

**The University of South Bohemia in České Budějovice
Faculty of Science**

**Evaluation of Self-Supervised Learning for Vehicle-Type
Detection in Autonomous Driving**

Master's thesis

Manaf Ahmed

Advisor: Prof. Dr. Patrick Glauner

České Budějovice 2023

M.W. Ahmed, "Evaluation of Self-Supervised Learning for Vehicle-Type Detection in Autonomous Driving" MS. thesis in English, Faculty of Applied Computer Science, Deggendorf Institute of Technology, Deggendorf Germany and Faculty of Science, University of South Bohemia, České Budějvice, Czech Republic, Aug. 2023, p.68

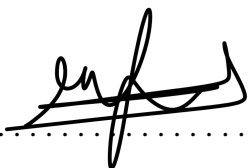
Annotation:

Self-Supervised Learning (SSL) method was implemented on vehicle-type detection task within autonomous driving context. The task involved evaluation of prominent SSL methods such as SimCLR and SimSiam against the conventional supervised method. The study explores the influence of different factors on the performance of SSL and its ability to generalize well under different circumstances.

I declare that I am the author of this qualification thesis and that in writing it I have used the sources and literature displayed in the list of used sources only.

Deggendorf, 30.08.2023

Date



Signature

Table of Content

ACKNOWLEDGEMENT	vii
ABSTRACT	viii
CHAPTER 1: INTRODUCTION	1
1.1 Overview:	1
1.2 Problem Statement:	2
1.3 Research Questions:.....	2
1.4 Scope of Study:.....	3
1.5 Objectives:.....	3
1.6 Methodology:.....	3
1.6.1 Research:.....	3
1.6.2 System Design:	4
1.6.3 System Development:	4
1.6.4 Testing and Troubleshooting:.....	4
1.7 Report Outline:.....	4
CHAPTER 2: RELATED WORK	6
2.1 Evolution of Vehicle-Type Detection in Autonomous Driving:	6
2.1.1 Early Approaches:.....	6
2.1.2 Deep Learning Approaches:	7
2.1.3 Self-Supervised Learning Approaches:.....	8
2.2 SSL Emergence:	9
2.2.1 Pretext Tasks:	9
2.2.2 Downstream Tasks:	10
2.3 SSL Paradigms in Object Detection:	10
2.3.1 Contrastive Learning Techniques for Object Detection:	10
2.3.1.1 SimCLR:.....	11
2.3.1.2 MoCo:	12
2.3.1.3 SwAV	14

2.3.2 Non-Contrastive Self-Supervised Learning (NCSSL):	15
2.3.2.1 BYOL	15
2.3.2.2 SimSiam	16
2.3.2.3 VICReg	17
2.4 SSL Compared to Other Approaches	19
2.5 Summary	20
CHAPTER 3: METHODOLOGY	21
3.1 Introduction.....	21
3.2 Dataset and Preprocessing.....	21
3.2.1 Description of Dataset.....	21
3.2.2 Preprocessing for Pretraining.....	23
3.2.3 Data Augmentation	24
3.2.3.1 SSL Augmentation	24
3.2.3.2 Additional Augmentation	24
3.3 Models Implementation.....	25
3.3.1 Backbone, Input, Batch Sizes.....	25
3.3.2 Loss Function.....	26
3.3.3 Optimizer.....	28
3.3.4 Projection Head.....	29
3.3.5 Predictor (in SimSiam).....	29
3.4 Pretraining Procedure	30
3.5 Linear Evaluation.....	30
3.6 Object Detection Preprocessing.....	31
3.7 Evaluation Metrics.....	32
3.7.1 Pretraining Metrics.....	32
3.7.2 Classification Metrics	32
3.7.3 Object Detection Metrics	33
3.8 Experimental Setup	34

3.9 Conclusion	35
CHAPTER 4: RESULTS.....	36
4.1 Introduction.....	36
4.2 Pretraining.....	36
4.2.1 High-level Overview	36
4.2.2 Detailed Visualization.....	37
4.3 Linear Evaluation.....	39
4.3.1 Evaluation Scores	39
4.3.2 Detailed Visualization.....	40
4.4 Evaluation of Transfer Learning	43
4.4.1 Transfer Learning Scores	43
4.4.2 Detailed Transfer Learning Visualizations	45
4.5 Discussion of Results	50
CHAPTER 5: CONCLUSION	52
5.1 Summary	52
5.2 Challenges	53
5.3 Limitations.....	54
5.4 Recommendations	55
REFERENCES	56
APPENDIX	63
Appendix.A CIFAR Pretraining	63
Appendix.B CIFAR Linear Evaluation.....	65
Appendix.C Further ImageNet Experiments.....	67

List of Figures

Figure 2.1 SimCLR architecture by Chen et al. (2020).....	12
Figure 2.2 MoCo architecture (He et al., 2019).....	13
Figure 2.3 SwAV architecture (Caron et al., 2020).....	14
Figure 2.4 BYOL architecture (Grill et al., 2020).....	16
Figure 2.5 SimSiam architecture (X. Chen & He, 2021).....	17
Figure 2.6 VICReg architecture (Bardes et al., 2022)	18
Figure 3.1 General workflow	21
Figure 3.2 ImageNet Subset_I classes.	22
Figure 3.3 ImageNet Subset_II classes	23
Figure 3.4 Two augmented views of the same image.....	24
Figure 3.5 SimSiam Model architecture	26
Figure 3.6 SimCLR Model architecture.....	26
Figure 3.7 Augmented images with their annotations	31
Figure 4.1 SimSiam pretraining	37
Figure 4.2 SimCLR pretraining	38
Figure 4.3 Comparison of methods (mean of 5 trials)	39
Figure 4.4 SimSiam linear evaluation	40
Figure 4.5 SimCLR linear evaluation.....	41
Figure 4.6 SSL methods evaluation	41
Figure 4.7 t-SNE features	42
Figure 4.8 Transfer learning accuracy comparison	43
Figure 4.9 Transfer learning loss comparison (right: bounding box, left: classifier).....	43
Figure 4.10 ImageNet_subset_II class distribution	45
Figure 4.11 SimSiam classification heatmap	46
Figure 4.12 SimCLR classification heatmap.....	46
Figure 4.13 Visualization of bounding boxes with different colors (blue: ground truth,.....	47
Figure 4.14 IoU values distribution comparison	48
Figure 4.15 SimSiam AUC- ROC	48
Figure 4.16 SimCLR AUC- ROC.....	49
Fig A.1 SimSiam CIFAR pretraining.....	63
Fig A.2 SimCLR CIFAR pretraining.....	64
Fig B.1 CIFAR mean accuracy Linear Evaluation.....	65

Fig B.2 CIFAR Linear Evaluation graph comparison.....	65
Fig B.3 CIFAR SimCLR linear evaluation.....	66
Fig B.4 CIFAR SimSiam linear evaluation.....	66
Fig C.1 ResNet50 linear evaluation on ImageNet_subset.....	67
Fig C.2 SimCLR ResNet50 linear evaluation graph on ImageNet_subset_I.....	67
Fig C.3 SimSiam ResNet50 linear evaluation graph on ImageNet_subset_I.....	68

List of Tables

Table 4.1 Pretraining Metrics	36
Table 4.2 Metrics comparison between methods	39
Table 4.3 Object detection scores on test set.....	44

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my supervisor, **Prof. Dr Patrick Glauner**, for his unwavering support and guidance throughout the course of my master's thesis. His profound expertise and insightful feedback have been invaluable in shaping this research. His patience and motivation have made this thesis possible and significantly contributed to my growth as a researcher. I am truly grateful for his mentorship and the opportunity to learn from him. His passion for the field of machine learning has been a constant source of inspiration for me. This invaluable guidance, support, and knowledge was a key factor in helping me to accomplish this project successfully.

ABSTRACT

Identifying vehicles' appearances with the existence of complex surroundings and occlusions is a crucial task in autonomous driving. Only in Germany, 47% of vehicles are passenger cars, the rest is distributed among trucks, buses, and other non-conventional types. Moreover, using supervised methods requires a significant amount of labelled data, which is expensive. Self-Supervised Learning offers an efficient solution which requires less labelling yet improves generalization on unseen data. The study focuses on two prominent self-supervised methods, SimCLR (contrastive) and SimSiam (non-contrastive) and compares their performance against conventional supervised learning. This work employs a comprehensive experimental setup on a domain-specific subset of ImageNet with five vehicle types, reproducing a similar pipeline to the original papers to fairly evaluate their transferability performance on vehicle-type detection task. Results demonstrated that both SimCLR and SimSiam score better accuracy than supervised learning. However, the performance dynamics vary, with SimCLR showing superior performance in classification task, while SimSiam surpassed in bounding box precision. The study empirically analyzes these SSL methods and finds that the choice of backbone architecture depth is dependent on the dataset size and image resolution. The findings of this thesis have important implications in tackling automotive industry domain gaps, offering better generalizability and robustness. The superior performance in vehicle-type detection can enhance the capabilities of these systems in trajectory planning, traffic flow analysis, and safety precautions. The study also opens new avenues for future research in SSL and its potential applications in more sophisticated object detection tasks within the autonomous driving context.

CHAPTER 1: INTRODUCTION

1.1 Overview:

Self-supervised learning (SSL) is a machine learning technique that enables models to learn from unlabelled data without explicit supervision. It has become a promising approach for addressing some of the limitations of supervised learning, including the need for a substantial amount of labelled data, overfitting, and generalization to unseen data. Self-supervised learning has the potential to revolutionize various industries, including Healthcare, Finance, Manufacturing, and autonomous driving.

In the domain of autonomous driving, vehicle-type detection is a critical task that involves identifying the type of vehicle in the environment. Accurate detection of vehicle types is essential for various autonomous driving tasks, including traffic flow analysis, trajectory planning, and collision avoidance. However, this task is challenging due to the complexity of the surrounding environment and variability in the appearance of different vehicle types.

SSL can be employed to learn effective representations for vehicle-type detection in autonomous driving. By leveraging large amounts of unlabelled data, SSL can capture the underlying structure and relationships within the data, leading to more accurate and robust detection of vehicle types. For instance, SSL can learn representations of various vehicle parts, such as wheels, headlights, and taillights, which can then be used to identify the type of vehicle based on these representations.

In this thesis, we investigate the use of self-supervised learning for vehicle-type detection in an autonomous driving context. We propose a self-supervised learning approach that leverages the structure and relationships within the data to learn effective representations for vehicle-type detection. We evaluate our approach on a dataset of autonomous driving scenarios and demonstrate its superior performance compared to existing state-of-the-art approaches for vehicle-type detection.

1.2 Problem Statement:

Identifying the type of vehicle in an autonomous driving environment is a crucial and challenging task due to the variations in the appearance of different types of vehicles and complex surroundings. In 2020, Germany had over 47 million registered vehicles according to the Federal Highway Research Institute (BASt), including 47% passenger cars, 16% trucks, and 7% buses, making accurate vehicle-type detection essential for managing and optimizing the flow of these vehicles.

Conventional supervised learning methods require a significant amount of labelled data, which is expensive and time-consuming to obtain. According to a report by MarketsandMarkets, the global data annotation market crossed 0.8 billion in 2022 and is expected to reach USD 3.6 billion by 2027, growing at a compound annual growth rate (CAGR) of 33.2%. This substantially increasing market indicates the huge expenditure on data annotation – whether in-house or outsourced – which is going to cost more and more as industries need additional data to keep their models' performance from declining. Therefore, an efficient and effective approach is needed to stop the financial drain. Self-supervised learning is a promising alternative that can utilize unlabelled data to learn effective vehicle-type representations.

Despite the potential of self-supervised learning, its utilization in autonomous driving remains underexplored. According to a report by Allied Market Research, the global market for autonomous driving technology is expected to grow rapidly from 2020 to 2027, with a projected market size of \$556.67 billion by 2027. This growth emphasizes the urgency of developing swift, efficient, and practical solutions for autonomous driving.

1.3 Research Questions:

- How does self-supervised learning compare to other approaches in terms of performance for vehicle-type detection in autonomous driving?
- What are the key factors influencing the effectiveness of self-supervised learning for vehicle-type detection in autonomous driving?
- Can self-supervised learning techniques effectively handle the domain gap and improve the generalization capabilities of deep learning models for detecting various vehicle types in autonomous driving scenarios?

1.4 Scope of Study:

The scope of this study is to investigate the effectiveness of self-supervised learning for vehicle-type detection in an autonomous driving context. Specifically, we aim to explore the use of self-supervised learning to learn representations of different vehicle parts and use them to identify the type of vehicle. We will evaluate our approach on a public dataset of autonomous driving scenarios and compare its performance with existing state-of-the-art approaches. The study will also investigate the impact of various factors on the performance of the self-supervised learning approach, such as the size of the training data and the complexity of the vehicle types.

The study will contribute to the growing body of research on self-supervised learning and its applications in Autonomous Driving (AD) and possibly add enhance Advanced Driver Assistance Systems (ADAS) which will pave the way for further research and development in this area. The findings of this study may have practical implications for the development of safer and more efficient autonomous driving systems.

1.5 Objectives:

1. To investigate the effectiveness of self-supervised learning techniques for vehicle-type detection in the context of autonomous driving.
2. To compare the performance of the proposed self-supervised learning approach with existing state-of-the-art methods for vehicle-type detection.
3. To evaluate the impact of various factors on the performance of the self-supervised learning approach and provide insights for improving vehicle-type detection in autonomous driving applications.
4. To Recommend improvements for all the possible drawbacks of the model and how to make it more robust and reliable.

1.6 Methodology:

1.6.1 Research:

Reviewing and analyzing earlier research papers that are related to this work, highlighting the key issues and the various methods used to address them. In addition,

researching various approaches to creating such a system is crucial to getting a clear understanding of how to begin the project without making the same mistakes as others.

1.6.2 System Design:

In order to have a robust and robust system, we first need to design it. This allows us to define how we want our system to operate and define the key functionalities. This is the stage where we define the flow of our system. This stage will define the system on which the final code is based.

1.6.3 System Development:

To achieve the desired system, it's crucial to accurately integrate the components. Each function can be created using standard coding practices, resulting in a system that runs smoothly, and effectively, and is easier to debug during troubleshooting.

1.6.4 Testing and Troubleshooting:

Numerous tests must be performed on the system once it has been fully developed to check for efficiency, robustness, and accuracy when used with real data. The system's shortcomings or the area that needs to be improved will be revealed by the testing results. Tests and adjustments conducted over time will eventually produce the desired outcomes and consistent performance.

1.7 Report Outline:

Chapter I:

Chapter 1 serves as an introduction to the study, providing an overview of the research topic. It defines the problem statement, research questions, scope, objectives of the study, and the overall methodology to approach this study, covering research, system design, development, testing, and troubleshooting. The chapter concludes with an outline of the report's structure and organization.

Chapter II:

Chapter two explores the evolution of vehicle-type detection in autonomous driving, from early approaches to deep learning techniques. It discusses the emergence of self-supervised learning (SSL) and its paradigms in object detection. SSL methods are compared with other approaches, providing insights into their significance for vehicle-type detection.

Chapter III:

This chapter presents the methodology employed to evaluate self-supervised learning for vehicle-type detection in autonomous driving. It outlines data preprocessing and the experimental setup. The chapter also covers evaluation metrics and the comparison with supervised learning. Data augmentation is utilized to improve performance in vehicle type detection.

Chapter IV:

Chapter four presents the evaluation of self-supervised learning methods for vehicle-type detection in autonomous driving. It includes an analysis of pretraining results, performance comparison with supervised learning, and the impact of transfer learning. The chapter concludes with an interpretation of the results and their implications for autonomous driving systems.

Chapter V:

Chapter Five concludes the thesis by summarizing the key findings of the study and their implications for machine learning and autonomous driving. It discusses the challenges and limitations encountered during the research. The chapter also provides recommendations for future research in self-supervised learning and potential improvements in autonomous driving systems.

CHAPTER 2: RELATED WORK

This chapter provides a comprehensive overview of the evolution of vehicle-type detection in autonomous driving, starting from early approaches, transitioning into deep learning methods, and finally focusing on the emergence of self-supervised learning (SSL) techniques. It dives into the different paradigms of SSL in object detection, including contrastive and non-contrastive methods, with a detailed examination of prominent techniques under each paradigm. The chapter also presents a comparison of SSL with other approaches and concludes with a summary of the key points.

2.1 Evolution of Vehicle-Type Detection in Autonomous Driving:

Autonomous driving has gained significant attention in recent years, with advancements in computer vision and deep learning techniques playing a crucial role. One of the fundamental tasks in autonomous driving is vehicle-type detection, which involves identifying and classifying different types of vehicles on the road. This section provides a general overview of the evolution of vehicle-type detection approaches in autonomous driving, highlighting the key milestones and contributions.

2.1.1 Early Approaches:

Early approaches focused on merely object detection relied on traditional computer vision techniques and handcrafted features such as Feature-based, Template Matching, and Model-based approaches.

Feature-based Approaches utilized handcrafted features, such as edges, corners, and texture, to distinguish between different vehicle types. One popular approach involved extracting gradient-based features, like Histograms of Oriented Gradients (HOG), to represent vehicle shapes ([Dalal & Triggs, 2005](#)). The HOG features were then fed into classifiers such as Support Vector Machines ([Boser et al., 1992](#)) to classify vehicle types. These methods showed promising results in certain scenarios but struggled with variations in lighting conditions, occlusions, and complex backgrounds.

Template-matching techniques were also employed for object detection, especially when Lim and Guntoro used them for vehicle-type detection ([2001](#)). These methods relied on comparing vehicle images with predefined templates or reference models to determine

their types (The templates could be created based on manually annotated images or through the use of geometric models. While template-matching approaches exhibited simplicity and efficiency, they were sensitive to changes in viewpoint, scale, and appearance, limiting their applicability in real-world scenarios.

