



UNIVERSIDADE D  
COIMBRA

Roberto Manuel Trindade Oliveira

**DEVELOPMENT OF A HIGHWAY TOLLING  
AND ENFORCEMENT SYSTEM USING  
CONVOLUTIONAL NEURAL NETWORKS  
AND FINE-GRAINED VISUAL  
CLASSIFICATION**

**Dissertação no âmbito do Mestrado Integrado em Engenharia  
Eletrotécnica e de Computadores, Ramo de Automação**

Outubro de 2020



DEPARTAMENTO DE  
ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES  
UNIVERSIDADE DE COIMBRA

DISSERTATION FOR THE PARTIAL FULFILLMENT OF THE  
DEGREE OF MASTER OF SCIENCE IN ELECTRICAL AND  
COMPUTER ENGINEERING

2019/2020

**Development of a Highway Tolling and  
Enforcement System Using Convolutional  
Neural Networks and Fine-Grained Visual  
Classification**

*Author:*

Roberto Manuel Trindade Oliveira

*Supervision by:*

Professor Jorge Manuel Moreira de Campos Pereira Batista, Phd

October, 2020



## List of Figures

1	<i>Example of parallel recognition systems for fraud detection. License plate and vehicle info are extracted and cross-examined. . . . .</i>	14
2	<i>SIFT and CNN comparison. Table taken from [14]. . . . .</i>	17
3	<i>VGG16's deep, yet simple, architecture. . . . .</i>	18
4	<i>A single Inception module. These are arranged sequentially and ultimately feed toward a fully-connect section that performs the final prediction. . . . .</i>	19
5	<i>A single Residual Block. Similarly to Inception, these blocks are arranged sequentially in variable depth sizes, leading to fully-connected layers. . . . .</i>	20
6	<i>A dense block. Previous layers feed to all subsequent layers in the block. Like Inception and residual networks, a variable sequential arrangement of these blocks is performed. Diagram taken from [18]. . . . .</i>	21
7	<i>Generalist binary confusion matrix example. . . . .</i>	24
8	<i>Specialized tasks within Object Recognition. Image taken from [53]. . . . .</i>	26
9	<i>Typical curve of error progression of training and testing data when overfitting is occurring, as seen in [20]. . . . .</i>	28
10	<i>Max Channel Pool (MCP) operator (left) vs Regular Max Pool (right). MCP results in a smaller depth of channels of feature maps. . . . .</i>	29
11	<i>Diagram representation of the solution put forward in [28]. . . . .</i>	30
12	<i>The hierarchical architecture of [29]. . . . .</i>	30
13	<i>The image quality enhancement stage of [34]. . . . .</i>	31
14	<i>The weakly supervised concept of [31]. . . . .</i>	31
15	<i>MV-CNN used in [36]. . . . .</i>	32
16	<i>MV-CNN architecture used in [37]. . . . .</i>	33
17	<i>Diagram representing the architecture of [47]. . . . .</i>	34
18	<i>The single pass architecture of [38]. . . . .</i>	35
19	<i>Regions in one input image that led to the top-5 predictions on a certain model [48]. . . . .</i>	36
20	<i>Example given on [49] of the surveillance sourced images. Given as an example of variety within images of the same viewpoint, it can be seen that the vehicles are still in identical poses and still easily recognizable. . . . .</i>	38

21	<i>Sample images of ShortBrisa.</i> . . . . .	39
22	<i>Example of contents in the LargeBrisa dataset. Notice the extra view.</i> . . . . .	39
23	<i>Pre and Post bounding box annotation for an Audi vehicle. Post bounding box vehicles are scale-invariant.</i> . . . . .	41
24	<i>Pre and Post data augmentation for one vehicle entry.</i> . . . . .	41
25	<i>MC-CNN. Top Network is typical CNN usage with RGB image. Bottom is the suggested multi-view over multi-channels.</i> . . . . .	42
26	<i>MV-CNN. Multi-view parallel convolutional neural network. In this work Resnet18 networks will be parallelized.</i> . . . . .	43
27	<i>Sample input.</i> . . . . .	44
28	<i>Resulting features from the convolutions applied to the sample input.</i> . . . . .	44
29	<i>One ROI extraction. The input image is passed in a resnet18 network trained to predict the vehicle model based on the non-cropped vehicle images. With CAM, a bounding box can be obtained via a weakly supervised method that corresponds to regions which lead to the obtained class prediction.</i> . . . . .	46
30	<i>Example of second extracted ROI, for the same view as previously. It follows the same pipeline of figure 29, using a resnet18 network trained to predict vehicle model based on the first extracted ROIs.</i> . . . . .	46
31	<i>Precision and recall curves for CompCars and ShortBrisa.</i> . . . . .	50
32	<i>Example of manually inserted noise.</i> . . . . .	54

# List of Tables

1	<i>Results obtained by [21], performed on the same dataset with the same training conditions, proved that transfer learning not only converges faster, but has potential for better results. . . . .</i>	23
2	<i>Example results from a model with very high, but misleading, accuracy. . . . .</i>	23
3	<i>Example of a model with worse accuracy, but better performance. . . . .</i>	24
4	<i>Composition of CompCars . . . . .</i>	38
5	<i>CompCars Results . . . . .</i>	49
6	<i>ShortBrisa Performance results. . . . .</i>	49
7	<i>Bounding Box plus Data Augmentation results. . . . .</i>	51
8	<i>Individual View performances. . . . .</i>	51
9	<i>Comparison of results in MC-CNN vs MV-CNN. . . . .</i>	52
10	<i>Results for MC-CNN. Numbers inside brackets were introduced in the network by increase the channel depth of the first layer. . . . .</i>	53
11	<i>Results for MV-CNN. Patches that were stacked in channel depth are separated by commas, while patches that were assigned to different Resnet18s are separated by brackets. New top-performance, accuracy wise, is achieved by the bold entries on table 11. . . . .</i>	54
12	<i>Noise tests over different architectures that had presented good results so far. . . . .</i>	55
13	<i>Comparative results of ROI architecture with MV-CNN and a simple resnet18. . . . .</i>	55
14	<i>Table 13's MV-CNN-8 figures of merit. . . . .</i>	56

# List of abbreviations

- Acc.** – Accuracy
- AP** – Accuracy Points
- AUCPR** – Area Under the Curve of Precision-Recall
- BIT** – Brisa innovation and technology
- CAM** – Class Activation Map
- CNN** – Convolutional Neural Network
- DL** – Deep Learning
- FGVC** – Fine-Grained Visual Classification
- FL** – Front-Left
- GPU** – Graphics Processing Unit
- ILSVRC** – ImageNet Large Scale Visual Recognition Challenge
- LPR** – License Plate Recognition
- LSVC** – Large-Scale Visual Classification
- MCP** – Max Channel Pooling
- MC-CNN** – Multi-Channel Convolutional Neural Network
- ML** – Machine Learning
- MV-CNN** – Multi-View Convolutional Neural Network
- NN** – Neural Network
- OV** – Overview
- ROI** – Region Of Interest
- RL** – Rear-Left
- SIFT** – Scale Invariant Feature Transform



# Contents

<b>1</b>	<b>Introduction</b>	<b>13</b>
1.1	Motivation and expectations . . . . .	13
1.2	Contributions . . . . .	15
<b>2</b>	<b>Deep Learning and Convolutional Neural Networks Concepts</b>	<b>17</b>
2.1	Introduction . . . . .	17
2.2	Convolutional Neural Network (CNN) Architectures . . . . .	17
2.2.1	VGGNet . . . . .	18
2.2.2	Inceptionv3 . . . . .	18
2.2.3	Residual Networks (Resnet18, Resnet50, etc) . . . . .	19
2.2.4	Densely Connected Convolutional Neural Networks (Densenet-121, Densenet-161, etc) . . . . .	20
2.3	Neural Network Training . . . . .	21
2.3.1	Supervised Learning . . . . .	21
2.3.2	Unsupervised Learning . . . . .	22
2.3.3	Weakly-Supervised Learning . . . . .	22
2.3.4	Transfer Learning . . . . .	22
2.4	CNN performance metrics . . . . .	23
2.4.1	Accuracy . . . . .	23
2.4.2	Confusion-Matrix . . . . .	24
2.4.3	Precision and Recall . . . . .	24
2.5	Large-Scale vs Fine-Grained Visual Classification (LSVC vs FGVC) . . . . .	25
2.6	Object Recognition . . . . .	26
2.6.1	Classification . . . . .	26
2.6.2	Localization and Detection . . . . .	26
2.6.3	Segmentation . . . . .	27
2.7	Data Augmentation . . . . .	27
<b>3</b>	<b>State-of-the-art</b>	<b>29</b>
3.1	FGVC Classification . . . . .	29

3.2	Multi-View approaches . . . . .	32
3.3	Object Detection State-of-the-art . . . . .	33
3.3.1	R-CNN . . . . .	33
3.3.2	YOLO . . . . .	34
3.3.3	RetinaNet . . . . .	35
3.3.4	Class Activation Maps . . . . .	35
<b>4</b>	<b>Methodology</b>	<b>37</b>
4.1	Datasets . . . . .	37
4.1.1	CompCars . . . . .	37
4.1.2	ShortBrisa . . . . .	38
4.1.3	LargeBrisa . . . . .	39
4.2	Assessing Baseline Performance . . . . .	40
4.2.1	CompCars Dataset . . . . .	40
4.2.2	ShortBrisa Dataset . . . . .	40
4.3	Performance Boosting Techniques . . . . .	40
4.3.1	Bounding Boxes . . . . .	40
4.3.2	Data Augmentations . . . . .	41
4.4	Multi-View Solutions . . . . .	42
4.4.1	Multi-Channel Convolutional Neural Network (MC-CNN) . . . . .	42
4.4.2	Multi-View Convolutional Neural Network (MV-CNN) . . . . .	43
4.5	Study of Descriptive Regions in Vehicles . . . . .	43
4.6	Region of Interest Extraction based in Class Activation Maps . . . . .	45
<b>5</b>	<b>Results</b>	<b>49</b>
5.1	CompCars preliminary tests . . . . .	49
5.2	ShortBrisa Preliminary tests . . . . .	49
5.3	Bounding Box and Data Augmentation addition results . . . . .	50
5.4	Multi-View tests . . . . .	51
5.4.1	MC-CNN vs MV-CNN . . . . .	51
5.5	Image Patch tests/study . . . . .	52
5.5.1	MC-CNN . . . . .	52

5.5.2	MV-CNN . . . . .	52
5.5.3	Image Patches and Multi-Views validation . . . . .	54
5.6	LargeBrisa Results With ROI extraction . . . . .	55
<b>6</b>	<b>Results Discussion</b>	<b>57</b>
6.1	CompCars Preliminary tests . . . . .	57
6.2	ShortBrisa results analysis . . . . .	57
6.3	Bounding Box and Data Augmentation Results . . . . .	58
6.4	Multiple Views tests . . . . .	58
6.5	Multiple Views with Image Patches . . . . .	59
6.6	ROI extraction . . . . .	59
<b>7</b>	<b>Closing Remarks</b>	<b>61</b>
<b>8</b>	<b>Future Work</b>	<b>63</b>
<b>9</b>	<b>Attachments</b>	<b>66</b>
9.1	Attachment A: MV-CNN-8 Prediction Errors 1 . . . . .	66
9.2	Attachment B: MV-CNN-8 Prediction Errors 2 . . . . .	67
9.3	Attachment C: MV-CNN-8 Prediction Errors 3 . . . . .	68
9.4	Attachment D: MV-CNN-8 Prediction Errors 4 . . . . .	69
9.5	Attachment E: MV-CNN-8 Prediction Errors 5 . . . . .	70
9.6	Attachment F: MV-CNN-8 Prediction Errors 6 . . . . .	71
9.7	Attachment G: MV-CNN-8 Prediction Errors 7 . . . . .	72
9.8	Attachment H: Patch Selection for Front View . . . . .	73
9.9	Attachment I: Patch Selection for Overview . . . . .	74
9.10	Attachment J: Patch Selection for Rear View . . . . .	75
9.11	Attachment K: Full ROI extraction architecture description . . . . .	76
9.12	Attachment L: Prediction-Recall data for Resnet50 with FL view, as seen in Table 7 (95.25% accuracy) . . . . .	77
9.13	Attachment M: Prediction-Recall results MV-CNN, as seen in table 9 (95.52% accuracy)	78
9.14	Attachment N: Prediction-Recall data for MC-CNN, as seen in Table 10 (95.52% accuracy)	79

9.15 Attachment O: Prediction-Recall results for MV-CNN-6, as seen in Table 11 (95.78% accuracy) . . . . .	80
9.16 Attachment P: Prediction-Recall results for MV-CNN-12, as seen in Table 11 (96.04% accuracy) . . . . .	81
9.17 Attachment Q: Precision and Recall statistics for MV-CNN-8 on LargeBrisa, as seen in Table 13 (98.02% accuracy) . . . . .	83
9.18 Attachment Q (continued): Precision and Recall statistics for MV-CNN-8 on LargeBrisa, as seen in Table 13 (98.02% accuracy) . . . . .	84
9.19 Attachment R: Class Distribution in ShortBrisa . . . . .	85
9.20 Attachment S: Class Distribution in LargeBrisa . . . . .	86

*Para os meus pais. Obrigado por tudo.*



## Abstract

Deep Learning and Convolutional Neural Networks have been staples in solving challenges related to Image Processing, Computer Vision and Pattern Recognition. Since their breakthrough in 2012 that no other method has come close, be it in overall results, consistency, but also computation capabilities, with the technology evolving and delivering more and more impressive results as researchers push the boundaries. As a consequence, we are seeing work in the domain of Deep Learning creep more and more in our every day lives, automating a variety of tasks, many of them unnoticeable.

In this work, Deep Learning will be applied to try and automate one such barely noticed task: toll collection. In Portugal, Brisa operates an automatic toll collection service, which despite their best efforts, is still fragile and subject to fraud. It is then in their interest that toll collection becomes as precise and reliable as possible. With this in mind, this document explores technology related to Vehicle Recognition, and, taking advantage of a multi-camera setup, applies state-of-the-art methodologies that identify key defining features in vehicles, fusing this multi-perspective information, ultimately delivering a solution that performs vehicle recognition with state-of-the-art competitive results. Moreover, the methodologies here applied can easily be replicated and should translate well to other applications where the main type of data is a visual, particularly where multi-perspective data entries are available, such as medical imaging based diagnosis.

**Key Words:** Deep Learning, Convolutional Neural Networks, Computer Vision, Vehicle Recognition, Multi-Perspective Data, Information Fusion

## Resumo

Deep Learning e Redes Neurais Convolucionais estabeleceram-se como as ferramentas mais comuns para resolver problemas relacionados com Processamento de Imagem, Visão por Computador e Reconhecimento de Padrões. Desde 2012, quando foi demonstrado o quão poderosas poderiam ser, nenhum outro método chegou perto dos resultados, consistência e capacidade de computação obtidos, sendo que a tecnologia continua a evoluir e a mostrar resultados cada vez mais impressionantes, à medida que investigadores exploram os limites. Como consequência, estamos a observar aplicações, muitas delas imperceptíveis, resultantes de trabalhos de investigação na área do Deep Learning a entrar cada vez mais nas nossas vidas e a automatizar diversas tarefas.

Neste trabalho, Deep Learning será aplicado de modo a automatizar uma dessas tarefas imperceptíveis: cobrança de portagens. Em Portugal, a Brisa opera um serviço de cobrança de portagens automático, chamado Via Verde, que apesar dos seus melhores esforços ainda é frágil e susceptível a fraude. É então no melhor interesse da Brisa que o processo de cobrança se torne o mais preciso e fiável possível. Tendo isto em conta, este documento explora tecnologia relacionada com reconhecimento de veículos, e, tirando partido de uma configuração multi-câmara, são aplicadas metodologias estado da arte que permitem identificar características altamente definidoras em veículos, fundindo a informação multi-câmara, e, em última instância, alcançando resultados que competem com o demonstrado pelo estado da arte no que toca a reconhecimento de veículos. Além disso, as metodologias aqui aplicadas e desenvolvidas podem ser facilmente replicadas e devem transportar para outras aplicações de redes neuronais e visão por computador os resultados obtidos, particularmente em situações em que existem dados de múltiplas perspectivas, como diagnóstico do cancro mamário.

**Palavras Chave:** Deep Learning, Redes Neurais Convolucionais, Visão por Computador, Reconhecimento de Veículos, Dados Múltipla Perspectiva, Fusão de Informação



## 1 Introduction

Coarse-grained classification problems are an established and matured field of research. An example that is a good indicator of this is the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)[1], and the progress observed over time in the results of the competition. ImageNet is a Large-Scale Visual Classification (LSVC) dataset, and in the ILSVR Ccompetition, teams of researchers attempt to achieve the highest accuracy classifying a trimmed version of ImageNet into one thousand different categories. Over time, mean accuracy points have progressed from 0.22581 in 2013, to 0.731392 in 2017. The state-of-the-art performance as of the writing of this document is a top-1 accuracy of 88.55%, 88.5% and 88.4% according to the works of [52],[51] and [2] respectively, with the work in [52] still under review.

However, real world problems are often not coarse-grained and thus require more specialized datasets and approaches. This means that there may arise situations where coarse-grained work and research may lead to non-optimal results, due to the fact that in these more specialized situations, the classes on a particular dataset usually have significant intra-class variation and inter-class similarity, along with smaller sample sizes [3]. As such, image classification research applied on these real-world problems has become a research field of its own, Fine-Grained Visual Classification (FGVC).

Several problems have become benchmark staples for FGVC such as [4], [5] or [6]. In addition to this more research-oriented investigation stream, more frameworks are being proposed and developed to target real-world problems such as face-recognition applied to surveillance [7] and road traffic analysis [8].

These FGVC frameworks and methods, specially those applied to vehicle recognition, will be of great interest to the work proposed in this document, since the scope of application is very similar.

### 1.1 Motivation and expectations

The work proposed in this document was developed within the scope of A-to-B, a company owned by Brisa Innovation & Technology, serving as the group's international brand responsible for developing and delivering solutions to mobility services operators.

Brisa is a transport infrastructure operator offering an electronic toll collection service called “Via Verde”, in which, if acquired, one installs a device in the windshield of one's car and is automatically charged for the distance travelled in paid highway roads, without having to stop at the toll. This

device is usually specific to one car only.

To make sure the correct car is passing the toll and no fraud is taking place, a License Plate Recognition (LPR) system is already in use. This system captures an image of the passing vehicle and extracts the corresponding license plate. With such information, it's easy to cross examine the database of registered vehicles and verify that the correct one is present. Although the current performance of the LPR system is good for ideal conditions, those conditions can't be guaranteed 100% of the time, with both natural (like rain, foggy days, sun-light reflection) and unnatural (such as degraded license plates) circumstances hindering performance, which could lead to False-Negatives and False-Positives.

It is not only accidental circumstances that challenge the performance of the LPR, more elaborate frauds may circumvent the system, such as a situation where one not only places the “Via Verde” device in another car, but also the license plate.



Figure 1: *Example of parallel recognition systems for fraud detection. License plate and vehicle info are extracted and cross-examined.*

To solve this situation, a tool was developed to automate the recognition and validation of vehicles passing “Via Verde” tolls. As said previously, the system is integrated in the scope of A-to-Be and will serve as a complement to the current LPR system, attenuating the effects of situations where the LPR isn’t providing satisfactory results. It will also work in parallel to other vision systems such as front seat occupancy detectors.

The results will be a more robust system, more immune to variations and disturbances in the acquired images, providing not only better results (leading to more efficient charging of customers) but also the added benefit of fighting fraudulent and criminal activities, since license plate fraud is a crime, in accordance with the Portuguese penal code (DL n.º 48/95, 15 de Março, Capítulo II, Secção II, article 256º).

## 1.2 Contributions

This is an FGVC focused work, and as such, provides insights in a subset area of Machine and Deep Learning, which is not where the bulk of DL applied to Image Processing and Computer Vision research is done, or at the very least, published. Particularly, it’s a problem of FGVC applied to vehicle make recognition, which is of emergent interest and where good results are being achieved. In this work, properties of Convolutional Neural Networks (CNN) are explored, as well as what vehicle patterns are important for CNNs to perform recognition. State-of-the-art methods that related work has shown to be of use in solving such a problem are employed and evaluated. As such, this work could be of interest to other students and researchers working on similar FGVC problems, particularly vehicle make and model recognition.

Ultimately, a weakly-supervised solution with results competitive with the state-of-the-art, that can be applied to any domain of computer vision where there are multiple perspectives of data, is proposed.



