are merged to get the final predictions. The detection task is similar to the localization, but background class is added for images that do not contain any objects.

YOLO [11] is one of most recent object detection approaches. Some previous works use classifiers to perform object detection. In contrast, the proposed method predicts bounding box coordinates along with class score within the same neural network. Unlike region proposal-based networks YOLO uses the whole image instead of separate regions to make decisions. The network architecture is based on GoogleLeNet where inception modules are replaced by reduction layers. Twenty convolutional layers are first pre-trained and then another four convolutional and two fully-connected layers are added to the model. Each input image is divided into a grid. Each grid cell predicts bounding box coordinates, a bounding box confidence and a class probability.

Ming Liang and Xiaolin Hu [12] propose a recurrent CNN. This solution is inspired by recurrent connections in recurrent neural networks. The authors develop a recurrent convolutional layer that is used instead of regular convolutional layers in the proposed model. Only the first convolutional layer of the network is not recurrent. The base network and training process were adapted from AlexNet. The depth of the network is increased by adding recurrent connections.

2.3 Object classification

In 2012 Krizhevsky et al. [7] make a breakthrough in large dataset image classification by introducing AlexNet. While there are several others machine learning methods used at the competition, the authors prove that CNNs are capable of managing millions of images and achieve high results. The key to final results is the depth of AlexNet. There is no preprocessing done on the dataset except for subtracting the mean. The proposed network consists of five convolutional and three fully-connected layers. The authors choose ReLU for the activation function since it drastically improved the speed of the training. Local response normalization is applied after the activation in first two layers. Moreover, overlapping pooling helped to reduce overfitting during the training. The final layer is a softmax layer with 1000 units. The training is done on two GPUs where each GPU has access to different layers. Data augmentation and dropout are used to reduce

overfitting of the network.

In 2014 [13] another successful CNN architecture was introduced – VGGNet. With this neural network, the authors won the first and the second places in the classification and localization tasks of the ILSVRC'14 competition. The base of the network is inspired by AlexNet. Similarly to AlexNet, the only preprocessing was mean extraction. The authors experimented with different network depths to understand how it affects the final results. One of the differences from previous architectures is the use of smaller kernel size 3×3 in all convolutional layers. In addition to that, the CNN contains multiple sequential convolutional layers without pooling layer in between. The authors tested six different configurations of different depth. The best results were achieved when the network consisted of 16 and 19 layers.

CNN-RNN [14] is a convolutional neural network combined with a recurrent neural network that is used for multi-label classification. The goal of multi-label classification is to predict the labels of multiple objects in the same image. The authors observed that adding RNN to the original network increased the accuracy. Long short term memory units are used as recurrent units of this network. The CNN part of the network is based on VGGNet and it is used to collect semantic information from images. The RNN part deals with the relationship of images and labels.

2.4 Vehicle detection and classification

This section gives a overview of vehicle detection and classification methods that either use CNNs or feature-based methods.

2.4.1 Using feature-based methods

Vehicle detection and classification has various solutions using descriptors like SIFT, SURF and HOG Compared to CNNs the features extracted by previously mentioned

methods are rather low-level. Typically a support vector machine (SVM) is used as the classifier.

In [15] the authors use a combination of SIFT and bag-of-words and perform the classification of vehicle make and model using SVM. Another approach first subtracts the background around the vehicle, then the image is segmented and SIFT is used to extract the features [16]. The authors perform feature matching to evaluate the results. Ma and Grimson [17] "adopt SIFT, with several key modifications tuned to vehicle classification, as descriptors for edge points".

Another vehicle make and model recognition is performed using a modified SURF algorithm – a symmetrical SURF descriptor [18]. The idea of this modification is to identify whether the input contains symmetrical objects. The proposed method works well with real-time data and does not need background subtraction.

X. Li and X. Guo [19] developed a forward vehicle detection system that uses HOG descriptor and SVM. Their idea relies on detecting shadows underneath vehicles to distinguish between vehicles and background. Moreover, this method proved to perform well in different lighting conditions. Similarly to the previous research, Sivaraman and Trivedi [20] take advantage of shadows underneath vehicles. The features are extracted using HOG and the proposed symmetrical HOG that detects symmetrical features. In their research Feng Han et. al [21] use HOG together with SVM to detect people and vehicles from different viewpoints. Firstly, the authors use stereo cue to detect either person or vehicle candidates. Then, the classification using HOG features is done with SVM. For each viewpoint, the authors develop separate classifiers that are applied simultaneously to get a joined result. Pablo Negri et. al [22] utilize HOG and Haar-like features in their vehicle detection research. In fact, the authors experiment with merging these descriptors and achieve accurate final results.

2.4.2 Using convolutional neural networks

Dong et al. [23] propose a semi-supervised CNN for detecting vehicles from frontal view images. The filters, used in convolutional layers, are learned using unlabeled data and the proposed sparse Laplacian filter learning. The output layer is a softmax classifier that is trained on a small labeled data. The final solution proved to be effective when used to classify vehicle types. A year before, Dong et al. [24] introduced unsupervised CNN for vehicle classification. They use CNN to learn features and then classify vehicles using softmax regression. The filters in their network are learned with sparse filtering method.

In [25] the authors use CNN with low resolution video frames to detect and classify vehicles. The pre-processing stage includes re-sizing each frame and adjusting the contrast with histogram equalization. The proposed CNN architecture detects both high-level and low-level features. The authors vary the number of filters and filter sizes as well as the number of hidden layers. In the paper by Heikki Huttunen et al. [26] a simple CNN outperforms the SIFT and SVM method when applied to vehicle classification.

Transfer learning is another approach used for vehicle detection and classification. It is a method where an already pre-trained model is used as a starting point or as a feature extractor. If needed the existing model can be fine-tuned further. In [27] the authors use fine-tuned YOLO [11] for vehicle detection and fine-tuned AlexNet model for vehicle classification. The authors additionally use AlexNet as feature extractor and perform classification on extracted features using linear SVM.

Jorge E. Espinosa et al. [28] compare the performance of AlexNet and Faster R-CNN [29]. Furthermore, Faster R-CNN was fine-tuned and modified by Quarfu Fan et al. [30] to perform better on vehicle detection. In [31] Fast R-CNN [32], the predecessor of Faster R-CNN, has been applied to vehicle detection, although the authors simplified it for their task.