Another category of early object detection involved model-based approaches. These methods utilized geometric and dynamic properties to detect and classify vehicles, building consistent and efficient 2D representations out of 3D range data ([Petrovskaya & Thrun, 2009](#)). By matching the observed image with the expected appearance based on the models, these approaches could estimate the position, velocity and shape of vehicles. However, model-based techniques often faced challenges in handling viewpoint variations, occlusions, and lighting conditions.

These methods focused on extracting discriminative features and employing classifiers to distinguish vehicles ([Sivaraman & Trivedi, 2013](#)). However, these approaches often faced challenges such as limited robustness, scalability, and difficulties in handling complex real-world scenarios.

2.1.2 Deep Learning Approaches:

The advent of deep learning revolutionized the object detection concept in autonomous driving, expanding to pedestrians, traffic signs, vehicle types, etc. Early Deep learning works started the use of Convolutional Neural Networks (CNNs) to extract discriminative features from raw data which are fed into a classifier. Notable works include the implementation of CNNs for object detection by [Hinton et al. \(2012\)](#) and AlexNet ([Krizhevsky et al., 2012](#)).

Region-based Convolutional Neural Networks (R-CNN) improved detection by leveraging region proposal algorithms to generate potential object regions for classification. The features within these regions were extracted using CNNs, and a classifier was trained for vehicle type identification. R-CNN-based methods, such as Fast R-CNN ([Girshick, 2015](#)), achieved remarkable accuracy in object detection tasks.

Single Shot Detectors (SSD) based approaches addressed the limitations of region proposal algorithms by directly predicting object bounding boxes and class probabilities from multiple feature maps at different scales. These methods offered faster inference times while maintaining competitive detection performance. The works of [Liu et al. \(2016\)](#) and [Redmon et al. \(2016\)](#) demonstrated the effectiveness of SSD.

Additionally, YOLOv3 ([Redmon and Farhadi, 2018](#)) offered better real-time performance by leveraging a single pass through the network for object detection. Recent advancements in deep learning-based detection have focused on improving efficiency and reducing computational complexity. On the other hand, EfficientDet ([Tan et al., 2020](#)) introduced a family of highly efficient and accurate object detectors by optimizing the network architecture and scaling strategies. These detectors demonstrated state-of-the-art performance in various vehicle detection benchmarks.

Although these tremendous potentials of deep learning have opened doors for customized object detection, especially in autonomous driving, challenges related to occlusions, lighting conditions, and generalization remained an issue on the path of advancing the capabilities of autonomous driving models to further improve.

2.1.3 Self-Supervised Learning Approaches:

To build upon previous works and address their challenges, self-supervised learning methods have been proposed for improving supervised tasks including detection. These methods aim to learn useful representations from unlabelled data, which can then be used for downstream tasks such as vehicle-type detection.

Contrastive learning is one of the SSL methods that appeared in works like the Simple Framework for Contrastive Learning of Visual Representations (SimCLR) by [Chen et al., \(2020\)](#), Momentum Contrast (MoCo) by [He et al., \(2020\)](#) which was improved later as MoCo v3 ([Chen et al., 2021](#)), and SwAV by Caron et al. ([2020](#))

The non-contrastive method illustrates promising results as well, works such SimSiam, ([X. Chen & He, 2021](#)), leverage non-contrastive learning to learn powerful representations without requiring explicit annotations. Similarly, the BYOL method, introduced by [Grill et al. \(2020\)](#), employs a different variant of self-supervised learning to train a model on large-scale unlabelled data, resulting in improved performance on downstream tasks. Not to forget [Bardes et al., \(2022\)](#) who proposed a self-supervised learning technique called VICReg, which uses variance-invariance-covariance regularization to learn pretext-invariant representations.

Moreover, self-supervised learning approaches have also been combined with other techniques such as domain adaptation to improve object detection in autonomous driving. For instance, [Munir et al., \(2021\)](#) proposed a domain-adaptive self-supervised learning

approach that leverages both unlabelled and labelled data from source and target domains. It was extensively evaluated on the FLIR-ADAS and the KAIST Multi-Spectral dataset.

All of the above-mentioned methods have shown remarkable performance in detecting different vehicle types with limited labelled data. As self-supervised learning continues to advance, we can expect further improvements in better vehicle perception and other autonomous driving tasks.

2.2 SSL Emergence:

2.2.1 Pretext Tasks:

Pretext tasks are auxiliary objectives that can be used to learn representations in a self-supervised manner. The idea is to design a task that allows the model to learn features that are useful for solving a downstream task such as colourization (; [Larsson et al., 2016](#)), placing image patches in the right place ([Doersch et al., 2015](#) ; [Noroozi & Favaro 2016](#)), placing frames in the right order ([Misra et al., 2016](#) [Lee et al., 2017](#)), inpainting ([Pathak et al., 2016](#)), or classify corrupted images ([Fang et al., 2022](#))

One of the main benefits of using pretext tasks is that they allow us to leverage large amounts of unlabelled data, which is often much easier to obtain than labelled data. By training on a pretext task, we can learn a set of features that are likely to be useful for many different tasks ([Yi et al., 2022](#)), without the need for task-specific labels. In this way, self-supervised learning has the potential to greatly improve the efficiency and scalability of machine learning systems.

Several other recent works have also explored the use of pretext tasks for self-supervised learning, including "Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles" ([Noroozi and Favaro, 2016](#)), "Learning Representations by Maximizing Mutual Information Across Views" ([Bachman et al., 2019](#)), and "Self-Supervised Learning of Pretext-Invariant Representations" by Misra and Van Der Maaten, ([2020](#)).

Choosing the right pretext task is vital for successful self-supervised learning. It must require an understanding of the data that is also needed for the downstream task ([Zaiem et al., 2022](#)). An example of this is the autoencoder, which compresses input images into a reduced form and then generates an output image as close to the original as possible. However, if the downstream task requires generating higher-quality images, the autoencoder would be a poor choice as it also regenerates noise in the original image.

Overall, pretext tasks provide a powerful tool for learning representations in a self-supervised manner. By leveraging large amounts of unlabelled data and designing tasks that encourage the model to learn useful features, we can greatly improve the efficiency and scalability of machine learning systems.

2.2.2 Downstream Tasks:

In the context of machine learning, downstream tasks refer to the primary objectives that a model is designed to achieve after learning the representations of data through a pretext task. As stated by Tanaka et al. (2022b), "A typical SSL model is trained from a large amount of unlabelled data. Then, the trained parameters are transferred into a model for specific tasks called downstream tasks." These tasks usually require labelled data and are specific to a particular application or problem.

Downstream tasks are crucial for evaluating the effectiveness of self-supervised learning. Automatic Speech Recognition (Tanaka et al., 2022b), image retrieval, and generative modelling are some of the most common tasks used to evaluate self-supervised learning. Notably, in the context of autonomous driving, there are various other tasks such as Object detection (Pototzky et al., 2021), semantic segmentation (Wang et al., 2022), video class agnostic segmentation (Siam et al., 2021), depth estimation (Xue et al., 2020), and action recognition (Zhou et al., 2023) which indicates the increasing significance of SSL in the automotive industry and autonomous driving in particular.

2.3 SSL Paradigms in Object Detection:

2.3.1 Contrastive Learning Techniques for Object Detection:

Contrastive learning has recently emerged as a popular technique in the field of machine learning for unsupervised representation learning. However, its roots can be traced back to the early work on Siamese neural networks in the 1990s (Bromley et al., 1994). The idea of using contrastive loss to train such networks was introduced by Hadsell et al. (2006b), where they proposed a metric learning approach that learned the similarity between pairs of images. This work led to the development of many Siamese-based architectures in the following years.

Wang and Gupta (2015) introduced the concept of Contrastive Convolutional Neural Networks for unsupervised feature learning. They trained a deep neural network to learn meaningful representations from unlabelled data. This work inspired the development of

many other contrastive-based learning approaches for unsupervised representation learning, including Contrastive Predictive Coding (CPC) by Hénaff et al. (2020) and SimCLR (Chen et al., 2020).

Another important work in contrastive learning is the triplet loss, which was proposed by Schroff et al. (2015). The triplet loss aims to learn embeddings that preserve the relative distances between samples in the input space. The triplet loss has been used in various applications, including image retrieval and person re-identification.

Recently, various contrastive learning methods have been developed, including InfoNCE loss (Oord et al., 2018), SimCLR (Chen et al., 2020), and MoCo (He et al., 2020). These methods have shown significant improvements in learning representations in both supervised and unsupervised settings. Contrastive learning is a promising approach to deep learning and has the potential to advance in various applications. We shall discuss some of its well-known approaches below.

2.3.1.1 SimCLR:

SimCLR is a contrastive learning method that was proposed by Chen et al. (2020). The basic idea of SimCLR is to learn representations that are invariant to different augmentations of the same image. This is achieved by using a contrastive loss function that maximizes the similarity between positive pairs and minimizes the similarity between negative pairs. The training process involves creating two views of each image and passing them through a shared encoder network. The representations of the two views are then compared using the contrastive loss function.

The success of SimCLR can be attributed to several factors. Firstly, it uses a simple yet effective training framework that is easy to implement and scale. Secondly, it uses data augmentation to create different views of each image, which increases the amount of training data and makes the learned representations more robust. Finally, it achieves strong transfer learning performance by fine-tuning the learned representations on downstream tasks.

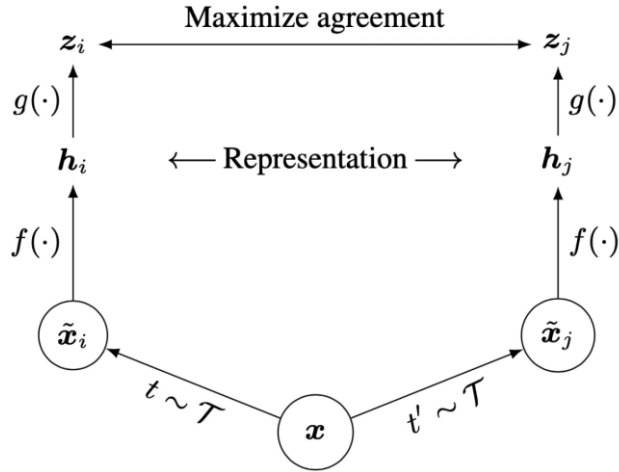


Figure 2.1 SimCLR architecture by Chen et al. (2020)

In the framework above, two distinct data augmentation operators are selected from a common augmentation set, named $t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$ applied to each data example to get two views. The encoder $f(\cdot)$ and a projection head $g(\cdot)$ are then trained to increase similarity via contrastive loss. After training is accomplished, the projection head $g(\cdot)$ is thrown away while encoder $f(\cdot)$ and representation h is used for downstream tasks.

SimCLR has been evaluated on several benchmark datasets, including CIFAR-10 (Krizhevsky & Hinton, 2009), ImageNet ILSVRC-2012 (Russakovsky et al., 2015), and COCO (Lin et al., 2014). The success of SimCLR has led to several follow-up studies that aim to improve the method further such as Chen et al. (2021) with MOCO v3, and Caron et al. (2020) with SwAV.

2.3.1.2 MoCo:

MoCo (Momentum Contrast) is a self-supervised learning framework for visual representations proposed by He et al. (2020). The framework achieved state-of-the-art performance on several benchmark datasets and was later improved with the introduction of MoCo v2 and MoCo v3.

The basic idea of MoCo is to learn a momentum-based contrastive loss function that maximizes the similarity between positive pairs and minimizes the similarity between negative pairs. Based on the work of He et al. (2020), the training process involves creating two views of each image, one as a query and the other as a key. The query image is passed through a query network, and the key image is passed through a key network. The key network is updated with a momentum update rule to make it more stable and improve the quality of the learned representations.

MoCo v2 ([X. Chen et al., 2020](#)) and MoCo v3 ([X. Chen et al., 2021](#)) improve upon the original MoCo framework by introducing several new techniques. MoCo v2 introduced a queue-based contrastive loss function that stores a large number of negative keys and uses them for training. MoCo v3 introduced a contrastive learning framework for vision transformers that leverages the structure of the transformer model and uses a different loss function that is based on the dot product between query and key representations.

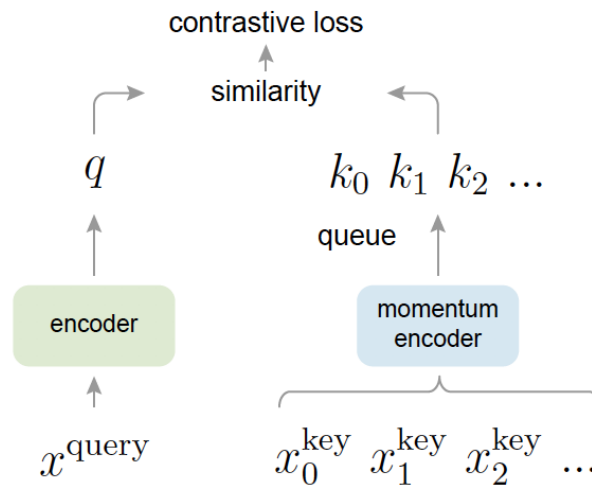


Figure 2.2 MoCo architecture (He et al., 2019)

In the above diagram, Momentum Contrast (MoCo) trains an encoder for visual representation in the provided example by applying a contrastive loss to match an encoded query, q , with a dictionary of encoded keys. The keys, denoted as $\{k_0; k_1; k_2; \dots\}$ are calculated on the fly from a collection of samples. The dictionary is managed like a queue, with the current mini-batch enqueued and the oldest one dequeued, guaranteeing that the dictionary size remains constant regardless of the mini-batch size. The keys are encoded with a slowly progressing encoder that is kept up to date with momentum by the query encoder. This method makes it easier to build a big and reliable vocabulary, which improves visual representation learning.

The success of MoCo ([He et al., 2020](#)) framework can be attributed to several factors. Firstly, it uses a contrastive loss function that is effective for learning visual representations in a self-supervised manner. Secondly, it uses momentum updates to make the learned representations more stable and robust. Finally, it has been shown to achieve strong performance on downstream tasks on datasets including ImageNet, CIFAR-10 ([Krizhevsky & Hinton, 2009](#)), and COCO ([Lin et al., 2014](#)).

2.3.1.3 SwAV

Swapping Assignments between multiple Views (SwAV) is an online algorithm for learning unsupervised visual representations introduced by Caron et al. (2020). It seeks to obtain features without the use of manual annotations and has demonstrated considerable performance increases when compared to supervised. The goal is to learn visual representations while simultaneously clustering the data and ensuring consistency between cluster assignments generated for various augmentations (or "views") of the same image. It accomplishes this without requiring specific instructions, pairwise feature comparisons, which can be computationally difficult.

SwAV employs a distinct prediction approach, where it predicts a view's code based on the representation of another view, rather than directly comparing image features. To convert scores into probabilities, SwAV incorporates a temperature parameter and computes assignments using the Sinkhorn-Knopp algorithm (Cuturi, 2013). The versatility of SwAV allows for training with both large and small batches (Caron et al., 2020), and it can efficiently scale to handle extensive datasets. Additionally, SwAV introduces the multi-crop data augmentation strategy, employing a mix of views with varying resolutions to enhance performance without increasing memory or computer demands.

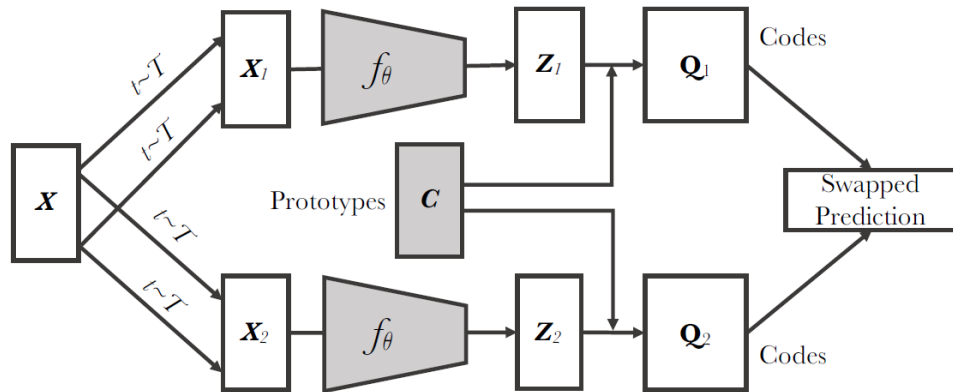


Figure 2.3 SwAV architecture (Caron et al., 2020)

As shown from the above architecture, SWAV algorithm produces multiple views of a collection of images (X_1 and X_2) by sequentially applying different augmentation operations. To create an embedding vector, these views go through feature extraction using a CNN backbone (ResNet50). The projection vector (designated as Z) is created by performing a shallow non-linear transformation on the embedding vector. A single linear prototype layer (designated as C) further processes the projection vector by mapping it to trainable prototype vectors. The output of the layer is calculated as the dot product of the

prototypes and the projection vector, with the weights of the layer serving as a learnable prototype bank. Sinkhorn-Knopp by Cuturi ([2013](#)) is used to assign clusters using a portion of the output from the linear layer, resulting in a swapped prediction problem. The output of the Sinkhorn-Knopp algorithm is denoted as Q .

2.3.2 Non-Contrastive Self-Supervised Learning (NCSSL):

Non-contrastive self-supervised learning (SSL) is a method of learning useful representations without requiring expensive target labels. It differs from contrastive SSL in that it does not require negative pairs ([Zhuo et al., 2023](#)), which are used to encourage representations of different objects to be further apart. Instead, non-contrastive SSL methods use positive pairs, where the hidden representations of two augmented views of the same object are brought closer together.