## 2 Deep Learning and Convolutional Neural Networks Concepts

### 2.1 Introduction

The general approach to solve FGVC problems is to find discriminant parts and extract features from the input images, and from these features some form of prediction method is used to perform classification. Different methods for this have been proposed, such as Scale Invariant Feature Transform (SIFT) descriptors and CNNs.

SIFT-based approaches have shown good results. [9] used a SIFT-based approach to obtain good results on datasets like Flower-102, LandUse-21 or Aircraft-100, being able to match a great number of features between different images of the same coarse-grained category. SIFT methods were the main method of feature extraction from 2003 to 2012.

Although the ground work to make fast CNN implementations was already laid in 2004 (by [10] with the suggestion to use GPU computation, taking advantage of its many cores and parallelism capabilities), it was not until 2012, when a CNN architecture called AlexNet was put forward and won the ILSVRC competition, that it became the standard method to solve image pattern recognition tasks [11]. Ever since, CNN-based approaches have dominated the field and achieved remarkable breakthroughs, as concludes [12]. [12] also remarks that CNNs yield competitive accuracy on various retrieval tasks and have advantages in efficiency. [13] remarks that CNNs outperform their counterparts (such as SIFT) by a considerable margin. [14] draws a comparison between SIFT and CNN methods, from which the following table is derived:

	GOOD	BAD
SIFT	<ul style="list-style-type: none"> <li>• Identification Tasks</li> <li>• Simple to Implement</li> <li>• Fast</li> </ul>	<ul style="list-style-type: none"> <li>• Poor Generalization</li> <li>• Not robust to non-linear transformations</li> </ul>
CNN	<ul style="list-style-type: none"> <li>• Classification Tasks</li> <li>• Strongly Bio-inspired</li> <li>• Very Good Generalization</li> </ul>	<ul style="list-style-type: none"> <li>• Processing power required</li> <li>• Requires large datasets</li> <li>• Parameters to set</li> </ul>

Figure 2: *SIFT and CNN comparison. Table taken from [14].*

### 2.2 Convolutional Neural Network (CNN) Architectures

In the following sections, an high-level analysis of some common and powerful CNN architectures is performed.

### 2.2.1 VGGNet

[15] introduces a CNN architecture which was submitted to the ILSVRC challenge in 2014, where it won first and second places in localisation and classification tasks, respectively. It's main contribution was investigating the effect that increased network depth had on network performance, by pushing the number of layers to a maximum of 19 layers (smaller architectures were also presented, such as a 16 layer model), with the convolution layers having smaller 3x3 convolution filters (the minimum to get a sense of left/right and up/down) and a convolutional stride of 1 pixel. Each convolutional layer is followed by a ReLU activation function and occasionally, throughout the pipeline, Max-Pooling is performed over a 2x2 pixel window with stride 2. Each time this operation is performed the convolutional layer width increases by a factor of 2, starting at 64 in the first layers and ranging to 512 in the last layers. The variable stack of convolutional layers are followed by three fully-connected layers, with the first two having 4096 channels and the last one performing the classification for the 1000 outputs related to the ImageNet dataset. It is also claimed that the network generalises well to other tasks and datasets. In sum, it was shown that deep architectures have potential for good performance while maintaining a simple, but computationally heavy structure.

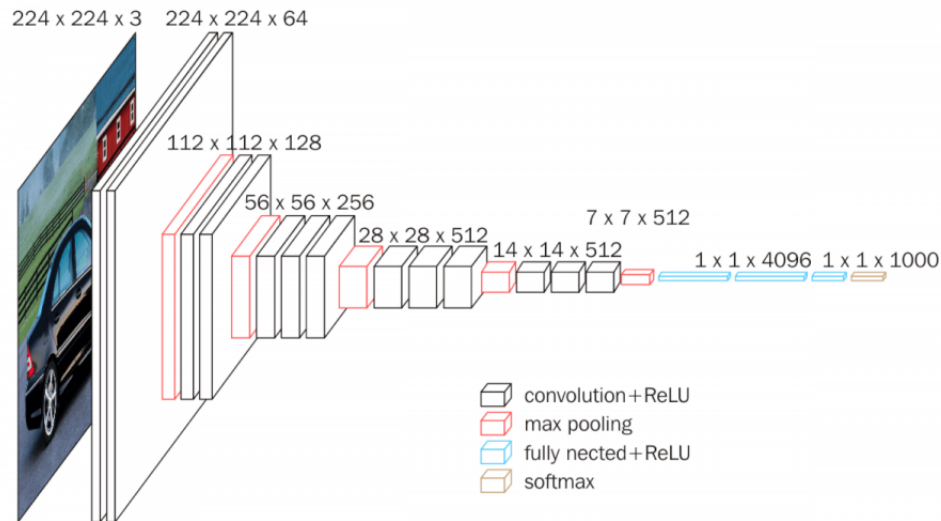


Figure 3: *VGG16's deep, yet simple, architecture.*

### 2.2.2 Inceptionv3

The CNN introduced in [16] attempted to address some shortcomings of very deep CNNs, like VGG16, which despite its architectural simplicity, pays a price in computing power, limiting its (and

other similarly deep networks) applicability in scenarios like mobile-vision. It is an attempt at scaling up CNNs utilizing added computation as efficiently as possible, and it is claimed that the resulting computing cost is lower than VGGNet.

The network is composed of so called Inception modules, arranged sequentially, which feature 1x1, 3x3 and 5x5 convolution filters (along with other common CNN layers like 3x3 MaxPool Layers). Different sizes of filter banks were bringing different benefits to neural networks and in this way, Inception manages to capture benefits from a variety of filter sizes.

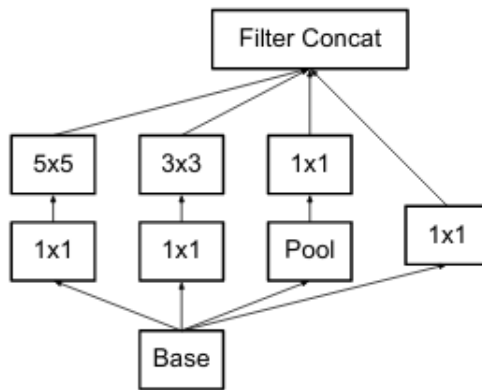


Figure 4: *A single Inception module. These are arranged sequentially and ultimately feed toward a fully-connect section that performs the final prediction.*

### 2.2.3 Residual Networks (Resnet18, Resnet50, etc)

Very deep CNNs are harder to train, require more epochs and more computational power. [17] aimed at maintaining depth in CNNs, while keeping computing cost low and easing the training process by proposing a reformulation of the convolution layers. The layers became residual functions that reference the input. It was empirically proven that residual networks are easier to optimize and gain accuracy from increased depth. Depths of up to 152 layers were achieved, which is 8 times the max depth in VGG networks, while maintaining lower computational cost.

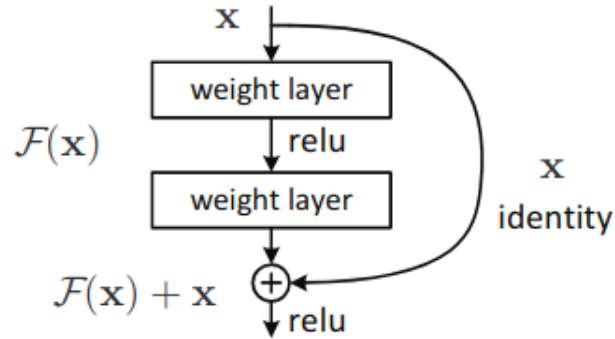


Figure 5: A single Residual Block. Similarly to Inception, these blocks are arranged sequentially in variable depth sizes, leading to fully-connected layers.

#### 2.2.4 Densely Connected Convolutional Neural Networks (Densenet-121, Densenet-161, etc

Expanding on the idea of residual networks and the fact that they can be deeper, more accurate and efficient to train (due to layers close to the output having a reference to the input values), [18] presented an approach that expands on this concept, where each layer connects to every other layer in the same (dense) block. In a standard network, for  $L$  layers there are  $L$  connections. With dense networks there are  $\frac{L(L+1)}{2}$  connections. This way, all feature maps resulting from previous layers are used as input to the current convolution layer, and the result from the current convolution layer is used as input in every subsequent layer (for layers in the same dense block). It is claimed, that among other benefits, this topology promotes feature propagation and reuse, thus improving feature detection, as well as reducing the number of parameters. The authors claim to reach state-of-the-art performance while needing less computation to achieve these results, when compared to other competitors.



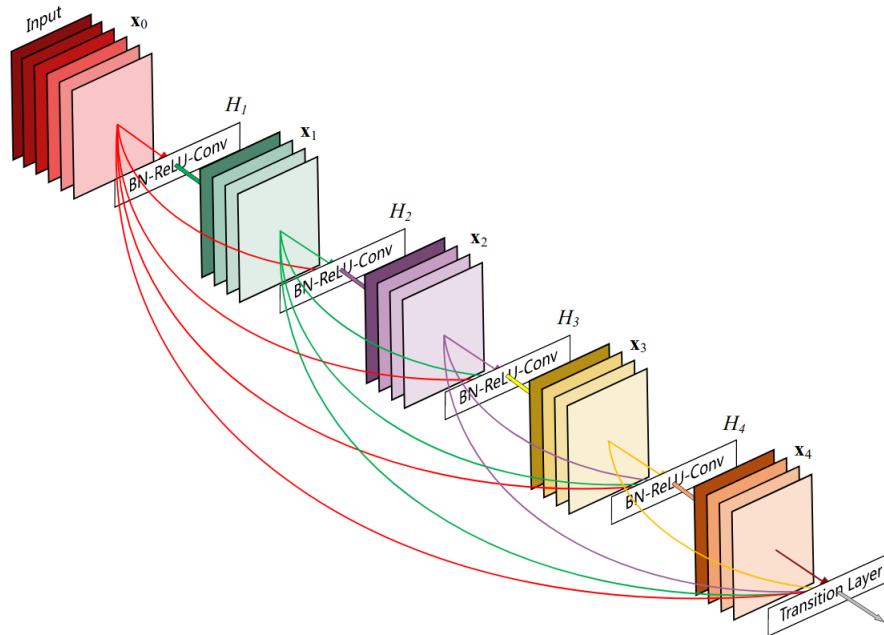


Figure 6: A dense block. Previous layers feed to all subsequent layers in the block. Like Inception and residual networks, a variable sequential arrangement of these blocks is performed. Diagram taken from [18].

## 2.3 Neural Network Training

### 2.3.1 Supervised Learning

Supervised Learning refers to Machine Learning work that is performed on labeled data. This has both advantages and disadvantages.

One advantage is that with this method one is in full control of the data. Assuming there wasn't any human error during the labeling of the data and that this person had sufficient expertise to perform this task, one can trust that the data is correctly labeled, and that there will be no "noisy" inputs to a model.

On the other hand, this is the most costly kind of learning/training method, as the time required for labeling can be quite significant, as was already mentioned, specially for tasks such as hand annotation of bounding boxes or contours for segmentation tasks.

### 2.3.2 Unsupervised Learning

This is used primarily to cluster data into categories, that may or may not be apparent to the naked eye. Usually the output data will then be interpreted and possibly labeled, but the labeling process is much less expensive, since one can label whole clusters of data instead of labeling data entries one by one. A disadvantage is that one is not in full control of the data and is relying on the assumed performance of whatever algorithm or model that was used to perform this clustering.

### 2.3.3 Weakly-Supervised Learning

Weakly supervision relates to tasks where both supervised and unsupervised methods are employed. It usually involves a label (or a weak label) that leads to some type of unsupervised function that is learned and taken advantage of.

One example of this, which will be used during the course of this work, would be labeling classes of a certain dataset (i.e., vehicle make) and then allowing whatever network or architecture one is using to learn by itself how to segregate the image portions that contain that class.

### 2.3.4 Transfer Learning

One interesting approach for CNN training is the widely used transfer learning methodology. As many other aspects of DL, it draws inspiration from human behaviour. If someone was to learn and master a particular skill, its safe to assume that that person would be able to learn new skills in the same domain or sub-domains of the original one due to this previous knowledge. For example, someone that can play an instrument (i.e., classical guitar) and has also learned music theory, can probably transfer the fine skills (guitar playing) and more general skills (music theory) to other learning tasks, such as playing the violin.

The idea behind transfer learning in CNNs is much the same. It has been observed that, independent of the task, CNNs tend to learn similar convolution filters in the convolution layers. Particularly, earlier layers tend to develop into general feature extractors (line detectors, gradient detectors, etc.), while later layers tend to specialize to the data at hand. As a concept, one can equate the earlier layers to the music theory knowledge in the previous example, and the later specialized layers to the guitar playing knowledge. One can also equate the transferring of knowledge from playing guitar to playing the violin, to the transfer of knowledge from a CNN model that differentiates animals to a CNN that

differentiates dog breeds.

It has been verified that however closer the domains (the original and the target domain) are, the more effective the transferred knowledge will be, lowering the amount of epochs required to converge to a solution. This is not always possible in very specialized datasets and there aren't readily available pre-trained models trained in all domains. As such, many transfer learning use cases use models pre-trained on the ImageNet dataset. These observations are supported by scientific surveys [22][23][24][25].

[21] performed experimental validation of performance with fine-tuned networks vs training from scratch and verified the following:

Table 1: *Results obtained by [21], performed on the same dataset with the same training conditions, proved that transfer learning not only converges faster, but has potential for better results.*

Architecture	Approach	Accuracy
Resnet50	from scratch	84.3%
Resnet50	transfer learning	92.0%
Resnet152	from scratch	35.3%
Resnet152	transfer learning	92.6%

## 2.4 CNN performance metrics

### 2.4.1 Accuracy

Accuracy ( $\frac{CorrectPredictions}{TotalInputs}$ ) is usually a very important (perhaps the most important) metric when measuring CNN model performance. However, there are situations where this is not only insufficient, but can also be quite misleading.

Table 2: *Example results from a model with very high, but misleading, accuracy.*

	True Class 1	True Class 2	True Class 3	Total Samples
Predicted Class 1	150	0	0	150
Predicted Class 2	0	1	7	8
Predicted Class 3	10	0	1	11

For the data in Table 2 there are 152 correct predictions and 169 total inputs. This represents

Table 3: *Example of a model with worse accuracy, but better performance.*

	True Class 1	True Class 2	True Class 3	Total Samples
Predicted Class 1	60	7	5	72
Predicted Class 2	4	30	7	41
Predicted Class 3	10	10	65	85

an accuracy of 89.95%. For the data on Table 3 there are  $\frac{155}{198} = 78.28\%$  accuracy. At first glance this could lead one to dismiss the model of Table 2, as superficially it appears less capable. However, when analyzing the individual class results, it can be seen that Table 2’s model performs poorly on classes 2 and 3, while Table 3’s model, despite not performing as well in class 1, outperforms Table 2’s model significantly on the other classes. Of these, which is then more capable? Intuitively, Table 3’s model seems better, but this can be quantified. This is a common problem in FGVC datasets, where representation may be lacking in certain classes.

### 2.4.2 Confusion-Matrix

The data in tables 2 and 3 is presented in a confusion matrix. A confusion matrix is a way to arrange the output data of a model vs the inputs, and easily visualize performance. One can easily check for True-Positives, False-Positives, True-Negatives and False-Negatives.

		Assigned Class	
		Positive	Negative
Actual Class	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Figure 7: *Generalist binary confusion matrix example.*

### 2.4.3 Precision and Recall

Precision and recall are useful measures for binary information retrieval tasks, that sometimes better describe performance than just accuracy.

Precision can be calculated as *True positives/Total number of positives predicted*. Recall can

be calculated as *True Positives/Total number of actual positives*. These can be combined into a single measure of performance called the F-score calculated by  $\frac{2*Recall*Precision}{(Precision+Recall)}$ .

These metrics can be extended to multi-class labels by considering a *one vs all* approach, binarizing the prediction, and thus getting individual prediction and recall values for each class.

Going back and applying the precision formulations it results prediction values of 100%, 12.5% and 9.1% for classes 1,2 and 3 respectively for Table 2. Comparing to table 3 it results 83.3%, 73.2% and 76.47% for classes 1,2 and 3. Applying the recall formulations, values of 93.75%, 100% and 12.5% are obtained for Table 2, and 81.1%, 63.83% and 84.41% for Table 3.

Plugging these values in the respective formula, an F-score of 0.97, 0.22 and 0.11 for classes 1,2 and 3 for Table 2's model is achieved. For Table 3, F-scores 0.82, 0.68 and 0.80 are derived. Averaging these results, it can be verified that Table 3 is actually the better performing model with an average F-score of 0.77, while Table 2's model averages 0.43.

## 2.5 Large-Scale vs Fine-Grained Visual Classification (LSVC vs FGVC)

Visual classification problems can generally be divided in two categories: LSVC and FGVC. What differentiates one from the other is the data within the dataset that one is working with and the outputs/classes that one is trying to predict.

In the case of LSVC datasets, they are generally composed of coarse classes, i.e., classes with low inter-class similarities, easily distinguishable by the general population. As such, building a LSVC dataset is easier and can be crowdfunded for labeling. Many of these LSVC datasets are so well-curated, that they become too well-curated, which in the end makes their application to real world problems limited, due to not addressing real world limitations such as poor lighting or varied object poses. Usually, class representation is well distributed, as the generality of the objects in these datasets makes it easy to find new samples. Properties like these can be found, for example, on the ImageNet dataset.

When it comes to FGVC however, classes are of more fine-grained nature (i.e., different bird species, or different vehicle brands, or even vehicle models), where inter-class similarities can be very high, and so can intra-class differences. This invalidates, for the most part, the option to crowd-source the labeling of an FGVC dataset, as one would usually need a person with the necessary expertise to do this.

Although there are datasets commonly used for FGVC research which are well-curated and of good

quality, many suffer from a lack of uniform representation of classes, making it harder to validate models. Images with poor lighting and generally more heterogeneous images are also more commonly found. These issues are exacerbated if one was to build a custom dataset applied to a particular real-world problem. This makes FGVC exceptionally challenging and still an important topic of research in the field of computer vision, specially when compared to LSVC tasks.

## 2.6 Object Recognition

Object Recognition is a general term used to describe one of the tasks CNNs are used for. It includes other sub-tasks that perform more specialized functions in the field of Image Processing and Computer Vision. Within Object Recognition, the tasks of Image Classification, Object Localization and Detection or Object Segmentation can be performed.

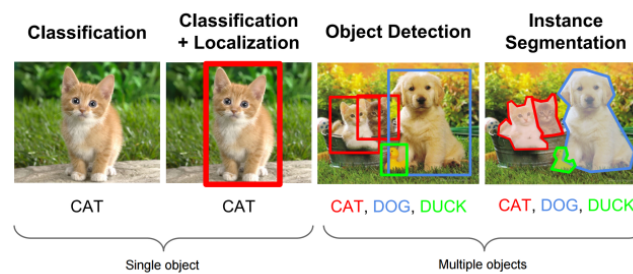


Figure 8: *Specialized tasks within Object Recognition. Image taken from [53].*

### 2.6.1 Classification

Classification has been somewhat covered in prior sections, and as shown in figure 8, it simply identifies one class, among several that are contained in a dataset without any extra information about the semantics of the input data. It is the most common task to perform, and generally, in Deep Learning development frameworks, the available pre-trained models (be it Resnets, or Inception networks or others) are classification networks [50]. Classification generally relies on previously labeled data.

### 2.6.2 Localization and Detection

This particular task provides additional information about the input data, giving information about the target class' location within the input image. Depending on the architecture and dataset, the task can be localization of a single object or various objects within the same context. Generally, these

localization networks adapt or take advantage of previously successful classification CNNs, such as [19], where an AlexNet is used as a feature extractor, while introducing other methodologies for localization. This was one of the first successful implementations of object localization with CNNs, introduced in 2014, hence the use of AlexNet, which was still somewhat the state-of-the-art architecture at the time.

While obviously useful in certain scenarios, this requires manual labeling of data (specifically, bounding boxes surrounding areas of interest), and if one is dealing with labeling more than one object per input, time required to label can increase dramatically.

### 2.6.3 Segmentation

Applying Object Detection Models can give us a bounding box around each relevant object in an image, however there are situations where one may be interested in a more descriptive output. Instance Segmentation provides a similar type of information to localization detectors, but outputting a mask over the pixels which have been deemed to contain the expected output. Applications of such models are more limited, and more importantly, the manual labour required to label the respective dataset is much greater, as one would need to identify all relevant contours for all relevant objects.