2.5 Conclusion

This chapter gave an overview of existing solutions in object detection field that utilize CNNs. Furthermore, two types of vehicle detection and classification approaches were introduced: feature-based and CNN based methods. Feature-based approaches depend on the precision of hand engineered features that are extracted using feature descriptors such as HOG, SIFT or SURF. Therefore, over past few years, several researchers have chosen CNNs over previously mentioned feature extractors. The related work includes a variety of CNN architectures, thus showing that CNN is a versatile tool within computer vision field.

3 Methodology

3.1 Introduction

It can be concluded from Chapter 2 that CNNs are a popular and useful tool for computer vision tasks. The aim of this work is to utilize CNNs to perform both vehicle classification and detection. Moreover, another focus of this work is how FFT affects the accuracy of a CNN. This chapter gives an overview of the methods behind the developed application.

The work process consists of four steps as illustrated in Figure 1. To begin with, the dataset is introduced. Next, this chapter describes the preprocessing techniques used in the implementation. The main difference with other common CNN preprocessing pipelines is the FFT usage. Then, the idea behind CNNs is introduced along with the proposed network architecture. Lastly, the network is trained and tested for both object classification and detection.

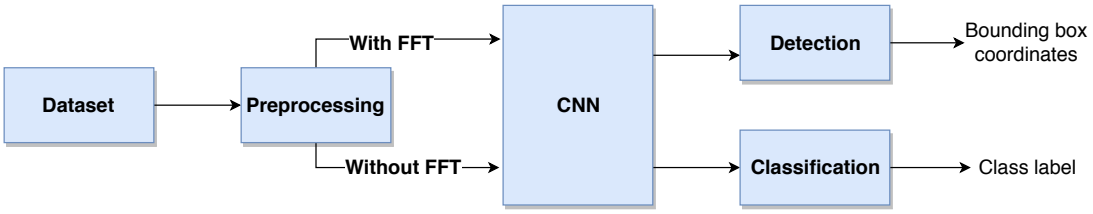

Figure 1. The pipline of the proposed solution.

3.2 Dataset

The dataset [33] by CVLab – EPFL contains 2299 images of cars from multiple angles as seen in Figure 2. The images of rotating cars were captured during the Geneva International Motor Show'08. There are at least 100 images of 20 different car models. Additionally, the dataset provides bounding box coordinates for each image. The

data for non-car samples is combined from multiple datasets: GRAZ-02 [34], Caltech Cars (Rear) background dataset [35] and INRIA car dataset [36]. In total there are 1866 various background images that illustrate highways, parking lots or buildings (see Figure 2).

Combining the vehicle and background images together results in a dataset with 4164 images. The dataset is split into training and testing set using 80:20 ratio. In the end, there are 3331 training samples and 833 testing samples.

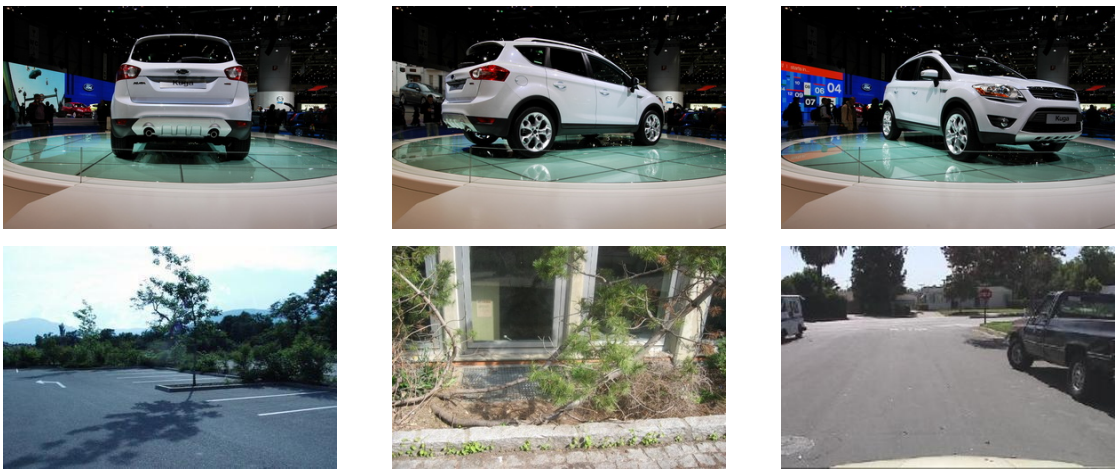

Figure 2. Sample images from the final dataset. The top row illustrates a vehicle from different angles. The bottom row contains random background images from non-car datasets.

3.3 Preprocessing

This section describes the data preprocessing steps of the final solution. This includes resizing and normalizing the data, generating more training data and finally applying FFT to images.

3.3.1 Resizing images

The dataset used in this thesis contains images of various sizes. This is due to combining the final dataset from different existing images. CNNs usually expect equally sized input images, thus the car and background images are all resized to 128×128. Although it is not necessary, most known CNN architectures seem to prefer square shaped input images. It is also necessary to scale the bounding boxes to gain accurate coordinates after the input images are resized.

3.3.2 Data normalization

All of the images are normalized using the standardization method. It is done by first subtracting the mean from every feature and then diving by standard deviation:

$$z = \frac{x - \mu}{\sigma}$$

where μ is the mean and σ is the standard deviation. The mean and standard deviation are computed over the training set and the same values are used for normalizing the test set [37]. Standardization ensures that the features are zero-centered and have a standard deviation of 1.

3.3.3 Data augmentation

The aim of data augmentation is to generate more training data from an already existing dataset. The size of a dataset has a great impact on the accuracy of a model and usually more data results in better models. The problem with small datasets is that models tend to overfit quicker [38]. Regularization can be used to solve this problem, such as adding dropout or L1/L2 regularization to a CNN. Data augmentation is another suitable solution for this problem [38, 37].

The process of data augmentation includes transforming original images in different ways to produce new data. For example, each input image can be rotated, zoomed in or

out, flipped vertically or horizontally, shifted in random directions. This way a CNN will learn from a wider variety of training examples. In this thesis, data augmentation is used only during the training phase of the vehicle classifier. The transformations that are used include zooming in and out, shifting both vertically and horizontally and shearing. Flipping vehicles and roads vertically would not make sense and horizontal flip is unnecessary since, for example, the vehicles are already depicted from multiple angles.