One advantage of non-contrastive SSL is that it does not require large batch sizes or memory queues to provide negative pairs ([Tian et al., 2021](#)), making it more efficient and conceptually simple. Notable methods of non-contrastive SSL include BYOL (Bootstrap Your Own Latent) and SimSiam, which employ a dual pair of Siamese networks to learn powerful representations using only positive pairs.

2.3.2.1 BYOL

Bootstrap Your Own Latent (BYOL) is a self-supervised learning approach introduced by Grill et al. ([2020](#)). BYOL aims to learn useful visual representations from unlabelled data, without the need for manual annotation or supervision.

The key idea behind BYOL is to train a neural network to predict the representations of a set of augmented views of an input image. Specifically, BYOL ([Grill et al., 2020](#)) consists of two neural networks: an online network and a target network. The online network is trained to predict the representations of a set of augmented views of an input image, while the target network is a copy of the online network that is updated with an exponential

moving average of the online network's weights.

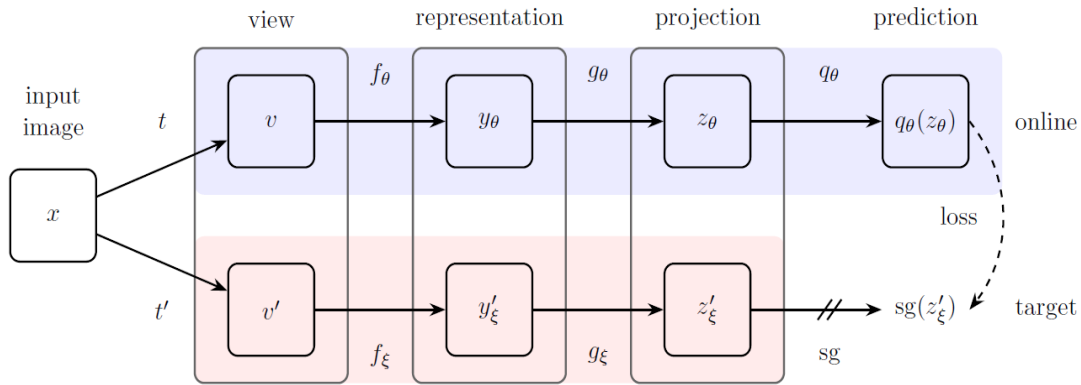


Figure 2.4 BYOL architecture (Grill et al., 2020)

As shown above, BYOL's architecture. BYOL reduces the similarity loss between $q_\theta(z_\theta)$ and z'_ξ , θ is the trained weights, ϵ refers to the exponential moving average of θ and sg refers to stop-gradient. Eventually, only f_θ is considered while the rest is disposed, and y_θ is considered as image representation.

BYOL's primary innovation lies in the training approach where the online network is responsible for predicting the target representations, while the target network does not undergo explicit training. Instead, the target network receives updates using an exponential moving average of the online network's weights. This unique approach enables the target network to acquire valuable representations without the need for direct supervision.

BYOL has achieved state-of-the-art performance on several benchmark datasets, including Birdsnap (Berg et al., 2014), the SUN397 scene dataset (Xiao et al., 2010), and others. It has also been shown to be effective for transfer learning, where a model trained on one dataset is fine-tuned on another dataset.

2.3.2.2 SimSiam

SimSiam is a self-supervised learning approach for training deep neural networks (X. Chen & He, 2021). SimSiam aims to learn useful representations from unlabelled data, without the need for manual annotation or supervision. The method involves training a neural network with a Siamese architecture, where two identical networks share the same weights. The input data is randomly augmented twice and then fed into the two networks, and the networks are trained to predict the similarity of the two outputs.

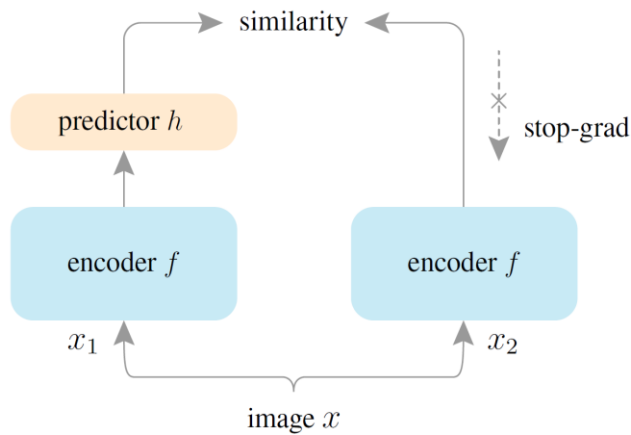


Figure 2.5 SimSiam architecture (X. Chen & He, 2021)

During the training of SimSiam, the same encoder network processes two augmented perspectives of the same image. Without the use of negative pairings or a momentum encoder, the model maximizes the similarity between both sides. (X. Chen & He, 2021). An important component of the training process is a stop-gradient operation, which is critical to prevent the model from collapsing to a constant. The stop-gradient operation treats one of the output vectors as a constant in the loss function, effectively creating two sets of variables that the model alternates between optimizing.

The key innovation in SimSiam is it doesn't use negative pairs, unlike many other self-supervised learning models that use negative sample pairs to avoid collapsing solutions, SimSiam directly maximizes the similarity of two views of the same image without using negative pairs. Secondly, it doesn't use momentum encoders, which are commonly used in other models to maintain consistency in the representations. Lastly, it doesn't require large batches as the model works well with typical batch sizes and does not rely on large-batch training.

SimSiam has achieved state-of-the-art performance on several benchmark datasets, including ImageNet and CIFAR-10. It has also been shown to be effective for transfer learning, where a model trained on one dataset is fine-tuned on another dataset.

2.3.2.3 VICReg

The VICReg algorithm (Bardes et al., 2022) introduces a regularization term that encourages the network to learn representations that are invariant to transformations of the input data, while also maintaining a high variance and low covariance. This is achieved by minimizing the variance of the representations, maximizing their invariance to

transformations, and minimizing the covariance between different dimensions of the representations.

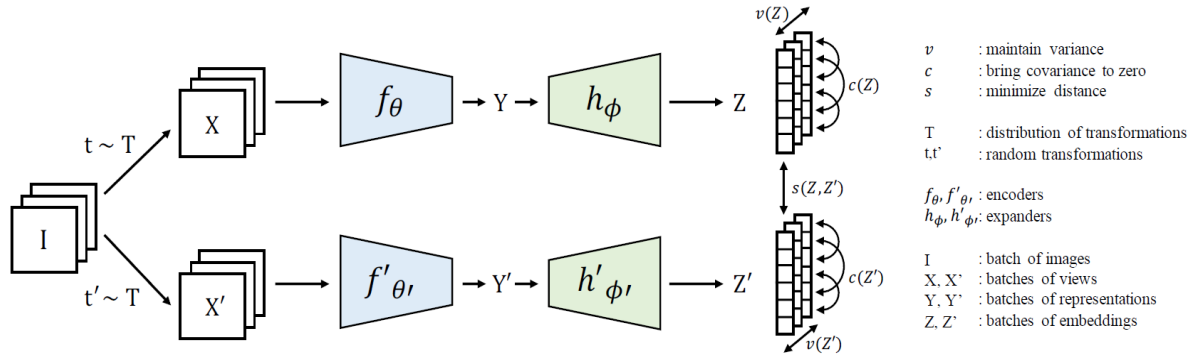


Figure 2.6 VICReg architecture (Bardes et al., 2022)

As illustrated above, having set I , two sets of diverse sees X and X' are created, which are at that point encoded into representations Y and Y' . The representations are sent to an expander, which produces the embeddings Z and Z' . The separation between two embeddings from the same picture is minimized, the fluctuation of each embedding variable over bunches is kept to a minimum, and the variance between sets of inserting factors over clusters is pulled in to zero, decorrelating the factors from each other.

During training, the VICReg algorithm uses a two-step process. First, it applies a series of transformations to the input data, and then it feeds these transformed inputs through a neural network to obtain a set of representations. The algorithm then computes the variance, invariance, and covariance of these representations and uses these values to compute a loss function, which it then minimizes. This process encourages the network to learn representations that are invariant to the transformations applied to the input data, while also ensuring that these representations have high variance and low covariance (Bardes et al., 2022).

The key innovation of the VICReg (Bardes et al., 2022) is its use of a regularization term that encourages the network to learn representations that are invariant to transformations of the input data, while also maintaining a high variance and low covariance. This approach allows the network to learn more robust and generalizable representations, which can improve performance on downstream tasks.

In summary, VICREG is a promising regularization technique for self-supervised learning that can improve the quality of learned representations by enforcing variance, invariance, and covariance constraints. The method has shown state-of-the-art performance

on several benchmarks and can be easily integrated into existing self-supervised learning frameworks.

2.4 SSL Compared to Other Approaches

SimCLR was evaluated across 12 natural image datasets. When fine-tuned, their self-supervised model outperformed the supervised baseline on five datasets, while the supervised baseline was superior on only 2 datasets. On the remaining five datasets, the models were statistically tied. In the semi-supervised setting, SimCLR was better than other cutting-edge semi-supervised methods with both 1% and 10% of the labels.

MoCo has been compared with ImageNet supervised pre-training, followed by fine-tuning on various tasks. The gaps to the ImageNet supervised pre-training counterpart were at least +0.5%. This suggests that MoCo has outperformed supervised pre-training in certain tasks like instance segmentation, pose estimation, and keypoint detection. It's worth noting that MoCo's improvement from ImageNet is noticeable, but relatively small.

BYOL has demonstrated superior performance compared to supervised baselines. This is true even when the baselines utilize more data or deeper models. When comparing BYOL to supervised ImageNet (Supervised-IN) across multiple benchmarks, BYOL outperforms the Supervised-IN baseline on 7 out of 12 benchmarks, while showing only marginally lower performance on the remaining 5.

SimSiam, has been found to yield superior results across various tasks. These tasks include *VOC 07 detection*: a Faster R-CNN ([Ren et al., 2015](#)) model was fine-tuned on the VOC 2007 trainval dataset and evaluated on the VOC 2007 test dataset; *COCO detection and instance segmentation*: a Mask R-CNN ([He et al., 2017](#)) model was fine-tuned on the COCO 2017 train dataset and evaluated on the COCO 2017 validation dataset. In all these tasks, SimSiam outperformed all other methods including supervised ones in AP_{50} , AP_{75} .

According to Caron et al. ([2020](#)), the SwAV algorithm shows great performance compared to supervised pretraining on various transfer tasks such as linear classification tasks on datasets like VOC07 ([Everingham et al., 2009](#)), Places205 ([Zhou et al., 2014](#)) and iNaturalist2018 ([Van Horn et al., 2018](#)). Moreover, SwAV paper was the first self-supervised method to outperform ImageNet supervised features on all transfer tasks and datasets, including object detection using Faster R-CNN ([Ren et al., 2015](#)) and DETR ([Carion et al., 2020](#)). Thus, SwAV demonstrates remarkable performance compared to supervised pretraining in these transfer scenarios.

VICReg had a good performance across various benchmarks. In linear classification tasks, VICReg outperforms the supervised method on mAP for VOC07 ([Everingham et al., 2009](#)), Places205 ([Zhou et al., 2014](#)) and iNat18 ([Van Horn et al., 2018](#)) datasets, except for a slight (+0.3%) margin on iNat18. For object detection on the VOC07+12 dataset plus another object detection and instance segmentation with Mask R-CNN model with an FPN backbone ([He et al., 2017](#)) on the COCO dataset ([Lin et al., 2014](#)), it outperforms other methods, improving performance by approximately 1%.

2.5 Summary

This chapter reviews self-supervised learning (SSL), especially its use in vehicle-type detection for autonomous driving. It traces the evolution of vehicle detection, from early to deep learning approaches, and the rise of SSL. It explores SSL paradigms in object detection, including contrastive techniques like SimCLR, MoCo, SwAV, and non-contrastive methods like BYOL, SimSiam, and VICReg. The chapter compares these SSL methods with supervised and semi-supervised approaches and discusses the CIFAR-10 and ImageNet ILSVRC-2012 datasets used in the thesis.

CHAPTER 3: METHODOLOGY

3.1 Introduction

This study aims to evaluate the effectiveness of self-supervised learning for vehicle-type detection in autonomous driving. The research questions focus on comparing self-supervised learning to other approaches, identifying key factors influencing its effectiveness, and assessing its ability to handle the domain gap and improve the generalization capabilities of deep learning models. This chapter’s methodology can be summarized in the following diagram:

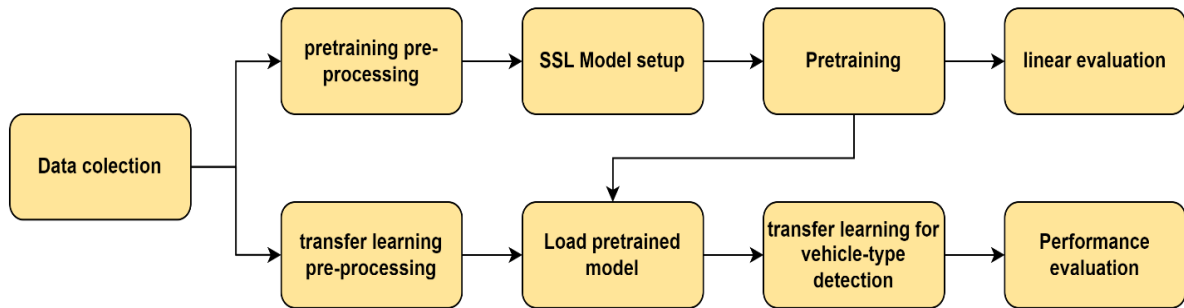


Figure 3.1 General workflow

The chapter is structured to provide a comprehensive understanding of the research process. It begins with a description of the dataset and preprocessing steps, followed by a detailed explanation of the self-supervised learning methods used. A supervised learning baseline is also established for comparison. The chapter then delves into the training procedure, evaluation, and experimental setup. The chapter concludes with a summary of the methodology used in the study.

3.2 Dataset and Preprocessing

This section discusses which datasets were used in this study, how they were pre-processed, augmented, and formatted for pre-training and Transfer learning.

3.2.1 Description of Dataset

The datasets used in this study are a subset of the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) dataset ([Russakovsky et al., 2015](#)) and CIFAR-

10 ([Krizhevsky, 2009](#)). However, most of the training and preprocessing was done on ImageNet.

The CIFAR-10 dataset is a collection of 60,000 32x32 coloured images in 10 classes, with 6,000 images per class. It is divided into 50,000 training images and 10,000 testing images. The dataset is widely used in machine learning and computer vision research. Although CIFAR-10 is less diverse and smaller in scale than ImageNet, its simplicity can be advantageous for initial model testing and rapid prototyping. The ten classes in CIFAR-10 are airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck ([Krizhevsky & Hinton, 2009](#)).

The ImageNet ILSVRC-2012 ([Russakovsky et al., 2015](#)) dataset is labelled with a single object class out of the 1000 possible classes, and the dataset is divided into several parts: a training set with around 1.28 million images with 732-1300 images per class, and bounding box annotated images ranging from 91-1268 per class, a validation set with 50,000 images, and a test set with 100,000 images ([Russakovsky et al., 2015](#)).

For the purpose of this study, we focus on vehicle-related classes in ImageNet. ImageNet's comprehensive dataset splits wheeled vehicles into ten car classes, five truck classes, and five van classes, making it a suitable choice for the task of vehicle-type detection in autonomous driving. These classes cover a wide range of vehicle types, which is crucial for measuring the ability to distinguish different types of vehicles in various autonomous driving scenarios. Two sets of five vehicles were used in this study – as shown below – the first set included a firetruck, golf cart, pickup truck, police van, and school bus, while the second set had an ambulance, cab, dustcart, jeep, and minivan.



Figure 3.2 ImageNet Subset_I classes.



Figure 3.3 ImageNet Subset_II classes

Both datasets, despite their differences, contribute to the study by providing a range of data complexities. This allows for a more comprehensive evaluation of the self-supervised learning methods under different conditions, especially ImageNet which aligns with the goal of evaluating self-supervised learning for vehicle-type detection. It provides a challenging and realistic benchmark for assessing the effectiveness of self-supervised learning methods.

3.2.2 Preprocessing for Pretraining

Given the organized nature of the CIFAR10 32x32 dataset, the preprocessing stage primarily involves splitting the data into training and validation sets (saved for linear evaluation). It's crucial to ensure that the split is stratified, meaning that each class is equally represented in both the training and validation sets. This stratified split helps maintain the balance of classes, preventing the model from becoming biased towards any particular class. It also ensures that the model's performance is evaluated fairly across all classes during the validation process.

The dataset underwent a rigorous preprocessing phase to ensure its quality and relevance. Initially, the images were programmatically and manually cleaned to remove any irrelevant images. This resulted in the exclusion of at least 10% of images from each class due to factors such as mislabeling, irrelevance, or ambiguity. Finally, the dataset was split in the same fashion as the CIFAR10.

To standardize the input for our models, all images were resized to a uniform size of 64x64 pixels. This size was chosen to balance computational efficiency and capability to preserve enough details for effective vehicle-type detection. In addition, images with a bounding box area ratio of 10% or less were excluded as well.

3.2.3 Data Augmentation

3.2.3.1 SSL Augmentation

In the process of pretraining, this study used SimCLR and SimSiam self-supervised learning methods. The data augmentations applied were similar to those used in their respective original papers. Data augmentation is a vital component of self-supervised learning, as it provides a source of variability and helps the model learn robust and invariant features.

For SimCLR, the augmentations included random cropping followed by resizing back to the original size, random horizontal flipping, colour distortions, and Gaussian blur. SimSiam also utilized a similar set of augmentations, as suggested in its original paper. These augmentations provide a rich and diverse set of training examples for the models to learn from.