## 2.7 Data Augmentation

Deep CNNs have established themselves as the best performing method for pattern recognition in image processing. They are however reliant on large amounts of training data to guarantee that the results observed on test data are credible and that the model has not overfit to the training data.

Overfitting is a consequence of low amounts of data, or uneven representation of data over the dataset's classes, which leads to a model learning to perfectly adapt itself to the training data, and only the training data. This is very prevalent in research areas where sourcing image data is more complicated, either because of scarce existing examples and/or privacy concerns, such as medical image analysis.



Figure 9: *Typical curve of error progression of training and testing data when overfitting is occurring, as seen in [20].*

A dependable Deep Learning Model should have equal, or very similar performance throughout the training and testing epochs. Data Augmentation is a powerful method to combat this shortcoming, allowing artificially increasing the available data via image processing techniques like geometric transformations, color space transformations, strategic noise placement, among others. Survey work performed by [20] and [21] has demonstrated that indeed data augmentation is essential for Deep Learning CNNs performance.



### 3 State-of-the-art

#### 3.1 FGVC Classification

The work in [26] presented a new Max Pool layer that performs the Max operation along the channel of a  $(x, y)$  positioned feature in the final convolution (before the fully-connected) layer. This method compresses the final feature tensor into a smaller dimensional feature space, while claiming to maintain classification performance. This means that the number of parameters of a particular network become reduced and the number of redundant features is reduced. With this change in network structure, state-of-the-art results have been achieved for both the CompCars and Car-196 dataset, with the best results being an accuracy of 97.82% and 93.71% respectively.

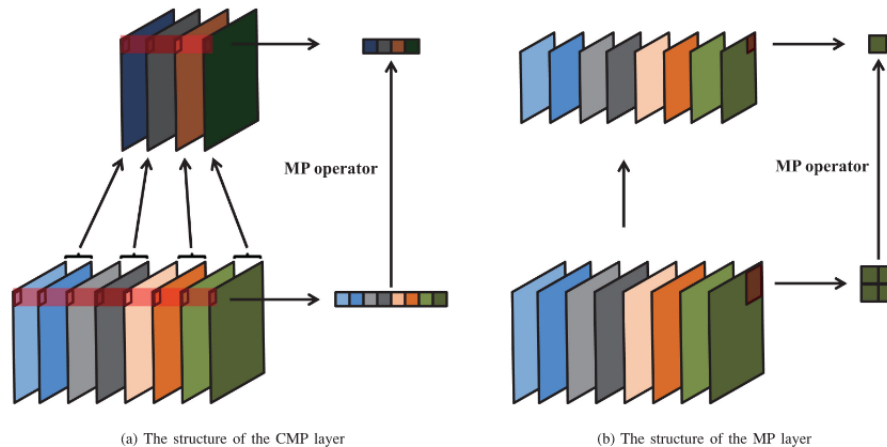


Figure 10: *Max Channel Pool (MCP) operator (left) vs Regular Max Pool (right). MCP results in a smaller depth of channels of feature maps.*

The work put forward in [28] suggests a non-CNN alternative for unsupervised Vehicle FGVC classification solution. It was developed and evaluated in a real-world acquired dataset, which features many of the problems one would expect real-life to present, that are not found in research datasets like CompCars or Stanford-Cars: sun-light reflections, varying weather and visibility conditions, poor lighting. It is also claimed that by avoiding a CNN solution, common CNN derived problems are not of such concern, like class under representation. A SIFT based feature extractor is utilized that analyses parts of the image in a grid fashion, identifying features by comparing them to a feature codebook/trained dictionary of features. This method achieved competitive results in the CompCars dataset (98.63%) and in the custom dataset (97.51%).

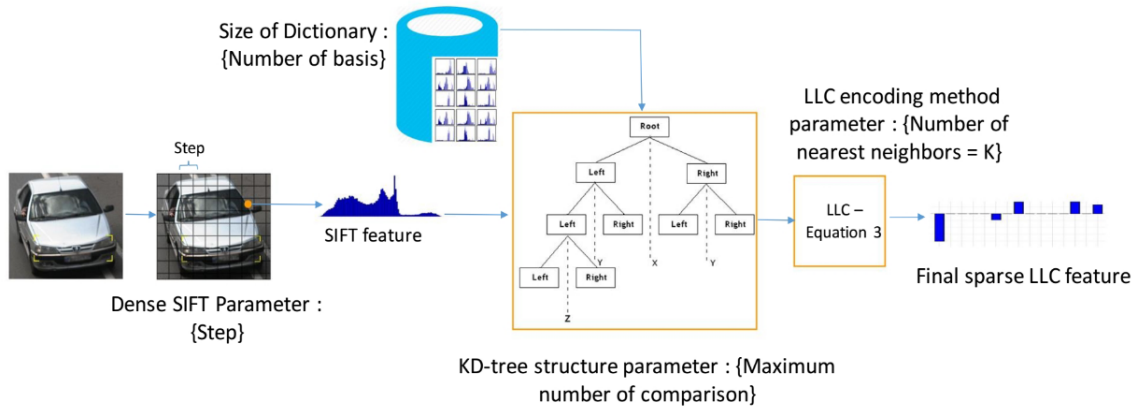


Figure 11: *Diagram representation of the solution put forward in [28].*

[29] introduces a simple, yet interesting way to deal with noisy and heterogeneous real-life vehicle images, which is to hierarchically feed the input images through networks dedicated to assess image quality. A threshold is defined and if the hierarchical section that classifies image quality attributes a quality score below the threshold, the image will be then processed and classified by a network dedicated to low quality image feature extraction, otherwise, the image will be classified by a standard quality network. Results were assessed on CompCars and an accuracy of 98.89% was achieved.

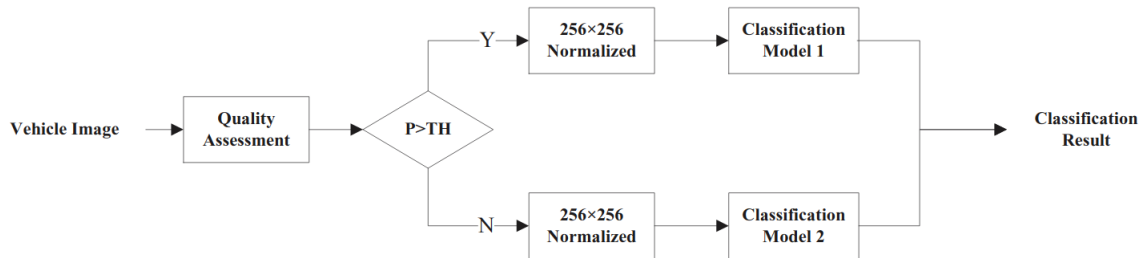


Figure 12: *The hierarchical architecture of [29].*

[34] is another method that aims at dealing with non uniform and noisy input images, instead of just evaluating performance on "good" datasets. Vehicle localization is performed and a bounding is output by a first stage, Then image processing is done on the found front view of the vehicle, which is then the input to a CNN for classification. The devised application was tested on a custom private dataset composed of 2191 test images and 26 classes. It was reported a performance of 97.31% accuracy for the task of vehicle make and model classification.

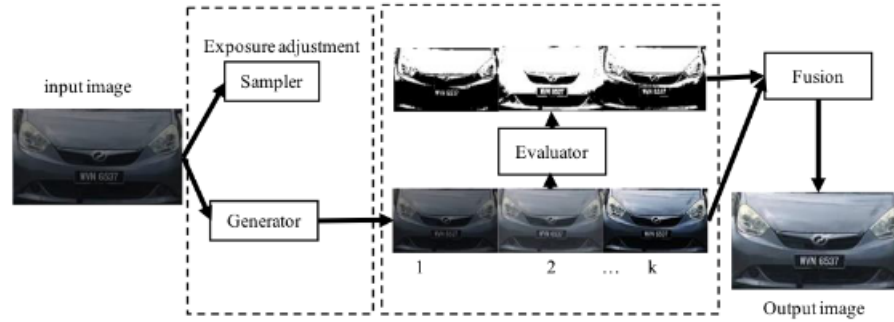


Figure 13: *The image quality enhancement stage of [34].*

A weakly supervised data augmentation network was suggested in [31]. It claims to increase performances by playing with high descriptive zones in the input images. After a high descriptive image region is discovered, that same input is modified either by "dropping" (i.e., occluding said zone) or "cropping" (i.e., zooming in on the feature, excluding surrounding zones). This resulting image is then fed again to a classification network. This is a form of weakly supervised data augmentation, that delivered state-of-the-art results in several non-vehicle fine-grained datasets. In the same vein, [32] introduces a weakly-supervised approach which aims to focus high descriptive zones, also achieving state of the art on four widely-used datasets. [33] also presents a novel attribute-aware model that focus descriptive regions for fine-grained datasets, where local and global features are learned in an end-to-end manner, resulting in a feature vector with better information, instead of noisy or irrelevant features. Here, state-of-the-art results are reached on the CARS196 dataset.

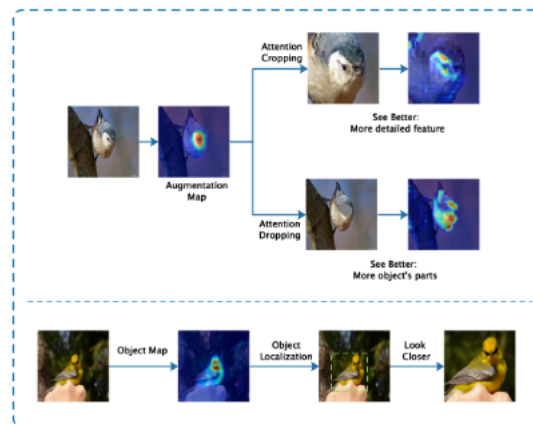


Figure 14: *The weakly supervised concept of [31].*

### 3.2 Multi-View approaches

Intuitively, there are situations where human observers may benefit from having several perspectives to perform predictions. It comes as no surprise that similar approaches have been tried on CNNs.

[35] and [36] took advantage of Multi-View CNN (MV-CNN) architectures for detection of mammary cancers, with both approaches delivering state-of-the-art results in their fields. The addition of different perspectives of data allows the MV-CNN to extract complementary information from several viewpoints, similar to the way a larger field of vision benefits human observers. In these approaches, each view is passed through a CNN that extracts that view's features. The outputs of these multiple feature extractions are then concatenated into a single feature vector and fed through a dense network.

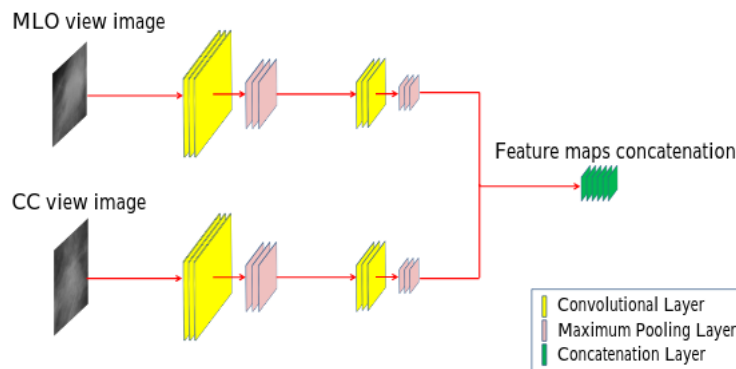


Figure 15: *MV-CNN used in [36]*.

The work of [37] is similar to that of [35] and [36], but applied to the same domain and similar scope of this work. It is again a multi-input configuration of CNNs. It takes one input which is resized to three different sizes. They then pass through dedicated convolution branches and feature extraction is performed. Additionally there is a local loss module after each branch, so that the parameters of these feature extracting branches can be updated during training. This makes it so that the model takes advantage of all predictions: the predictions from the three input branches and the final prediction performed by the fully connected layers, which is placed in the network after the features extracted from the three convolution branches are concatenated. Again, the authors mention that one of the systems this architecture draws inspiration from is human vision.

The devised architecture was trained on a dataset acquired and labeled for the purpose of the work and tested on 7 fine-grained classes with each being composed of about 1000 samples. In this dataset

an accuracy of 94.9% was achieved, while on the CompCars dataset Top-1 accuracy was 91%. These results are not state-of-the-art at the time of the writing of this dissertation, but this still builds a good base for working with multiple image inputs.

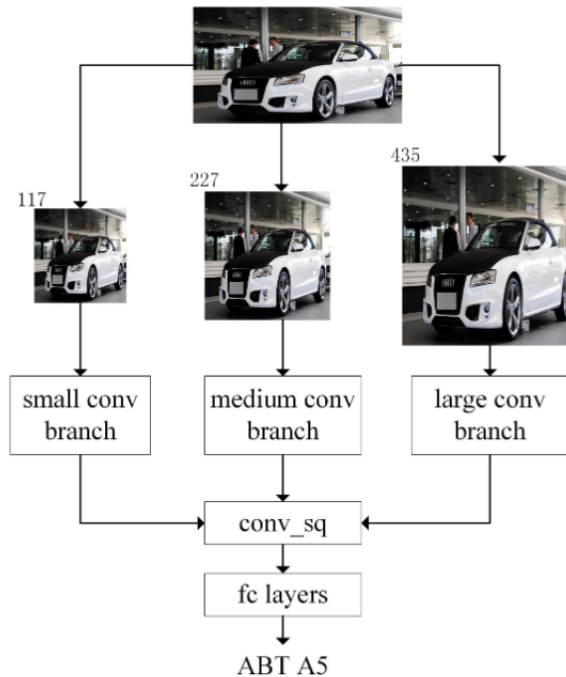


Figure 16: *MV-CNN architecture used in [37].*

### 3.3 Object Detection State-of-the-art

#### 3.3.1 R-CNN

[44] presents the work that lead to one of the first successful implementations of object detection using CNNs. It's a region proposal based system that selects parts of the image and passes them through a CNN, where the proposed class is extracted. This is quite a computationally heavy solution due to the several passes being made on the classification CNN, however this was, at the time, a significant breakthrough and the state-of-the-art for object detection.

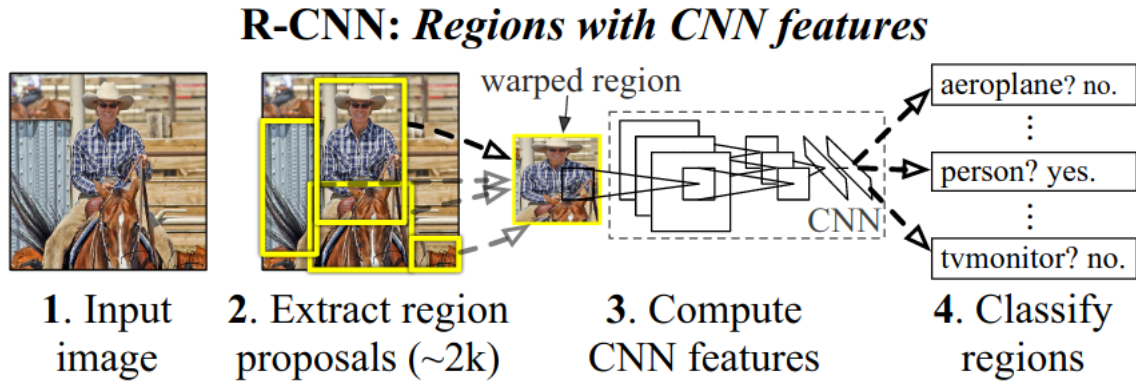


Figure 17: *Diagram representing the architecture of [47].*

The work in [44] was eventually built upon by the same team of researchers, and results reported in [45] and [46] showed that this achieved state-of-the-art results accuracy wise, although only near real-time utilization was possible. [47] is a report by the same team, where it is shown how a branch for predicting a mask inside the previous bounding box detected by [46] achieves good instance segmentation results, outperforming similar architectures.

### 3.3.2 YOLO

The YOLO object detection architecture introduced in [38] a frame object detector that delivers a bounding box and prediction probability. It took a different approach when compared to other detection systems that re-purposed classifiers to perform detection. While re-purposed classifier based systems will usually evaluate an image at various locations with the classifier network and check if that image part has a certain object, (i.e., [41] proposes a sliding window approach where the classifier network is run in equally spaced intervals over the input image) YOLO suggests a single pass through the CNN (hence its name: you only look once (YOLO)). This also has the advantage of reducing the overall complexity found in R-CNN.

In YOLO, a single convolution network predicts the bounding boxes and the supposed object, which means that YOLO is faster than its competitors. It is also claimed that the system is more immune to background noise when compared to Fast R-CNN, due to looking at the input image as a whole and not with region proposals, making less than half of background errors. Despite this, YOLO does not beat state-of-the-art methods on accuracy due to difficulties in detecting smaller objects, with the main contribution being the fast execution time and real-time application capabilities.

Over time, improvements have been made to the architecture leading to it delivering state-of-the-art results: [39], among other improvements, increased the number of predicted classes and [40] also reports other minor improvements that lead to better localization of small objects, which leads to results competitive with [42].

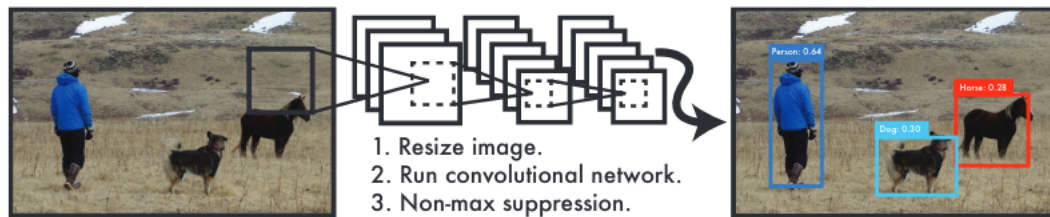


Figure 18: *The single pass architecture of [38].*

### 3.3.3 RetinaNet

RetinaNet was introduced in [42] and, like [38], its a one-stage object detector that aimed, again like [38], at handling not only the fact that the best performing object detectors were, at the time, two-stage detectors, but the existing one-stage methods offered worse performance with the trade-off of faster execution. The main goal was then to develop a one-stage object detector that delivered state-of-the-art accuracy and fast execution.

It was assumed that the main issue driving lower results for one-stage architectures to be class imbalance. The suggested solution was a modified loss function that down-weights the losses derived from well performing and well represented classes.

Afterwards, the works presented in [17] and [43] served as the backbone for the object detection system, with the proposed loss function change implemented. Results achieved from this implementation matched state-of-the-art for one-stage object detectors for the first time.

### 3.3.4 Class Activation Maps

While not directly an object detection technique, [48] suggests that neural networks have good innate object localization ability. This allows the visualization of which parts of the input image led specifically to a certain prediction score on the output of the neural network. Experimental results showed that this method performs good localization. And although this may not be the state-of-the-art for this task, it has the big advantage of being a weakly-supervised method, since one only needs to

annotate the output class and not the bounding box itself. A disadvantage is that this way it is not directly obtained a bounding box as the output, but a heat-map of importance in the image. From here, by establishing a threshold, a bounding-box can be built, but it is not as straightforward.

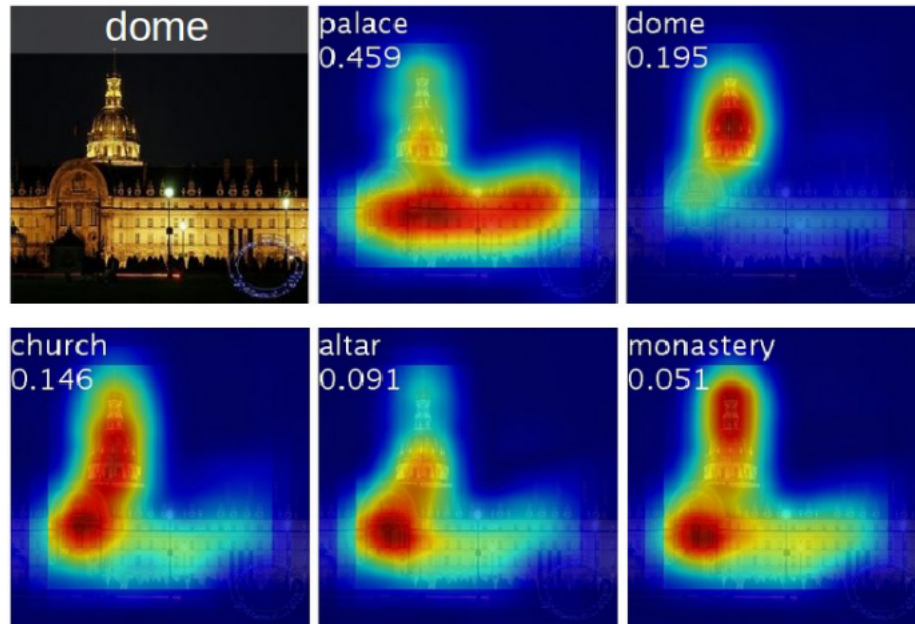


Figure 19: *Regions in one input image that led to the top-5 predictions on a certain model [48].*



## 4 Methodology

### 4.1 Datasets

In this section, the datasets used for the development of the work are summarized.