3.3.4 Fast Fourier Transform

This section is based on information from [39] and [40]. Fast Fourier Transform (FFT) is an algorithm to speed up the calculation of Discrete Fourier Transform (DFT). Although widely used in signal processing FFT can also be used with images. For example, to reduce noise in images. For an image of size $M \times N$ the 2-D DFT is defined by:

$$F(u,v) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) \exp\left[-i2\pi\left(\frac{mu}{M} + \frac{nv}{N}\right)\right]$$

where $f(m,n)$ is the original image and $F(u,v)$ is the image in the frequency domain. Similarly, the inverse of DFT is given by:

$$F(m,n) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} f(u,v) \exp\left[i2\pi\left(\frac{mu}{M} + \frac{nv}{N}\right)\right].$$

The inverse function transforms the image back to the spatial domain. In fact, since 2-D DFT is separable, the final output can be computed by first computing 1-D DFT for every image row and then computing 1-D DFT for every column.

The output of 2-D DFT is complex-valued, i.e. consists of both real and imaginary parts. Therefore, the range of values is larger than that of the original image. Either logarithmic or square root function (see Equation (1) and (2)) can be applied to the output for better visualization:

$$Q(u,v) = log(|F(u,v)|) \tag{1}$$

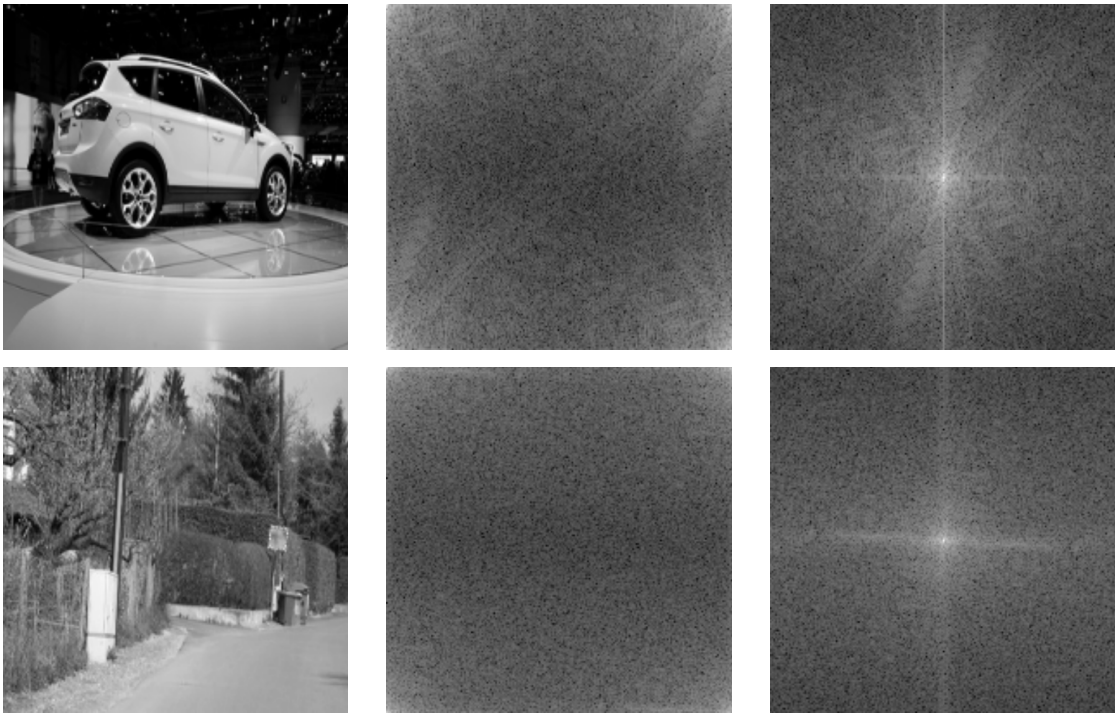

Figure 3. Visualizing spectrum after applying FFT to images. The first column contains original images. The second column illustrates the spectrums after computing FFT. The last column is the shifted spectrums.

$$Q(u,v) = \sqrt{|F(u,v)|}. \tag{2}$$

The DC value $F(0,0)$ is often shifted to the center of an image, as illustrated in Figure 3. Doing so displays low frequencies in the center of the image and high frequencies further away from it.

3.4 Convolutional Neural Network

Convolutional Neural Network (CNN) is a machine learning technique that is inspired by the human brain. In 1989 and 1990 Yann LeCun et al. [41, 6] introduced several aspects of modern CNNs, such as feature maps, sub-sampling, and shared weights. The

proposed method successfully recognized handwritten digits. The famous LeNet-5 [8] was presented in 1998, the CNN designed to perform digit recognition as well.

Since then multiple new CNN architectures have emerged, e.g. AlexNet [7], VGGNet [13], GoogLeNet [42], ZF Net [43], and ResNet [44]. One thing these networks have in common is that they are all trained on the ImageNet data [45] for the ILSVRC [46]. While LeNet-5 architecture was rather simple, its successors experiment with more deep network structures. From this, it can be concluded that deep network architectures improve the accuracy in certain cases. Despite the differences in network architectures, however, all CNNs contain similar components described further. The sample CNN architecture is illustrated in Figure 4.

3.4.1 Activation function

Activation functions in deep learning models introduce non-linearity. In CNN models an activation function is applied after every convolutional layer. Most recent CNN architectures use rectified linear unit (ReLU):

$$f(x) = max(0, x)$$

Other possible activation function are sigmoid and hyperbolic tangent, illustrated in Equation (3) and Equation (4) respectively.

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

$$f(x) = tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{4}$$

3.4.2 Convolutional layer

Feature detection and extraction happens in convolutional layer. Each convolutional layer has its own set of kernels that are convolved with the layer's input resulting in multiple

feature maps. Let the input of ith feature map of layer l be x_i^l, w_{ij}^l the kernel and f the activation function, then

$$x_i^l = f(\sum_j w_{ij}^l * x_j^{l-1})$$

In this work, the input is made up of images. Thus, each input and feature map is three dimensional and has the size of $W \times H \times D$ [47]. The hyperparameters of a convolutional layer are the number of kernels and the stride and kernel sizes. If we convolve an input of size $W \times H \times D$ with N kernels of size $W_k \times H_k$ the size of the output feature map will be $W_o \times H_o \times N$ where $W_o = W - W_k + 1$ and $H_o = H - H_k + 1$ [47]. If the stride size S is higher than 1 then $W_o = \frac{W-W_k}{S} + 1$ and $H_o = \frac{H-H_k}{S} + 1$.