Figure 3.4 Two augmented views of the same image

By closely following the augmentation strategies used in the original papers, we aimed to replicate the conditions under which these self-supervised learning methods have been shown to perform well. This allows for a fair comparison of their performance in the context of vehicle-type detection in autonomous driving.

3.2.3.2 Additional Augmentation

To increase the training samples for the five classes, we utilized data augmentation techniques to generate additional training images. This was accomplished using the data generator function from the Keras library, which allows for real-time data augmentation.

The generated images – 26,000 images out of 6,500 – underwent a series of transformations to obtain more data. These included rotation of the image by up to 40 degrees, shifting the image width and height by up to 20%, applying random shear

transformations with a shear intensity of 0.2, random zooming by 20%, and random flipping. These augmentations not only increased the size of the training set but also introduced a variety of transformations that the models need to be invariant to, thereby improving their ability to generalize to unseen data.

3.3 Models Implementation

The study utilized two self-supervised learning methods: SimSiam and SimCLR. SimSiam is a non-contrastive method that doesn't employ negative samples. Instead, it focuses on learning similar representations for two augmented views of the same image ([Chen & He 2021](#)). On the other hand, SimCLR is a contrastive method that makes use of negative examples to learn representations. It aims to maximize agreement between differently augmented views of the same image while minimizing agreement with negative samples ([Chen et al., 2020](#)). In the following subsections, detailed settings of both models shall be discussed.

3.3.1 Backbone, Input, Batch Sizes

Both methods used specific architectures based on the ResNet 50 model ([He et al., 2015](#)), a widely used deep residual network in computer vision tasks. The ResNet 50 has impressive performance on the ImageNet dataset, which was used for pretraining.

All Baseline settings were replicated, but certain modifications were made to adapt them to the requirements of this study. To ensure consistency in training both methods employed a unified batch size of 256; a higher batch size was unfeasible due to resource constraints.

Considering the small size of the images (64x64), the ResNet-18 architecture (see figures below) was used instead of ResNet-50. This decision was based on observations that deeper architectures like ResNet 50 may not perform as well on small images due to excessive downsampling and loss of important details ([He et al., 2015](#)). By utilizing ResNet 18. With a shallower architecture, the study aimed to preserve more spatial information and potentially enhance performance in vehicle-type detection tasks.

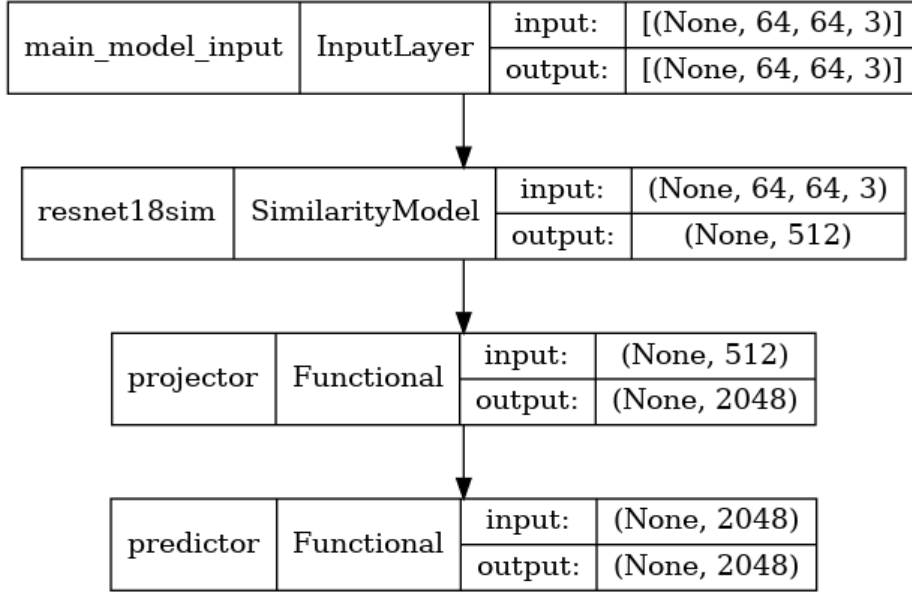


Figure 3.5 SimSiam Model architecture

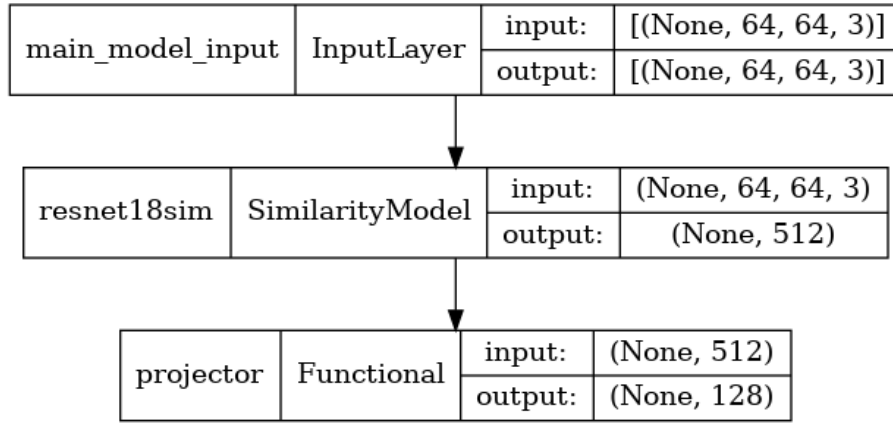


Figure 3.6 SimCLR Model architecture

3.3.2 Loss Function

SimCLR uses a contrastive loss function, specifically the NT-Xent (Normalized Temperature-Scaled Cross Entropy) loss. This function encourages the model to maximize agreement between differently augmented views of the same image while minimizing agreement with negative samples.

The NT-Xent loss for a positive pair of examples (i.e., two augmented views of the same image) is given by:

$$L_{i,j} = - \log \frac{\exp(\text{sim}(z_i, z_j) / \tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(\text{sim}(z_i, z_k) / \tau)} \quad (1)$$

Where:

- z_i and z_j are the representations of two augmented views of the same image (i.e., a positive pair), produced by the projection head of the model.
- τ is a temperature parameter that controls the concentration of the distribution. A higher value of τ produces a softer distribution (i.e., the probabilities are more uniform), while a lower value produces a harder distribution (i.e., the probabilities are more concentrated on a few examples).
- The sum in the denominator is over all pairs of examples in the batch (of size $2N$), and $1_{\{k \neq i\}}$ is an indicator function that is 1 when $k \neq i$ and 0 otherwise. This sum computes the normalization term for the softmax function, which ensures that the probabilities sum to 1.
- $\text{sim}(\cdot, \cdot)$ is the cosine similarity function, which measures the cosine of the angle between two vectors. It is defined as:

$$\text{sim}(u, v) = \frac{u^T \cdot v}{(\|u\| \|v\|)} \quad (2)$$

where u and v are vectors, u^T is the transpose of u , this equation denotes the dot product between $L2$ normalized u and v . The goal of the NT-Xent loss is to make the representations of positive pairs more similar (i.e., increase their cosine similarity) and the representations of negative pairs less similar (i.e., decrease their cosine similarity). This is achieved by minimizing the NT-Xent loss during training.

SimSiam loss, on the other hand, uses a symmetric negative cosine similarity loss. The goal of this loss function is to maximize the similarity between two differently augmented views of the same image. When sampling an input image x from the dataset upon which we create two augmented views, let's call them x_1 and x_2 . Both views are passed to the encoder model f which will output z_1 and z_2 , the representations of x_1 and x_2 respectively.

The representations are then passed to the prediction MLP h that will transform the representation of one view and matches it to the other view's representation. For example,

taking the representation z_1 and transforms it to $p_1 = h(z_1)$ and then compare it to z_2 . This distance is minimized using the following loss function:

$$L = -\frac{1}{2}(D(p_1, z_2) + D(p_2, z_1)) \quad (3)$$

Where $D(p_1, z_2)$ is the negative cosine similarity function, defined as:

$$D(p_1, z_2) = -\frac{p_1}{\|p_1\|_2} \cdot \frac{z_2}{\|z_2\|_2} \quad (4)$$

$\|\cdot\|$ denotes the $L2$ norm. The loss function is minimized when the cosine similarity of (p_1, z_2) , and (p_2, z_1) is maximized. z_2 is treated as a constant by using gradient-stop (detaching the variable from the graph):

$$L = \frac{1}{2}D(p_1, \text{stopgrad}(z_2)) + \frac{1}{2}D(p_2, \text{stopgrad}(z_1)) \quad (5)$$

Here the encoder on x_2 receives no gradient from z_2 in the first term, but it receives gradients from p_2 in the second term (and vice versa for x_1).

3.3.3 Optimizer

SimCLR uses the LARS (Layer-wise Adaptive Rate Scaling) optimizer ([You et al., 2017](#)). LARS is a learning rate policy that computes individual learning rates for each layer of the model. It's designed to improve the performance and stability of large-scale and distributed training. The update rule for LARS is as follows:

1. Compute the local learning rate for each layer:

$$\eta_l = \eta \frac{|w_l|}{|\nabla L(w_l)| + \beta |w_l|} \quad (6)$$

Here, (η) is the global learning rate, (w_l) are the weights of layer (l), ($\nabla L(w_l)$) is the gradient of the loss function with respect to and the weights of layer (l), and (β) is a weight decay term.

2. Update the weights of each layer:

$$w_l^{t+1} = w_l^t - \eta_l \times (\nabla L(w_l^t) + \beta \times w_l^t) \quad (7)$$

Here, (w_l^t) are the weights of layer (l) at time step (t), and $(\nabla L(w_l^t))$ is the gradient of the loss function with respect to the weights of layer (l) at time step (t).

This optimizer allows each layer to have a different learning rate, which can help to stabilize the training process and potentially improve the final performance of the model.

3.3.4 Projection Head

In SimCLR, the projection head is a 2-layer MLP that maps the output of the base encoder (e.g., a ResNet) to a space where the contrastive loss is applied. The purpose of the projection head is to project the representations to a 128-dimensional latent space where they are more suitable for contrastive learning. The base encoder $h(\cdot)$ maps an input image x to a representation vector $h = h(x)$. Then the projection head $g(\cdot)$ maps the representation h to a vector $z = g(h)$.

Similarly in SimSiam, there is a projection head but with a 3-layer MLP that maps the output of the backbone network to a 2048-dimensional latent space where the loss function is applied. Its output function has no ReLU.

3.3.5 Predictor (in SimSiam)

The prediction head is another 2-layer MLP that is used to predict the output of the projection head of the other view of the same image. If we denote the prediction head as $h(\cdot)$, the output p of the prediction head for an input image x can be represented as:

$$p = h(g(f(x))) \quad (8)$$

Here, $g(f(x))$ is the output of the projection head, and $h(\cdot)$ maps this output to another space where the loss is computed.

The loss function in SimSiam is then computed as the negative cosine similarity between z and p for two different views of the same image. This encourages the model to learn representations such that the output of the prediction head for one view is similar to the output of the projection head for the other view.

3.4 Pretraining Procedure

The pretraining procedure followed the guidelines provided in the original SimCLR and SimSiam papers. Both methods were pretrained using a mutual batch size of 256. This batch size was chosen as it was used in both papers, ensuring a fair comparison between the two methods. The pretraining was run for 100 epochs, and other hyperparameters, such as the learning rate, weight decay, and temperature parameter (for SimCLR), were kept similar to the values recommended in their respective papers. This was done to adhere as closely as possible to the original methods, and to ensure that any differences in performance could be attributed to the methods themselves, rather than differences in the training setup.

When using ImageNet data, the pretraining procedure was set up on a domain-specific subset of ImageNet, referred to as “ImageNet_Subset_I”. Choosing domain-specific pretraining can give substantial and statistically significant performance gains and lead to better results ([Li et al., 2022](#) ; [Zheng et al., 2021](#)). This subset was carefully selected to align with the specific task of vehicle-type detection in autonomous driving scenarios.

By following this pretraining procedure, the study aimed to provide a fair and rigorous comparison of the performance of SimCLR and SimSiam for the task of vehicle-type detection in autonomous driving scenarios.

3.5 Linear Evaluation

After pretraining the models in this study, i.e., SimCLR and SimSiam, the parameters are fixed, meaning that their weights are not updated during the subsequent training phase. Instead, a linear layer is added on top of this frozen backbone, and only the weights of this linear layer are trained. This setup allows us to evaluate the quality of the representations learned by the SSL models during the pretraining phase. If these representations are good, then a simple linear classifier should be able to use them to achieve good performance on the vehicle-type detection task.

This approach is compared with a supervised model of ResNet18 trained in a supervised manner, followed by a simple linear layer. This comparison allows us to assess the effectiveness of the self-supervised pretraining approach relative to a more traditional supervised learning approach.

3.6 Object Detection Preprocessing

Following the pretrained model, the image size of 64x64 was maintained. In addition, images with a tiny bounding box area ratio were excluded as well. The process of extracting labels and bounding boxes from the .xml files was applied through a custom script to parse these .xml files and extract the necessary information.

However, there was a discrepancy between the number of annotations and the number of images per class, with the number of annotations ranging from 91 to 1268, compared to 732 to 1300 images per class. This discrepancy indicates that not all images had corresponding annotations, which decreased the data size to half and could potentially hinder the training of the model.

To address this issue, data augmentation techniques were employed to generate more annotated data. These techniques included random transformations such as HSV, horizontal flipping, scaling, translation, Rotation, and shear, which can create new, varied examples from the existing images. The bounding boxes were also adjusted accordingly to match the transformations applied to the images.



Figure 3.7 Augmented images with their annotations

This resultant dataset is then used to train the SSL model consisting of a ResNet18 backbone followed by two heads: one for classification and one for regression. This approach is compared with a supervised model of ResNet18, this comparison allows us to assess the effectiveness of the self-supervised pretraining approach relative to a more traditional supervised learning approach.

3.7 Evaluation Metrics

In the study, different evaluation metrics are used at different stages to assess the performance of the models:

3.7.1 Pretraining Metrics

In the *Pretraining* stage, the metrics are used to monitor their performance:

- **Loss:** a function that the models are trying to minimize during training, which its formula was discussed earlier
- **Projector Loss:** computed in the projection space of the model. It's specific to the self-supervised learning methods used in this study.
- **Projector standard deviation:** This measures the standard deviation of the outputs in the projection space. It provides insight into the diversity of the learned representations.
- **Predictor standard deviation (for SimSiam):** This measures the standard deviation of the outputs in the prediction space. It's specific to SimSiam and provides insight into the diversity of the learned representations.

3.7.2 Classification Metrics

As for *Classification*, the following metrics are used to evaluate its performance:

- **Categorical cross-entropy:** This loss is used in multi-class classification tasks. The categorical cross-entropy loss for a single instance is calculated as follows:

$$L = - \sum_{i=1}^C y_i \log(\hat{y}_i) \quad (9)$$

Here, L is the categorical cross-entropy loss, C is the number of classes. y_i is the true label for class i . \hat{y}_i is the model's predicted probability that the instance belongs to class i .

- **Accuracy:** This measures the proportion of correctly classified instances, calculated as follows:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \quad (10)$$

- **Precision, Recall, F1 Score:** These metrics provide a more detailed view of the classifier's performance, taking into account both false positives and false negatives. Their respective formulas are:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (11)$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (12)$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (13)$$

- **AUC-ROC:** This is the area under the receiver operating characteristic curve. It measures the classifier's ability to distinguish between positive and negative instances, regardless of the classification threshold it is calculated as the area under the curve formed by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings:

$$\text{True Positive Rate (TPR)} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (14)$$

$$\text{False Positive Rate (FPR)} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}} \quad (15)$$

3.7.3 Object Detection Metrics

In *Object Detection*, the following metrics are used:

- **Mean Squared Error (MSE):** is a common loss function used for regression tasks. It measures the average squared difference between the actual and predicted values. The formula for MSE is:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2 \quad (16)$$

Here, n is the number of samples, Y_i is the actual value for the i -th sample, and \hat{Y}_i is the predicted value for the i -th sample. The squaring operation ensures all differences are positive and that larger differences are penalized more. The sum of these squared differences is divided by the number of samples to get the average.

- **Intersection over Union (IoU):** This measures the overlap between the predicted bounding boxes and the true bounding boxes. It's a common metric for evaluating the performance of object detection models.

$$\text{IoU} = \frac{\text{Area of Overlap}(BB_{\text{pred}}, BB_{\text{true}})}{\text{Area of Union}(BB_{\text{pred}}, BB_{\text{true}})} \quad (17)$$

- **mean Average Precision (mAP):** This is a summary metric for object detection models. It computes the average precision (AP) for each class and then takes the mean over all classes.

$$\text{AP} = \frac{1}{11} \sum_{r \in \{0.0, 0.1, \dots, 1.0\}} \max_{\tilde{r}: \tilde{r} \geq r} p(\tilde{r}) \quad (18)$$

Here, r is the recall, p is the precision, and the max operation is over all recall levels \tilde{r} greater than or equal to r . The mAP is then the mean of the AP values for all classes:

$$\text{mAP} = \frac{1}{\text{Number of Classes}} \sum_{\text{class}=1}^{\text{Number of Classes}} \text{AP}_{\text{class}} \quad (19)$$

In the context of vehicle-type detection in autonomous driving, these metrics provide a comprehensive evaluation of the models' performance. Here's a table summarizing the stages and their corresponding metrics:

Stage	Metrics
Pretraining	Loss, Projector Loss, Projector standard deviation, Predictor standard deviation (for SimSiam)
Classification	Loss, Accuracy, Precision, Recall, F1 Score, AUC-ROC
Object Detection	Loss, MSE, IoU, mean Average Precision

3.8 Experimental Setup

The experiments were conducted at the Deggendorf Institute of Technology's Deep Learning Lab, utilizing AI workstations with the following specifications:

- **CPU:** Intel Xenon W-1390P, 8 cores and 16MB cache, with 3.5 - 5.3 GHz.
- **RAM:** 128GB DDR4-UDIMM-ECC-RAM, divided across four 32GB modules.