#### 4.1.1 CompCars

For part of the work that will be outlined in coming sections, the dataset proposed in [49] (which is publicly available) was used.

Much of the FGVC research currently performed on vehicle classification is validated on this dataset. It features data acquired both from the web and from surveillance cameras. The purpose of the surveillance data was to add more heterogeneity to the data, with images with more variation in quality, and generally more similar to what one would find in real life applications. However, even these surveillance sourced images are still quite similar, with the images being of similar pose (for a particular view) and even where there may be some kind of noise, vehicle features are still easily discernible. As such, state-of-the-art research has largely solved the problem of classifying the vehicles of this dataset, with one of the next steps, which isn't made possible with this dataset, being the validation of FGVC methods in a real-life scenario.

The dataset contains 163 annotated car makes and 1716 annotated car models. A total of 136716 images of the cars are available, with another 27618 images of car parts (headlights, interiors, dashboards, etc.). The 136716 set of images are divided in 5 viewpoints (Front, Rear, Side, Front-Side and Front-Rear), with the distribution per-class being quite balanced.

In sum, this is quite a well curated dataset, with overall very good quality images, and, within the the same viewpoint, most images present the vehicle in the same pose, which facilitates classification performance, and thus, for certain tasks, accuracy achieved on this dataset may not be transferable to a real-life application.

Table 4: *Composition of CompCars*

Viewpoint	Total	No. per Model
Front	18431	10.9
Rear	13513	8.0
Side	23551	14.0
Front-Side	49301	29.2
Rear-SideS	31150	18.5



Figure 20: *Example given on [49] of the surveillance sourced images. Given as an example of variety within images of the same viewpoint, it can be seen that the vehicles are still in identical poses and still easily recognizable.*

#### 4.1.2 ShortBrisa

This dataset is comprised of images of vehicles passing Brisa’s tolls. Thus, the acquisition of these images were all subject to real world constraints, that are evident upon analyzing the data.

One can verify that vehicle pose has significant variation, as well as other visible constraints, like lighting, due to some images being acquired at different times of the day.

The train set of data is made up of 4560 samples for the three views (thus, 1520 sets of viewpoints per vehicle). The test set of data is made of 1137 samples for the three views (379 individual entries). There are 15 labeled classes. Class distribution for this dataset can be consulted in attachment R.

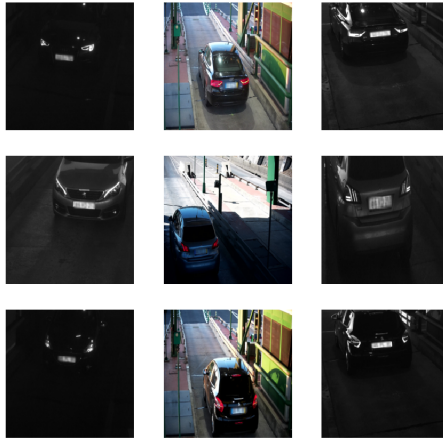


Figure 21: *Sample images of ShortBrisa.*

### 4.1.3 LargeBrisa

LargeBrisa is a much more complete version of ShortBrisa. It contains all the images and classes of ShortBrisa but extends the number of classes to 39, has an additional viewpoint, has 1286 sets of 4 view samples per vehicle for testing and 5864 samples for training. As a much bigger dataset, results achieved in it should be more credible.

This more complete dataset was only available later in development, as such, not all tests performed on ShortBrisa were repeated (and many would be redundant and unnecessary), but this dataset is the one where the final results were validated. Class distribution is demonstrated in attachment S.



Figure 22: *Example of contents in the LargeBrisa dataset. Notice the extra view.*

## 4.2 Assessing Baseline Performance

### 4.2.1 CompCars Dataset

Preliminary tests were performed using the CompCars dataset. Using the front-view available within the dataset, training was performed on a few established classification CNNs, following the previously introduced training strategy of transfer learning. The purpose of this was not only to analyse individual performance of each selected network, but other performance metrics such as training time. It also served as a primary point of reference, as due to the homogeneity of the image composition in this dataset and how well it was curated, good results are expected, as shown by state of the art research.

### 4.2.2 ShortBrisa Dataset

Identical tests were performed on the ShortBrisa dataset. In this case, the tests were performed individually for each available view. Here, the purpose is mainly to verify the expected downgrade in performance when utilizing the front-view (due to the factors exposed on 4.1.2), and also to verify what kind of performances the remaining views deliver.

## 4.3 Performance Boosting Techniques

This section describes techniques aimed at boosting performance and reducing incorrect predictions.

### 4.3.1 Bounding Boxes

As mentioned previously, image composition is much more chaotic in ShortBrisa when compared to CompCars, which one can presume should lead to poorer results. One of contributing factors is that ShortBrisa images were acquired "organically" (i.e., some sort of system, unknown to us, captured a picture of a car passing a toll station, via some triggering mechanism, with minimal supervision), as such, vehicles were captured in a variety of poses (some closer to the camera, some farther away). To deal with this issue, bounding boxes were annotated and added to the ShortBrisa dataset. Images were then re-scaled, before they were passed through a network for classification, leading to a more homogeneous image composition, both due to the exclusion of background noise, and the re-scaling making all the vehicles close to the same size.

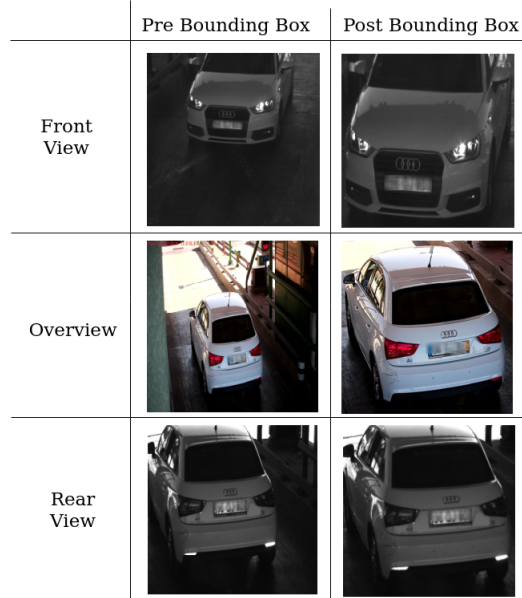


Figure 23: *Pre and Post bounding box annotation for an Audi vehicle. Post bounding box vehicles are scale-invariant.*

### 4.3.2 Data Augmentations

The data augmentations strategies in the following figure were implemented and inserted in the dataset. This should compensate certain shortcomings of the dataset, such as small representation of a certain class.










DA Technique	Horizontal Flip	Random Erasing	Affine Transform	Color Jitter	Sample of Resulting Image
Description	50% Chance of horizontally flipping image	Random erasing of image parts	Image Rotation, Translation, Scaling and Shear Deformation	Changes Image Brightness, Contrast, Saturation and Hue	
Original Image					
Post DA					

Figure 24: *Pre and Post data augmentation for one vehicle entry.*

## 4.4 Multi-View Solutions

Extra views from different poses are available for each vehicle on both CompCars and ShortBrisa. In CompCars these extra views were not chosen to be part of testing because results of over 99% were already being achieved, and no conclusions could be derived from including more views.

However, with ShortBrisa, performance is expected to degrade. In this case, and inspired by similar solutions to similar problems, it was attempted to perform predictions fusing the information available from the three available views on ShortBrisa. In the following two subsections the methodologies chosen for this shall be explained.

### 4.4.1 Multi-Channel Convolutional Neural Network (MC-CNN)

The typical utilization of a neural network for image analysis sees the input being of a certain dimension in width and height (i.e., 224x224 pixels) and channel depth. Usually then, the input is of size 224x224xChannel\_Depth, where Channel\_Depth is usually of size 3 for each of the RGB layers of a colored image. To insert multiple views in a single network, the Channel\_Depth RGB channels will be replaced each with a vehicle view by transforming each image to a single channel gray level image, as exemplified in the following figure.

This is not limited to 3 views/channels only, as it is possible to change this first layer to accommodate as many input channels/views/sources of information as needed. Such feature will be taken advantage of later in this work.



Figure 25: MC-CNN. Top Network is typical CNN usage with RGB image. Bottom is the suggested multi-view over multi-channels.

#### 4.4.2 Multi-View Convolutional Neural Network (MV-CNN)

Following the work described in [37], a Resnet18 implementation was developed, featuring parallel Resnet18 networks, with each of them processing a different input image. The architecture is composed of standard Resnet18 networks and each of them takes a single different input image, from which three predictions are extracted, along with the corresponding loss values for the training data. The fusion of the information from each input happens when, before the fully connected layers of each Resnet18, the feature vectors of size 512 resultant from each input are extracted and merged. After this, a max operation is performed, originating a final feature vector which will be the input to the final Fully Connected layer. The computed loss will be a function of the previous losses of each Resnet18's prediction for its respective image input, and a function of the final loss that results from the final prediction based on the final feature vector. This is evident in the diagram below.

One thing of note, is that this architecture is not limited to three views/inputs only. It is possible to have as many parallel inputs as needed and the general functioning stays the same. This will, like the previously showcased method, be taken advantage of in upcoming sections.

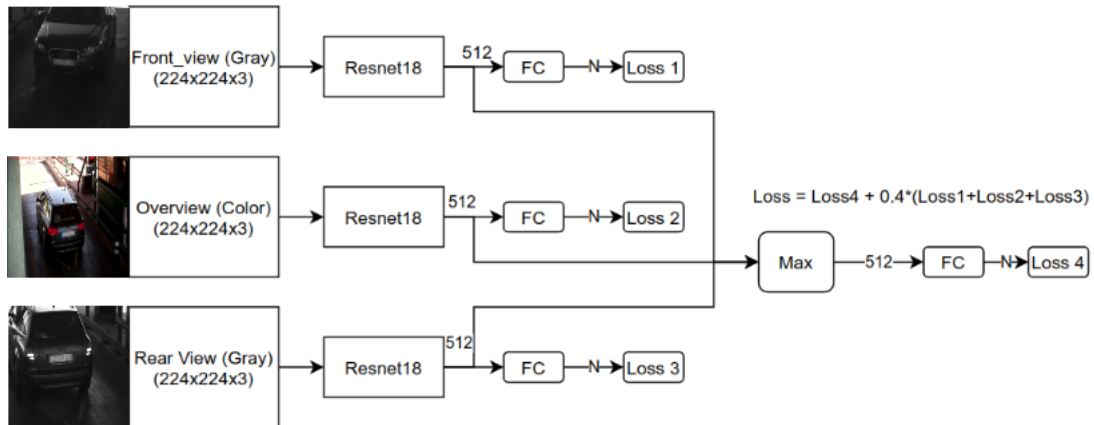


Figure 26: *MV-CNN. Multi-view parallel convolutional neural network. In this work Resnet18 networks will be parallelized.*

#### 4.5 Study of Descriptive Regions in Vehicles

As explored in [38], one interesting approach relating to visual classification is finding relevant parts in images, and steer the network towards focusing more on that particular part, discarding extra or unnecessary information. One such approach, that will be followed and analyzed in the coming

section, is finding these descriptive regions within the image itself in order to feed them together with the original image for classification.

To get some insight for what regions are descriptive for convolutional neural networks performing this task, a small neural network of 10 convolution layers was realized and trained on the CompCars dataset. Such a small network actually achieves decent performance on this dataset (92.3% accuracy), but that is not the intent of its use. Its aim is to extract the output of each convolution filter to determine which regions of a vehicle are discriminatory and which info is being passed to the classifier.

After training on the CompCars dataset, an image of a vehicle was fed through the network. The outputs of each filter and the original input are shown below:

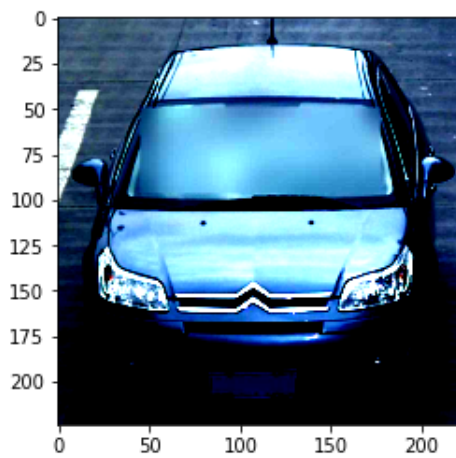


Figure 27: *Sample input.*

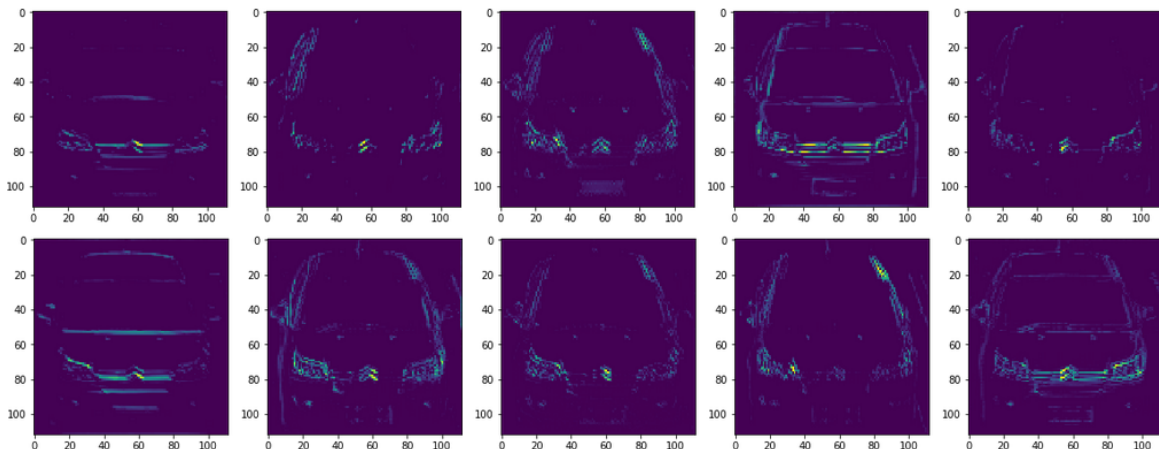


Figure 28: *Resulting features from the convolutions applied to the sample input.*



The obtained images highlight descriptive regions. Based on this information, tests will be performed with zoomed in image regions, taking advantage of the Multi-View solutions presented in section 4.4. Image patches will be cropped out of the original image, as shown in Attachments H, I and J, and a varied combination of these patches and the original images will be fed as input to both multi-view solutions. The intent is that this "zoom" on features will allow the network to put more emphasis on these parts that are known to be more descriptive and thus, lead to better performance. It could also bring the added benefit of not losing image resolution: when performing training, the original image is resized to a 224x224 image, which leads to loss of pixel resolution. By cropping regions of the image of size 224x224, one allows, for example, the region of the make badge to be fed to the network with its resolution intact.

#### 4.6 Region of Interest Extraction based in Class Activation Maps

One of the things that most improved system performance was the definition of bounding boxes. It is now of interest to automate the definition of these regions of interest (ROI) in the input image. To that end, Class Activation Maps (CAM) will be used, following the procedure outlined in [48].

For a given input image,  $f_x(x, y)$  represents the activation of unit  $k$  in the last convolution layer of a CNN (in this case, it is layer4 of the Resnet18) at spatial location  $(x, y)$ . For unit  $k$ , global average pooling,  $F^k$ , is performed as  $\sum_{x,y} f_k(x, y)$ . The input to the softmax to get the resulting class is  $S_c = \sum_k w_k^c F^k$ , where  $w_k^c$  is the weight corresponding to class  $c$  for unit  $k$ , and it indicates the importance of  $F_k$  for class  $c$ . One can then get the output via  $\exp(S_c) / \sum_c \exp(S_c)$ . Plugging  $F^k = \sum_{x,y} f_k(x, y)$  in the class score ( $S_c$ ) expression, one obtains  $S_c = \sum_k w_k^c \sum_{x,y} f_k(x, y) = \sum_{x,y} \sum_k w_k^c \cdot f_k(x, y)$ .

The Class Activation Map is defined by  $M_c(x, y) = \sum_k w_k^c f_k(x, y)$ , and thus  $S_c = \sum_{x,y} M_c(x, y)$ . This expression shows how Class Activation Map  $M_c(x, y)$  directly represents the importance that pixel position  $(x, y)$  represents for a class  $c$ .

In sum, getting the CAM is a matter of, in the case of resnet18, extracting the  $f_k$  tensor of feature blobs from the end of layer4 and the  $w_k^c$  tensor from the linear layer in the fully connected section, which lead to the class prediction of class  $c$ .

With the extracted CAM bounding box, a first region of interest will be extracted. This ROI will then be an input to another resnet18 which will be trained on these new ROIs, and will output a second ROI. This will be done for all views. A diagram representing the full network and the interactions that lead to a class prediction can be found in Attachment K. The architecture relating just to the

ROI extraction for a single view is presented below.

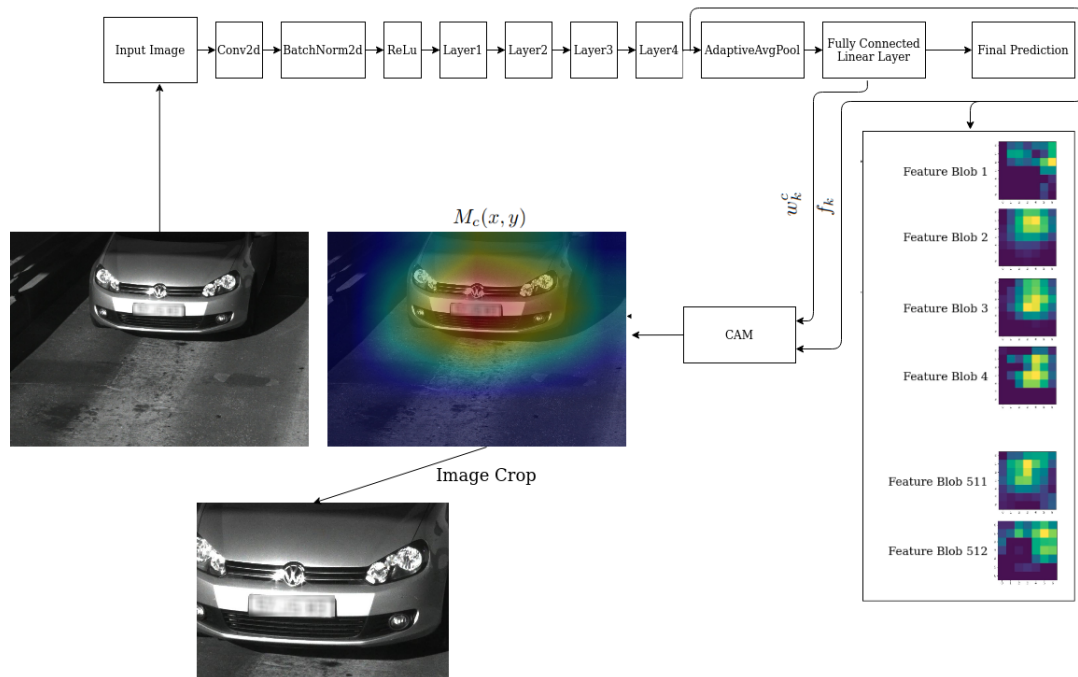


Figure 29: *One ROI extraction. The input image is passed in a resnet18 network trained to predict the vehicle model based on the non-cropped vehicle images. With CAM, a bounding box can be obtained via a weakly supervised method that corresponds to regions which lead to the obtained class prediction.*

Following this same pipeline, another ROI will be extracted from the resulting first ROI. For the input exemplified previously it would result in the following:

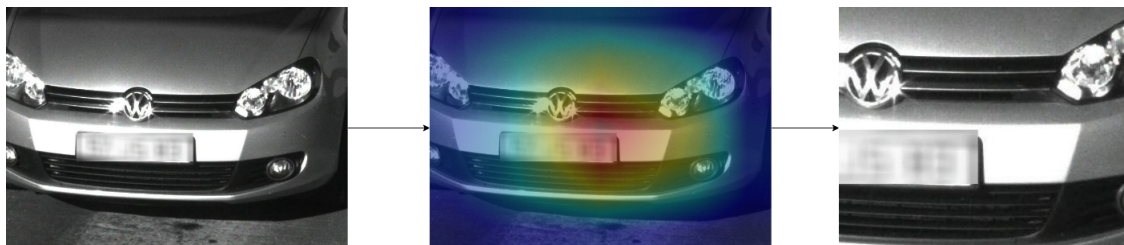


Figure 30: *Example of second extracted ROI, for the same view as previously. It follows the same pipeline of figure 29, using a resnet18 network trained to predict vehicle model based on the first extracted ROIs.*

This is not the state of the art for object detection and bounding box extraction, but it was chosen

because it is a weakly-supervised solution that doesn't require manually labelling bounding boxes, which saves long hours of repetitive work, since only two bounding boxes per input are actually being used.

