UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

Object detection in automotive vehicle domain based on real and synthetic data

Author:
Roman ILECHKO

Supervisor:
Viktor SDOBNIKOV

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2022

Declaration of Authorship

I, Roman ILECHKO, declare that this thesis titled, "Object detection in automotive vehicle domain based on real and synthetic data" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Master of Science

Object detection in automotive vehicle domain based on real and synthetic data

by Roman ILECHKO

Abstract

In recent years, we have seen an incredible increase in deep learning. With increasing interest, we also have an increasing number of bold and even revolutionary studies that drive progress and boost models performance. Despite the fact that the number of large and high-quality data-sets is growing rapidly, we could often observe that models need even more data for many domains and tasks. Usually, additional data is needed not only for giant models. Even though the domains like autonomous vehicles, which mainly focus on lightweight models, require extra data. We should state that sometimes data labeling is not a panacea. Especially for autonomous vehicles, as the data provided must have a great variety and low error risk. The additional synthetic could be an excellent booster for existing approaches or even a must-have part of training data. For example, simulators give the ability to manage the scene's complexity by controlling the number of objects, their size, and their interaction with the environment, which could be very helpful for such tasks as object detection. Nowadays, the researcher should intuitively balance the ratio of natural and generated data simultaneously, considering the possibility of gaps between the two domains. Despite the fact that the mentioned task is not evident, constraints like model size and count of classes could bring additional unclarity. In this paper, we precisely analyze the impact of synthetic data on the training process, cover possible training strategies, and provide guidance on defining the amount of artificial data with existing constraints.

Acknowledgements

I want to thank my supervisor, Viktor Sdobnikov, for mentoring and supporting me during my diploma project.
Additionally, I want to express my gratitude to Oleksii Molchanovskyi for diploma meetings and regular support. Separate thanks to my good friend Roman Vei.

Contents

List of Figures

List of Tables

List of Abbreviations

CARLA	**C**a**r** **L**earn to **A**ct
RADAR	**Ra**dio **D**etection **A**nd **R**anging
LIDAR	**Li**ght **D**etection **A**nd **R**anging
CNN	**C**onvolution **N**eural **N**etwork
ML	**M**achine **L**earning
AI	**A**rtificial **I**ntelligence
IMU	**I**nertial **M**easurement **U**nits
IOU	**I**ntersection **O**ver **U**nion
AR	**A**verage **R**ecall
AP	**A**verage **P**recision
mAP	**m**ean **A**verage **P**recision
SOTA	**S**tate **o**f **t**he **A**rt

Dedicated to those who defend our freedom…

Chapter 1

Introduction

Over the last decade, artificial intelligence was deeply integrated into the route life. The field of machine learning experienced exaltation and oblivion due to many reasons. The domain of computer vision expected the same participation. One of the most significant challenges during all periods which researchers met was the question of feature extraction. Starting from the '50s of the last century, researchers worldwide have been working on an algorithm to present the images to the computer system and teach them to understand as humans do. Two pioneers of AI started the ambitious program - the Summer Vision project Papert, 1966. The main goal of the two-month project was to develop a computer system that could recognize the objects on the images. Such a simple task for the human recognition system was an insurmountable task. Despite the fact that the Summer Vision project Papert, 1966 did not provide the revolution approach, it caused the discussion. A few years later, the computer vision solution will be proposed based on neural science research named neocognitron Fukushima and Miyake, 1982. Inspired by this idea, the CNN for digit recognition was presented LeCun et al., 1999. The first layer reveals basic things such as vertical and horizontal edges. As you move inside the neural network, the layers reveal more complex features, including angles and shapes. The latest layers of CNN reveal specific things such as faces, doors, and cars. The source layer of CNN provides a table of numerical values that represent the probability of detecting a particular object in the image. Decades after, when we dramatically improved the approach to visual understanding, we still can state that we are far away from ideal. In recent years the deep learning domain rapidly grew. While the quality of brought solutions increased, simultaneously, the model's complexity rose dramatically. Despite the fact that available public datasets are an acceptable solution for many machine learning and research projects, there is always a temptation to increase the training data with the aim to attain a more pleasing result. Even more, domain-specific cases often require additional data, which is usually not publicly available. Data labeling is an obvious solution in the faced situation; it also has weaknesses. Manually collected and labeled data is not the panacea due to time-consuming and high error risk. The synthetical data could be a suitable solution based on the facts above but also have drawbacks. Nowadays, the researcher should intuitively balance the ratio of natural and generated data simultaneously, considering the possibility of gaps between the two domains. Despite the fact that the mentioned task is not evident, constraints like model size and count of classes could bring additional unclarity. In this work, I precisely analyze the impact of synthetic data on the training process, cover the possible training strategies, and provide guidance on defining the amount of artificial data with existing constraints.

In Simonyan and Zisserman, 2015, the researchers proposed to focus on the depth of the convolution neural network with small filters, as this strategy could improve the accuracy of the image classification taskDeng et al., 2009 significantly.

In comparison, GoogleNetSzegedy et al., 2015 introduced the new Inception module, which allows rich state-of-the-art performance by concentrating on the design of the network and keeping in mind not only depth but either width of the model. Paper He et al., 2015 demonstrated one of the solutions to overcome the problem of vanishing gradient. While the mentioned convolution neural network provides the ability to extract features more efficiently, the pay for improvements was high. With increasing the performance, the training time increased dramatically. Including the fact that robust but straight-forward methods such as batch normalization Ioffe and Szegedy, 2015 nowadays defacto is a gold standard for most of the current works, as it provides the ability to reduce the amount of training data and improve the model convergence by normalizing the layer inputs, still the training process of such networks requires a considerable amount of diverse data. Nevertheless, mentioned models have a strong ability for generalizing, and they are perfect for feature extracting. Moreover, using such architecture as the backbone of Redmon et al., 2015; Liu et al., 2016 for object detection will provide the coveted result, in conjunction with the GPU algorithm will work in real-time. However, the execution time at embedded and mobile devices could be insupportable. Due to the noted constraints, the embedded and mobile devices require a particular approach. One of the possible solutions, which is now the subject of active research, is neural network quantization Hubara et al., 2016; Choi et al., 2019; Jin, Yang, and Liao, 2020. Such a tricky approach provides the ability to increase the model performance by reducing not only the model size but also the mode speed. Apparent that fact, the system with performance limitations needs a slightly different strategy in feature extracting, especially in backbone architecture designing. The architectures like MobileNet Howard et al., 2017; Sandler et al., 2019; Howard et al., 2019 were mainly designed to satisfy the described demand. With the increasing interest to the domain of automotive driving, the stated milestones become crucial. Despite legal and ethical issues, the automotive driving system must be safe, aching a massive amount of data. While data labeling is time-consuming and costly, one of the possible solutions was proposed in Munoz, Bagnell, and Hebert, 2010; Farabet et al., 2013. Another approach is to use generated data. As a matter of fact, different data augmentation methods and strategies Buslaev et al., 2020; Xie et al., 2020; Zoph et al., 2020 are commonly used in computer vision. The synthetic data generated from simulators like CARLA Dosovitskiy et al., 2017 is more inherent in the research automotive driving area. This source of training data is very attractive because of its versatility, in Figure 1.1, we can see examples of the outputs from different sensors. Besides, they are also essentially free. On the other hand, such a strategy also has a drawback - domain gap. Our goal is to provide hands-on recommendations for convolution neural network training, focusing on object detection tasks.