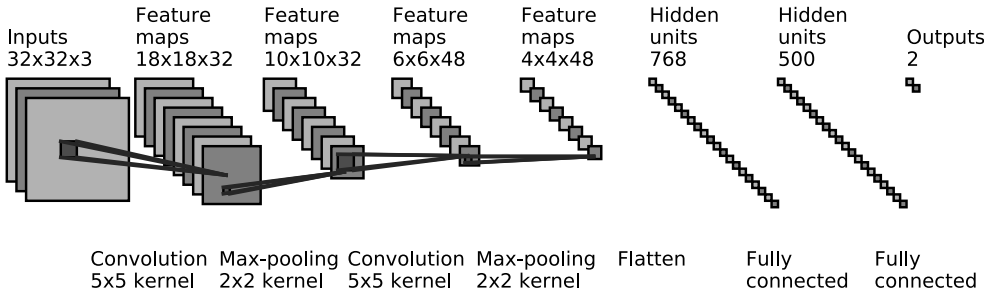

Figure 4. A sample CNN architecture[1]. This network consists of two convolutional layers, two pooling layers using the max pooling function and 2 fully-connected layers.

3.4.3 Pooling layer

A pooling layer, if used, usually follows a convolutional layer. As stated by LeCun et al. [8] the precise position of every detected feature is irrelevant. Important is the fact that some feature is present. Thus, pooling layers are used to eliminate unimportant

[1]This figure is generated by the code from `https://github.com/gwding/draw_convnet`

information from feature maps. This is achieved by using a pooling function, such as max pooling, average pooling or L^2-norm pooling [48]. Recently max pooling has become the most popular pooling function, although the original LeNet-5 [8] used average pooling.

The pooling function is applied to feature map regions of size $n \times n$. For example, a maximum value is chosen from each region to be included in the output feature map. The regions can be either overlapping or not depending on the stride size. The idea of max pooling is illustrated in Figure 5. If the size of a feature map is $W \times H \times N$, then after performing pooling of size $W_p \times H_p$ the output size will be $W_o \times H_o \times N$ where $W_o = W - W_p + 1$ and $H_o = H - H_p + 1$ [47]. Similarly to the convolution, if the stride size S is higher than 1 then $W_o = \frac{W - W_p}{S} + 1$ and $H_o = \frac{H - H_p}{S} + 1$.

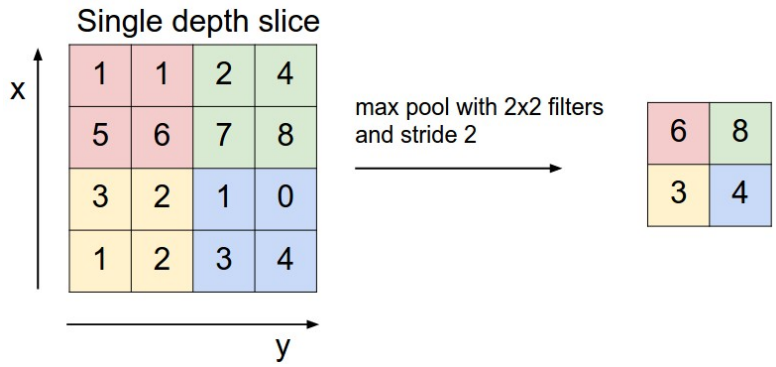

Figure 5. Example of 2×2 max pooling with a stride of 2 [47].

While most CNN architectures include pooling layers there has been proposed an all convolutional architecture [49]. The authors concluded that omitting max pooling layers does not ruin the accuracy if the stride of convolutional layers is increased. On the other hand, using pooling layers can help to reduce overfitting since it reduces the number of parameters as well as the size of the feature maps. For example, the authors of AlexNet observed that overlapping pooling helped to control overfitting.

3.4.4 Fully-connected layer

Fully-connected layers are used as final layers in CNNs. The last layer outputs the estimations. Multi-class networks use softmax classifier in the final layer while binary classification is done using the sigmoid function. Similarly to regular neural networks, the computational units in a fully-connected layer are connected to every unit of the previous layer. Unlike convolutional and pooling layers fully-connected layers are one-dimensional.

3.4.5 Proposed network architecture

The CNN architecture is developed using the Keras [50] deep learning framework with Python and Tensorflow backend. The same network is later used for vehicle classification and detection with minor changes in the output layer. The proposed CNN contains 6 convolutional layers and 5 max pooling layers. The only fully-connected layer is the final layer and the layer after the last convolutional layer. The input layer takes in batches of images of size $128 \times 128 \times 3$. ReLU is used between the hidden layers and a sigmoid activation is used in the output layers of both classification and detection networks.

The first convolutional layer has 32 filters of size 3×3. It is followed by a pooling layer that uses the max pooling operation with the size of 2×2. The second convolutional layer has 64 filters with a size of 3×3. Similarly to the first convolutional layer, it is followed by a max-pooling layer with the kernel size of 2×2. The third convolutional layer has 128 filters with the same kernel size as previous convolutional layer. The 2×2 max pooling is applied yet again. The fourth convolutional layer is identical to the third one but no max pooling is used. The final two convolutional layers have 256 filters of size 3×3 and they are followed by the same max pooling layer like previous convolutional layers. The parameters of the described layers are also illustrated in Table 1. The output layer has one unit in the classification task and predicts whether image is vehicle or not. The output layer of the detection network contains four units for bounding box coordinates.

Table 1. Parameters of convolutional and pooling layers.