The expectation with this method is that the first ROI will act and bring benefits in a similar way to the manually annotated bounding boxes of before, and the second ROI will help boost performance by zooming on known descriptive regions, inspired by [31],[32] and [33].



## 5 Results

All the following tests were performed under the same conditions: 16 batch size, Stochastic Gradient Descent as the optimizer, a learning rate of 0.001, momentum of 0.9 and Cross Entropy as the loss criterion. Results on the same tables all performed either 300 epochs, or enough epochs that the learning gradient reached 0.

### 5.1 CompCars preliminary tests

Below are the results relating to testing standard CNNs with no changes, with the front-view of the CompCars dataset.

Table 5: *CompCars Results*

CNN Architecture	Acc.
VGG16	99.75%
Resnet50	99.87%
Resnet18	99.73%
Densenet161	99.84%
InceptioV3	99.82%

### 5.2 ShortBrisa Preliminary tests

Replicating the tests done on CompCars to ShortBrisa, the results in Table 6 are achieved.

Table 6: *ShortBrisa Performance results.*

CNN Architecture	Pre-trained model (Test/Train)	From scratch (Test/Train)
Resnet18	78.36%(100%)	30.87%(100%)
Resnet50	86.81%(100%)	26.65%(100%)
InceptionV3	91.56%(100%)	44.06%(95.72%)
Densenet161	91.03%(100%)	61.48%(100%)
VGG16	84.38%(100%)	41.69%(100%)

It is observable how performance drops significantly, with a great degree of overfitting. This is due to the issues already discussed that usually affect FGVC datasets. One can also confirm the conclusions already discussed regarding transfer learning, that pre-trained models outperform models trained from scratch.

With the predictions and respective ground truths of both datasets, the precision and recall curves of each dataset can be plotted, for an easier visualization of the difference in performance.

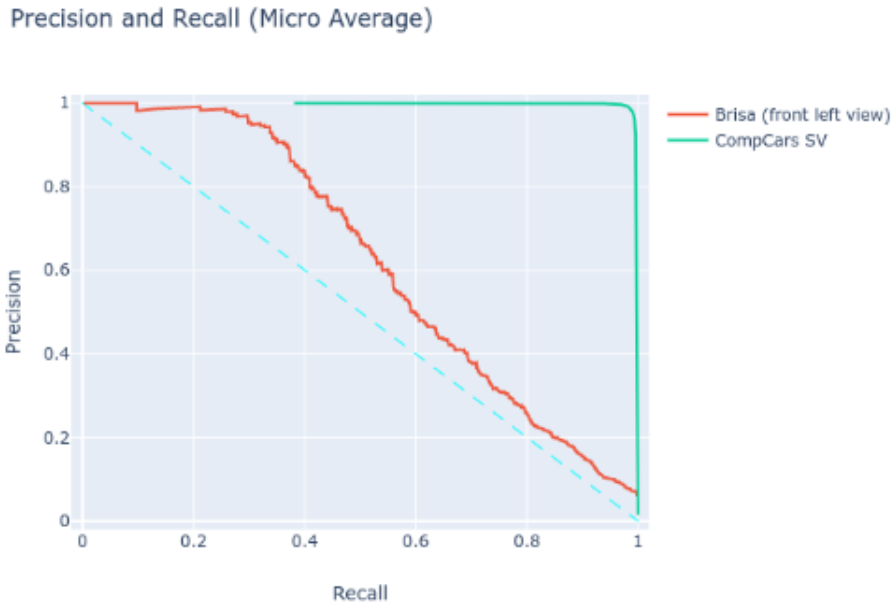


Figure 31: *Precision and recall curves for CompCars and ShortBrisa.*

Following the guidelines laid out in section 2.4.3, it can be visually inferred that on the CompCars dataset performance is better, as the Area Under the Curve of the Precision-Recall plot (AUCPR) is greater than that of ShortBrisa, which indicates better values of Precision and Recall pairs for the predicted classes.

### 5.3 Bounding Box and Data Augmentation addition results

Employing the dataset augmentations discussed and the annotated bounding boxes, the results in Table 7 were achieved.

This confirms the assumptions laid out by the state-of-the-art analysis: that bounding boxes and data augmentation are paramount to good results in FGVC CNN based image classification. Precision and recall data for each class and curve for the Resnet50 results can be consulted in attachment L.

Table 7: *Bounding Box plus Data Augmentation results.*

CNN Architecture	Acc.
VGG16	93.40%
<b>Resnet50</b>	<b>95.25%</b>
Resnet18	91.56%
InceptioV3	94.99%
Densenet161	93.93%

## 5.4 Multi-View tests

In section 4.1.2 the available dataset was detailed and it was verified that there are 3 available views. Previously, in sections 5.1, 5.2 and 5.3 tests were performed to assess baseline performance and starting performance. These tests were performed using only the front-view because intuitively it is the pose with which it would be easier for a human observer to identify the make of a vehicle, and, by association, given that neural networks "mimic" an organic neuron, it would be expected that it is also the view which provides better results with CNNs. To verify that this is indeed the case, the dataset with each view, individually, was trained on the residual networks used so far.

 Table 8: *Individual View performances.*

CNN Architecture	Front View (FL)	Rear View (RL)	Overview (OV)
Resnet50	95.25%	84.17%	79.68%
Resnet18	91.56%	78.24%	71.14%

The results confirm the assumption that the front-view is the most descriptive by a good margin. However it is evident that there are still descriptive features in other views for vehicle make recognition. The following subsections showcase the results achieved when fusing information from multiple views.

### 5.4.1 MC-CNN vs MV-CNN

Table 9 shows results obtained when implementing and testing the MC-CNN and MV-CNN architectures.

From the results, one can see that the MC-CNN idea has not delivered the hoped for results, with

Table 9: *Comparison of results in MC-CNN vs MV-CNN.*

CNN Architecture	Input	Acc.
MC-CNN-Resnet18	FL-OV-RL	88.39%
MC-CNN-Resnet50	FL-OV-RL	92.08%
MV-CNN	FL-RL	94.72%
MV-CNN	FL-OV	92.61%
MV-CNN	RL-OV	84.70%
<b>MV-CNN</b>	<b>FL-OV-RL</b>	<b>95.52%</b>

performance actually degrading significantly on both the Resnet50 and Resnet18, when comparing the results with those obtained in Table 7.

The MV-CNN on the other hand has delivered new top performance when fusing information from FL, OV and RL views. Precision and Recall data and plot for this particular result can be consulted in attachment M.

## 5.5 Image Patch tests/study

To study the impact that the addition of image patches containing zoomed features could have on performance, tests were carried out. The results are detailed below.

### 5.5.1 MC-CNN

In table 10 are presented the results for MC-CNN, where each sequence of numbers inside brackets represents an arrangement of image cropped patches, according to the numbering that can be found in Attachments H, I and J. As explained previously, in the case of MC-CNN, these patches are inserted and added channel wise, and so the input to the network will be of size 224x224xNumber\_of\_patches.

Evidenced in this table of results, is the fact that MC-CNN, on contrary to what was seen previously, can also achieve top-performance (95.52%) under the right circumstances. Precision and Recall data for this architecture can be seen in attachment N.

### 5.5.2 MV-CNN

Similarly, tests were carried out using the MV-CNN. Results are visible in Table 11.



Table 10: *Results for MC-CNN. Numbers inside brackets were introduced in the network by increase the channel depth of the first layer.*

CNN Architecture	Patches Selected	Acc.
MC-CNN-Resnet18	[30,61,92]	88.39%
MC-CNN-Resnet18	[30,61,30,92,12,14,20,30,21,46,72,79,61]	93.14%
MC-CNN-Resnet18	[30,6,12,13,14,18,19,20,21,30,22,24,25,26,27,28,30]	93.14%
<b>MC-CNN-Resnet18</b>	<b>[30,61,92,12,14,20,21,46,72,79]</b>	<b>94.20%</b>
MC-CNN-Resnet18	[30,12,13,14,18,19,20,21,24,25,26,27]	93.67%
MC-CNN-Resnet18	[12,13,14,18,19,20,21,24,25,26,27]	93.40%
MC-CNN-Resnet50	[30,61,92]	92.08%
MC-CNN-Resnet50	[30,61,30,92,12,14,20,30,21,46,72,79,61]	94.20%
<b>MC-CNN-Resnet50</b>	<b>[30,6,12,13,14,18,19,20,21,30,22,24,25,26,27,28,30]</b>	<b>94.72%</b>
<b>MC-CNN-Resnet50</b>	<b>[30,61,92,12,14,20,21,46,72,79]</b>	<b>94.72%</b>
<b>MC-CNN-Resnet50</b>	<b>[30,12,13,14,18,19,20,21,24,25,26,27]</b>	<b>95.52%</b>
MC-CNN-Resnet50	[12,13,14,18,19,20,21,24,25,26,27]	93.67%

Some things to know for correct interpretation of Table 11: all patches are gray level images (i.e., only one channel).

Patch numbers 30,61 and 92 are gray level versions of Front-Left (FL), Overview (OV) and Rear-Left (RL), respectively (i.e., only one channel). FL, OV and RL are three channel images, with the OV view being an RGB Image, while the others are three channel gray images.

In the "CNN Architecture" column, the network name is codified in a way to tell the number of parallel networks. For example, MV-CNN-6 denotes 6 parallel networks.

In the "Images" column, individual network inputs are within brackets (i.e., [12,14,20] means that patches 12,14 and 20 were fed in a 224x224x3 tensor, as an input to one of the parallel Resnet18s). Likewise, an arrangement of patches like [30][61][92][12][14][20][46][52][53][72][78][79] means that each of these patches were fed to a dedicated Resnet18 network as a 224x224x1 tensor.

In sum, patches that were stacked in channel depth are separated by commas, while patches that were assigned to different Resnet18s are separated by brackets. Precision and Recall data and plot for model MV-CNN-6 can be consulted in attachment O, and in attachment P for model MV-CNN-12.

Table 11: *Results for MV-CNN. Patches that were stacked in channel depth are separated by commas, while patches that were assigned to different Resnet18s are separated by brackets. New top-performance, accuracy wise, is achieved by the bold entries on table 11.*

CNN Architecture	Images	Acc.
MV-CNN-3	[FL][OV][RL]	95.25%
MV-CNN-6	[14,20,30][46,53,61][72,78,92]	94.46%
MV-CNN-12	[30][12][13][14][18][19][20][21][24][25][26][27]	93.14%
MV-CNN-2	[FL][14,15,20]	94.20%
<b>MV-CNN-6</b>	<b>[FL][OV][RL][12,14,20][46,52,53][72,78,79]</b>	<b>95.78%</b>
<b>MV-CNN-12</b>	<b>[30][61][92][12][14][20][46][52][53][72][78][79]</b>	<b>96.04%</b>
MV-CNN-2	[FL][19,20,21]	93.67%

### 5.5.3 Image Patches and Multi-Views validation

So far, good results and progress have been achieved. However, it seems accuracy is stagnating at around 95.5%, which leads to the question: are the multiple views and image zoomed features actually improving performance or is it negligible? After all, it adds an extra layer of complexity, and the gains are not too significant, because due to the small amount of test data (379 entries), a wrong or right prediction corresponds to a change of 0.26% in accuracy, and so, the difference from the 95.52% achieved with resnet50 and the Front-View only, to the 96.04% achieved with MV-CNN-12 corresponds to only 2 more right predictions.

Thus a test was performed where noise was introduced in one of the views as shown below, to evaluate the effect that this has on performance.

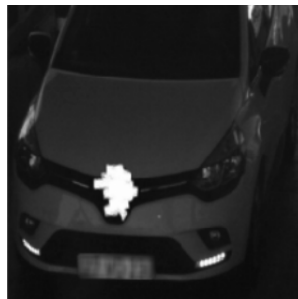


Figure 32: *Example of manually inserted noise.*

Table 12: *Noise tests over different architectures that had presented good results so far.*

CNN Architecture	Image/View arrangement	Original Acc.	Noise Acc.
Resnet50	[FL]	95.25%	74.41%
MC-CNN-Resnet50	[30,12,13,14,18,19,20,21,24,25,26,27]	95.52%	48.81%
MV-CNN-3	[FL][OV][RL]	95.52%	91.56%
MV-CNN-6	[FL][OV][RL][12,14,20][46,52,53][72,78,79]	95.78%	87.60%
MV-CNN-12	[30][61][92][12][14][20][46][52][53][72][78][79]	96.04%	89.18%

## 5.6 LargeBrisa Results With ROI extraction

The ROI extraction method was tested with the newly available LargeBrisa dataset, with results as shown in Tables 13 and 14.

Table 13: *Comparative results of ROI architecture with MV-CNN and a simple resnet18.*

Views	CNN Architecture	Acc.
Front-Left(Not Cropped)	Resnet18	94.78%
Front-Left(First ROI)	Resnet18	95.81%
Front-Left(Second ROI)	Resnet18	96.46%
Front-Left(First ROI + Second ROI)	MV-CNN-2	96.03%
4 Views (Not Cropped)	MV-CNN-4	96.20%
4 Views (First ROI)	MV-CNN-4	97.31%
4 Views (Second ROI)	MV-CNN-4	97.15%
<b>4 Views (First ROI + Second ROI)</b>	<b>MV-CNN-8</b>	<b>97.77%(98.02%*)</b>

Once again new top performance for this task is achieved, and, upon review of the missed predictions it was found that actually 3 inputs were mislabeled, and although they were flagged as incorrect, they were actually correct predictions. Adjusting for this, the new top accuracy becomes **98.02%\***. This correction was not performed for other results of Table 13, because this error was found only after analysing results in detail, and the models that lead to the other results were not saved.

Precision and recall data for MV-CNN-8 can be checked in attachment Q. The 1.98% failed predictions can be verified in attachments A through G.

Table 14: *Table 13's MV-CNN-8 figures of merit.*

Accuracy Metric	Acc.
top-1 Accuracy	98.02%
top-2 Accuracy	98.65%
top-3 Accuracy	98.89%
top-5 Accuracy	99.52%

## 6 Results Discussion

### 6.1 CompCars Preliminary tests

On Table 5, one can verify that all network architectures achieve over 99% accuracy on CompCars, which is not surprising, since, as was said previously, the dataset is, for the most part, composed of high quality, homogeneous images. As such, not many conclusions can be extracted, apart from confirmation that FGVC of vehicles is achievable.

### 6.2 ShortBrisa results analysis

As can be observed in table 6, performance drops significantly when the FGVC dataset provided by Brisa is used, under the same exact conditions that CompCars was tested with. This is due to the issues evidenced in 4.1.2.

To verify the benefits of transfer learning, the models were also trained without initialized weights, which proved that in this case performance drops dramatically with a great degree of overfitting, which confirms the observations of previous works on transfer learning and its importance for Deep Learning. During training, residual networks were also noticeable faster both to converge to a solution, and performing a complete training epoch.

In sum, this section of results allowed us 3 main takeaways:

- 1: It is possible to perform FGVC for vehicles with the right pre-conditions (i.e., sizeable, well curated dataset and the right neural network architecture).
- 2: Residual Neural Networks are good for this task, offering a good trade-off between performance and lightweight footprint.
- 3: Using Pre-Trained models allows us to take advantage of general feature extraction capabilities of CNNs trained on ImageNet and, through transfer learning, specialize them on custom fine-grained datasets.

Observing the plotted precision and recall curves for the CompCars dataset and ShortBrisa dataset (both on the resnet50 model), a starting point is measured and defined, and so the final objective can be set: to get a Precision-Recall curve with ShortBrisa dataset as similar as possible to the one observable with the CompCars dataset.

### 6.3 Bounding Box and Data Augmentation Results

With the annotated bounding boxes, significant performance gains were achieved, as was expected. Again, the trend of residual networks having good performance and the lowest training time was maintained, which leads to the possibility that these architectures are possibly the most interesting solution because, as can be seen, Resnet50 performs the best so far with 95.25% accuracy, with even the Resnet18 (which has an even lighter foot print) showing good promise and performance increases. Other networks perform well too, but the trade-off between performance and computing power required seems to be more advantageous with the residual networks. As such, the bulk of the work done after these tests was performed on residual networks.

### 6.4 Multiple Views tests

In section 4.1.2 the available dataset was detailed and it was seen that there are 3 available views. Prior to this point, the tests performed only took advantage of a single, frontal view. And it was shown that it was indeed possible to achieved good vehicle recognition based on only one view.

Individual tests on the other views were performed, to verify if it was possible to perform vehicle recognition with them. Intuitively, the front-view should deliver best results, because it is the pose with which it would be easier for a human observer to identify the make of a vehicle, and, by association, given that neural networks "mimic" an organic neuron, it would be expected that it is also the view which provides better results with CNNs. To verify that this is indeed the case, the dataset, with each view individually, was trained on the same residual networks used so far, as seen in Table 8.

The results confirm the assumption that the front-view is the most descriptive by a good margin. However it is evident that there are descriptive features in other views for vehicle make recognition.

Results with MC-CNN and MV-CNN (Table 9), which merge the information from the different views, although not absolutely groundbreaking (mainly in the case of MC-CNN) still improved performance somewhat (in the case of MV-CNN), giving a new best performance of 95.52%. This isn't that much of a significant improvement because, as seen previously, it corresponds only to 1 more correct prediction when compared to the 52.25% results of Resnet50. It is still progress nonetheless, and most of all, it provides a base to experiment different combinations of data, as it will be seen next.

## 6.5 Multiple Views with Image Patches

Due to related work that proved such strategies successful, efforts were then focused on using zoomed in features, with the patch selection being based on the results obtained in 4.5.

With the patch solution explained previously, it was possible to match 95.52% accuracy on MC-CNN (Table 10), and surpass it on MV-CNN (Table 11) with an accuracy of 95.78% and 96.04%. This seems to show that forcing the network to focus on zoomed in features does bring performance benefits. It was noticed however that it is critical to select the appropriate patch with the appropriate info, because in situations where the selected patch was non-descriptive, it had a noticeable negative effect on performance.

Due to the apparent saturation of accuracy in the region of 95 to 96%, other avenues to explore the usefulness of feature zooms were explored, as shown in 5.5.3 (in Table 12). These dataset noise tests seem to indicate a clear gain in system reliability when using multiple views, compared to using the single view. The test tried to mimic a situation where sun light would be reflecting off the make badge of the car (which is definitely possible, if not even common), and in this situation, the single-view solution dropped performance drastically, by over 20 percentile points. The other architectures, where data from the other inputs is present, also drop percentage points in accuracy, but still maintain an acceptable level of performance.

Interestingly, the second row on Table 12 shows how if one makes the network rely too much on a single feature, accuracy degrades even more than with the single-view.

Taking all these observations into account, it can be concluded that multi-view inputs definitely offer, along with performance, system reliability. One can also conclude that while zoomed patches seem to benefit performance, if too much focus is placed in a single feature, the network performs a kind of overfitting, where it can only recognize a given object if that feature is there, i.e., it will ignore all other features.

## 6.6 ROI extraction

It was then implemented a method that takes advantage of CAM to sequentially extract smaller ROIs that are mostly guaranteed to contain a variety of good features, and not just one feature in particular. At this time it was also made available a larger dataset, with more samples and classes.

The results derived from both these additions lead to a performance level which is competitive to similar state-of-the-art approaches.

With the new LargeBrisa dataset, the results so far achieved are further validated, as with more possible output classes and types of vehicles to identify, performance is maintained. It is also strongly believed that this addition of data contributed heavily to the improved performance, due to less under represented classes. This shows how a well-curated and complete dataset is the most important factor when working with Deep Learning in general, and Deep Learning applied to pattern recognition in particular.