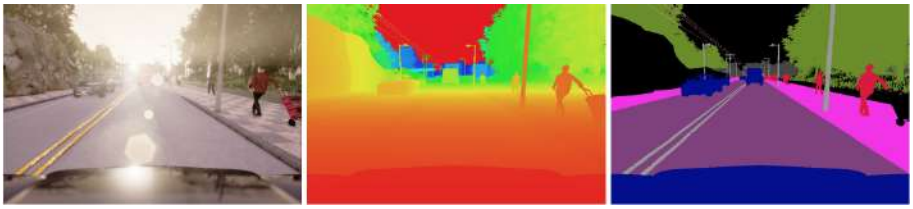

FIGURE 1.1: Example of different sensors output Dosovitskiy et al., 2017 simulator. From left to right: monocular camera, depth camera, semantic segmentation

One of the most crucial parts of the manuscript is the analysis of the existing

works. In Chapter 2 reveals the state-of-the-art techniques and briefly introduces the existing domain gap, helps expose the most crucial parts of research and outlines the intended contribution. The Chapter 3 and Chapter 4 mainly focus on covering the existed public available datasets and method of model evaluation, here, we briefly described metrics we will use to compare the results. In Chapter 5 was described the trained experiments. The last section contains the conclusions which describe the current achievements their potential impact and sketch the future steps and improvement.

Chapter 2

Related work

Synthetic data is widely used for numerous tasks in Computer Vision. It is a common strategy for object detection Hinterstoisser et al., 2019, instance segmentationFeng et al., 2019, face verificationKortylewski et al., 2018, and even 3D object detectionBrekke, Vatsendvik, and Lindseth, 2019. One of the significant benefits of synthetic data is the ability to generate specific scenes. Such an advantage is essential, for example, for the robotics domainTremblay et al., 2018. Simulators like CARLA Dosovitskiy et al., 2017 provide the ability to generate different pictures with various scenarios. By adjusting the number of road users, researchers could control the complexity of the scene. In addition to a complexity control, researchers also already have free labeled data. Despite the fact that suitable data-sets Yu et al., 2020; Fritsch, Kuehnl, and Geiger, 2013; Caesar et al., 2019, which contain examples of different daytime and weather conditions, are available for research purposes, they are often insufficient. On another side, there is always a temptation to train the model on more comprehensive data. Evident of that fact, data from artificial and real sources vary a lot. Such a combination of natural and synthetic data could provide unpredictable results. The final ratio between the sources of data and the learning strategy rests squarely on the shoulders of the researcher. Comparing the benchmark result for object detection task on realYu et al., 2020; Fritsch, Kuehnl, and Geiger, 2013; Caesar et al., 2019 and syntheticGaidon et al., 2016; Cabon, Murray, and Humenberger, 2020; Wrenninge and Unger, 2018a; Richter, Hayder, and Koltun, 2017a data-sets could provide some clearness about the impact of artificial data but do not provide answers entirely. In contrast, Hartwig and Ropinski, 2019 considered this problem from another angle and proposed reducing the difference between the domain of data instead of finding the perfect balance. In Hartwig and Ropinski, 2019 was investigated three approaches in generating photo-realistic reflecting objects:

- Reflection Approximation
- Domain Randomization
- Physically-Based Rendering with Domain Randomization

Essential to notice the fact that mixing data sources is the hottest topic not only for object detection with RGB image input but also for thermal imagesBongini et al., 2021. The impact of the data generated from a military simulator was provided at the researchÖhman, 2019. The difference between research areas does not allow projecting the results to the domain of automotive vehicles. In Gao, Tang, and Wang, 2021 was researched YOLOv4Bochkovskiy, Wang, and Liao, 2020, CenterNetDuan et al., 2019, and Faster-RCNN Ren et al., 2015 models trained on synthetic data. Since evaluation was made on the generated data, the behavior on the real-world example is not covered. In contrast, in Nowruzi et al., 2019 demonstrated the performance of

a few object detectors trained with different data ratios in detail. Additionally, was covered the variants with transfer learning and model fine-tuning. We should remember that the automotive vehicle area has few constraints. First of all, the amount of diverse labeled data is not enough for real-life scenarios. Additionally, the model performance should be at a sky-high level. Moreover, object detection algorithms must run in real-time at the embedded devices. Due to that point, we could not wholly project the result of the previous work. From another point of view, the mentioned papers do not provide the general advice or guidelines which could be at least partly applicable for the specific field.

From the mentioned fact, we can understand that previous works cover the subject only partly and do not disclose the impact of synthetic data for object detection in the automotive vehicle domain. As my work consists of the analysis of the performance of different models, obviously that we need firstly to cover the necessary algorithms. This section mainly focuses on the algorithms that could be a significant part of experiments and reveal their strong and weak sides. The goal is to construct a list of suitable algorithms to research the impact of synthetic data for object detecting tasks in the automotive vehicle domain.

2.1 VGG

One of the most impactful works in computer vision was Simonyan and Zisserman, 2015. In this publication, the researcher proposed going deeper with a convolutional neural network and investigating the impact of increased model depth on the accuracy of the network. In contrast to AlexNet Krizhevsky, Sutskever, and Hinton, 2012, where was actively used large filters like 7×7, scientists introduced the using an architecture with a small 3 convolution filter. While the convolution filter of 7×7 size has a large receptive field, it also has a considerable number of learnable parameters. Researchers proved that it could be effectively replaced by 3 3×3 filters, which reduce the number of parameters by 81%. Additionally, the 1×1 convolution layers were actively used with the aim of improving the decision function by increasing non-linearity. As an activation function was used ReLU. Also, after some convolutional block performed 2×2 max-pooling layer, with a stride 2, which helps to cover high-level information and decrease the dimensionality. The whole architecture of the model is displayed in Figure 2.1.

As this model took participation in the ImageNet challenge, where SOTA results were gained, it was trained for the classification task. The convolutional part of the pretrained on this challenge model was commonly utilized as a feature extractor in many computer vision tasks, such as object detection, image segmentation, and even style transfer Gatys, Ecker, and Bethge, 2015.

While this model performed an excellent result, the main disadvantage is the giant network size. The total number of learnable parameters in this model is 138 million, which causes a long inference time and is unsuitable for the automotive vehicle domain. Weighing all the pros and cons, we decided not to use this model as a part of the detector in our experiment.

2.2 MobilenetV1

While AI from day to day becomes a routine part of our life, the issue of deployment of models for phones and portable gadgets is acute. While the are a lot of web services that allow using GPU for fast inference even of large neural networks, the

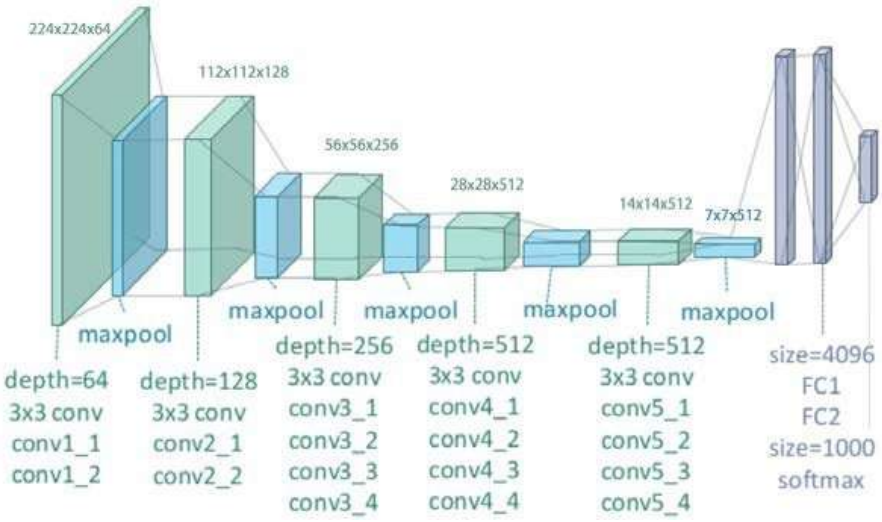

FIGURE 2.1: VGG architecture

portable devices or systems like automotive vehicles often cannot use such a benefit. At the same time, the general trend was focused on achieving a higher result in the challenges like ImageNet. The acute was a question of the accurate and fast model which could perform in real-time on the inference stage.To solve such problem was introduced a new type of efficient feature extractor - MobileNet.