	Kernel size	**Number of kernels**	**Stride size**	**Output size**
Convolution 1	3x3	32	1	126 x 126 x 32
Max pool 1	2x2	-	2	63 x 63 x 32
Convolution 2	3x3	64	1	61 x 61 x 32
Max pool 2	2x2	-	2	30 x 30 x 64
Convolution 3	3x3	128	1	28 x 28 x 128
Max pool 3	2x2	-	2	14 x 14 x 128
Convolution 4	3x3	128	1	12 x 12 x 128
Convolution 5	3x3	256	1	10 x 10 x 256
Max pool 4	2x2	-	2	5 x 5 x 256
Convolution 6	3x3	256	1	3 x 3 x 256
Max pool 5	2x2	-	2	1 x 1 x 256

3.5 Conclusion

This chapter described the main methods and steps of the implementation of vehicle recognition and classification. The workflow starts with obtaining a suitable dataset. Then, all of the images are preprocessed. Images are resized, and normalized. To obtain more training data, data augmentation method is used for the classification task. Additionally, FFT is applied to input images after previous preprocessing steps are done. Finally, the preprocessed data is fed to the proposed CNN. The output is either a label or four bounding box coordinates depending on the task.

4 Results and discussion

This chapter introduces the results of the experiments. Firstly, a CNN without FFT preprocessing and a CNN with FFT preprocessing are trained and tested for vehicle classification. The final classifier has to perform a binary classification between vehicles and non-vehicles. The second part involves the results of vehicle detection. The aim of the detection network is to either detect a vehicle or to understand that there is none present. The vehicle detection is also tested using a CNN without FFT and with FFT.

4.1 Vehicle classification

For vehicle classification task, the network had to distinguish between vehicles and non-vehicles (background). Two CNN networks were trained and tested: one CNN without FFT preprocessing step and one with FFT. Both networks were trained for 100 epochs using Adam optimizer and learning rate of 0.001. Loss function was binary cross-entropy due to binary classification. The training and testing set contained 3331 and 833 images respectively.

The confusion matrices of both networks are illustrated in Figure 7 and Figure 6. From both confusion matrices, it is obvious that the classification accuracy of both networks is almost identical. The only difference is the number of false negatives: the network without FFT processing has less false negatives. Recall and precision (see Equation (5) and (6)) are used as classification evaluation metrics:

$$recall = \frac{true\,positives}{true\,positives + false\,negatives} \tag{5}$$

$$precision = \frac{true\,positives}{true\,positives + false\,positives}. \tag{6}$$

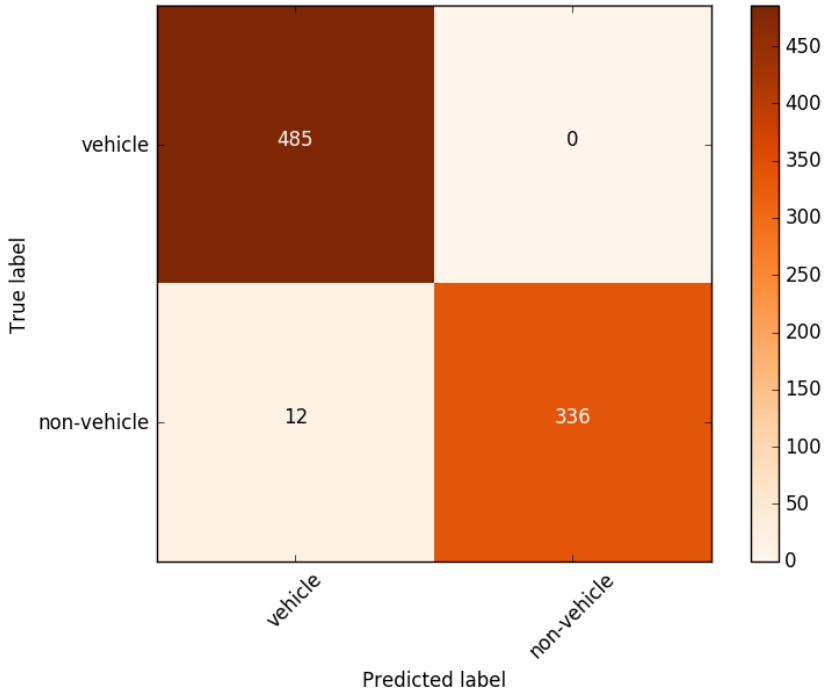

Figure 6. Confusion matrix of the CNN with FFT preprocessing.

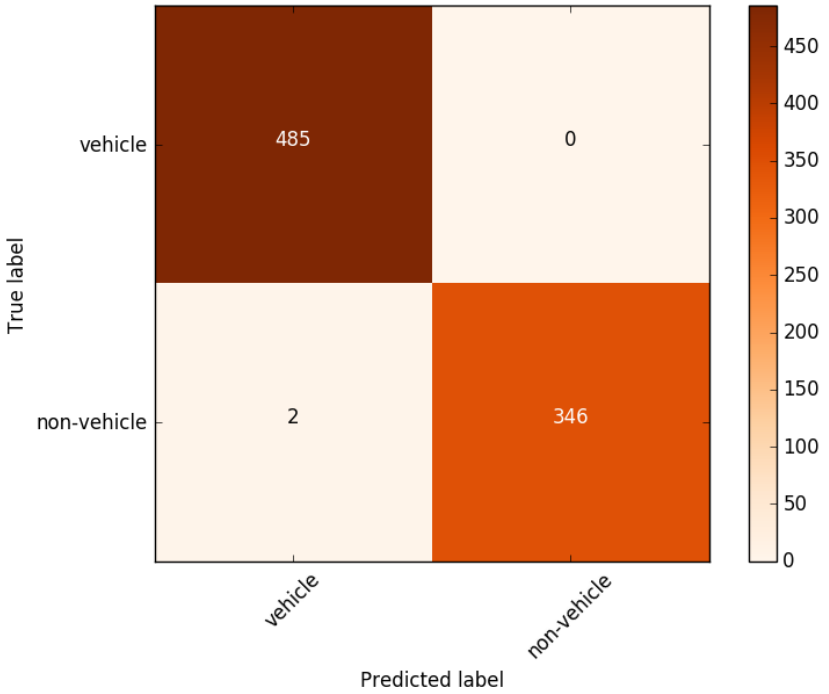

Figure 7. Confusion matrix of the CNN without FFT preprocessing.

The explanation of these results may lie in the quality of the dataset. The images of the vehicles have the same background of an exhibition hall while the background images contain mostly the scenes from the outside. As a consequence, it is not difficult to distinguish between the images that contain vehicles and those that do not. In other words, it would be harder for the network to learn from images that contain vehicles with a road in the background. The results are provided in Table 2.