- **SSD:** Two 2TB SSDs for fast and reliable data storage.
- **GPU:** NVIDIA RTX A5000 with 24GB memory, based on the latest Ampere architecture, offering up to 10 times faster compared to the previous generation.
- **Operating System:** Ubuntu 20.04, a robust operating system for AI tasks.

3.9 Conclusion

This chapter starts with dataset preparation and preprocessing, including image cleaning, resizing, and data augmentation. It then describes the self-supervised learning methods, SimCLR and SimSiam, detailing their architectures and hyperparameters. The pretraining procedure on a domain-specific subset of ImageNet is also discussed. Evaluation metrics for pretraining, classification, and object detection are explained. The chapter wraps up with the experimental setup, detailing the hardware and software used. This methodology provides a solid basis for assessing self-supervised learning in vehicle-type detection for autonomous driving.

CHAPTER 4: RESULTS

4.1 Introduction

The purpose of this chapter is to present the results obtained from our evaluation, which aimed to assess the effectiveness of self-supervised learning methods in addressing the domain gap and improving vehicle-type detection performance. We conducted extensive experiments using benchmark datasets and evaluated the performance of self-supervised learning models against traditional supervised learning approaches. The findings presented here shed light on the potential of self-supervised learning techniques and their practical implications in the field of autonomous driving.

The results presented in this chapter contribute to the existing knowledge in the field of autonomous driving and provide valuable insights into the potential benefits of self-supervised learning for vehicle-type detection. These findings have implications for the development of more advanced and reliable autonomous driving systems, ultimately aiming to enhance safety and efficiency on the roads.

4.2 Pretraining

4.2.1 High-level Overview

In the pretraining stage of my thesis, two self-supervised learning methods, SimSiam and SimCLR, were evaluated through 100 epochs of unsupervised training and their losses were monitored and recorded.

Method	Loss	Projector standard deviation	Projector loss	Predictor standard deviation
SimSiam	0.1703	0.0435	0.0852	0.0432
SimCLR	1.3249	0.0865	-	-

Table 4.1 Pretraining Metrics

The table above shows that SimSiam model achieved a loss of 0.1703, which is significantly lower than the loss of 1.3249 observed for the SimCLR model. This suggests that the SimSiam model was more effective at learning representations that align with the self-supervised learning objective during pretraining. The projector standard deviation for SimSiam was slightly better than SimCLR. The projector loss and predictor standard deviation for SimSiam was also reported in the table. The SimCLR model does not have a predictor, so these values are not applicable.

4.2.2 Detailed Visualization

This section provides an in-depth examination of the pretraining process, the first phase in the self-supervised learning pipeline, demonstrating elaborative insights on the performance of the SimSiam and SimCLR models during this phase.

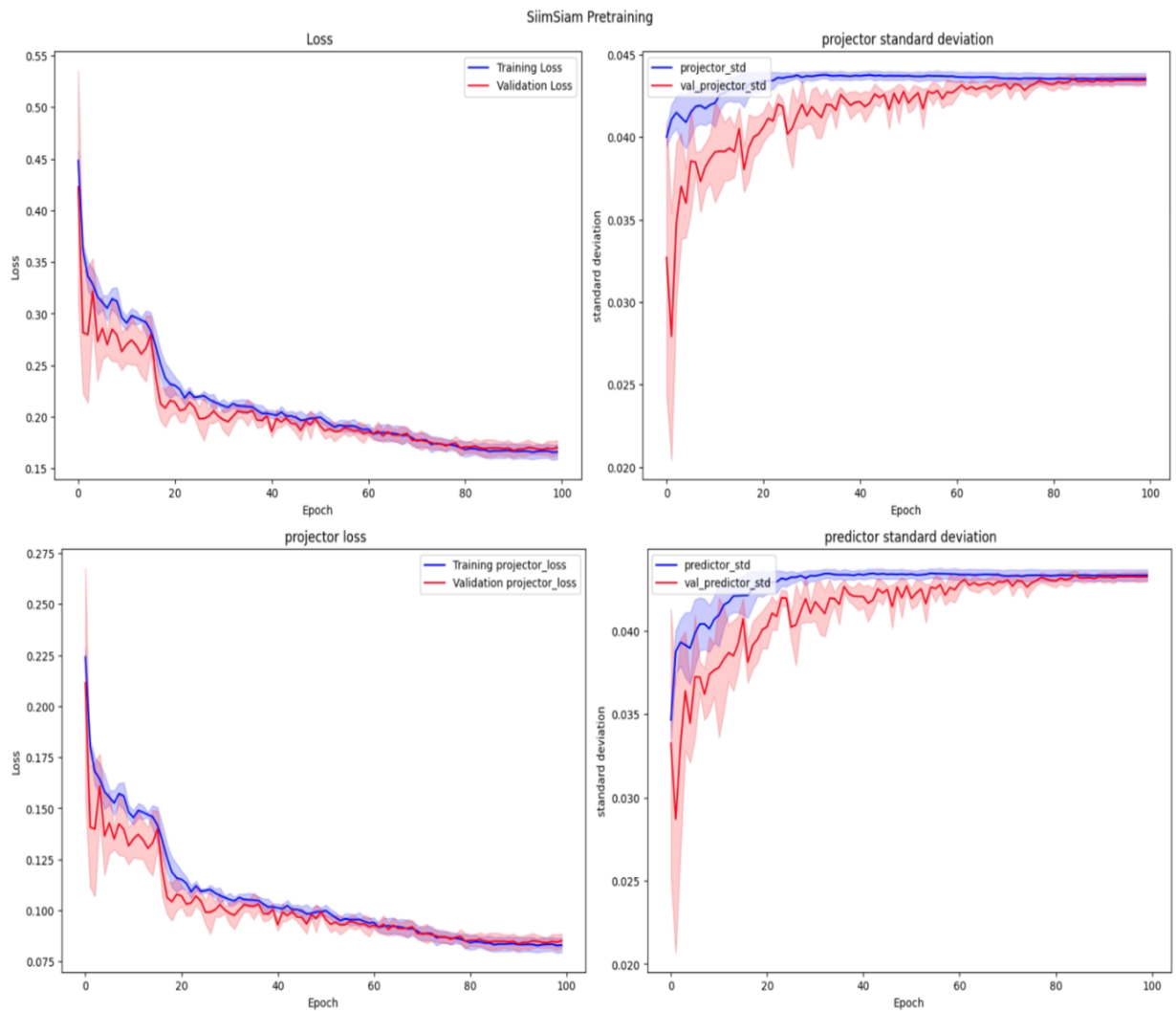


Figure 4.1 SimSiam pretraining

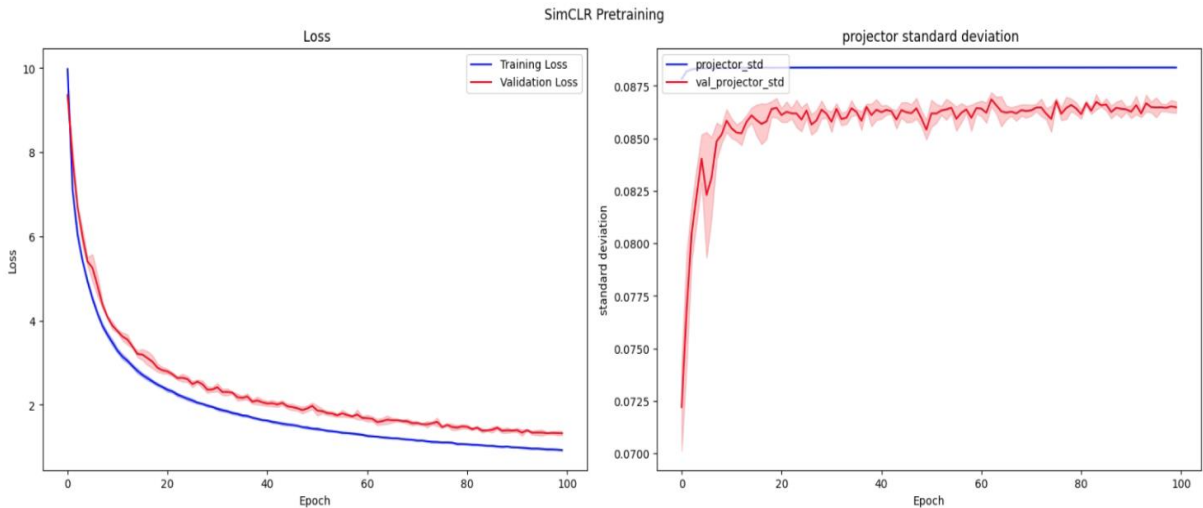


Figure 4.2 SimCLR pretraining

In all the above graphs, the shaded areas represent the range of values obtained from five cross-validation trials, while the actual curve represents the mean value. This representation allows for a visual assessment of the variability in the model’s performance across different trials.

The SimSiam model's loss curve shows an initial period of instability during the first 20 epochs. This instability could be attributed to the model adjusting to the data and learning the initial representations. However, after this initial period, the SimSiam model's loss curve becomes much smoother and eventually results in a lower loss than the SimCLR model. This implies that despite the initial instability, the SimSiam model is able to learn effective representations that minimize the loss.

The projector loss curve for the SimSiam model follows a similar trend to the model loss curve, but the losses are almost halved. This indicates that the projector, which maps the representations to a lower-dimensional space, is able to maintain the essential information while reducing the dimensionality.

The graphs showing the standard deviation of the activations of the final layer in the projector and predictor models provide insights into the diversity of the learned representations. The above standard deviations reach value $\approx \frac{1}{\sqrt{d}}$ (where d is the output dimension) illustrates that there is no output collapse, indicating more diverse representations, disseminated on the hypersphere, which can potentially lead to better performance on the downstream tasks.

Although – at first glance – there is an indication that one model is better than the other, this does not necessarily mean that the quality of its representations is better, linear evaluation and transfer learning tasks would give more insights on each model’s performance.

4.3 Linear Evaluation

4.3.1 Evaluation Scores

This section gives a high-level overview of the scores that each method has achieved in five cross-validated trials.

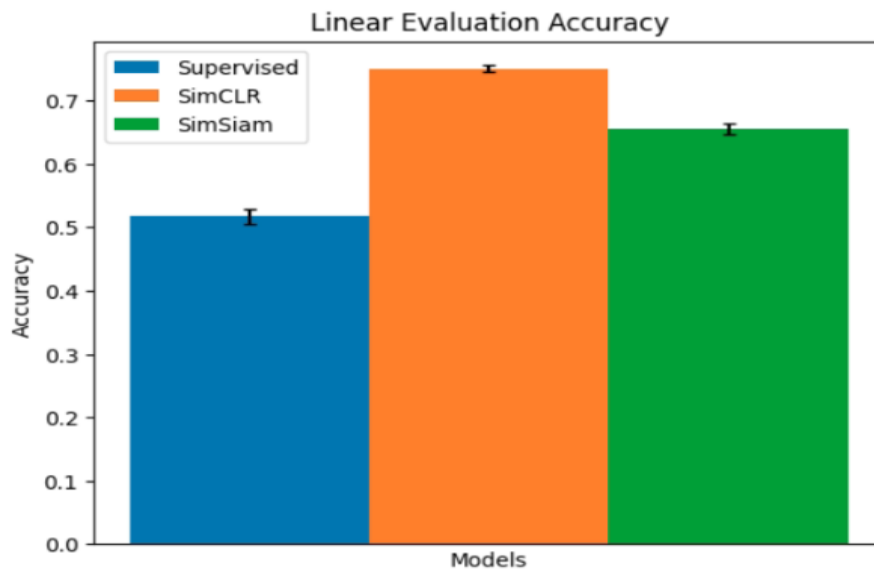


Figure 4.3 Comparison of methods (mean of 5 trials)

Method	Loss	Accuracy	Precision	F1 score
Supervised	1.2611	50.85%	73.51%	28.27%
SimSiam	0.9122	66.88%	80.17%	65.34%
SimCLR	0.6602	75.85%	83.69%	74.37%

Table 4.2 Metrics comparison between methods

The Supervised Learning method – using Resnet18 – showed the highest loss and the lowest accuracy among the three methods. This suggests that it struggled to capture the complexity of the vehicle-type detection task. Its precision was relatively high, indicating that when it did make correct predictions, they were often reliable. However, the low F1 score suggests that the model had difficulty in achieving a balance between precision and recall.

On the other hand, the SimSiam model showed a significant improvement over the Supervised Learning method. Its accuracy, precision, and F1 score were all substantially higher than those of the Supervised Learning method, demonstrating its strength in both identifying the correct classes and balancing precision and recall. However, during the linear evaluation stage, SimSiam was outperformed by SimCLR despite its lower pretraining loss. This could mean that representations might not have been generalized as well.

The SimCLR model outperformed both the Supervised Learning and SimSiam methods. Despite having a higher loss during pretraining, it achieved the lowest loss in the vehicle-type detection task, suggesting that the representations it learned were highly effective. It also achieved the highest accuracy, precision, and F1 score, indicating its superior performance in classifying vehicle types and its ability to balance precision and recall.

4.3.2 Detailed Visualization

This section provides an in-depth examination of the linear evaluation process, a crucial phase that follows pretraining in the self-supervised learning pipeline, offering a more granular perspective on the performance of the SimSiam and SimCLR models during this phase.

The below two graphs depict the progression of the linear evaluation process for both models. Again, the shaded areas in these graphs represent the range of values obtained from five cross-validation trials, while the actual curve denotes the mean value. This representation provides a visual understanding of the variability in the models' performance across different trials.

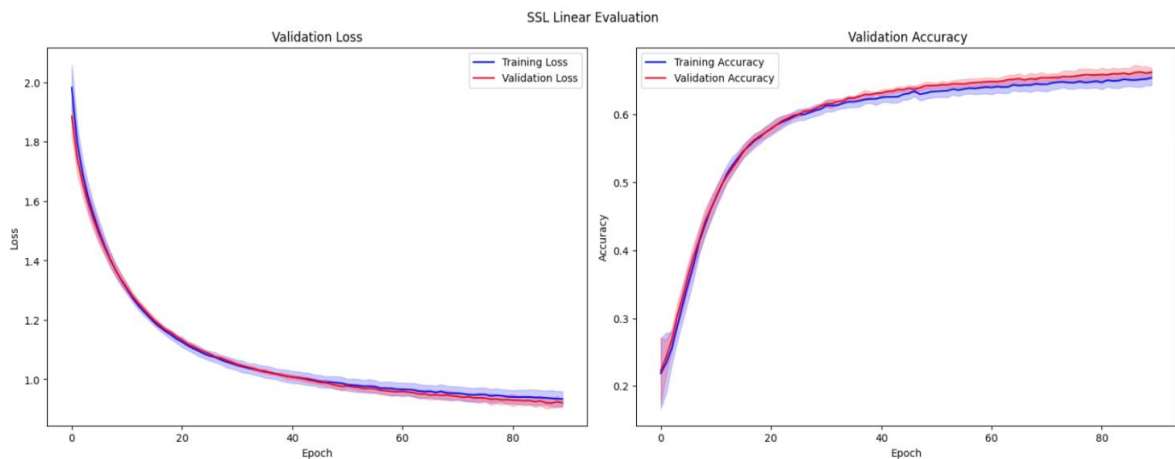


Figure 4.4 SimSiam linear evaluation

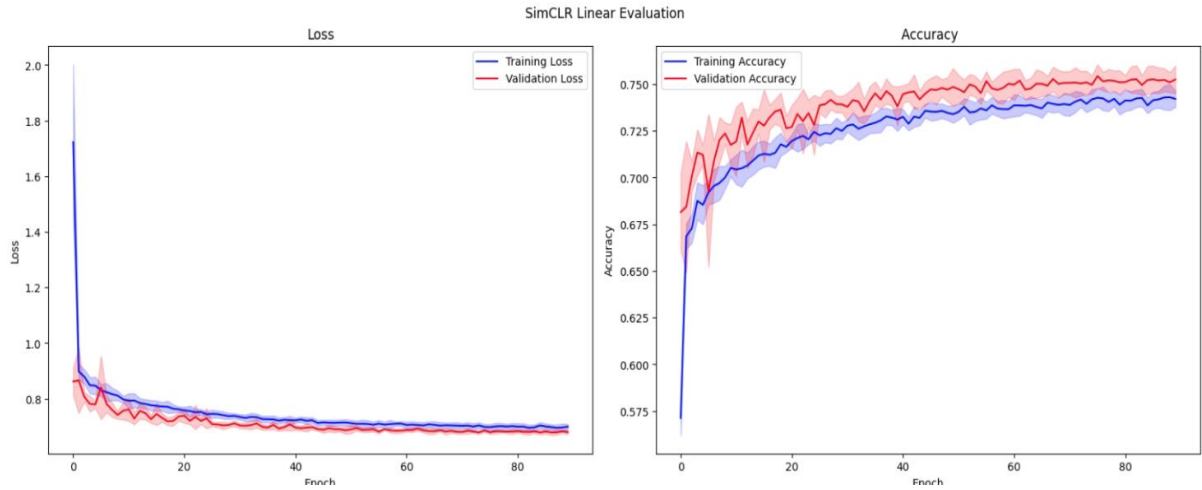


Figure 4.5 SimCLR linear evaluation

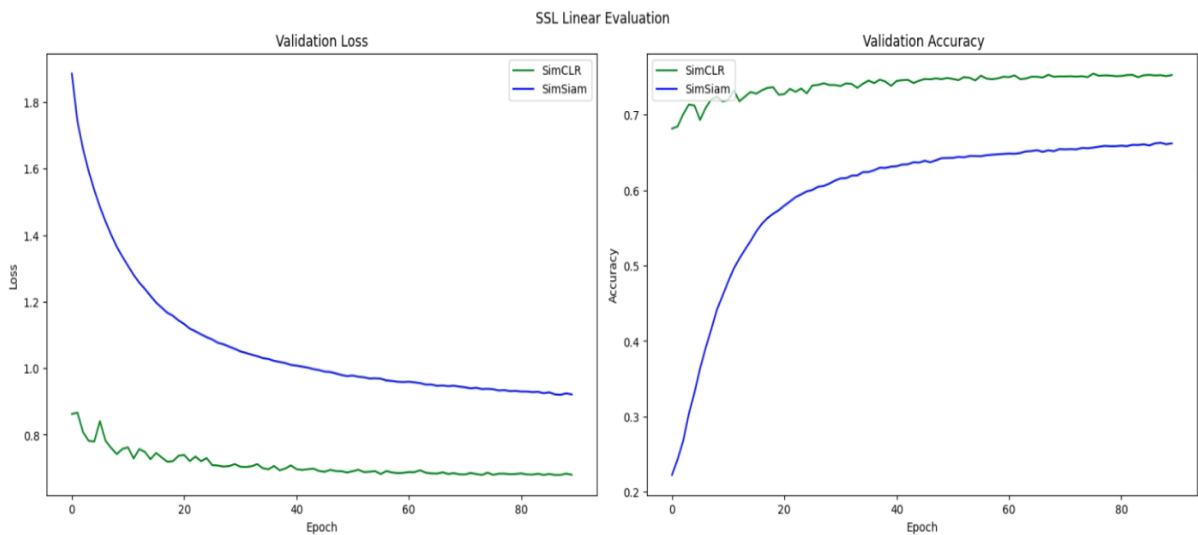


Figure 4.6 SSL methods evaluation

The evaluation of the models included multiple trials to ensure the robustness of the results. A bar plot summarizing the average accuracy across five trials for each model is demonstrated below, the error bar denotes the standard deviation. This visualization offers a clear comparison of the overall performance of the Supervised Learning, SimSiam, and SimCLR methods.