The main novelty which optimize thearchitecture of the mentioned model is the following:

- Depthwise convolution using
- Introducing pointwise convolution
- Model width multiplier
- Model resolution multiplier

2.2.1 Depthwise Separable Convolution

The architecture of MobileNet is mostly conducted from a block called - Depthwise Separable Convolution, which consists of:

- Depthwise convolution
- Pointwise convolution

The depth-wise convolution works the following, for each input channel, this block applies the single filter. After the mentioned operation is succeeded, the 1×1 convolutional filter, also known as pointwise convolution, will be applied. If we go deeper, the standard convolutional layer makes this operation into one step, while the newly depthwise separable convolution layer performs it in two stages:

- Depthwise convolution part is responsible for filtering the input
- Pointwise convolution layer is responsible for combining the result of depth-wise convolution layer.

As a result, we gain a lower number of parameters. Let us compute the number of parameters in the default convolution block. Let D_K be the size of the kernel, D_F - the size of the feature map, and M, N stands for a number of input and output channels, respectively. Then the total number of parameters will be computed using the next formula:

$$D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F \tag{2.1}$$

As a depth wise convolution works on the channel wise level, the total number of parameters will not depend on the number of output channel N, as in the previous case, and will be the following:

$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F \tag{2.2}$$

The overall complexity of sepearable depthwise convolution will be computed as eqution and number of parameters in the pointwise layer, which computed as:

$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F \tag{2.3}$$

As a result the reduction will be:

$$\frac{1}{N} + \frac{1}{D_K^2}$$

As a result the reduction will be:

$$\frac{1}{N} + \frac{1}{D_K^2}$$

2.2.2 Width multiplier

The MobiliNet model is comparatively tiny, but sometimes there is a need to make the model even smaller. For such purposes, a new hyper-parameter α was introduced, which is responsible for thining the models layer. §the number of input channel M and output channel N will be multiplied by α.

2.2.3 Resolution multiplier

The last but not least parameter that can reduce the computational cost is the resolution multiplier, which reduces the resolution of the input image and feature map by value ρ.

Such a feature of MobileNet made it extremely useful for domains that require SOTA performance and low latency. That's why the mentioned model will participate in our experimental process and help refine the effect of generated data.

2.3 MobileNetV2

The first generation of MobileNet architecture showed impressive results in terms of accuracy and computational complexity, but it is always a temptation to improve the work. That is how MobileNetV2Sandler et al., 2019 was introduced. The building block in MobileNetV2Sandler et al., 2019 as in its predecessor is a depthwise separable convolution but slightly modified. The main innovations in this paper were the following: Inverted residual blocks Linear bottleneck Using ReLU6 instead of ReLU

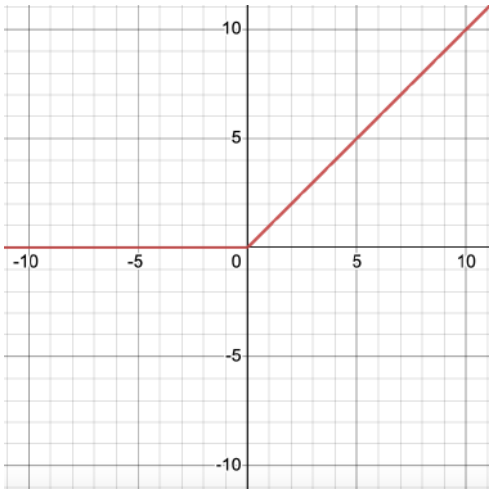

FIGURE 2.2: ReLU activation function.

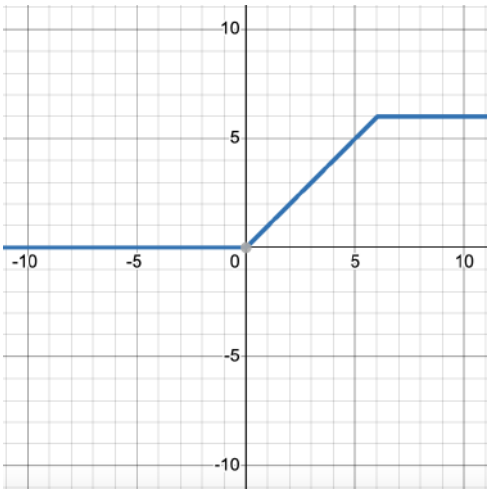

FIGURE 2.3: ReLU6 activation function.

2.3.1 Inverted residual blocks

The initial idea of residual blocks was mainly focused on preventing the gradient from vanishing. At the input of the block, the information was compressed by a convolution filter with size 1×1. The 3×3 filter goes further, and as the information increased, we compressed it again using 1×1. After all these operations, the output signal is added to input one. In the MobileNetV2, the sequence is inverted, as instead of the default convolution block, the depthwise convolution is used. The described idea of using skip connection was introduced as an inverted residual and had a lower number of parameters.

2.3.2 Linear bottleneck

To capture the data's non-linear behavior, the researcher uses the deep neural network with a non-linear activation function. Usually, the role of such a function accrues the ReLU activation function, which cuts off the negative values to zero. Apparent the fact that we have the information loss with such an operation, commonly, to level the impact of this operation researcher uses a larger number of channels. The inverted residual operation does the opposite. To tackle the impact of ReLU and the inverted residual block researcher proposed to use the linear activation at the end of inserted residual.

2.3.3 ReLU6

Additionally to another modification, instead of using the default ReLU(Fig. 2.2) function as an activation after the convolution layer researcher recommend using ReLU6(Fig. 2.3), which cuts off the values greater than 6.

As in previous cases with the lightweight model, the SquuezeNet is suitable for the automotive vehicle domain due to its fast and accurate inference. Our experiments will use it as a feature extractor in the object detection algorithm.

2.4 SqueezeNet

The willingness to create an efficient and accurate model is very desirable among many researchers. In the Iandola et al., 2016 researcher has the motivation to create the small and robust network as it has the next benefits:

- The process of distributed training is more straightforward.
- Easier process of exporting the new model from the cloud to an automotive car.
- Suitable for hardware with a small amount of memory

The main building block of SqueezeNet is called Fire(Fig. 2.4). It consists of two-part:

- squeeze
- expand

2.4.1 Fire module

The squeeze part of the fire module is made from a 1×1 convolutional layer. The hyper parameter for this building block of the fire module is a number of 1×1 layer. After processing the input squeeze part feeding the data to the expand part. This module includes not only 1×1 layers but also 3×3 layers. The hyper-parameters for this part of the fire module are responsible for the number of 1×1 and 3×3 layers. The sum of layers in expanding section must be bigger than in the squeeze part.