Table 2. Precision and recall of both experiments.

	recall	**precision**
CNN with FFT	0,976	1
CNN without FFT	0,996	1

4.2 Vehicle detection

For vehicle detection, the proposed CNN was trained on vehicle and background data. The ground truth boxes of vehicles were provided with the dataset. For background, the whole image was considered correct detection. Similarly to the classification, two experiments were conducted: training and testing the CNN without FFT preprocessing step and with it. In both cases, the network was trained for 200 epochs using Adam optimizer and learning rate of 0.0001. Mean squared error was used as the loss function. The training set contained 3331 images and testing was performed on 833 samples.

Intersection over Union (IoU) is used to evaluate the detection results. The IoU algorithm computes how much the ground truth and predicted bounding boxes overlap. If the overlap value exceeds the 50% threshold, then the estimation is considered valid [51], true positive. Everything under the threshold is false positive. The area of intersection is divided by the area of union to compute the IoU [51]:

$$a_o = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})}.$$

As seen in Figure 8, the results of the CNN without FFT preprocessing are highly accurate on the test set: the predicted pink bounding boxes are almost completely overlapping with the ground truth boxes. Since the ground truth for the background images was the whole image, then no pink boxes in background images indicate the correct detection. From the results in Table 3, it is evident that there were no incorrect estimations on the test set judging by the IoU output.

Figure 8. Detection outputs of the CNN without FFT preprocessing. The green square indicates the ground truth bounding box and the pink one the predicted bounding box.

Table 3. Evaluation of CNN without FFT preprocessing.

IoU value	**Background detection**	**Vehicle detection**
IoU > 0.5	348	485
IoU < 0.5	0	0

The accuracy graph of the model is provided in Figure 9. As seen in the Figure, the testing accuracy is slightly higher than the training accuracy during approximately 40 epochs. The accuracy stops rising a bit before the 100th epoch. The final score of the model is 96% on the testing set.

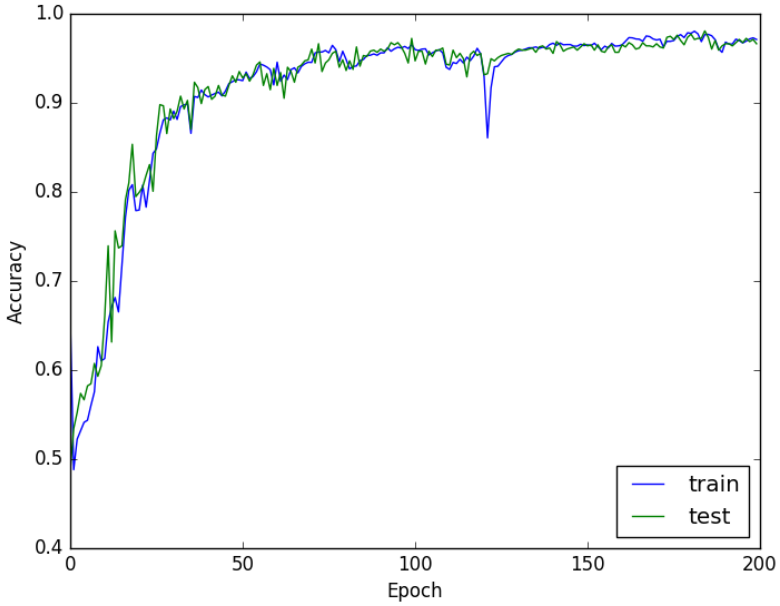

Figure 9. CNN without FFT accuracy on train and test sets.

The CNN with FFT preprocessing achieved considerable results as well. As illustrated in Figure 10, the network was mostly able to guess the position of vehicles in the test images. However, compared to the previous CNN the position of some bounding boxes is quite off. One example is the most rightmost image in the top row of Figure 10. Additionally, sometimes the network proposed bounding boxes in the background images as seen in Figure 10. Overall, based on the IoU values (see Table 4) the network predicted correctly 214 backgrounds and 373 vehicles.

The accuracy of this model is illustrated in Figure 11. The starting accuracy is slightly higher compared to the accuracy of the CNN without FFT. Nevertheless, the network's final accuracy also exceeds 90%. Similarly, it can be concluded that the accuracy did not

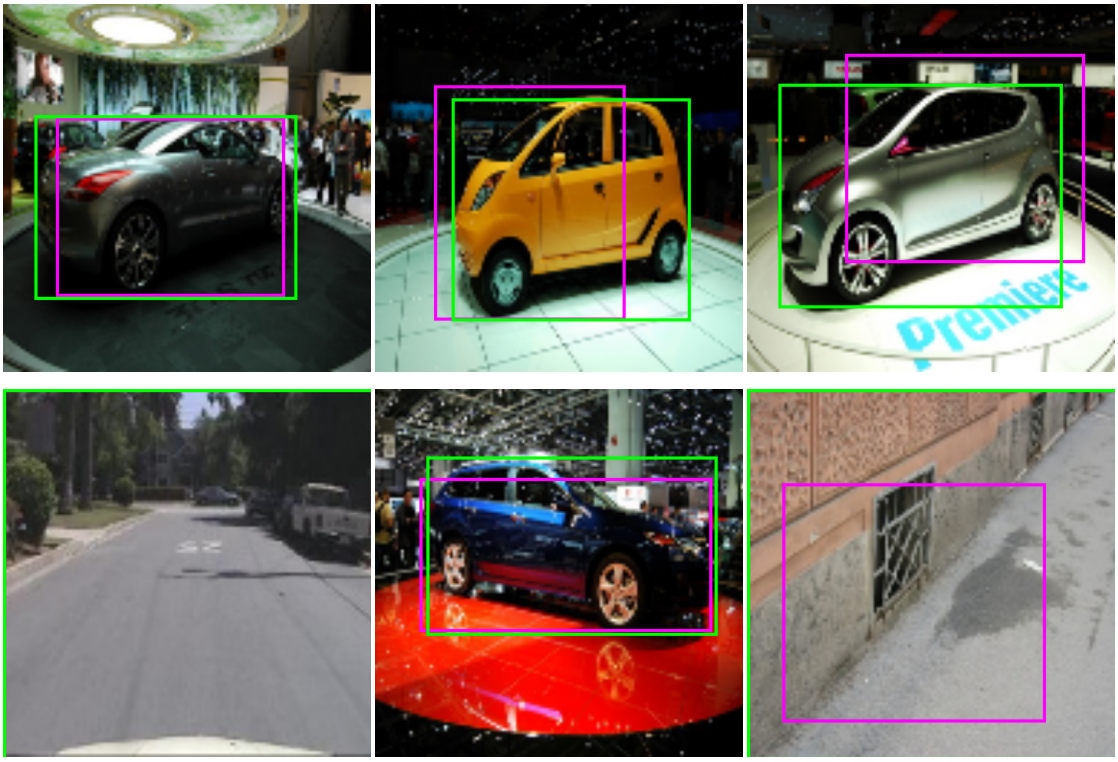

Figure 10. Detection outputs of the CNN with FFT preprocessing. The green rectangle indicates the groundtruth bounding box and the pink one the predicted bounding box.