Interestingly, the SimCLR model's loss curve exhibits a degree of fluctuation throughout the entire training process, which contrasts with its behaviour during pretraining. Despite this fluctuation, SimCLR manages to achieve a lower loss than SimSiam by the end of the training process. This is evident in Figure B.3, where the green curve on the left (SimCLR) dips below the blue curve (SimSiam). Moreover, on the right side of the figure, SimCLR proves quick impressive results from the very beginning, outperforming SimSiam

in terms of accuracy, despite having a higher loss during the pretraining phase. This suggests that SimCLR's representations may be more robust or better suited to the downstream task.

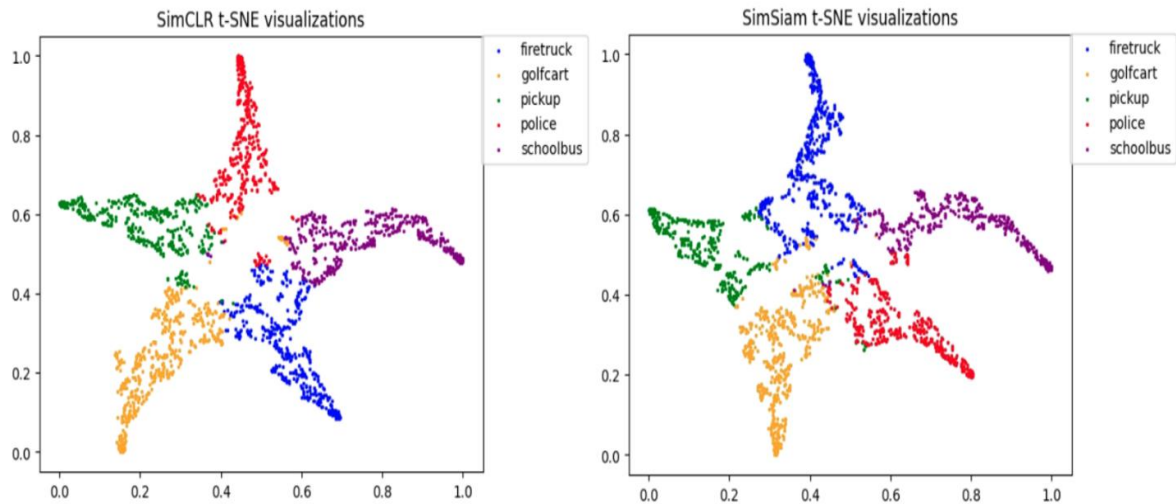


Figure 4.7 t-SNE features

The study also dives into the t-SNE visualizations of the learned representations. t-SNE (t-Distributed Stochastic Neighbor Embedding) is a dimensionality reduction approach that is particularly appropriate for the visualization of high-dimensional datasets. It works by estimating the likelihood that two data points in a high-dimensional space are neighbours and then selecting a low-dimensional embedding that produces a comparable distribution. t-SNE is used to visualize the data points for the ImageNet_subset_I classes. Each point in the t-SNE plot corresponds to an image, and the spatial proximity of points reflects the similarity of their high-dimensional features. The classes' features look somewhat distinct, but a notable part of the features are not that distant from each other, suggesting that the representation learned could be further improved.

In conclusion, while the Supervised Learning method struggled with vehicle-type detection task as they are all subclasses of vehicles, both SimSiam and SimCLR showed strong performance, with SimCLR being the most effective. These results highlight the potential of self-supervised learning methods for complex tasks like vehicle-type detection. However, it's worth noting that the performance of these methods can be influenced by factors such as the complexity of the task, the quality of the training data, and the choice of hyperparameters.

4.4 Evaluation of Transfer Learning

4.4.1 Transfer Learning Scores

Here we compare all models on object detection on very similar vehicle types, more visualizations and insights in this section. The following graphs are comparisons between different methods in transfer learning averaged over five trials:

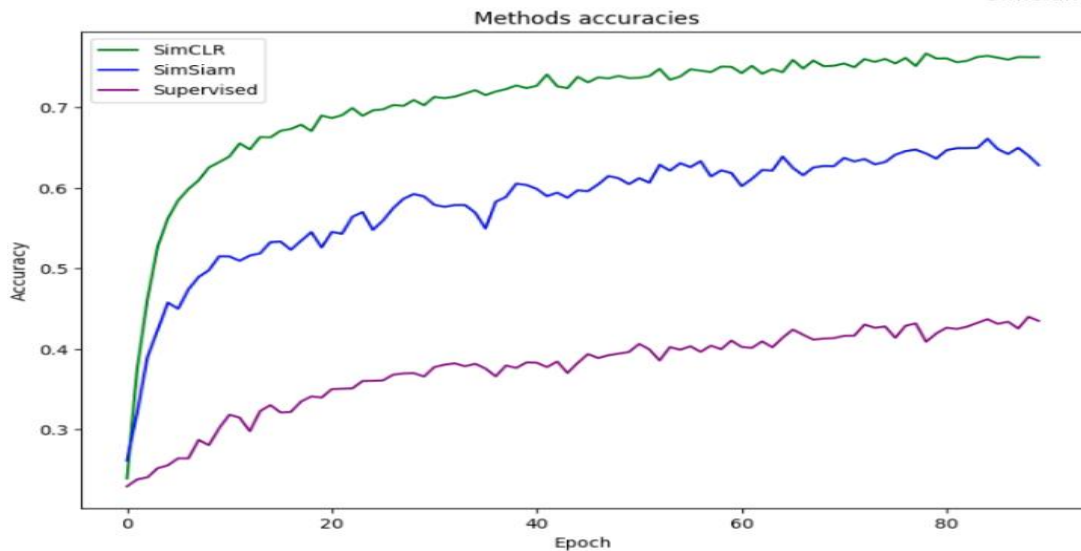


Figure 4.8 Transfer learning accuracy comparison

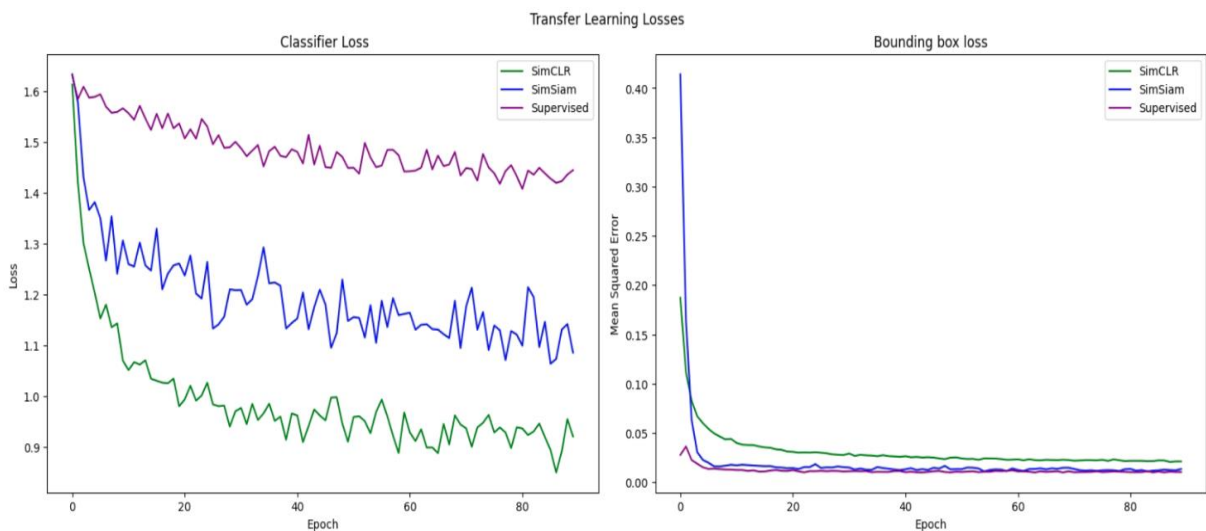


Figure 4.9 Transfer learning loss comparison (right: bounding box, left: classifier)

The first graph, which presents the accuracy of the three methods, shows that the supervised method had the lowest accuracy, falling in the 40s range. This suggests that the supervised method was less effective at correctly classifying the vehicle types. SimSiam, a self-supervised learning method, outperformed the supervised method, indicating that it was

more successful in learning useful representations for the task. However, SimCLR had the highest accuracy, demonstrating its superior performance in the vehicle-type detection task.

The second graph presents two types of loss: classification loss and bounding box mean squared error (MSE). In terms of classification loss, the supervised method had the highest loss, followed by SimSiam, and then SimCLR. This suggests that SimCLR was most effective at minimizing classification errors, while the supervised method struggled the most.

The right side of the second graph shows the bounding box MSE, which is a measure of the model’s performance in predicting the location of the vehicles. Interestingly, SimCLR, which had the highest accuracy and lowest classification loss, had the highest bounding box MSE. This suggests that while SimCLR was effective at classifying the vehicles, it struggled with accurately predicting their locations. The supervised method had the lowest bounding box MSE, indicating that it was most successful at this aspect of the task. However, as the number of epochs increased, the difference in bounding box MSE between the supervised method and SimSiam diminished, implying that SimSiam was improving in its ability to predict vehicle locations over time.

For further understanding of the methods’ performances, the table below provides further metrics to properly evaluate the vehicle-type detection task:

Method	accuracy	precision	recall	F1 score	mAP_{50}
Supervised	41.2 %	43.36 %	39.27 %	39.09 %	46.86 %
SimSiam	65.55 %	65.77 %	64.24 %	64.256 %	65.13%
SimCLR	71.85 %	71.22 %	70.99 %	71.01 %	50.73 %

Table 4.3 Object detection scores on test set

The table above demonstrates comparisons among various metrics, in terms of accuracy, precision, recall, and F1 score, SimCLR outperformed both SimSiam and the supervised method. These metrics all measure different aspects of the model's ability to correctly classify instances, and SimCLR's superior performance suggests that it was most effective at this aspect of the task. SimSiam, while not as effective as SimCLR, still outperformed the supervised method, indicating the potential benefits of self-supervised learning methods for this task.

However, the results for the mAP_{50} metric, which measures the model's performance at a stricter intersection over union (IoU) threshold, tell a different story. Here, SimSiam was the best performer, outperforming SimCLR by a significant margin of 15%. This suggests

that while SimCLR was better at classifying instances overall, SimSiam was more effective at precisely localizing the objects when a higher IoU threshold was required. This could be due to differences in the way the two models learn representations of the data, with SimSiam perhaps learning representations that are more effective for precise localization.

The supervised method was the least effective across all metrics, further highlighting the potential advantages of self-supervised learning methods for this task.

4.4.2 Detailed Transfer Learning Visualizations

This section provides the specifics of the transfer learning process, a critical phase in the self-supervised learning pipeline, providing a more detailed view of the performance of the SimSiam and SimCLR models during this phase.

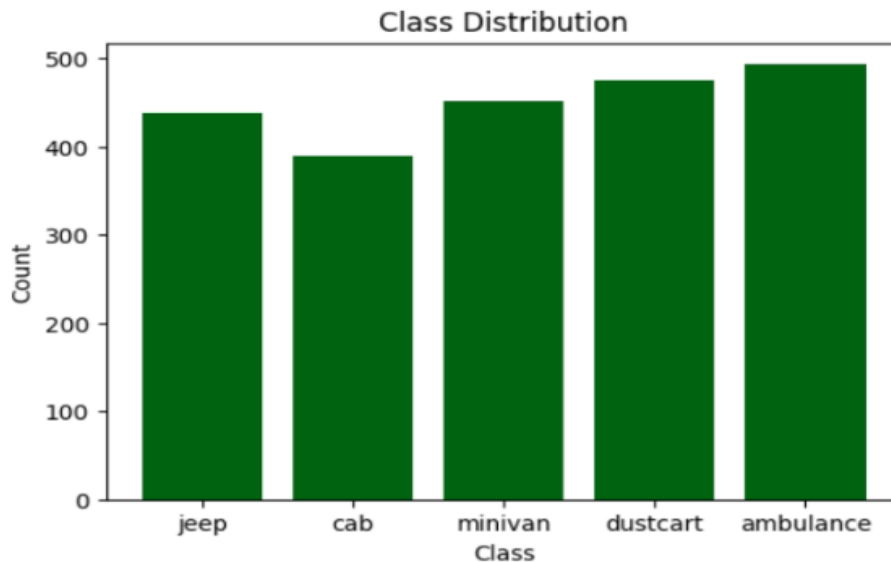


Figure 4.10 ImageNet_subset_II class distribution

A general analysis was done on the ImageNet_subset_II dataset, which includes five classes: *jeep*, *cab*, *minivan*, *dustcart*, and *ambulance*. The distribution of images across these classes is presented in Figure 4.10, a bar chart that provides a clear visual representation of the dataset's composition. The dataset underwent further analysis, with 173 bad files discarded, leaving 2248 valid image cases. These valid cases were then split into 1798 training images and 238 test images.

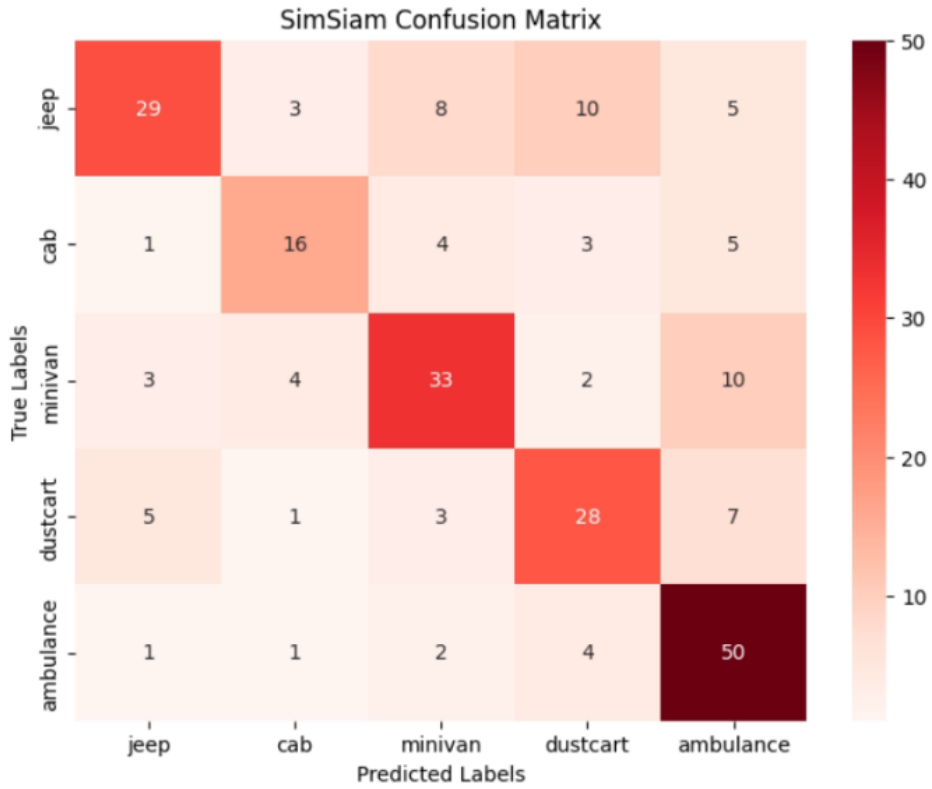


Figure 4.11 SimSiam classification heatmap

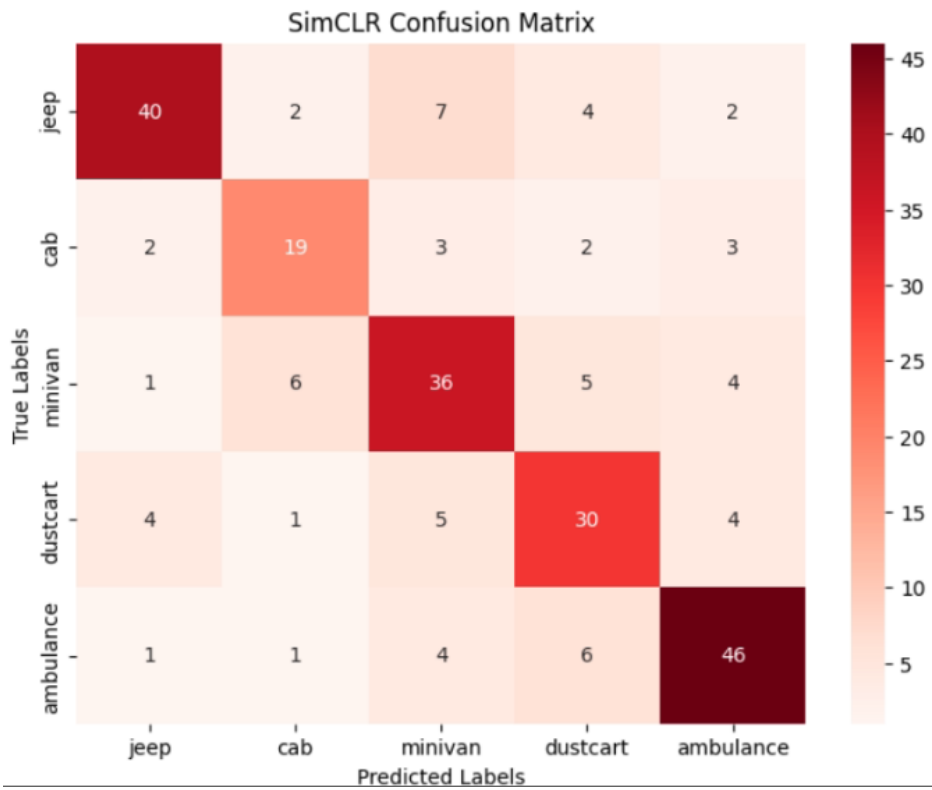


Figure 4.12 SimCLR classification heatmap

The two figures above show classification heatmap for the self-supervised learning methods. Heatmaps are a powerful tool for visualizing data and can provide valuable insights into the behaviour of each self-supervised learning method. In this case, the heatmap shows that SimCLR outperforms SimSiam in the classification task, indicating that SimCLR's learned representations may be more effective for this task.