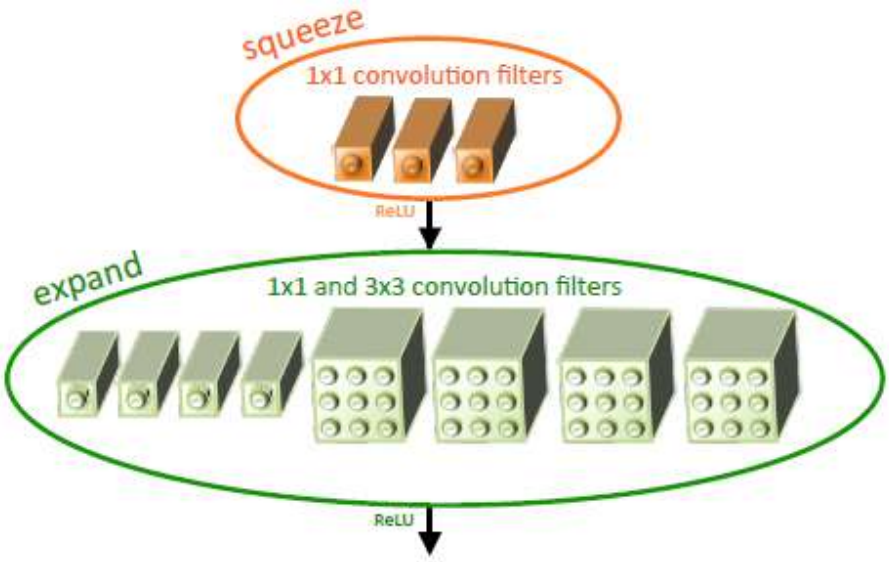

FIGURE 2.4: Fire module

2.4.2 SqueezeNet modification

The idea of skip connection is not new, and many researchers were impressed by this approach. The SqueezeNet also decided to implement this idea as a modification of the SqueezeNet model. Additionally, two models were proposed SqueezeNet with simple bypass and SqueezeNet with complex bypass(Fig. 2.5).

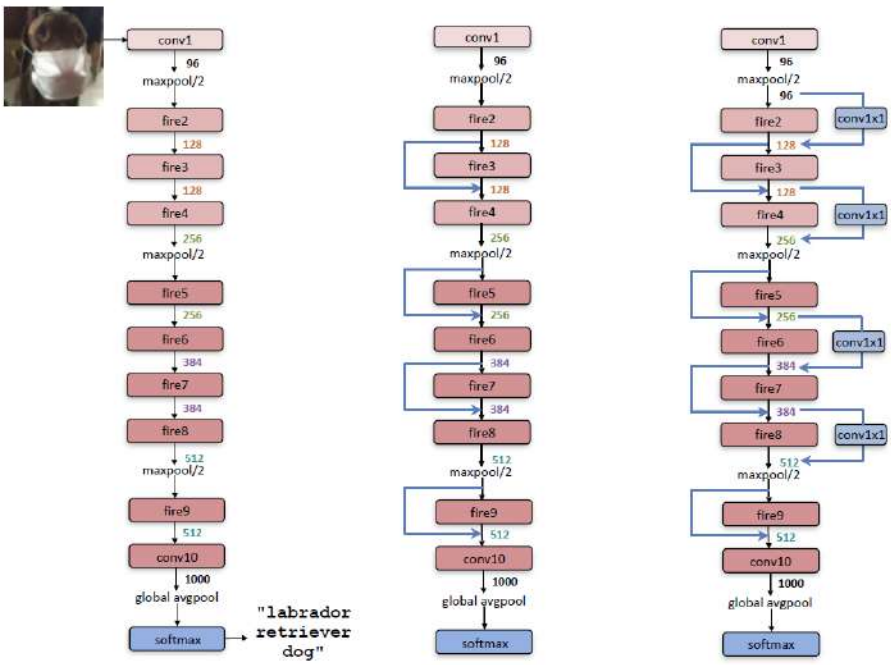

FIGURE 2.5: SqueezeNet mmodification from left to right: default SqueezeNet, SqueezeNet with simple bypass, SqueezeNet with complex bypass

2.5 SSD

The reviewed feature extraction method could be a backbone part of object detection. As was mentioned earlier, one of the model's main characteristics is the latency for the automotive vehicle. Obviously, an object detector as part of such a system must be lightweight and real-time. For this task, suitable are object detection methods that predict the objects and their location from a single run.

The SSD model was designed for real-time object detection and consisted of two-part:

- SSD feature extractor
- SSD head

We are interested in the SSD head part.

2.5.1 Grid cells

To find objects in the image, we want to cover the whole picture obviously that the sliding window approach will take too much time. For such a purpose, the researcher proposed the idea of splitting the image into parts. Each of the cells has

the default bounding boxes with different orientations and aspect ratios. By splitting an image into many parts and generating the multiple default boxes for each cell, we could cover the image with the predefined boxes. After the passing image through the feature extractor, features from different levels will be collected and passed through the blocks where the bounding boxes predict. The reason why we collect features from different backbone levels is very straightforward - as we want to capture features for large and for small objects as well. After that, the predefined default boxes will be corrected by corresponding correcting values. There is a linked vector with per class confidence for each of the boxes. After that, corrected bounding boxes from each level will be collected and passed to the Non-Maximum Suppression algorithm.

2.6 YOLO

As in the case of the SSD model, the YOLO network is also on stage object detector, which could perform in real-time. We could also divide the YOLO model into a few parts, such as:

- feature extractor
- detection head

As a feature extraction model often researcher takes a pretrained, for example, on ImageNet, CNN model, which could effectively extract different level features from the image. After the image is passed through the feature extractor module, it goes to the learnable module, which is responsible for object detection. This module consists of a few rapidly convolution layers and two fully connected layers. At the end output of the fully connected layer will be reshaped to tensor $7 \times 7 \times N$. In contrast to the SSD network, the YOLO model takes features only from one feature level, which causes some limitations in detecting small objects. Each of the 49 vectors from the tensor describes the respective area at the input image. The output dimension of the vector could be calculated as $N = 5 * D + C$, where d is the number of default boxes per cell and C is the number of classes. The end bounding box confidence is computed as $P(obj_i|objinbox)$. After this step, all bounding boxes will be filtered with non Maximum Suppression algorithm.

2.7 Algorithms summary

To conclude the algorithm research work, in sets of experiments to identify the impact of generated data for object detection tasks in the automotive vehicle domain, as a feature extraction will be used the models:

- MobileNetV1
- MobileNetV2
- SqueezeNet

The SSD algorithm was chosen to detect objects in the outdoor environment.

Chapter 3

Datasets

The dataset is one of the main parts of each research. The quality of the dataset and its size influence the algorithm's performance. The automotive vehicle systems could process different types of data, solving various tasks. The complex system could work with signals from RADAR, LiDAR, and monocular images, solving a long list of tasks. This work focused on the object detection part of the automotive vehicle system. As discussed earlier, the large amount of high-quality data commonly could be a bottleneck not only for many studies but for the industry at all. This chapter describes the available datasets, their limitation, and advantages for both actual and synthetic data.

The domain gap problem is a common issue we could meet in various deep learning tasks. Such difficulties appear when data significantly differ and are collected from different domains. For example, mentioned problem manifests itself in the automotive vehicle field, when the model is trained mainly on the data from one region and testing in another. Additionally, we could face troubles with the domain gap when a dataset is created from actual and synthetic data. To mimic the natural data, we could improve it to make it as close to the real as possible. At the same time, the mentioned improvements could direct different properties, such as visual and scenario similarity. We divided the type of synthetic data by the following criteria to determine the potential gains.