Table 4. Evaluation of CNN with FFT preprocessing.

IoU value	Background detection	Vehicle detection
IoU > 0.5	214	373
IoU < 0.5	134	112

change a lot after approximately 100 epochs.

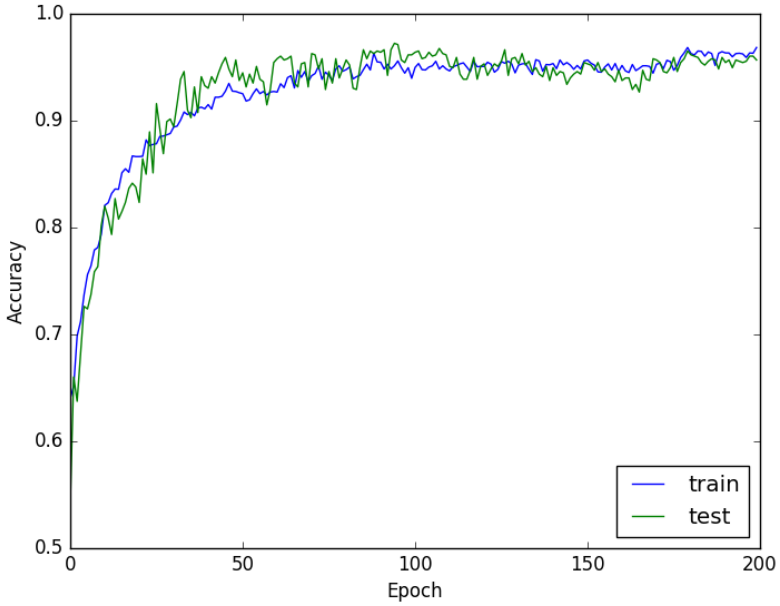

Figure 11. CNN + FFT accuracy on train and test sets.

4.3 Conclusion

This chapter included the results of vehicle classification and detection. One goal of the experiments was to investigate whether applying FFT to images in the preprocessing step could affect the accuracy of the network. To achieve that, one CNN was trained without using FFT to preprocess the images and the other one with FFT. Furthermore, the overall performance of the developed solution was of interest.

From the results, it appears that with the proposed network and chosen hyperparameters the FFT preprocessing did not affect the accuracy of the classifier. Both networks achieved high results and could easily distinguish between vehicles and non-vehicles. However, such high results may be caused by the dataset selection. The vehicle and background images illustrated different environments.

Interestingly enough the vehicle detection showed different results. Compared to the classifier the vehicle detector performed better when to FFT preprocessing was utilized. The latter neural network correctly detected every vehicle and background. Although the CNN with FFT preprocessing performed worse, it was able to detect over half test images.

5 Conclusion

Vehicle detection and classification have great influence on the advances in the field of Intelligent Transportations Systems. This computer vision task help in developing better road systems by analyzing the traffic and assist in preventing or detecting traffic accidents. While it is helpful in many ways, detecting and classifying vehicles is not an easy task.

This thesis investigates two main approaches to vehicle detection and classification. The related work shows that feature extraction algorithms paired with a classifier like SVM have been the method of choice for vehicle detection and classification for years. The main disadvantage is the complexity of extracting the right features and therefore, these solutions depend a lot on preprocessing. Fortunately, object detection and classification have become more accurate and fast with the rise of popularity of convolutional neural networks. These neural networks can produce high results on large datasets without the need of additional feature extraction and extensive preprocessing in most cases.

In this thesis, the vehicle detection and classification is done utilizing convolutional neural networks as they have proved to be a suitable choice for the task. The vehicle classification is a binary classification task where the network has to differentiate between two types of image data: vehicles and non-vehicles. For vehicle detection, the suggested neural network is trained to predict bounding box coordinates of the vehicles. This task involves understanding when there is no vehicle present in an image. The same network architecture is used to solve both problems. Minor changes are made to the structure of the output layer to either predict a label or coordinates.

In addition to regular preprocessing, such as data augmentation, resizing and normalizing images, this thesis proposes to apply Fast Fourier Transform to images. In other words, the images are processed with two-dimensional Discrete Fourier Transform. To see whether this improves the final results the experiments were carried out using inputs that were previously transformed with FFT and using the original images. Both developed

vehicle classifier and detector were tested in such a way.

Lastly, the results of the mentioned experiments were obtained. The convolutional networks were able to perform the tasks with both FFT preprocessing applied and without it. The classification precision and recall were high in case of both experiments. The detection, however, was performed better by the convolutional network where no FFT preprocessing was used. From this, it can be concluded that in current experiments the FFT preprocessing did not significantly affect the final results.

5.1 Future perspectives

The future work could include fine-tuning the current CNN architecture and FFT preprocessing. The current detection results with FFT preprocessing were somewhat worse than with a basic CNN. Consequently, the algorithm needs more work to be more efficient. Moreover, the thesis focused on detecting a single vehicle. Therefore in the future, the current CNN could be modified to detect multiple vehicles in images or videos.

References

[1] J. Zhang, F. Y. Wang, K. Wang, W. H. Lin, X. Xu, and C. Chen, "Data-Driven Intelligent Transportation Systems: A Survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, pp. 1624–1639, Dec 2011.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[4] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 886–893, IEEE, 2005.