Although the two heatmaps seem visually indistinguishable, especially in the diagonals, SimSiam was having difficulties in precisely recognizing the “Jeep” class, making it lag in the overall precision score. It is obvious also that “Ambulance” class representations were the best learned by both models.



Figure 4.13 Visualization of bounding boxes with different colors (blue: ground truth, green: predicted box with true label, red: predicted box with false label)

After training the added classifier and regressor for object detection, a visualization of the detected objects has been performed on a few test images to understand how the system is performing as illustrated above. Additionally, Intersection over Union (IoU) is demonstrated on each image with its actual and predicted bounding box, showing a green box when the predicted class matches the actual and red when it doesn't.

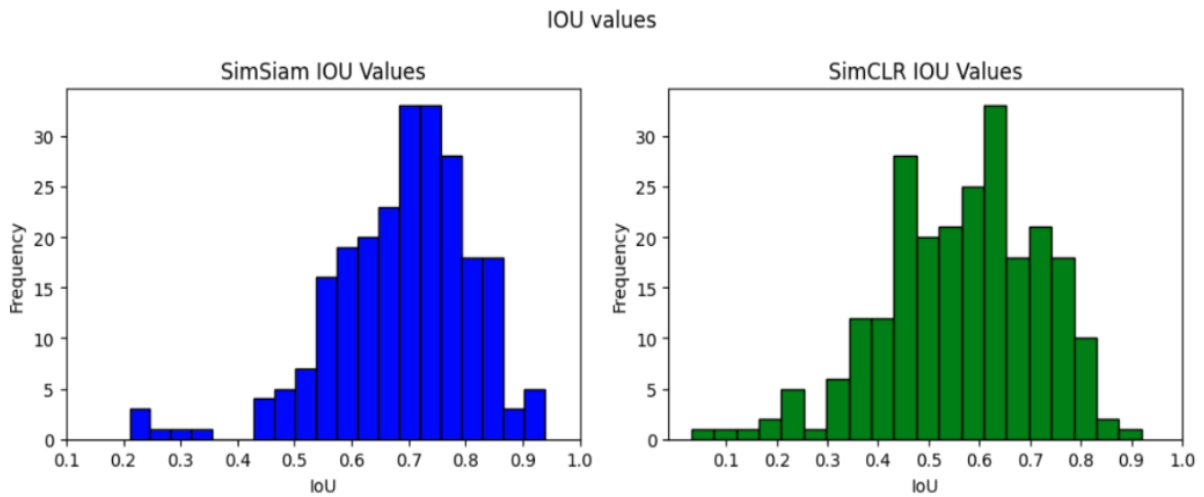


Figure 4.14 IoU values distribution comparison

The distribution of Intersection over Union (IoU) values is also presented in the histogram above. IoU is a common metric for evaluating the quality of object detection models, as it measures the overlap between the predicted bounding box and the ground truth box. The histogram shows that SimSiam performs well in precisely locating the vehicle in the image, providing further evidence of the effectiveness of its learned representations.

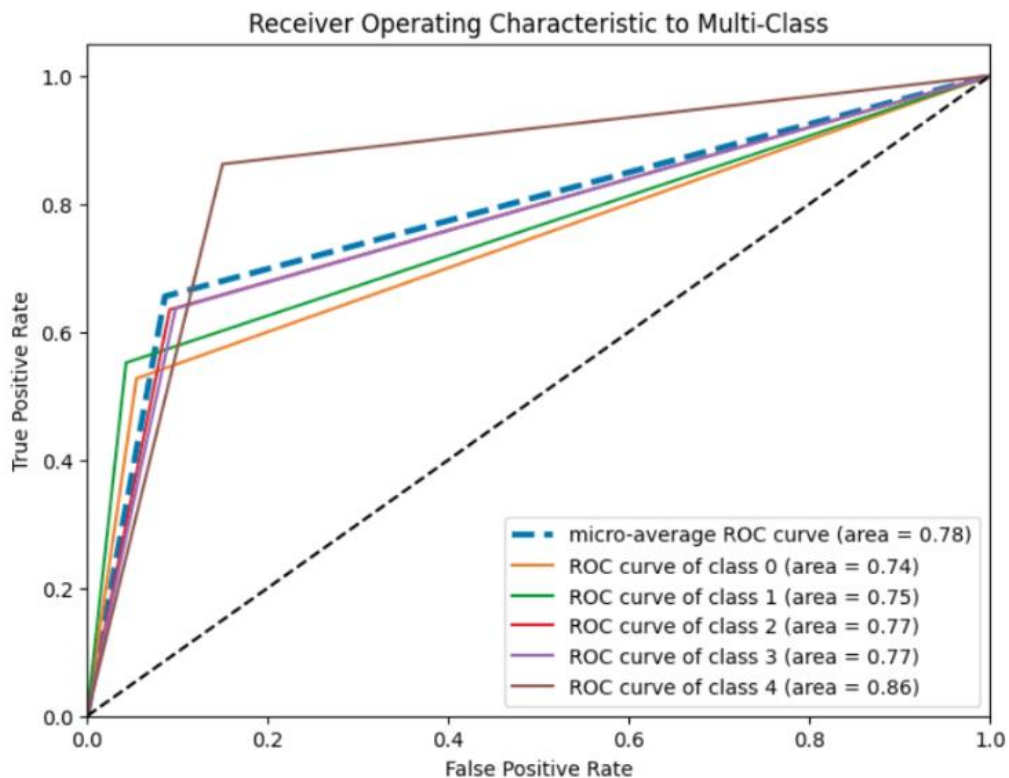


Figure 4.15 SimSiam AUC- ROC

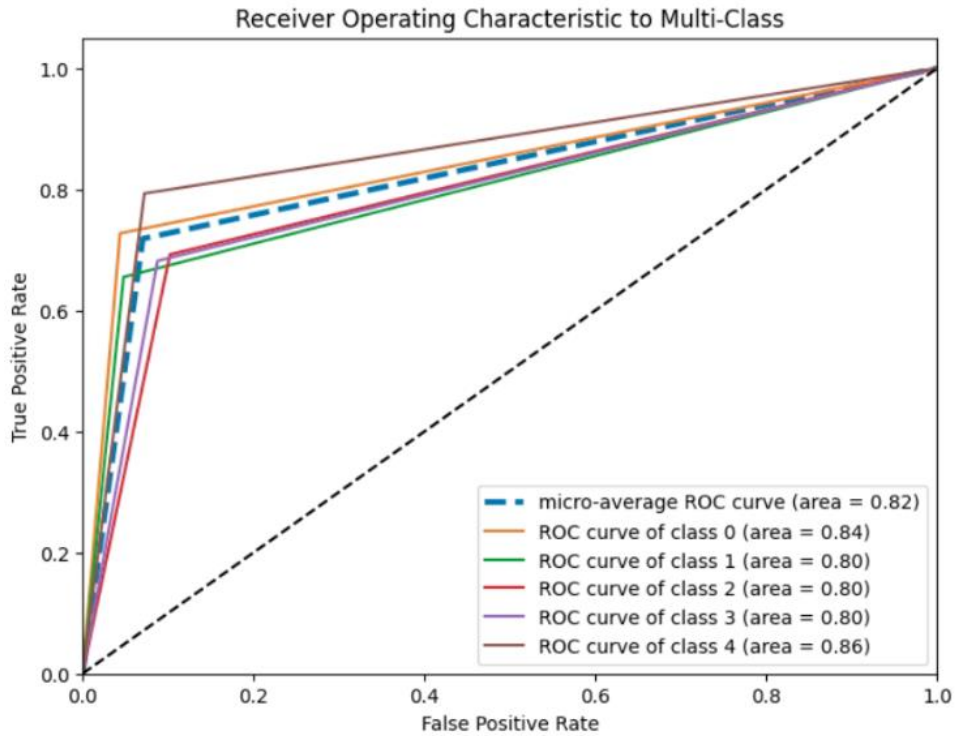


Figure 4.16 SimCLR AUC- ROC

Additionally, the Area Under the Receiver Operating Characteristic (AUC-ROC) curves for each self-supervised learning method was added, including the supervised ResNet18 method. The area under the curve (AUC) captures the performance of the model at various threshold levels and provides a comprehensive assessment of that performance in a single value. The AUC value ranges from 0 to 1, with 1 denoting perfect classification and 0.5 denoting random guessing. The AUC-ROC curves provide insights into the models' behaviour, particularly their trade-off between sensitivity (true positive rate) and specificity (false positive rate).

Here SimSiam average sensitivity was 0.66 while specificity was 0.91, on the other hand, SimCLR scored 0.72 sensitivity and 0.93 specificity. The blue dashed line represents the average curve for all classes. The classes from 0 to 4 are jeep, cab, minivan, dustcart, and ambulance respectively. From the area under the curve of each class, both SSL methods have the same area for the ambulance class, i.e., both methods are equally able to distinguish whether the class is an ambulance or no not. However, with regard to other classes, SimCLR is more capable.

4.5 Discussion of Results

The comparison results revealed that self-supervised learning methods, specifically SimSiam and SimCLR, outperform traditional supervised methods in vehicle-type detection tasks. This is a significant finding as it underscores the potential of self-supervised learning in scenarios where labelled data is scarce or expensive to acquire, reaching very good results even when using 10% of training data compared to the supervised method.

SimCLR, despite having a higher pretraining loss, excelled in classification metrics such as accuracy, precision, recall, and F1 score. This shows that SimCLR's contrastive learning approach is effective in learning robust representations that generalize well to the downstream task. However, when it comes to precise object localization, as measured by mAP_{50} , SimSiam outperforms SimCLR. This indicates that SimSiam's approach, which does not rely on negative pairs, might be more effective in learning representations that are useful for precise localization tasks.

The previous sections provided a profound dive into the implementation details, shedding light on the intricacies of the methods and the nuances of their performance. The pretraining graphs elucidated the trajectory of learning for both SimSiam and SimCLR. While SimSiam was promising during pretraining, the true quality was tested during the linear evaluation phase.

The t-SNE cluster visualizations offered a visual representation of the feature spaces learned by the models. These clusters provided insights into the separability and distinctiveness of the representations, with tighter, well-separated clusters indicating more discriminative features. The cluster visualizations indicated in general that both methods learned representations very well, which is a good indication that pretraining was making positive progress, but it's hard to tell how good it is unless it is transferred to a downstream task – vehicle-type detection in this case – to comprehend their generalizability quality.

The transfer learning graphs further expanded on the models' adaptability and showcased the versatility of the learned representations. The performance of both SimSiam and SimCLR in this phase was indicative of their potential, each one of them surpasses the other in certain areas. SimSiam was outstanding in the localization task while SimCLR excelled in classification, this showcase that the quality of representations learned is more important than just reaching a lower pretraining loss faster.

The classification heatmap, on the other hand, provided a significant representation of the models' classification decisions, allowing for a broader understanding of areas where the model excels or falters. SimSiam was having difficulties in precisely recognizing the “Jeep” class, failing to accurately detect this class almost by 50%. In other classes, the difference was somehow tiny between the two SSL methods except for the *Ambulance* class where SimSiam has shown some competition and even surpassed SimCLR slightly, but this came with a penalty of making double false positive detections compared to SimCLR.

Lastly, the AUC-ROC curve and classification heatmap provided a holistic view of the models' classification prowess. The AUC-ROC curve, highlighted the true positive rate against the false positive rate in which both methods are equally powerful when it comes to the detection of *Ambulance* class, but unlike SimSiam, SimCLR remained consistent and maintained a high score on the other classes. This curve supports the interpretation of *Ambulance* class from the heatmap, showing that despite the better sensitivity in *Ambulance* class of SimSiam over SimCLR (86% against 79%), the specificity was rather low with 85% against 92.8%.

In light of these findings and the detailed insights from all statistics and visualizations, it's evident that self-supervised learning, especially the methods explored in this study, holds significant promise for the field of machine learning and autonomous driving. The pretraining process of the SSL models is a crucial step to push the technological advancement level further in this area, followed by a linear evaluation to validate the quality of pretraining. Moreover, to validate whether the model has a better generalizability and tackling the domain gap, transfer learning experiments are essential. Additionally, utilizing such SSL models for transfer learning to other implementations in autonomous driving scenarios besides vehicle-type detection would definitely boost the efficiency of that industry.

Overall, those various metrics in vehicle-type detection are vital for spotting exactly the small areas where the SSL model fails or does not perform as expected. This would allow faster and more efficient troubleshooting processes whether it was in data preparation, pretraining, or fine-tuning phase, utilizing such technology to the maximum extent, and pushing the productivity upwards.

CHAPTER 5: CONCLUSION

5.1 Summary

The study embarked on a comprehensive exploration of self-supervised learning methods, particularly SimSiam and SimCLR, and their applicability in the realm of vehicle-type detection for autonomous driving systems. The results from Chapter Four underscored the potential of these methods, with both outperforming traditional supervised learning in various metrics.

These findings have significant implications for autonomous driving systems. Vehicle-type detection is a crucial component of these systems, contributing to safer and more efficient navigation. Knowing the type of vehicle can inform the autonomous system's decision-making process in several ways:

1. **Trajectory Planning:** Different types of vehicles have different driving behaviours and characteristics. For instance, trucks typically move slower and have larger turning radii than smaller vehicles. By identifying the type of vehicle, the autonomous system can better predict its future positions and plan its trajectory accordingly.
2. **Traffic Flow Analysis:** Understanding the composition of traffic, such as the proportion of cars, trucks, and motorcycles, can help the autonomous system anticipate the general flow of traffic and adjust its driving strategy.
3. **Priority Vehicles:** Identifying priority vehicles like ambulances or fire trucks is crucial for autonomous driving systems. These vehicles have the right of way in all circumstances, and the system must be able to recognize them and yield appropriately.
4. **Safety Precautions:** Certain types of vehicles, like trucks or buses, may require extra safety precautions due to their size and potential blind spots. Recognizing these vehicles can prompt the autonomous system to maintain a safe distance.

Beyond vehicle-type detection, the self-supervised learning models and transfer learning approach used in this study could be applied to other object detection tasks within the autonomous driving context. For instance, they could be used for pedestrian detection, traffic sign recognition, or lane detection. The ability of these models to learn useful

representations from unlabelled data could be particularly valuable in these tasks, where obtaining large amounts of labelled data can be challenging.

In conclusion, this study highlights the potential of self-supervised learning methods do not only validate the efficacy of these methods, but also pave the way for their adoption in broader applications within the autonomous driving ecosystem.