- A zero-level synthetic dataset contains generated data without enhancements to reduce the domain gap.
- A first-level dataset contains photorealistic data, which, for example, considers the impact of lighting and weather conditions on the resulting image.
- Second-level additionally focuses on a complex and realistic scenario generation from surroundings.
- Third-level focuses on domain randomization to create the data with randomized domain properties.

This work focused on the first-level synthetic data with photorealistic images generated with a base scenario without imitating the complex scenes.

3.1 Berkeley Deep Drive

The sizeable high-quality data is one of the main aspects of significant scientific research which push the domain forward in progress. It provides the ability for the experimenter to build a complex model. BDD100K Yu et al., 2020 is basically the

largest latest dataset of videos for automotive driving. This dataset contains 100K driving videos and ten tasks. In Figure 3.1, we can see examples of the images from the dataset. Collected data are accurately labeled on people, objects, road markings, sidewalks, and road signs for automotive driving. Also, the dataset retains a lot of variety of data in different fields such as climate, environment, and geography. Because of these, we exclude the model from falling into unknown conditions. The automotive vehicle system is complex and could contain a vast amount of computer vision models, each of them focuses on a particular task, like:

- Object detection
- Line Segmentation
- Object tracking
- Instance Segmentation

To teach the system to perform well on the different mentioned task, scientist requires a significant amount of various labeled data. It is a common situation when researchers meet the limitations of the dataset. The great advantage of this dataset - it contains labels for all mentioned tasks, which reduces the models development time a lot. The process of collection of real-world data was made in the USA. For such purposes was chosen the for cities:

- New York
- Berkeley
- San Jose
- San Francisco

The dataset also has a wide range of classes.

Car	Sign	Light	Person	Truck	Bus	Bike	Rider	Motor	Train
1021857	343777	265906	129262	42963	16505	10229	6461	4296	179

TABLE 3.1: The class distribution

The table with the classes is presented in Table 3.1.

To provide the ability to train the robust model and cover the highest amount of possible situations, the dataset was collected during different weather conditions during different times of the day. The dataset contains the following weather condition:

- clear weather
- overcast
- snowy
- rainy

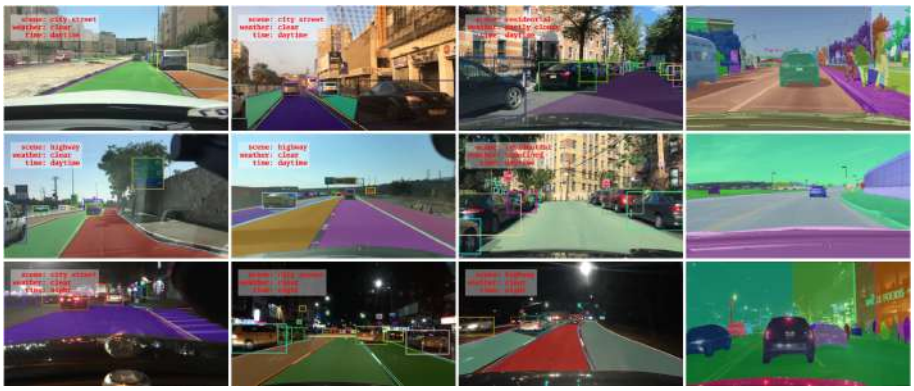

FIGURE 3.1: Images from BDD Yu et al., 2020 dataset

- cloudy
- foggy

No less critical to notice that an automotive vehicle is a complex system that takes the various information from the surrounding environment. Also, the data was captured in different locations, such as city streets, tunnels, highways, and parking.

3.2 Kitti-CityScapes

Kitti-CityScapes Geiger, Lenz, and Urtasun, 2012 is a combined dataset that consists of some samples from CityScapes and KITTI. The samples of this dataset are displayed in Figure 3.2. The dataset contains a large amount of images that provide information about urban street scene understanding. This dataset contains high-quality examples of complex scenes with fine annotation on pixel and instance levels. While there are numerous great object detection datasets, there are a comparatively lower amount of datasets that address the problem of image segmentation in the urban domain. The dataset contains the following classes, which belongs to groups shown in Table 3.2.

Group	**Classes**
flat	road, sidewalk, parking, rail track
human	person, rider
vehicle	car, truck, bus, on rails, motorcycle, bicycle, caravan, trailer
construction	building, wall, fence, guard rail, bridge, tunnel
object	pole, pole group, traffic sign, traffic light
nature	vegetation, terrain
sky	sky

TABLE 3.2: The groups and classes in dataset

The data were collected during different daytime with different weather conditions. Moreover, the images were captured over several months in over 50 cities.

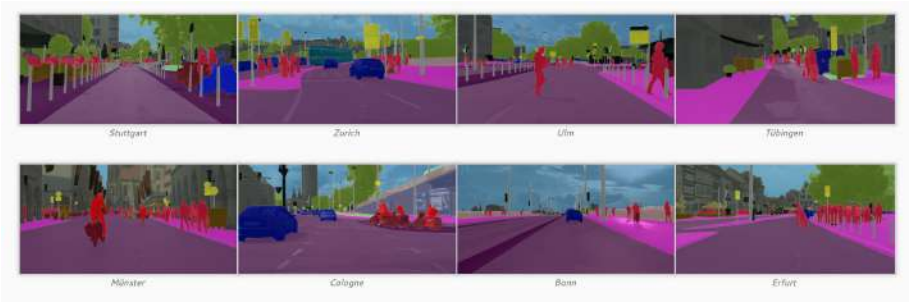

FIGURE 3.2: Images from Kitti-CityScapes Geiger, Lenz, and Urtasun, 2012 dataset

3.3 NuScenes

NuScenes Caesar et al., 2019 dataset were camptured of 1000 automotive driving scenes, each of them has 20s duration. The automotive driving systems could make the streets safer and improve the driver experience. While the monocular camera is an excellent type of sensor which could provide various information about the surrounding environment, it also has drawbacks. For example, the processed output of cameras could provide information about the location of the agents in the domain and their type. From another hand has difficulties measuring the distance to the objects. The solution for such a problem is to use multimodal data from different sensors, which will overcome each other weak sides. The system which combines the lidars, which could measure the distance to close objects, radars, which estimate the far objects, and monocular cameras, which help identify and track the objects using their semantic information precisely, could more accurately classify the status of the surrounding environment and as a result, make the decision faster. For such purposes the NuScenes Caesar et al., 2019 dataset was perfectly designed. One data collection platform consist of:

- Front camera
- Left front camera
- Right front camera
- Front RADAR
- Left front RADAR
- Right front RADAR
- Back camera
- Left back camera
- Right back camera
- Back RADAR
- Left back RADAR
- Right back RADAR
- LIDAR on the top

- IMU

This dataset has 3d object annotations (Fig. 3.3). Also, provide detailed map information on two cities, Boston and Singapore.