The devised double ROI method also helps to attenuate two shortcomings identified previously: it allows to avoid manually labeling bounding boxes (although manual labeling of vehicle make still has to be performed, as this is only a weakly supervised approach and not an unsupervised one), and guarantees good patch placement and feature content, since only regions that contributed to correct identification in the first stage can be selected. Furthermore, analysing top-2, top-3 and top-5 scores, one can further assess how valid this method is for the proposed task. Since this will not be the only recognition system in place, and is to be used in cross-reference with other systems, a top-2 or top-3 score can still guarantee that the correct vehicle has passed the toll system, if one of the other validation systems reports the same vehicle being reported by the solution proposed in this work.

The system also achieves both good precision and recall, important metrics for unbalanced datasets, that lend credibility to the results. In the end, the objective of approximating the Precision and Recall curve of the devised solution to the curve obtained on the CompCars dataset was achieved.



## 7 Closing Remarks

The work here presented explored several state-of-the-art research works in the field of Deep Learning applied to Computer Vision. With the knowledge acquired from analysing these documents, a strategy was formulated to successfully tackle a FGVC vehicle recognition problem.

The current state-of-the-art for LSVC and FGVC was verified, and it is evident that Deep Learning and CNNs are an established and valid solution, that has made remarkable progress in the last 8 years, increasingly drawing the attention of researchers from all areas of science where pattern recognition and data analysis is of interest.

It was also noted that while state-of-the-art research on FGVC is progressing at an accelerated pace, much of the research being made evaluates results on research oriented datasets. It was established that research datasets do not often reproduce real-life conditions that one encounters when solving a real-life problem, such as incomplete and noisy data. As such, it is very possible to get good results on these research datasets, but harder to guarantee that they will translate well to real applications.

The before mentioned real-life conditions were present on the dataset provided to us by Brisa, and it was confirmed how challenging achieving good performance can be. To tackle this, several methodologies were adopted.

It was confirmed the importance of data, or more importantly, how negatively lack of data impacts CNN performance. Data Augmentation was employed, and together with Bounding Box annotations, this combination offered a very significant improvement in performance, proving why these strategies are a staple in most recognition tasks.

It was further proved that data is the most important factor in CNN tasks when more of it was made available to us by Brisa, allowing us to build a second, more complete dataset. Accuracy was stagnating at around 96% for every architecture that was tried, and it wasn't until it was possible to expand the dataset that this barrier was broken.

The fact that multiple viewpoints of the data were available also contributed greatly in optimizing the solution. With these extra views it was possible to create a more robust solution, when compared to the single-view alternative. Taking in account that this is to become a real-life application, running 24/7, replicated over several toll stations, it is almost inevitable that external factors will interfere with predictions (i.e., camera malfunctions). It was proven how multiple sensors (cameras) acquiring multiple perspectives of data can decrease the effects of sensor failure. In addition to this advantage

in reliability, the multi-view neural network architectures implemented also demonstrated that multi-perspective information can be used and fused together, in a way that these perspectives complement each other, which in the case of this work, lead to the best results in every architecture tested, when compared to single-perspective inputs.

It was verified that further emphasizing descriptive regions can lead to significant accuracy increase, even after achieving apparent peak performance for a given architecture, as long as these regions are carefully and correctly chosen, and that this emphasis is not so great that it makes the CNN exclude other features. Doing this allows the chosen neural-network architecture to "focus" on the important features, lending less importance to noisy or less-relevant/irrelevant data.

Inspired by all the related-work read and results observed during the course of the work, a final weakly-supervised method was proposed. It incorporates the multi-view aspect (thus guaranteeing reliability) and ROI extraction, that both excludes non-descriptive regions and emphasizes descriptive features, while not requiring manually labeling bounding boxes and selecting feature patches, thus reducing human error and time required to implement, guaranteeing that the ROIs are only placed in regions that were perceived as descriptive by the first stage of the architecture, since their placement is based solely on data from within the network.

In sum, it is believed that this system fulfils the requirements to solve the problem at hand and that it can easily and rapidly be applied to any other field where multi perspectives are useful and available, specially if the categories are fine-grained. The results seem to be competitive with the state-of-the-art, with the two ROI extraction stages proving their effectiveness in increasing performance.

## 8 Future Work

The work presented delivers good results, but the architecture has become quite complex with an heavy footprint. There are 8 Resnet18 networks before the networks of the MV-CNN, which gives us a total of 16 Resnet18 networks working in cooperation. It is possible that this number can be reduced. The dataset has four perspectives: two front-views and two rear-views. It is likely that the front and rear views can share the ROI extraction network and still obtain good ROIs. It may also be possible to develop a training scheme to create a single network for ROI extraction, for each level of ROI. Another possibility is that, since the classifications that these first stage Resnet18s provide are not of real interest, such a deep CNN may not be needed. It may be possible to either use another shallower model, or implement a custom network for this.

The dataset entries that proved to be the most challenging to correctly predict were mostly some type of non-standard vehicle: either trucks or work vans with some specialized feature (like a cargo area). The approach by [28] could be used, creating a dedicated class for these types of vehicles, although then the question becomes if one isn't simply circumventing the problem, since at that point vehicle make recognition isn't being performed. One could also explore the idea of more classification networks for different types of vehicles as a solution to this (i.e., a dedicated network to classify trucks, a dedicated network to classify vans, etc.), although this increases complexity, footprint and development time. The system also struggled on some image sets where viewing conditions were poorer, and in this case one could further explore the work in [34].

The acquisition and labeling of more data lead to good breakthroughs in performance, and thus, this task is one where continuous data searching and improvement is always a benefit. In the particular case of the architecture here presented, benefits would be exponentially good: more data would, by itself, improve classification performance, but it would also improve the ROI extraction stage, which would also improve classification.

Finally, the system could be extended to perform not just make recognition but also model recognition. However, labeling for this task could prove much more challenging, due to situations where labelling make was already hard (due to lighting condition or others), and because labeling models needs more expertise knowledge, or at the very least requires more time researching and comparing vehicle models. This would also increase the number of classes dramatically, spreading the number of samples per class curve, and would probably lead to many under represented classes, which


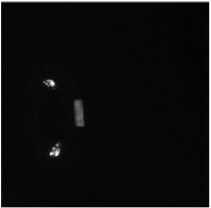
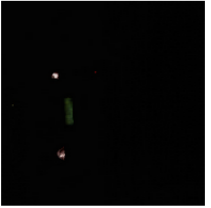
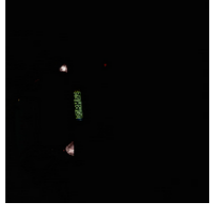




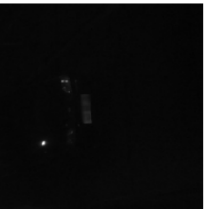




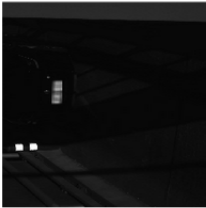


again shows how important the gathering of data is.

In sum, it is recommended that future work focuses on simplifying the devised architecture where possible, gathering more data and extending functionality.

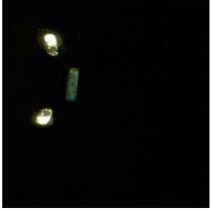
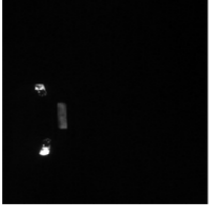
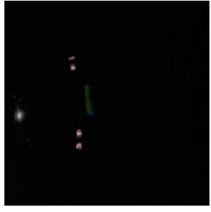
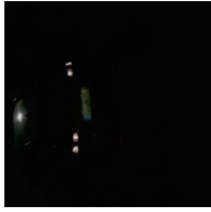

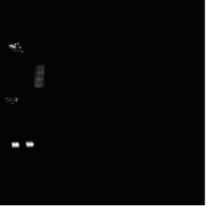







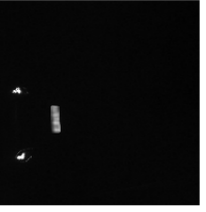
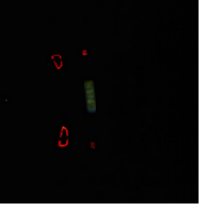



## 9 Attachments


### 9.1 Attachment A: MV-CNN-8 Prediction Errors 1

Front View 1	Front View 2	Rear View 1	Rear View 2	Prediction	Ground Truth(*)	Correct Prediction In
				Renault	Ford	2 Tries
				Volkswagen	Hyundai	18 Tries
				Renault	Fiat	4 Tries
				Peugeot	Jaguar	5 Tries

### 9.2 Attachment B: MV-CNN-8 Prediction Errors 2

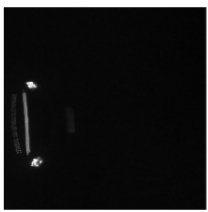
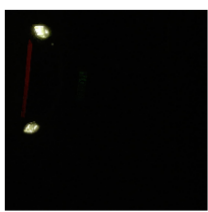
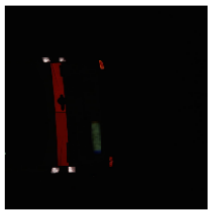






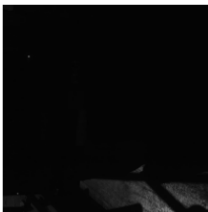


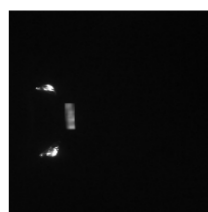

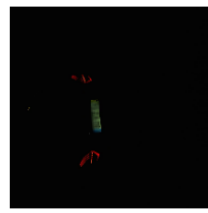
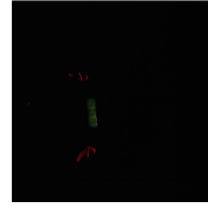
Front View 1	Front View 2	Rear View 1	Rear View 2	Prediction	Ground Truth(*)	Correct Prediction In
				Volkswagen	Citroen	5 Tries
				Kia	Hyundai	3 Tries
				Peugeot	FIAT	3 Tries
				Renault	Ford	5 Tries

### 9.3 Attachment C: MV-CNN-8 Prediction Errors 3

Front View 1	Front View 2	Rear View 1	Rear View 2	Prediction	Ground Truth(*)	Correct Prediction In
				Renault	Jeep	9 Tries
				Setra	MAN	2 Tries
				Mercedes	MAN	4 Tries
				Sertra	MAN	2 Tries



### 9.4 Attachment D: MV-CNN-8 Prediction Errors 4

Front View 1	Front View 2	Rear View 1	Rear View 2	Prediction	Ground Truth(*)	Correct Prediction In
				Volkswagen	Mercedes	2 Tries
				Volvo	Nissan(Volvo)*	*
				Scania	Nissan	18 Tries
				Seat	Peugeot(Ford)*	2 Tries(6 Tries)*

\* Correct class.

### 9.5 Attachment E: MV-CNN-8 Prediction Errors 5






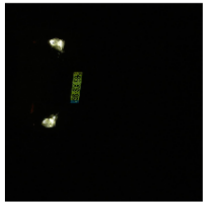

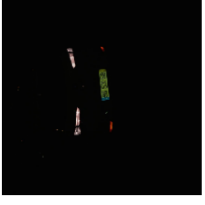
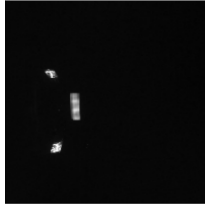
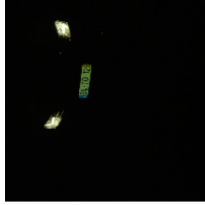
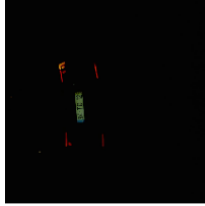


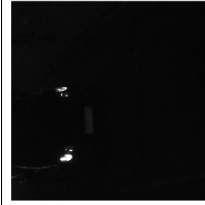

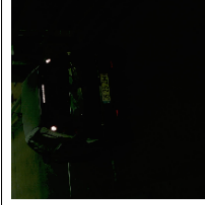
	Front View 1	Front View 2	Rear View 1	Rear View 2	Prediction	Ground Truth(*)	Correct Prediction In
					Mitsubishi	Peugeot	5 Tries
					Volkswagen	Peugeot (Volkswagen)(*)	*
					Fiat	Porsche	5 Tries
					Citroen	Renault	3 Tries

\* Correct class.

### 9.6 Attachment F: MV-CNN-8 Prediction Errors 6

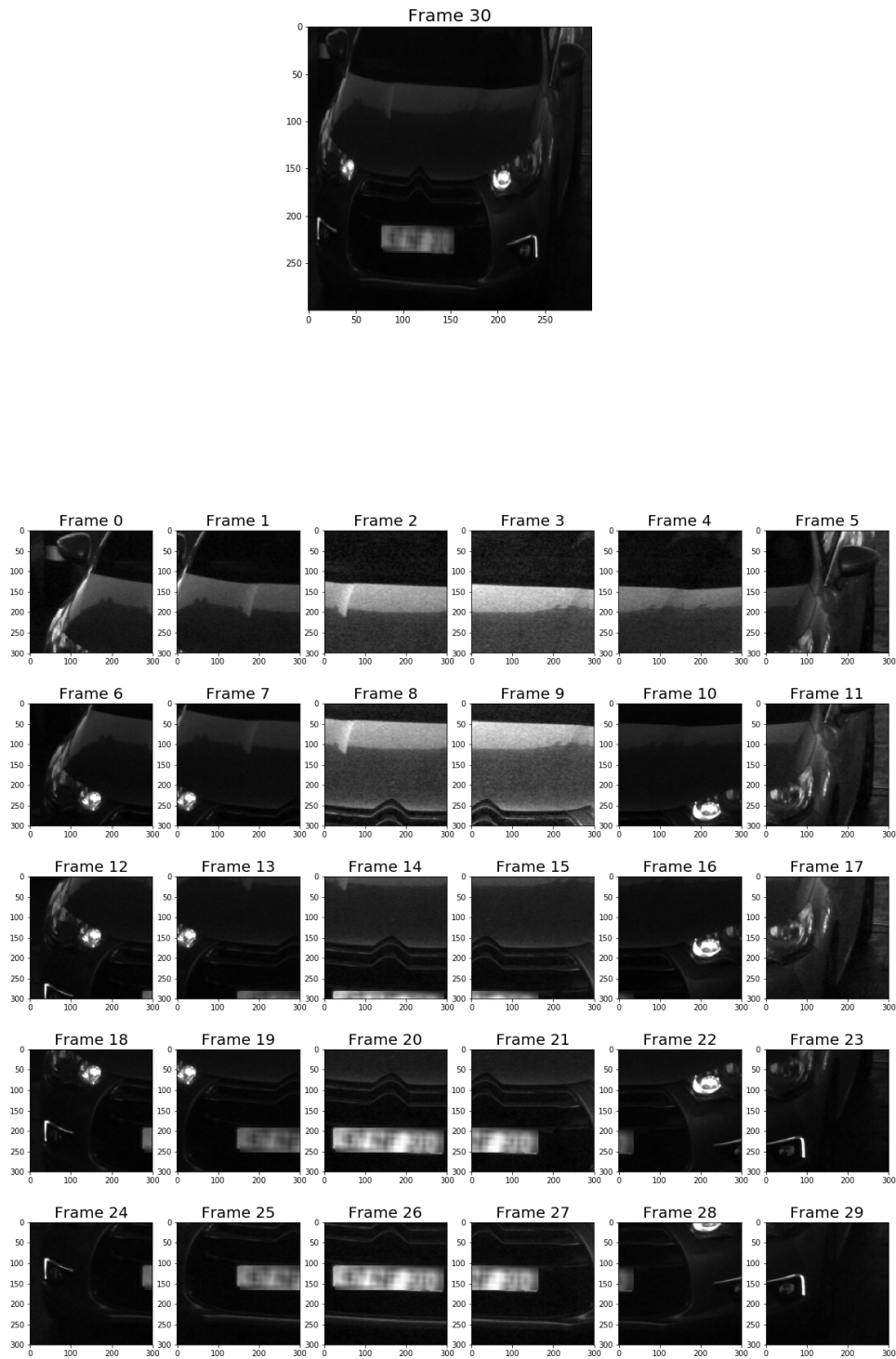
Front View 1	Front View 2	Rear View 1	Rear View 2	Prediction	Ground Truth(*)	Correct Prediction In
				Mitsubishi	Renault	2 Tries
				Smart	Renault	2 Tries
				Renault	Scania	2 Tries
				Setra	Scania	2 Tries

### 9.7 Attachment G: MV-CNN-8 Prediction Errors 7

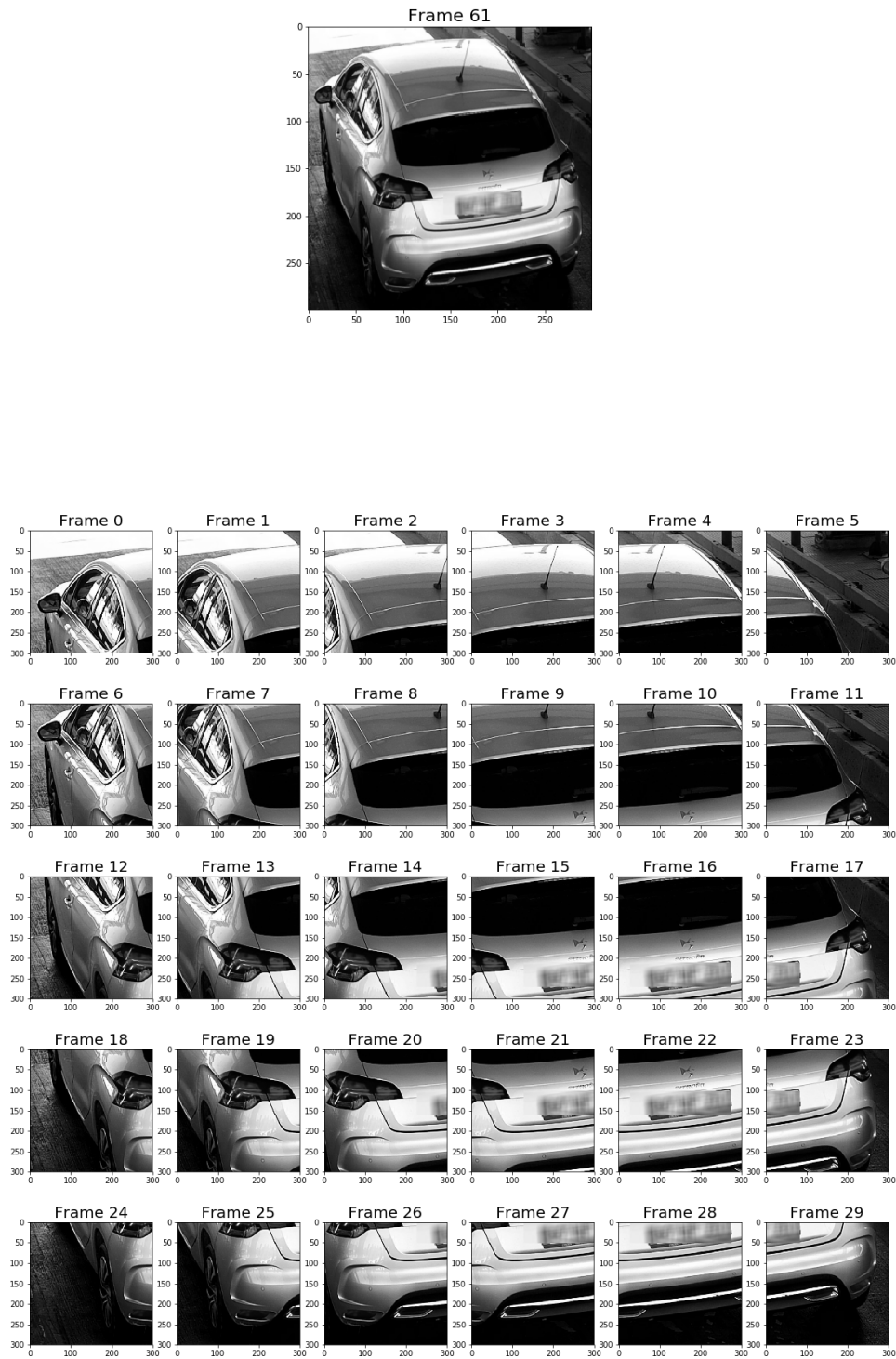
Front View 1	Front View 2	Rear View 1	Rear View 2	Prediction	Ground Truth(*)	Correct Prediction In
				Peugeot	Seat(Peugeot)(*)	*
				Opel	Seat	4 Tries
				Opel	Volkswagen (Citroen*)	3 Tries(12 Tries)*
				Fiat	Smart	9 Tries

\* Correct class.