[5] S. Srinivas, R. K. Sarvadevabhatla, K. R. Mopuri, N. Prabhu, S. S. S. Kruthiventi, and R. V. Babu, "A Taxonomy of Deep Convolutional Neural Nets for Computer Vision," *CoRR*, vol. abs/1601.06615, 2016.

[6] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in neural information processing systems*, pp. 396–404, 1990.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[9] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *CoRR*, vol. abs/1311.2524, 2013.

[10] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks," *CoRR*, vol. abs/1312.6229, 2013.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[12] M. Liang and X. Hu, "Recurrent convolutional neural network for object recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3367–3375, 2015.

[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[14] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "Cnn-rnn: A unified framework for multi-label image classification," in *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pp. 2285–2294, IEEE, 2016.

[15] M. A. Manzoor and Y. Morgan, "Vehicle Make and Model classification system using bag of SIFT features," in *Computing and Communication Workshop and Conference (CCWC), 2017 IEEE 7th Annual*, pp. 1–5, IEEE, 2017.

[16] M. C. Narhe and M. Nagmode, "Vehicle classification using SIFT," *International Journal of Engineering Research and Technology ESRSA Publications*, 2014.

[17] X. Ma and W. E. L. Grimson, "Edge-based rich representation for vehicle classification," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2, pp. 1185–1192, IEEE, 2005.

[18] J.-W. Hsieh, L.-C. Chen, and D.-Y. Chen, "Symmetrical SURF and its applications to vehicle detection and vehicle make and model recognition," *IEEE Transactions on intelligent transportation systems*, vol. 15, no. 1, pp. 6–20, 2014.

[19] X. Li and X. Guo, "A HOG feature and SVM based method for forward vehicle detection with single camera," in *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2013 5th International Conference on*, vol. 1, pp. 263–266, IEEE, 2013.

[20] S. Sivaraman and M. M. Trivedi, "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1773–1795, 2013.

[21] F. Han, Y. Shan, R. Cekander, H. S. Sawhney, and R. Kumar, "A two-stage approach to people and vehicle detection with HOG-based SVM," in *Performance Metrics for Intelligent Systems 2006 Workshop*, pp. 133–140, 2006.

[22] P. Negri, X. Clady, S. M. Hanif, and L. Prevost, "A cascade of boosted generative and discriminative classifiers for vehicle detection," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, p. 136, 2008.

[23] Z. Dong, Y. Wu, M. Pei, and Y. Jia, "Vehicle type classification using a semisupervised convolutional neural network," *IEEE transactions on intelligent transportation systems*, vol. 16, no. 4, pp. 2247–2256, 2015.

[24] Z. Dong, M. Pei, Y. He, T. Liu, Y. Dong, and Y. Jia, "Vehicle Type Classification Using Unsupervised Convolutional Neural Network," in *2014 22nd International Conference on Pattern Recognition*, pp. 172–177, Aug 2014.

[25] C. M. Bautista, C. A. Dy, M. I. Mañalac, R. A. Orbe, and M. Cordel, "Convolutional neural network for vehicle detection in low resolution traffic videos," in *Region 10 Symposium (TENSYMP), 2016 IEEE*, pp. 277–281, IEEE, 2016.

[26] H. Huttunen, F. S. Yancheshmeh, and K. Chen, "Car type recognition with deep neural networks," in *Intelligent Vehicles Symposium (IV), 2016 IEEE*, pp. 1115–1120, IEEE, 2016.

[27] Y. Zhou, H. Nejati, T.-T. Do, N.-M. Cheung, and L. Cheah, "Image-based vehicle analysis using deep neural network: A systematic study," in *Digital Signal Processing (DSP), 2016 IEEE International Conference on*, pp. 276–280, IEEE, 2016.

[28] J. E. Espinosa, S. A. Velastin, and J. W. Branch, "Vehicle Detection Using Alex Net and Faster R-CNN Deep Learning Models: A Comparative Study," in *International Visual Informatics Conference*, pp. 3–15, Springer, 2017.

[29] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, pp. 91–99, 2015.

[30] Q. Fan, L. Brown, and J. Smith, "A closer look at Faster R-CNN for vehicle detection," in *Intelligent Vehicles Symposium (IV), 2016 IEEE*, pp. 124–129, IEEE, 2016.

[31] S.-C. Hsu, C.-L. Huang, and C.-H. Chuang, "Vehicle Detection using Simplified Fast R-CNN,"

[32] R. B. Girshick, "Fast R-CNN," *CoRR*, vol. abs/1504.08083, 2015.

[33] M. Ozuysal, V. Lepetit, and P.Fua, "Pose Estimation for Category Specific Multiview Object Localization," in *Conference on Computer Vision and Pattern Recognition*, (Miami, FL), June 2009.

[34] A. Opelt, A. Pinz, M. Fussenegger, and P. Auer, "Generic object recognition with boosting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 3, pp. 416–431, 2006.

[35] R. Fergus, P. Perona, and A. Zisserman, "Object class recognition by unsupervised scale-invariant learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, (Madison, Wisconsin), pp. 264–271, June 2003.

[36] P. Carbonetto, G. Dorkó, C. Schmid, H. Kück, and N. De Freitas, "Learning to recognize objects with little supervision," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 219–237, 2008.

[37] F. Chollet, *Deep Learning with Python*.

[38] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," *CoRR*, vol. abs/1712.04621, 2017.

[39] M. Sonka, V. Hlavac, and R. Boyle, *Image processing, analysis, and machine vision*. Cengage Learning, 2014.

[40] R. Fisher, S. Perkins, A. Walker, and E. Wolfart, "Fourier transform." `https://homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm`.

[41] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[42] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, *et al.*, "Going deeper with convolutions,"

[43] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," *CoRR*, vol. abs/1311.2901, 2013.

[44] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[45] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255, IEEE, 2009.

[46] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[47] A. Karpathy, "Cs231n convolutional neural networks for visual recognition," *Neural networks*, vol. 1, 2016.

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

[49] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[50] F. Chollet *et al.*, "Keras." `https://keras.io`, 2015.

[51] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International journal of computer vision*, vol. 111, no. 1, pp. 98–136, 2015.

Appendix

I. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Viktoria Plemakova**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:

   1.1 reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

   1.2 make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

   of my thesis

   Vehicle Detection Based on Convolutional Neural Networks

   supervised by Amnir Hadachi, PhD

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, 21.05.2018