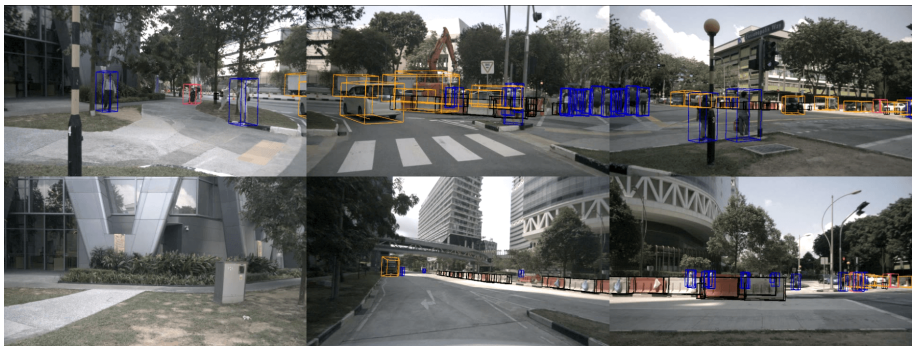

FIGURE 3.3: Images from NuScenes Caesar et al., 2019 dataset

3.4 Synscapes

Synscapes Wrenninge and Unger, 2018b is a dataset that consists of 25K unique images with all detail captured from sunlight and clouds to the composition and angles of the city(Fig. 3.4). The current level of technologies and algorithms could make the impressively realistic simulation, which could be, maybe, not an alternative but at least a good solution for the situation when the amount of data is relatively small. The Synscapes Wrenninge and Unger, 2018b dataset was generated with the focus on the photo and physic realism of the images. The generation algorithm included the impact of sun rays and weather conditions. Additionally, was taken into account the geometric of the scene, the reflecting ability of the materials, motion blur, and optics deformation. It is a common situation when the dataset for automotive vehicles is generated using the sequences. For some tasks, like object tracking, such a feature could be only beneficial. On the other hand, we have many images with pretty similar information. One of the unique characteristics of this dataset is that all 25k images were procedurally generated with unique scene and environment conditions.

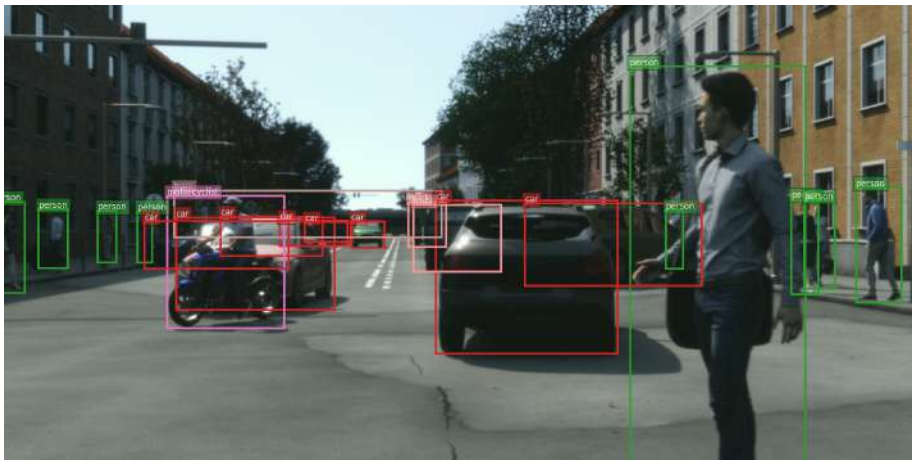

FIGURE 3.4: Images from Synscapes Wrenninge and Unger, 2018b dataset

3.5 Playing for Benchmark

Playing for Benchmark Richter, Hayder, and Koltun, 2017b is the next benchmark dataset which consists of synthetic images. This benchmark was generated using high-quality frames from around 250k videos. It could be used for numerous amount of tasks, such as:

- Optical flow
- Object detection
- Tracking
- Image segmentation
- Object-level 3D scene layout
- Visual odometry.

Each of the frames contains the annotated data, which was collected during the autopilot exploring of the virtual world(Fig. 3.5). The generation of the scene takes into account the composition of the environment agent and closely matches the natural environment. Moreover, all videos are photorealistic, which reduces the gap between domains.

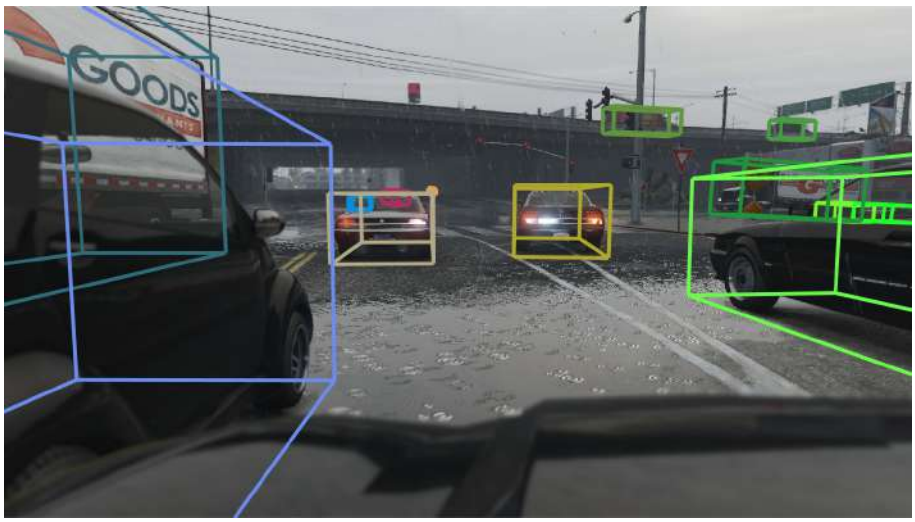

FIGURE 3.5: Images from Playing for Benchmark Richter, Hayder, and Koltun, 2017b dataset

Summarising the dataset research, we could state that there is an acceptable amount of labeled data that could be used to train models for an automotive vehicle system to solve a wide range of tasks using different data inputs. At the same time, datasets like Kitti-CityScapes Geiger, Lenz, and Urtasun, 2012 provide large-high-quality images and segmentation labels for them, which give the ability to understand the urban street scene but are not suitable for the object detection task. NuScenes Caesar et al., 2019 dataset is also a great source of real data as it contains multimodal data from different sensors, which is quite useless in my research topic. The datasets like BDD100K Yu et al., 2020, will be used as a source of real data, as

it contains a considerable amount of suitable quality images and labels for the object detection task. It is important to remember that this dataset was captured in the USA cities, so it will be interesting to validate it on images taken from another location. On the other hand, all generated datasets that were reviewed earlier could not be used as a source of data. Although the mentioned spawned data contains high-quality photorealistic images, we cannot manually control the environment, which provides a limitation in further research. A synthetic part of the dataset will be created with images generated from the CARLA Gao, Tang, and Wang, 2021 simulator.

Chapter 4

Evaluation

4.1 IoU

To make a quantitive comparison of the training network, we need to measure their performance. It is the place where the metrics come to the scene. In object detection, models predicting not only the location of the object but also the class of the detected instance. To count the rightly detected object, we will IoU, following the next logic. Intersection over union is a metric for evaluating the performance of the object detection model by comparing predicted and ground truth bounding boxes. Obvious that fact that the ideal prediction should follow the ground truth label by covering the same area with the same bounding box size.

$$\text{IoU} = \frac{area(gt \cap pd)}{area(gt \cup pd)} \tag{4.1}$$

Equation 4.1 considered the ratio between the intersection of the ground truth and predicted bounding box, and the union of actual and predicted bounding boxes. It is also known as the intersection over the union. IoU changes from 0 to 1, and the higher value, the better prediction was generated. This metric gives us the ability to set the threshold value. The output wich crossed this value will be counted as correctly classified, which will help us evaluate the model. As a threshold value for IoU, we use 0.5 and 0.75. That means the following, each bounding box with IoU higher than the threshold value will be marked as positive detection.