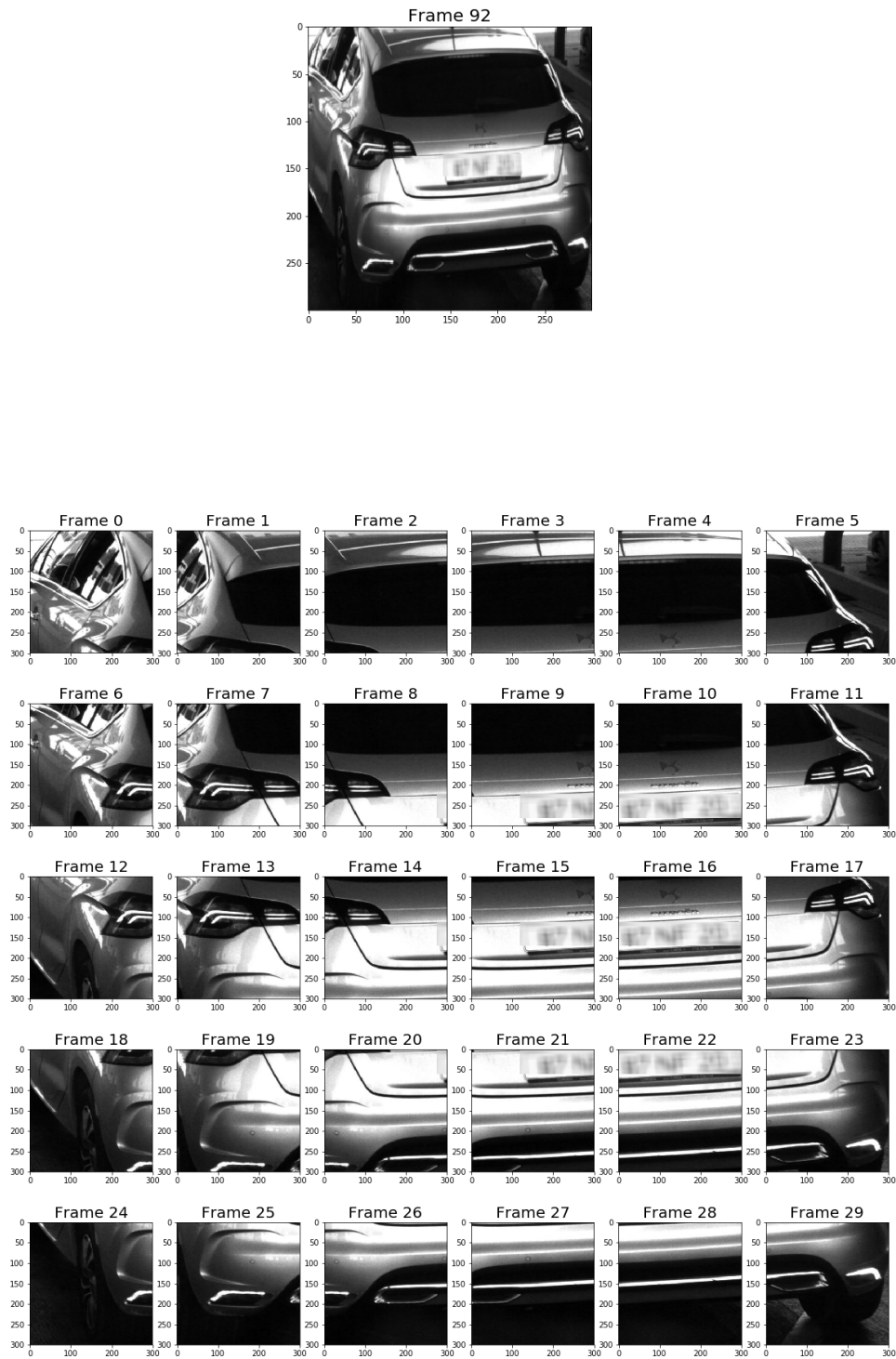
## 9.8 Attachment H: Patch Selection for Front View



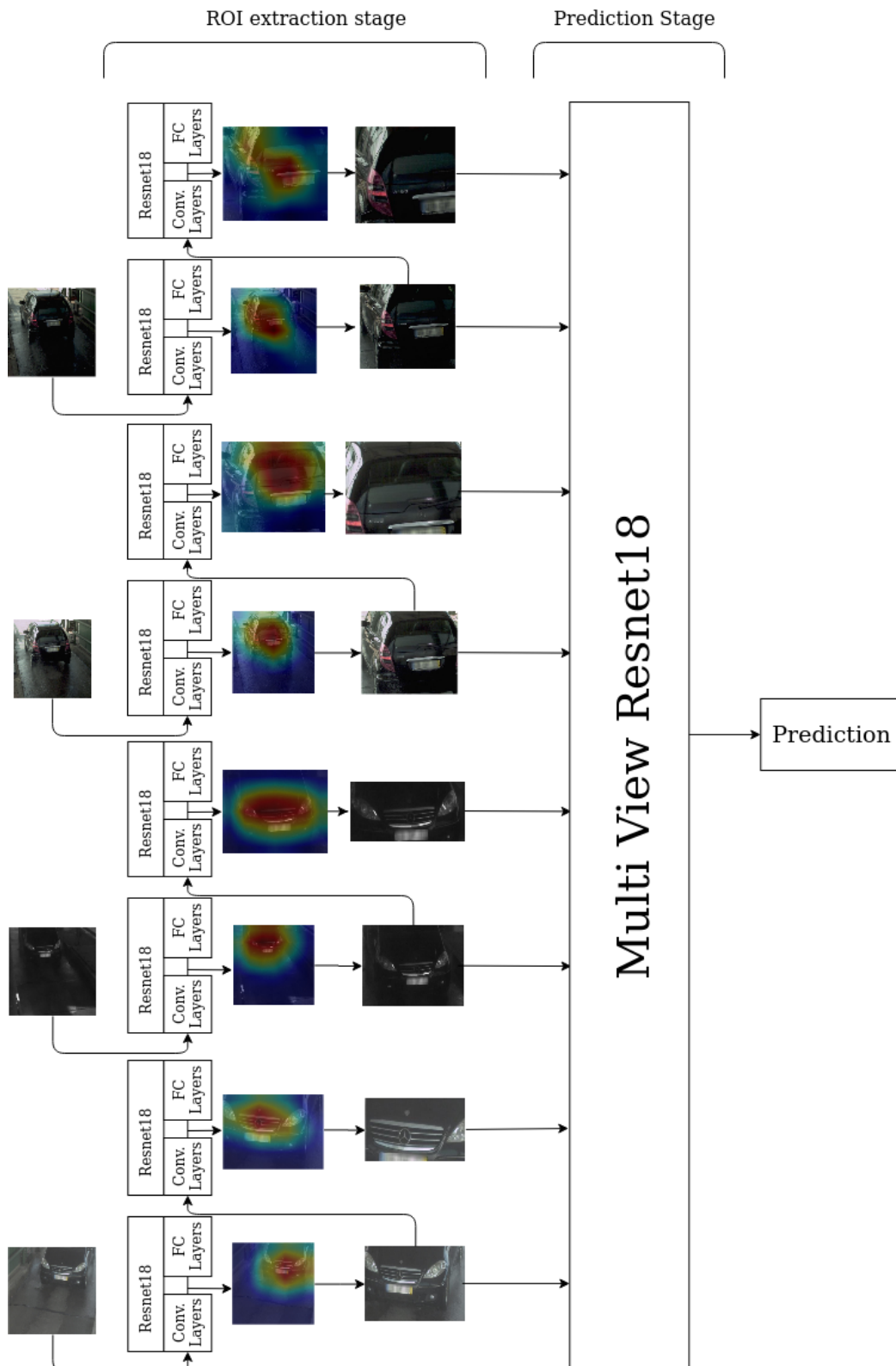
### 9.9 Attachment I: Patch Selection for Overview



### 9.10 Attachment J: Patch Selection for Rear View



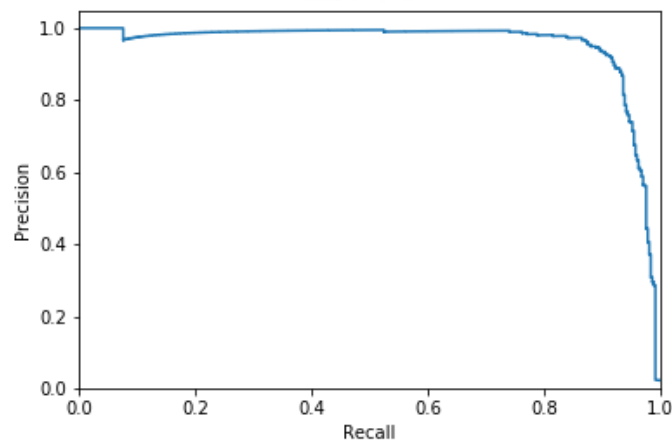
### 9.11 Attachment K: Full ROI extraction architecture description





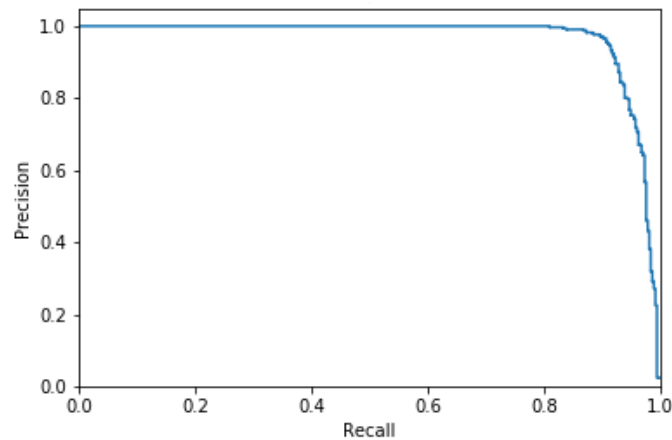
**9.12 Attachment L: Prediction-Recall data for Resnet50 with FL view, as seen in Table 7 (95.25% accuracy)**

	precision	recall	f1-score	support
Audi	1.000	1.000	1.000	14
BMW	1.000	1.000	1.000	20
Citroen	0.929	1.000	0.963	26
Fiat	0.880	1.000	0.936	22
Ford	0.947	1.000	0.973	18
Mercedes	0.947	1.000	0.973	54
Mitsubishi	0.909	0.909	0.909	11
Nissan	0.786	0.786	0.786	14
Opel	0.955	0.875	0.913	24
Peugeot	0.970	0.941	0.955	34
Renault	0.982	0.947	0.964	57
Scania	0.917	0.846	0.880	13
Toyota	1.000	0.833	0.909	18
Volkswagen	1.000	1.000	1.000	32
Volvo	0.955	0.955	0.955	22
accuracy			0.953	379
macro avg	0.945	0.939	0.941	379
weighted avg	0.954	0.953	0.952	379



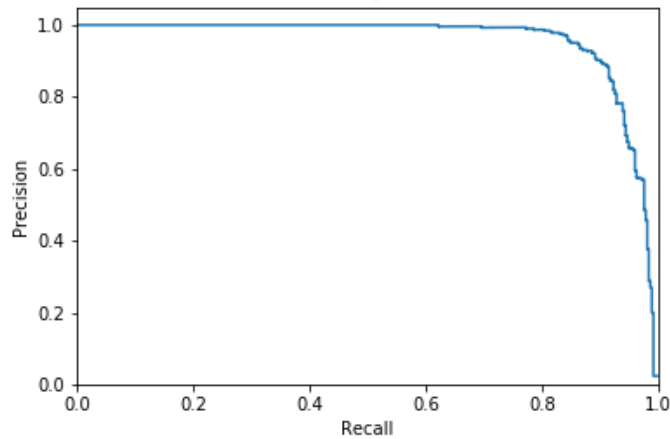
**9.13 Attachment M: Prediction-Recall results MV-CNN, as seen in table 9 (95.52% accuracy)**

	precision	recall	f1-score	support
Audi	1.000	1.000	1.000	14
BMW	1.000	1.000	1.000	20
Citroen	0.963	1.000	0.981	26
Fiat	0.955	0.955	0.955	22
Ford	1.000	1.000	1.000	18
Mercedes	0.962	0.944	0.953	54
Mitsubishi	0.800	0.727	0.762	11
Nissan	1.000	0.857	0.923	14
Opel	1.000	0.958	0.979	24
Peugeot	0.971	0.971	0.971	34
Renault	0.919	1.000	0.958	57
Scania	0.643	0.692	0.667	13
Toyota	1.000	1.000	1.000	18
Volkswagen	1.000	1.000	1.000	32
Volvo	1.000	0.909	0.952	22
accuracy			0.955	379
macro avg	0.948	0.934	0.940	379
weighted avg	0.957	0.955	0.955	379



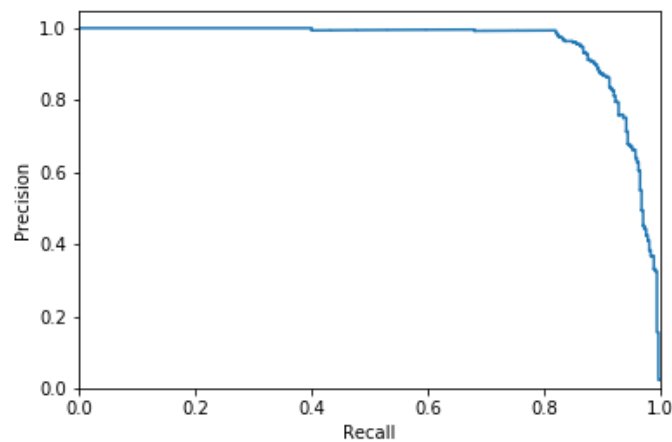
9.14 Attachment N: Prediction-Recall data for MC-CNN, as seen in Table 10  
 (95.52% accuracy)

	precision	recall	f1-score	support
Audi	1.000	1.000	1.000	14
BMW	1.000	1.000	1.000	20
Citroen	0.963	1.000	0.981	26
Fiat	0.955	0.955	0.955	22
Ford	0.947	1.000	0.973	18
Mercedes	0.857	1.000	0.923	54
Mitsubishi	1.000	1.000	1.000	11
Nissan	0.917	0.786	0.846	14
Opel	0.957	0.917	0.936	24
Peugeot	0.943	0.971	0.957	34
Renault	0.981	0.930	0.955	57
Scania	1.000	0.615	0.762	13
Toyota	1.000	1.000	1.000	18
Volkswagen	1.000	1.000	1.000	32
Volvo	1.000	0.955	0.977	22
accuracy			0.955	379
macro avg	0.968	0.942	0.951	379
weighted avg	0.958	0.955	0.954	379



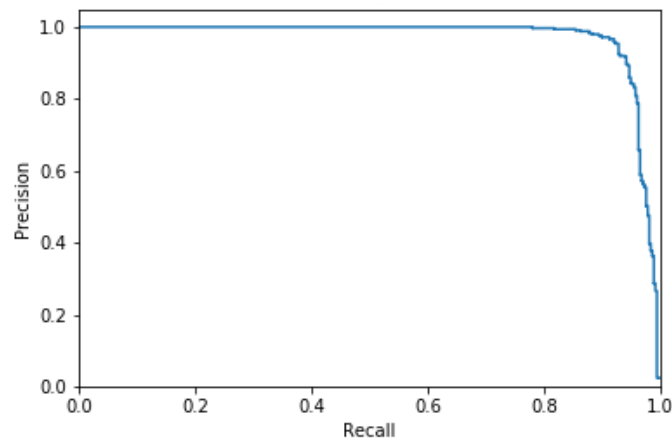
**9.15 Attachment O: Prediction-Recall results for MV-CNN-6, as seen in Table 11 (95.78% accuracy)**

	precision	recall	f1-score	support
Audi	1.000	1.000	1.000	14
BMW	1.000	1.000	1.000	20
Citroen	0.963	1.000	0.981	26
Fiat	1.000	0.955	0.977	22
Ford	0.944	0.944	0.944	18
Mercedes	0.947	1.000	0.973	54
Mitsubishi	0.818	0.818	0.818	11
Nissan	0.846	0.786	0.815	14
Opel	1.000	0.917	0.957	24
Peugeot	1.000	0.971	0.985	34
Renault	0.965	0.965	0.965	57
Scania	0.800	0.923	0.857	13
Toyota	0.895	0.944	0.919	18
Volkswagen	1.000	1.000	1.000	32
Volvo	1.000	0.909	0.952	22
accuracy			0.958	379
macro avg	0.945	0.942	0.943	379
weighted avg	0.959	0.958	0.958	379



**9.16 Attachment P: Prediction-Recall results for MV-CNN-12, as seen in Table 11 (96.04% accuracy)**

	precision	recall	f1-score	support
Audi	1.000	1.000	1.000	14
BMW	1.000	1.000	1.000	20
Citroen	1.000	1.000	1.000	26
Fiat	1.000	0.955	0.977	22
Ford	1.000	0.944	0.971	18
Mercedes	0.945	0.963	0.954	54
Mitsubishi	1.000	0.818	0.900	11
Nissan	1.000	0.857	0.923	14
Opel	1.000	0.958	0.979	24
Peugeot	0.971	1.000	0.986	34
Renault	0.889	0.982	0.933	57
Scania	0.818	0.692	0.750	13
Toyota	0.900	1.000	0.947	18
Volkswagen	1.000	1.000	1.000	32
Volvo	1.000	0.955	0.977	22
accuracy			0.960	379
macro avg	0.968	0.942	0.953	379
weighted avg	0.962	0.960	0.960	379