4.2 Precision and Recall

Analyzing the model output, we could have four situations:

- True Positive (TP) — correctly detected object.
- False Positive (FP) — detected the object did not exist.
- False Negative (FN) — object not detected by the object detector.
- True Negative (TN) —covers the cases when the background was correctly not detected by the model. This metric is not used in object detection.

Precision describes the degree the model detects the relevant object. Calculated as a ratio between correctly detected bounding boxes and all detected bounding boxes. Equation 4.2

$$P = \frac{TP}{TP + FP} \tag{4.2}$$

Recall describes the proportion all bounding boxes and calculated as correctly detected bounding box to all ground truth bounding box. Equation 4.3

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.3}$$

The excellent model has high precision and recall value. By rasing the threshold value of the IoU, the correctly classified will be objects with the most accurate prediction. In this case, we will have a higher value of FN. In the opposite situation, we will have a higher FP, as a result low precision and high recall values.

4.3 AP

Another way how to evaluate the performance of the trained model is to compute the Average Precision. The AP@α is an area under the curve that is computed for specific IoU threshold value α and computed as in equation 4.4. Important to note that the high value of the area under the curve means that model has considerable precision and recall value.

$$AP@\alpha = \int_0^1 p(r)dr \tag{4.4}$$

4.4 mAP

As was mentioned earlier in this section, object detection algorithms predict not only the location of the object but also the class of the belonging instance. Obviously that this kind of prediction is also needed to be measured with an aim to compare the model performance correctly.

$$\text{mAP}\alpha = \frac{1}{n}\sum_{i=1}^{n}\text{AP}_i \tag{4.5}$$

For such purposes, the mAP@α was used. This metric takes into account the precision and recall for all classes. As a result, it will have few AP values, which will be averaged. The equation 4.5 displaying the whole formula, where n stands for the number of classes.

Chapter 5

Experiments

. The SSD was chosen as an object detection algorithm due to its accuracy and low inference time, which is suitable for the automotive vehicle domain. As mentioned earlier, one of the main characteristics we should focus on while looking for an appropriate algorithm is low computational requirements. Keeping in mind this constraint was chosen three networks as a backbone for the SSD algorithm:

- MobileNetV1
- MobileNetV2.
- SqueezeNet.

5.0.1 Training details

At the very beginning, we decided to understand which training strategy was the best one. We launched experiments with 75% of actual data and 25% synthetic. The first set of experiments trained, as usual, the second one trained on synthetic and finetuned on real data. The result of experiments with finetuning. The approach with finetuning stage, on average, has 20% worse results compared to the default strategy. To avoid wasting time and computational resources e decided to research the default training strategy.

As a matter of fact, the training o a deep neural network is a computationally expensive process. Frameworks with parallel computing on GPU are highly used to reduce the training time. While there are many great choices, we decided to use the PyTorch framework due to its simplicity and python-like style. The training was done using two servers with 2 2080ti on the board, and the training time for each experiment was around one week. Pre-trained ImageNet models were used for the feature extraction part. The model training was launched with the following parameters(Table 5.1):

During the experiment phase of the research was interesting to investigate the impact of different amounts of data on the models performance. First of all, we tested on the actual dataset. After that, we launched the experiment on 75% of real data and 25% synthetic to understand the impact of synthetic data on the model. After the mentioned set of experiments were done, we started the experiments with an equal amount of real and generated data. The first pack of experiments was done for all classes(Table 5.2). We could see that models with 25% of synthetic data have comparable results in the case of the SqueezeNet model and even better for MobileNetV2. After that, we decided to analyze the results deeper. One of the main difficulties was the next, model could not effectively detect the small objects, especially at the night time of the day. We decided to launch experiments without traffic

Parameter name	**Value**
Learning-rate	10^{-2}
Momentum	0.9
Weight decay	$5 * 10^{-4}$
Gamma	0.1
Scheduler	cosine
Milestones for MultiStepLR	80,100
T_{max} value for Cosine Annealing Scheduler	200
Batch size	48
Number of epoch	200
Optimizer	SGD

TABLE 5.1: Training parameters

Data Ratio	mAP@50		
	SqueezeNet	MobileNetV1	MobileNetV2
100% of real data	18.33	35.43	23.8
75% of real data + 25% of synthetic	17.85	27.27	25.61
50% of real data + 50% of synthetic	15.51	26.49	24.6

TABLE 5.2: Launched experiments 1

lights and traffic signs(Table 5.3). As in previous experiments, we could see a similar picture. The last pack of experiments was launched only for classes like car, bus, person, and truck(Table 5.4). We could see that experiments with 25% of data look pretty promising. Often it is preferable to generate the last 25% of data, as it is cheaper and less time-consuming.

After that, we wanted to figure out the effect of synthetic data on the model, so we launched the experiments with 75% of the actual data(Table 5.5). We could see that for MobileNetV1, both models show comparable results, while the SqueezeNet and MobileNetV2 were improved by training with synthetic data. The Table 5.6 displays the performance of the model in the EU region.

Data Ratio	mAP@50		
	SqueezeNet	MobileNetV1	MobileNetV2
100% of real data	28.9	34.4	34.25
75% of real data + 25% of synthetic	26.13	32.2	33.74
50% of real data + 50% of synthetic	26.04	25.21	26.93

TABLE 5.3: Launched experiments 2

Data Ratio	mAP@50		
	SqueezeNet	MobileNetV1	MobileNetV2
100% of real data	38.25	62.63	45.70
75% of real data + 25% of synthetic	34.38	54.90	50.33
50% of real data + 50% of synthetic	33.79	44.78	46.80

TABLE 5.4: Launched experiments 3

Data Ratio	mAP@50		
	SqueezeNet	MobileNetV1	MobileNetV2
75% of real data + 25% of synthetic	34.38	54.90	50.33
75% of real data	30.62	55.09	46.23

TABLE 5.5: Launched experiments 4

Data Ratio	mAP@50		
	SqueezeNet	MobileNetV1	MobileNetV2
100% of real data	31.67	59.02	39.84
75% of real data + 25% of synthetic	30.87	51.67	44.92
50% of real data + 50% of synthetic	24.07	41.93	39.60

TABLE 5.6: Evaluation on data from another region

Chapter 6

Conclusions

6.0.1 Result Summary

The evaluated experiments show that generated synthetic data could be a reasonable solution for object detection tasks in the automotive vehicle domain. While providing additional data will be a long and expensive process, the combination of natural and synthetic data can give a desirable result. The best training strategy is regular training with natural and artificial data. From launched experiments, we could state that 25% of the end dataset could be replaced with synthetic data without a dramatic decrease in the model performance. On another side, the model trained with additional generated data on average outperforms the model without artificial data.

6.0.2 Future work

This work researched the impact of synthetic data on training object detection algorithms in the automotive domain. As mentioned earlier, this work focus on the implications of first-level synthetic data. As a future improvement, the subsequent work could investigate synthetic data's effect with more advanced generation scenarios, which will direct the complex cases in the real world. Additionally, the impact of domain randomization could also be included in the evolution of this work.

Bibliography

Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao (2020). "YOLOv4: Optimal Speed and Accuracy of Object Detection". In: *ArXiv* abs/2004.10934.

Bongini, Francesco et al. (2021). "Partially Fake it Till you Make It: Mixing Real and Fake Thermal Images for Improved Object Detection". In: *ACM Multimedia*. ACM, pp. 5482–5490.

Brekke, Åsmund, Fredrik Vatsendvik, and Frank Lindseth (2019). *Multimodal 3D Object Detection from Simulated Pretraining*. arXiv: 1905.07754 [cs.CV].

Buslaev, Alexander et al. (2020). "Albumentations: Fast and Flexible Image Augmentations". In: *Information* 11.2. ISSN: 2078-2489. DOI: 10.3390/info11020125. URL: https://www.mdpi.com/2078-2489/11/2/125.

Cabon, Yohann, Naila Murray, and Martin Humenberger (2020). *Virtual KITTI 2*. arXiv: 2001.10773 [cs.CV].

Caesar, Holger et al. (2019). "nuScenes: A multimodal dataset for autonomous driving". In: *arXiv preprint arXiv:1903.11027*.

Choi, Jungwook et al. (2019). "Accurate and Efficient 2-bit Quantized Neural Networks". In: *Proceedings of Machine Learning and Systems*. Ed. by A. Talwalkar, V. Smith, and M. Zaharia. Vol. 1, pp. 348–359. URL: https://proceedings.mlsys.org/paper/2019/file/006f52e9102a8d3be2fe5614f42ba989-Paper.pdf.

Deng, Jia et al. (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255.

Dosovitskiy, Alexey et al. (2017). "CARLA: An Open Urban Driving Simulator". In: *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 1–16.

Duan, Kaiwen et al. (2019). "CenterNet: Keypoint Triplets for Object Detection". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6568–6577.

Farabet, Clement et al. (Aug. 2013). "Learning Hierarchical Features for Scene Labeling". In: *IEEE transactions on pattern analysis and machine intelligence* 35, pp. 1915–1929. DOI: 10.1109/TPAMI.2012.231.

Feng, Di et al. (Feb. 2019). *Deep Multi-modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges*.

Fritsch, Jannik, Tobias Kuehnl, and Andreas Geiger (2013). "A New Performance Measure and Evaluation Benchmark for Road Detection Algorithms". In: *International Conference on Intelligent Transportation Systems (ITSC)*.

Fukushima, Kunihiko and Sei Miyake (1982). "Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position". In: *Pattern Recognit.* 15.6, pp. 455–469. DOI: 10.1016/0031-3203(82)90024-3. URL: https://doi.org/10.1016/0031-3203(82)90024-3.

Gaidon, Adrien et al. (2016). "Virtual worlds as proxy for multi-object tracking analysis". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 4340–4349.

Gao, Weihua, Jiakai Tang, and Taotao Wang (2021). "An object detection research method based on CARLA simulation". In: *Journal of Physics: Conference Series* 1948.

Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge (2015). *A Neural Algorithm of Artistic Style*. DOI: 10.48550/ARXIV.1508.06576. URL: https://arxiv.org/abs/1508.06576.

Geiger, Andreas, Philip Lenz, and Raquel Urtasun (2012). "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hartwig, Sebastian and Timo Ropinski (2019). "Training Object Detectors on Synthetic Images Containing Reflecting Materials". In: *ArXiv* abs/1904.00824.

He, Kaiming et al. (2015). "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385. arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

Hinterstoisser, Stefan et al. (2019). *An Annotation Saved is an Annotation Earned: Using Fully Synthetic Training for Object Instance Detection*. arXiv: 1902.09967 [cs.CV].

Howard, Andrew et al. (2019). *Searching for MobileNetV3*. arXiv: 1905.02244 [cs.CV].

Howard, Andrew G. et al. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. arXiv: 1704.04861 [cs.CV].

Hubara, Itay et al. (2016). *Quantized Neural Networks: Training Neural Networks with Low Precision Weights and Activations*. arXiv: 1609.07061 [cs.NE].

Iandola, Forrest N. et al. (2016). *SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size*. arXiv: 1602.07360 [cs.CV].

Ioffe, Sergey and Christian Szegedy (2015). *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. arXiv: 1502.03167 [cs.LG].

Jin, Qing, Linjie Yang, and Zhenyu Liao (2020). *AdaBits: Neural Network Quantization with Adaptive Bit-Widths*. arXiv: 1912.09666 [cs.CV].

Kortylewski, Adam et al. (2018). *Training Deep Face Recognition Systems with Synthetic Data*. arXiv: 1802.05891 [cs.CV].

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

LeCun, Yann et al. (1999). "Object Recognition with Gradient-Based Learning". In: *Shape, Contour and Grouping in Computer Vision*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 319–345. ISBN: 978-3-540-46805-9. DOI: 10.1007/3-540-46805-6_19. URL: https://doi.org/10.1007/3-540-46805-6_19.

Liu, Wei et al. (2016). "SSD: Single Shot MultiBox Detector." In: *ECCV (1)*. Ed. by Bastian Leibe et al. Vol. 9905. Lecture Notes in Computer Science. Springer, pp. 21–37. ISBN: 978-3-319-46447-3. URL: http://dblp.uni-trier.de/db/conf/eccv/eccv2016-1.html#LiuAESRFB16.

Munoz, Daniel, J. Bagnell, and Martial Hebert (July 2010). "Stacked Hierarchical Labeling". In: vol. 6316, pp. 57–70. ISBN: 978-3-642-15566-6. DOI: 10.1007/978-3-642-15567-3_5.

Nowruzi, Farzan Erlik et al. (2019). "How much real data do we actually need: Analyzing object detection performance using synthetic and real data". In: *ArXiv* abs/1907.07061.

Öhman, Wilhelm (2019). "Data augmentation using military simulators in deep learning object detection applications". In.

Papert, Seymour (1966). "The Summer Vision Project". In.

Redmon, Joseph et al. (2015). *You Only Look Once: Unified, Real-Time Object Detection*. cite arxiv:1506.02640. URL: http://arxiv.org/abs/1506.02640.

Ren, Shaoqing et al. (2015). "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39, pp. 1137–1149.

Richter, Stephan, Zeeshan Hayder, and Vladlen Koltun (Oct. 2017a). "Playing for Benchmarks". In: pp. 2232–2241. DOI: 10.1109/ICCV.2017.243.

Richter, Stephan R., Zeeshan Hayder, and Vladlen Koltun (2017b). *Playing for Benchmarks*. arXiv: 1709.07322 [cs.CV].

Sandler, Mark et al. (2019). *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. arXiv: 1801.04381 [cs.CV].

Simonyan, Karen and Andrew Zisserman (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *International Conference on Learning Representations*.

Szegedy, Christian et al. (2015). "Going Deeper with Convolutions". In: *Computer Vision and Pattern Recognition (CVPR)*. URL: http://arxiv.org/abs/1409.4842.

Tremblay, Jonathan et al. (2018). *Deep Object Pose Estimation for Semantic Robotic Grasping of Household Objects*. arXiv: 1809.10790 [cs.RO].

Wrenninge, Magnus and Jonas Unger (Oct. 2018b). *Synscapes: A Photorealistic Synthetic Dataset for Street Scene Parsing*.

— (2018a). "Synscapes: A Photorealistic Synthetic Dataset for Street Scene Parsing". In: *ArXiv* abs/1810.08705.

Xie, C. et al. (2020). "Adversarial Examples Improve Image Recognition". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, pp. 816–825. DOI: 10.1109/CVPR42600.2020.00090. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR42600.2020.00090.

Yu, Fisher et al. (2020). *BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning*. arXiv: 1805.04687 [cs.CV].

Zoph, Barret et al. (Nov. 2020). "Learning Data Augmentation Strategies for Object Detection". In: pp. 566–583. ISBN: 978-3-030-58582-2. DOI: 10.1007/978-3-030-58583-9_34.