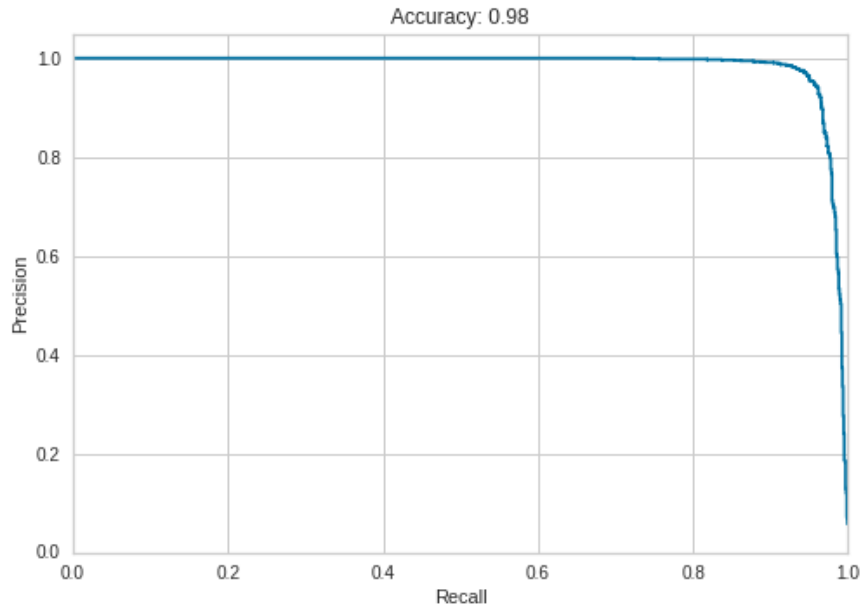




9.17 Attachment Q: Precision and Recall statistics for MV-CNN-8 on LargeBrisa, as seen in Table 13 (98.02% accuracy)

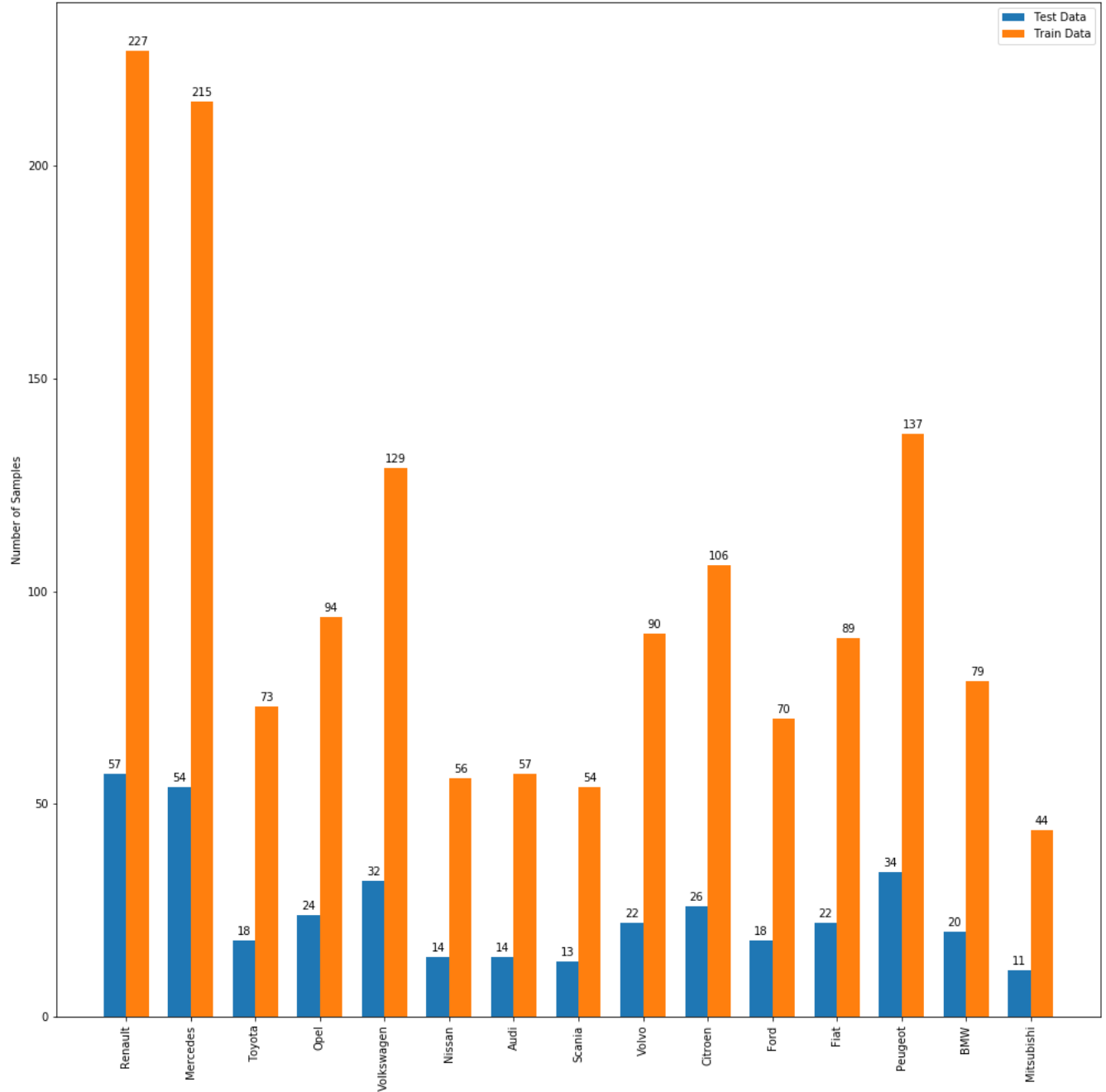
	precision	recall	f1-score	support
Alfa Romeo	1.000	1.000	1.000	5
Audi	1.000	1.000	1.000	48
BMW	1.000	1.000	1.000	70
Chevrolet	1.000	1.000	1.000	4
Citroen	0.988	0.988	0.988	81
DAF	1.000	1.000	1.000	8
Dacia	1.000	1.000	1.000	10
Fiat	0.967	0.967	0.967	61
Ford	1.000	0.970	0.985	67
Honda	1.000	1.000	1.000	14
Hyundai	1.000	0.833	0.909	12
Isuzu	1.000	1.000	1.000	2
Iveco	1.000	1.000	1.000	11
Jaguar	1.000	0.500	0.667	2
Jeep	1.000	0.500	0.667	2
Kia	0.944	1.000	0.971	17
Lancia	1.000	1.000	1.000	2
Land-Rover	1.000	1.000	1.000	5
Lexus	1.000	1.000	1.000	3
MAN	0.000	0.000	0.000	3
Mazda	1.000	1.000	1.000	9
Mercedes	0.992	0.992	0.992	129
Mini	1.000	1.000	1.000	10
Mitsubishi	0.909	1.000	0.952	20
Nissan	1.000	0.972	0.986	36
Opel	0.973	1.000	0.986	71
Peugeot	0.983	0.983	0.983	120
Porsche	1.000	0.500	0.667	2
Renault	0.971	0.982	0.977	171
Scania	0.889	0.800	0.842	10
Seat	0.968	0.968	0.968	31
Setra	0.400	1.000	0.571	2
Skoda	1.000	1.000	1.000	14
Smart	0.933	0.933	0.933	15
Suzuki	1.000	1.000	1.000	2
Tesla	1.000	1.000	1.000	2
Toyota	1.000	1.000	1.000	57
Volkswagen	0.968	0.989	0.979	93
Volvo	1.000	1.000	1.000	39
accuracy			0.980	1260
macro avg	0.946	0.920	0.923	1260
weighted avg	0.979	0.980	0.979	1260

9.18 Attachment Q (continued): Precision and Recall statistics for MV-CNN-8 on LargeBrisa, as seen in Table 13 (98.02% accuracy)

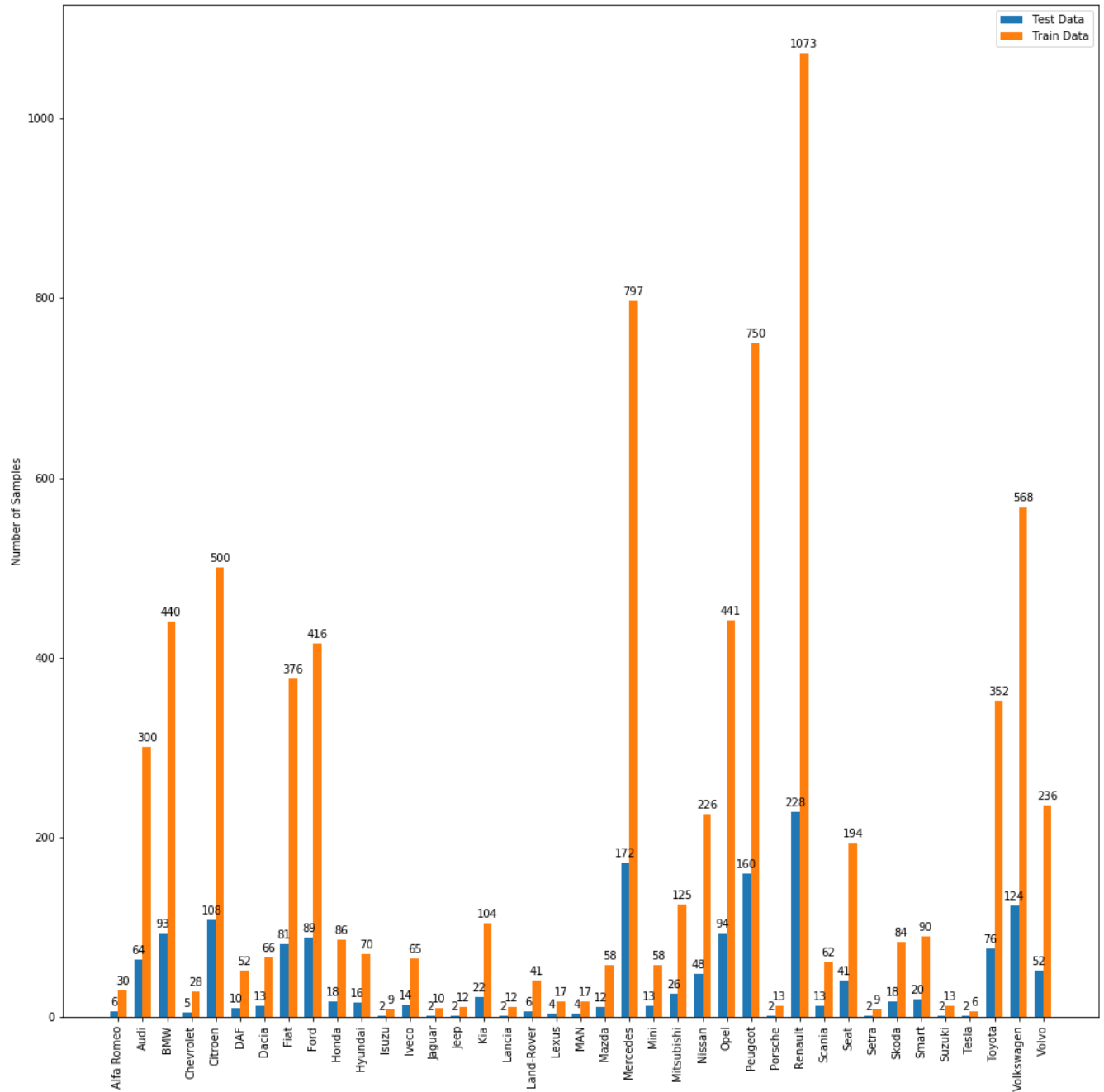




### 9.19 Attachment R: Class Distribution in ShortBrisa



### 9.20 Attachment S: Class Distribution in LargeBrisa



## References

- [1] O. Russakovsky *et al.* "ImageNet Large Scale Visual Recognition 17Challenge". Int. J. Comput. Vis., 2015. doi: 10.1007/s11263-015-0816-y.
- [2] Xie, E. Hovy, M.-T. Luong, and Q. V. Le. "Self-training with Noisy Student improves ImageNet classification". 2019.
- [3] A. Dubey, O. Gupta, P. Guo, R. Raskar, R. Farrell, and N. Naik. "Pairwise confusion for fine-grained visual classification". LectureNotes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2018, doi: 10.1007/978-3-030-01258-8\_5
- [4] B. Englert and S. Lam. "The Caltech-UCSD Birds-200-2011 Dataset". IFAC Proc. Vol., 2009, doi: 10.3182/20090902-3-US-2007.0059.
- [5] M. E. Nilsback and A. Zisserman. "Automated flower classification over a large number of classes". Proceedings -6th Indian Conference on Computer Vision, Graphics and Image Processing, ICVGIP 2008, 2008, doi: 10.1109/ICVGIP.2008.47.
- [6] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li. "Novel Dataset for Fine-Grained Image Categorization: Stanford Dogs". First Workshop on Fine-Grained Visual Categorization (FGVC), IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011.
- [7] A. R. Chowdhury, T. Y. Lin, S. Maji, and E. Learned-Miller. "One-to-many face recognition with bilinear CNNs". 2016 IEEE Winter Conference on Applications of Computer Vision, WACV 2016, 2016, doi: 10.1109/WACV.2016.7477593
- [8] A. Nazemi, Z. Azimifar, M. J. Shafiee, and A. Wong. "Real-Time Vehicle Make and Model Recognition Using Unsupervised Feature Learning". IEEE Trans. Intell. Transp. Syst., vol. PP, pp. 1–11, 2019, doi: 10.1109/tits.2019.2924830
- [9] L. Xie, Q. Tian, J. Wang, and B. Zhan. "IMAGE CLASSIFICATION WITH MAX-SIFT DESCRIPTORS". ol. 2, no. 2, pp. 54–65, doi: 10.5874/jfsr.2.2\_54
- [10] K. S. Oh and K. Jung. "GPU implementation of neural networks". Pattern Recognit., 2004, doi: 10.1016/j.patcog.2004.01.013.

- [11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “*ImageNet classification with deep convolutional neural networks*”. Advances in Neural Information Processing Systems, 2012.
- [12] L. Zheng, Y. Yang, and Q. Tian. “*SIFT Meets CNN: A Decade Survey of Instance Retrieval*”. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2018, doi: 10.1109/T-PAMI.2017.2709749.
- [13] Y. Wang, V. I. Morariu, and L. S. Davis. “*Learning a Discriminative Filter Bank Within a CNN for Fine-Grained Recognition*”. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2018, doi: 10.1109/CVPR.2018.00436.
- [14] Josip Josifovski. “*Object Recognition SIFT vs Convolutional Neural Networks.*”. [online]:[[https://tams.informatik.uni-hamburg.de/lectures/2015ws/seminar/ir/pdf/slides/JosipJosifovski-Object\\_Recognition\\_SIFT\\_vs\\_Convolutional\\_Neural\\_Networks.pdf](https://tams.informatik.uni-hamburg.de/lectures/2015ws/seminar/ir/pdf/slides/JosipJosifovski-Object_Recognition_SIFT_vs_Convolutional_Neural_Networks.pdf)]. University of Hamburg. 21-11-2015
- [15] K. Simonyan, A. Zisserman. “*Very Deep Convolutional Networks for Large-Scale Image Recognition*”. ICLR 2015
- [16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens. “*Rethinking the Inception Architecture for Computer Vision*”. Computer Vision and Pattern Recognition 2016. 2016, doi: 10.1109/CVPR.2016.308
- [17] K. He, X. Zhang, S. Ren and J. Sun. “*Deep Residual Learning for Image Recognition*”. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.
- [18] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger. “*Densely Connected Convolutional Networks*”. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 2261-2269, doi: 10.1109/CVPR.2017.243.
- [19] GR. Girshick, J. Donahue, T. Darrell and J. Malik. “*Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation*”. 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 580-587, doi: 10.1109/CVPR.2014.81.
- [20] Shorten, C., Khoshgoftaar, T.M. “*A survey on Image Data Augmentation for Deep Learning*”. J Big Data 6, 60 (2019). <https://doi.org/10.1186/s40537-019-0197-0>

- [21] A Schumann, L.W. Sommer, K. Valev, J. Beyerer. “*A systematic evaluation of recent deep learning architectures for fine-grained vehicle classification*”. Pattern Recognition and Tracking XXIX. 2018, doi: 10.1117/12.2305062
- [22] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, Q. He. “*A Comprehensive Survey on Transfer Learning*”. ArXiv, abs/1911.02685. 2019
- [23] K. Weiss, T. Khoshgoftaar, D. Wang. “*A survey of transfer Learning*”. J Big Data (2016) 3:9, doi: 10.1186/s40537-016-0043-6
- [24] J. Yosinski, J. Clune, Y. Bengio., H. Lipson. “*How transferable are features in deep neuralnetworks?*”. Advances in Neural Information Processing Systems 27 (NIPS 2014)
- [25] Y. Cui, Y. Somng, C. Sun, A. Howard, S. Belongie. “*Large Scale Fine-Grained Categorization and Domain-Specific TransferLearning*”. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). DOI: 10.1109/CVPR.2018.00432
- [26] Z. Ma, D. Chang, J. Xie, Y. Ding, S. Wen, X. Li, Z. Si, J. Guo. “*Fine-Grained Vehicle Classification With ChannelMax Pooling Modified CNNs*”. IEEE Transactions on Vehicular Technology, vol. 68, no. 4, pp. 3224-3233, April 2019, doi: 10.1109/TVT.2019.289997
- [27] Z. Ma, D. Chang, J. Xie, Y. Ding, S. Wen, X. Li, Z. Si, J. Guo. “*Fine-Grained Vehicle Classification With ChannelMax Pooling Modified CNNs*”. IEEE Transactions on Vehicular Technology, vol. 68, no. 4, pp. 3224-3233, April 2019, doi: 10.1109/TVT.2019.289997
- [28] A. Nazemi, M. Shafiee, Z. Azimifar, A. Wong. “*Unsupervised Feature Learning Toward a Real-timeVehicle Make and Model Recognition*”. IEEE Trans. Intell. Transp. Syst., vol. PP, pp. 1–11, 2019, doi: 10.1109/tits.2019.2924830
- [29] Q. Zhang, L. Zhuo, X. Hu and J. Zhang. “*Fine-grained vehicle recognition using hierarchical fine-tuning strategy for Urban Surveillance Videos*”. 2016 International Conference on Progress in Informatics and Computing (PIC), Shanghai, 2016, pp. 233-236, doi: 10.1109/PIC.2016.7949501.
- [30] L. Yang, P. Luo, C. C. Loy and X. Tang. “*A large-scale car dataset for fine-grained categorization and verification*”. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 3973-3981, doi: 10.1109/CVPR.2015.7299023.

- [31] T. Hu, H. Qi, Q. Huang, Y. Lu. “*See Better Before Looking Closer: Weakly Supervised Data Augmentation Network for Fine-Grained Visual Classification*”. 2019
- [32] Y. Peng, X. He, J. Zhao. “*Object-Part Attention Model for Fine-Grained Image Classification*”. IEEE Transactions on Image Processing, vol. 27, no. 3, pp. 1487-1500, March 2018, doi: 10.1109/TIP.2017.2774041.
- [33] K. Han, J. Guo, C. Zhang, M. Zhu. “*Attribute-Aware Attention Model for Fine-grained Representation Learning*”. 2018 ACM Multimedia Conference doi: 10.1145/3240508.3240550
- [34] A. Abbas, U. Ullah, C. Zhang, M. Mohd. “*Recognition of vehicle make and model in low light conditions*”. Vol.9, No.2, April 2020, pp. 550 557ISSN: 2302-9285, DOI: 10.11591/eei.v9i2.1865
- [35] T. Kyono and Fiona J. Gilbert, M. V. D. Schaar. “*Multi-view Multi-task Learning for Improving Autonomous Mammogram Diagnosis*”. MLHC, 2019
- [36] L. Sun, J. Wang, Z. Hu, Y. Xi, Z. Cui. “*Multi-view convolutional neural networks for mammographic image classification*”. IEEE Access, vol. 7, pp. 126273-126282, 2019, doi: 10.1109/ACCESS.2019.2939167.
- [37] Z.. Chen, C. Ying, C. Lin, S. Liu, W. Li. “*Multi-view Vehicle Type Recognition with Feedback-enhancement Multi-branch CNNs*”. IEEE Transactions on Circuits and Systems for Video Technology PP(99):1-1, doi: 10.1109/TCSVT.2017.2737460
- [38] J. Redmon, S. Divvala, R. Girshick, A. Farhadi. “*You Only Look Once: Unified, Real-Time Object Detection*”. 2016. [online]: <https://pjreddie.com/darknet/yolov1/>
- [39] J. Redmon, A. Farhadi. “*YOLO9000: Better, Faster, Stronger*”. 2016. [online]: <https://pjreddie.com/darknet/yolov2/>
- [40] J. Redmon, A. Farhadi. “*YOLOv3: An Incremental Improvement*”. 2018. [online]: <https://pjreddie.com/darknet/yolo/>
- [41] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, D. Ramanan. “*Object Detection with Discriminatively Trained Part-Based Models*”. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 9, pp. 1627-1645, Sept. 2010, doi: 10.1109/TPAMI.2009.167.

- [42] T. Lin, P. Goyal, R. Girshick, K. He, P. Dollár. “*Focal Loss for Dense Object Detection*”. 2017 IEEE International Conference on Computer Vision (ICCV), Venice, 2017, pp. 2999-3007, doi: 10.1109/ICCV.2017.324.
- [43] T.-Y. Lin, P. Doll ar, R. Girshick, K. He, B. Hariharan, S. Belongie. “*Feature pyramid networks for object detection*”. CVPR, 2017.
- [44] R. Girshick, J. Donahue, T. Darrell, J. Malik. “*Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation*”. 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 580-587, doi: 10.1109/CVPR.2014.81.
- [45] R. Girshick, J. Donahue, T. Darrell, J. Malik. “*Fast R-CNN*”. 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 1440-1448, doi: 10.1109/ICCV.2015.169.
- [46] S. Ren, K. He, R. Girshick, J. Sun. “*Faster R-CNN: Towards Real-Time ObjectDetection with Region Proposal Networks*”. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 1 June 2017, doi: 10.1109/TPAMI.2016.2577031.
- [47] K. He, G. Gkioxari, P. Dollár and R. Girshick. “*Mask R-CNN*”. 2017 IEEE International Conference on Computer Vision (ICCV), Venice, 2017, pp. 2980-2988, doi: 10.1109/ICCV.2017.322.
- [48] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, A. Torralba. “.”. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 2921-2929, doi: 10.1109/CVPR.2016.319.
- [49] L. Yang, P. Luo, C. C. Loy, X. Tang. “*A large-scale car dataset for fine-grained categorization and verification*”. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 3973-3981, doi: 10.1109/CVPR.2015.7299023.
- [50] *Pytorch available models source-code..* [online] <https://pytorch.org/docs/stable/torchvision/models.html>
- [51] H. Touvron, A. Vedaldi, M. Douze, H. Jégou. “*Fixing the train-test resolution discrepancy: FixEfficientNet*”. 2020.
- [52] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby. “*An image is worth 16x16 words: transformers for image recognition at scale*”. 2020.

[53] [online] [https://sci-hub.se/10.1007/978-0-387-30164-8\\_652](https://sci-hub.se/10.1007/978-0-387-30164-8_652)