5.2 Challenges

The journey to the successful outcomes of this study was not a straightforward path. It was marked by numerous challenges that required innovative problem-solving and perseverance. These obstacles were not only integral to the research process but also provided valuable insights that enriched the study. Some of the key challenges encountered during the study are included in the table below:

Challenge	Description	How it was tackled
Dataset	Selecting a dataset was a tough task as available datasets do not have the luxury of subdividing vehicles into various classes	Through rigorous inspection, ImageNet was selected, it has various 20 vehicle classes while others barely have 3
Data pre-processing	Data were not very clean, it contained irrelevant, mislabelled images, tiny bounding boxes, and notable class imbalance	By handling inconsistent data programmatically and adding augmentation when needed to compensate for deficit
SSL Model tuning	Applying hyperparameters did not always yield optimal performance, downgrading data size and resolution due to computational restrictions has influenced performance	Through various experiments, methods were slightly adjusted to ensure the best possible performance for such a customized dataset
Deep learning lab devices	All images of Ubuntu had problems in verifying CUDA installations, preventing GPU utilization in training	A clean image was set up from scratch with admin rights to properly use GPU

Transfer learning performance	In the pursuit of better results, data augmentation and grid search were used to get the best results, but the outcome was not good	Data augmentation was removed because the actual bounding boxes were not perfectly aligned, and augmenting bounding box has additional error margin, this confuses the model instead of helping it
-------------------------------	---	--

5.3 Limitations

Despite the significant findings and contributions of this study, it's important to acknowledge the limitations that may have influenced the outcomes and our understanding of the results. These limitations offer opportunities for future research to build upon this work. Some of the key limitations include:

1. **Limited Dataset:** While this ImageNet subset was carefully chosen and preprocessed, it may not fully represent the diversity and complexity of real-world scenarios in autonomous driving. This limitation could impact the generalizability of the findings since the number of images per class was low, and bounding box annotations were even lower.
2. **Hyperparameter Selection:** Although authors have put effort to optimize the hyperparameters of SSL models, there is no standardized way to select the best hyperparameters based on dataset quantity and quality, except for some rare occasions.
3. **Computational Constraints:** Despite the use of high-performance workstations, computational resources were still a very limiting factor. More extensive experiments, such as larger models or longer training times, might have been possible with more computational resources.

These limitations, while constraining, provide valuable directions for future research. By addressing these limitations, future studies can further advance our understanding of self-supervised learning and its applications in autonomous driving.

5.4 Recommendations

The findings of this study, while promising, also highlight areas for further exploration and improvement. The following recommendations are proposed for future work in this area:

1. **Expanding the Dataset:** To enhance the generalizability of the models, future studies could incorporate more diverse and complex datasets that better represent real-world scenarios in autonomous driving. Or create their own dataset for specific domain studies.
2. **Exploring Other Models:** While SimSiam and SimCLR were the focus of this study, many other SSL models could be explored in future research. These models could potentially yield better performance.
3. **Leveraging More Computational Resources:** If more computational resources are available, future studies could explore larger models, longer training times, or more extensive hyperparameter tuning, which could lead to improved results.
4. **Incorporating Additional Evaluation Metrics:** Future research could incorporate additional evaluation metrics to provide a more comprehensive assessment of the models' performance.
5. **Refining Transfer Learning Implementation:** Future work could explore different approaches to transfer learning, such as fine-tuning more layers to potentially improve the performance of the downstream task.
6. **Fixed Architecture:** This work – as well as other SSL works – used a fixed architecture (ResNet) for the models. Other architectures might yield different results and offer more insights into the behaviour of SSL models.

In conclusion, this study has made significant strides in exploring the potential of self-supervised learning for vehicle-type detection in autonomous driving. However, there is still much to learn and many opportunities for further research in this exciting field. By building on the findings of this study and addressing its limitations, future research can continue to advance our understanding of self-supervised learning and its applications in autonomous driving, ultimately contributing to safer and more efficient autonomous driving systems.

REFERENCES

- [1] Bachman, P., Hjelm, R. D., & Buchwalter, W. (2019). Learning Representations by Maximizing Mutual Information Across Views. In *Neural Information Processing Systems* (Vol. 32, pp. 15509–15519). <https://papers.nips.cc/paper/2019/file/ddf354219aac374f1d40b7e760ee5bb7-Paper.pdf>
- [2] Bardes, A., Ponce, J., & LeCun, Y. (2022). VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning. In *HAL (Le Centre pour la Communication Scientifique Directe)*. French National Centre for Scientific Research. <https://hal.inria.fr/hal-03541297>
- [3] Berg, T., Liu, J., Lee, S. H., Alexander, M. L., Jacobs, D. R., & Belhumeur, P. N. (2014). *BirdsNap: Large-scale fine-grained visual categorization of birds*. <https://doi.org/10.1109/cvpr.2014.259>
- [4] Boser, B. E., Guyon, I., & Vapnik, V. (1992). A training algorithm for optimal margin classifiers. *SpringerLink*. <https://doi.org/10.1145/130385.130401>
- [5] Bromley, J., Guyon, I., LeCun, Y., Sackinger, E., & Shah, R. (1994). SIGNATURE VERIFICATION USING A “SIAMESE” TIME DELAY NEURAL NETWORK. In *Series in machine perception and artificial intelligence* (pp. 25–44). World Scientific. https://doi.org/10.1142/9789812797926_0003
- [6] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-End Object Detection with Transformers. arXiv (Cornell University). <https://arxiv.org/pdf/2005.12872.pdf>
- [7] Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., & Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. *HAL (Le Centre Pour La Communication Scientifique Directe)*. <https://hal.science/hal-02883765>
- [8] Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. E. (2020). A Simple Framework for Contrastive Learning of Visual Representations. In *International Conference on Machine Learning* (Vol. 1, pp. 1597–1607). <http://proceedings.mlr.press/v119/chen20j/chen20j.pdf>

- [9] Chen, X., Fan, H., Girshick, R. G., Kaiming He, & He, K. (2020, March 9). *Improved Baselines with Momentum Contrastive Learning*. arXiv.org. <https://arxiv.org/abs/2003.04297v1>
- [10] Chen, X., & He, K. (2021). *Exploring Simple Siamese Representation Learning*. <https://doi.org/10.1109/cvpr46437.2021.01549>
- [11] Chen, X., Xie, S., & He, K. (2021). An Empirical Study of Training Self-Supervised Vision Transformers. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. <https://doi.org/10.1109/iccv48922.2021.00950>
- [12] Cuturi, M. (2013). Sinkhorn Distances: Lightspeed computation of optimal transportation distances. arXiv (Cornell University). <http://export.arxiv.org/pdf/1306.0895>
- [13] Dalal, N. S., & Triggs, B. (2005). *Histograms of Oriented Gradients for Human Detection*. <https://doi.org/10.1109/cvpr.2005.177>
- [14] Deng, J., Dong, W., Socher, R., Li, L., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/cvpr.2009.5206848>
- [15] Doersch, C., Gupta, A., & Efros, A. A. (2015). *Unsupervised Visual Representation Learning by Context Prediction*. <https://doi.org/10.1109/iccv.2015.167>
- [16] Everingham, M., Van Gool, L., Williams, C., Winn, J., & Zisserman, A. (2009). The Pascal Visual Object Classes (VOC) challenge. *International Journal of Computer Vision*, 88(2), 303–338. <https://doi.org/10.1007/s11263-009-0275-4>
- [17] Fang, Y., Dong, L., Bao, H., Wang, X., & Wei, F. (2022). Corrupted Image Modeling for Self-Supervised Visual Pre-Training. arXiv (Cornell University). <https://doi.org/10.48550/arxiv.2202.03382>
- [18] Girshick, R. (2015). *Fast R-CNN*. <https://doi.org/10.1109/iccv.2015.169>
- [19] Grill, J., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., Piot, B., Kavukcuoglu, K., Munos, R., & Valko, M. (2020). Bootstrap Your Own Latent: A new approach to self-supervised learning. In *HAL (Le Centre pour la Communication Scientifique Directe)*. French National Centre for Scientific Research. <https://hal.inria.fr/hal-02869787>

- [20] Hadsell, R., Chopra, S., & LeCun, Y. (2006). *Dimensionality Reduction by Learning an Invariant Mapping*. <https://doi.org/10.1109/cvpr.2006.100>
- [21] He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2019, November 13). *Momentum contrast for unsupervised visual representation learning*. arXiv.org. <https://arxiv.org/abs/1911.05722>
- [22] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. In *arXiv (Cornell University)*. Cornell University. <https://arxiv.org/pdf/1703.06870v3>
- [23] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *arXiv (Cornell University)*. <https://doi.org/10.48550/arxiv.1512.03385>
- [24] Hénaff, O. J. (2020). Data-Efficient Image Recognition with Contrastive Predictive Coding. In *International Conference on Machine Learning* (Vol. 1, pp. 4182–4192). <http://proceedings.mlr.press/v119/henaff20a/henaff20a.pdf>
- [25] Hinton, G. E., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. S., Jaitly, N., W, A., Senior, Vanhoucke, V., Nguyen, P. L., Sainath, T. N., & Kingsbury, B. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, 29(6), 82–97. <https://doi.org/10.1109/msp.2012.2205597>
- [26] Krizhevsky, A. (2009). *Learning Multiple Layers of Features from Tiny Images*. <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>
- [27] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In *Neural Information Processing Systems* (Vol. 25, pp. 1097–1105). http://books.nips.cc/papers/files/nips25/NIPS2012_0534.pdf
- [28] Larsson, G., Maire, M., & Shakhnarovich, G. (2016). Learning Representations for Automatic Colorization. In *Lecture Notes in Computer Science* (pp. 577–593). Springer Science+Business Media. https://doi.org/10.1007/978-3-319-46493-0_35
- [29] Lee, H. Y., Huang, J., Singh, M., & Yang, M. (2017). *Unsupervised Representation Learning by Sorting Sequences*. <https://doi.org/10.1109/iccv.2017.79>
- [30] Li, A. C., Efros, A. A., & Pathak, D. (2022). Understanding collapse in non-contrastive Siamese representation learning. In *Lecture Notes in Computer Science* (pp. 490–505). https://doi.org/10.1007/978-3-031-19821-2_28

- [31] Li, H., Li, Y., Zhang, G., Liu, R., Huang, H., Zhu, Q., & Tao, C. (2022). Global and Local Contrastive Self-Supervised Learning for Semantic Segmentation of HR Remote Sensing Images. *IEEE Transactions on Geoscience and Remote Sensing*, 60, 1–14. <https://doi.org/10.1109/tgrs.2022.3147513>
- [32] Lim, R., & Guntoro, A. T. (2001). Type of Vehicle Recognition Using Template Matching Method. *Petra*. https://www.academia.edu/476356/Type_of_Vehicle_Recognition_Using_Template_Matching_Method
- [33] Lin, T., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft Coco: Common Objects in context. In *Lecture Notes in Computer Science* (pp. 740–755). https://doi.org/10.1007/978-3-319-10602-1_48
- [34] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S. M., Fu, C., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In *Lecture Notes in Computer Science* (pp. 21–37). Springer Science+Business Media. https://doi.org/10.1007/978-3-319-46448-0_2
- [35] Misra, I., & Van Der Maaten, L. (2020). *Self-Supervised Learning of Pretext-Invariant Representations*. <https://doi.org/10.1109/cvpr42600.2020.00674>
- [36] Misra, I., Zitnick, C. L., & Hebert, M. (2016). Shuffle and Learn: Unsupervised Learning Using Temporal Order Verification. In *Lecture Notes in Computer Science* (pp. 527–544). Springer Science+Business Media. https://doi.org/10.1007/978-3-319-46448-0_32
- [37] Munir, F., Azam, S., & Jeon, M. (2021). SSTN: Self-Supervised Domain Adaptation Thermal Object Detection for Autonomous Driving. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. <https://doi.org/10.1109/iros51168.2021.9636353>
- [38] Noroozi, M., & Favaro, P. (2016). Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles. In *Lecture Notes in Computer Science* (pp. 69–84). Springer Science+Business Media. https://doi.org/10.1007/978-3-319-46466-4_5
- [39] Pathak, D., Krähenbühl, P., Donahue, J., Darrell, T., & Efros, A. A. (2016). *Context Encoders: Feature Learning by Inpainting*. <https://doi.org/10.1109/cvpr.2016.278>
- [40] Petrovskaya, A., & Thrun, S. (2009). Model based vehicle detection and tracking for autonomous urban driving. *Autonomous Robots*, 26(2–3), 123–139. <https://doi.org/10.1007/s10514-009-9115-1>

- [41] Pototzky, D., Sultan, A., Kirschner, M., & Schmidt-Thieme, L. (2021). Self-supervised Learning for Object Detection in Autonomous Driving. In *Lecture Notes in Computer Science* (pp. 484–497). Springer Science+Business Media. https://doi.org/10.1007/978-3-030-92659-5_31
- [42] Redmon, J., Divvala, S. K., Girshick, R., & Farhadi, A. (2016). *You Only Look Once: Unified, Real-Time Object Detection*. <https://doi.org/10.1109/cvpr.2016.91>
- [43] Redmon, J., & Farhadi, A. (2018, April 8). *YOLOv3: An Incremental Improvement*. arXiv.org. <https://arxiv.org/abs/1804.02767>
- [44] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: towards real-time object detection with region proposal networks. In *Neural Information Processing Systems* (Vol. 28, pp. 91–99). <https://papers.nips.cc/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf>
- [45] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. S., Berg, A. C., & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3), 211–252. <https://doi.org/10.1007/s11263-015-0816-y>
- [46] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). *FaceNet: A unified embedding for face recognition and clustering*. <https://doi.org/10.1109/cvpr.2015.7298682>
- [47] Siam, M. (2021, May 7). *Video Class Agnostic Segmentation with Contrastive Learning for Autonomous Driving*. arXiv.org. <https://arxiv.org/abs/2105.03533>
- [48] Sivaraman, S., & Trivedi, M. M. (2013). Looking at Vehicles on the Road: A Survey of Vision-Based Vehicle Detection, Tracking, and Behavior Analysis. *IEEE Transactions on Intelligent Transportation Systems*, 14(4), 1773–1795. <https://doi.org/10.1109/tits.2013.2266661>
- [49] Tan, M., Pang, R., & Le, Q. V. (2020). *EfficientDet: Scalable and Efficient Object Detection*. <https://doi.org/10.1109/cvpr42600.2020.01079>
- [50] Tanaka, T., Masumura, R., Sato, H., Ihuri, M., Matsuura, K., Ashihara, T., & Moriya, T. (2022). Domain Adversarial Self-Supervised Speech Representation Learning for Improving Unknown Domain Downstream Tasks. In *Interspeech 2022*. <https://doi.org/10.21437/interspeech.2022-11414>

- [51] Tian, Y., Chen, X., & Ganguli, S. (2021). Understanding self-supervised learning dynamics without contrastive pairs. In *International Conference on Machine Learning* (pp. 10268–10278). <http://proceedings.mlr.press/v139/tian21a/tian21a.pdf>
- [52] Van Den Oord, A., Li, Y., & Vinyals, O. (2018, July 10). *Representation Learning with Contrastive Predictive Coding*. arXiv.org. <https://arxiv.org/abs/1807.03748>
- [53] Van Horn, G., Mac Aodha, O., Yang, S., Chen, Y., Sun, C., Shepard, A., Adam, H., Perona, P., & Belongie, S. (2018). The iNaturalist Species Classification and Detection Dataset. arXiv (Cornell University). <https://doi.org/10.48550/arxiv.1707.06642>
- [54] Wang, X., & Gupta, A. (2015). Unsupervised Learning of Visual Representations Using Videos. *University of Toronto*. <https://doi.org/10.1109/iccv.2015.320>
- [55] Wang, Y., Zhuo, W., Li, Y., Wang, Z., Ju, Q., & Zhu, W. (2022, February 24). *Fully Self-Supervised Learning for Semantic Segmentation*. arXiv.org. <https://arxiv.org/abs/2202.11981>
- [56] Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., & Torralba, A. (2010). *SUN database: Large-scale scene recognition from abbey to zoo*. <https://doi.org/10.1109/cvpr.2010.5539970>
- [57] Xue, F., Zhuo, G., Huang, Z., Fu, W., Wu, Z., & Ang, M. H. (2020). *Toward Hierarchical Self-Supervised Monocular Absolute Depth Estimation for Autonomous Driving Applications*. <https://doi.org/10.1109/iros45743.2020.9340802>
- [58] Yi, J. S. K., Seo, M., Park, J., & Choi, D. (2022). PT4AL: Using Self-supervised Pretext Tasks for Active Learning. In *SpringerLink* (pp. 596–612). https://doi.org/10.1007/978-3-031-19809-0_34
- [59] You, Y., Ginsburg, B., & Gitman, I. (2017, August 13). *Large Batch Training of Convolutional Networks*. arXiv.org. <https://arxiv.org/abs/1708.03888>
- [60] Zaiem, S., Parcollet, T., Essid, S., & Heba, A. (2022). Pretext tasks selection for multitask self-supervised audio representation learning. *IEEE Journal of Selected Topics in Signal Processing*, 16(6), 1439–1453. <https://doi.org/10.1109/jstsp.2022.3195430>
- [61] Zhang, R., Isola, P., & Efros, A. A. (2016). Colorful Image Colorization. In *Lecture Notes in Computer Science* (pp. 649–666). Springer Science+Business Media. https://doi.org/10.1007/978-3-319-46487-9_40

- [62] Zheng, L., Guha, N., Anderson, B. R., Henderson, P., & Ho, D. E. (2021, April 18). *When does pretraining help? Assessing Self-supervised learning for Law and the Casehold Dataset*. arXiv.org. <https://arxiv.org/abs/2104.08671>
- [63] Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., & Oliva, A. (2014). Learning Deep Features for Scene Recognition using Places Database. *NeurIPS Proceedings*, 27, 487–495.
- [64] Zhou, Y., Duan, H., Rao, A., Su, B., & Wang, J. (2023, February 17). *Self-supervised Action Representation Learning from Partial Spatio-Temporal Skeleton Sequences*. arXiv.org. <https://arxiv.org/abs/2302.09018>
- [65] Zhuo, Z., Wang, Y., Ma, J., & Wang, Y. (2023). Towards a unified theoretical understanding of non-contrastive learning via rank differential mechanism. arXiv (Cornell University). <https://doi.org/10.48550/arxiv.2303.02387>

APPENDIX

This appendix elaborates on a variety of experimentations in order to provide further insights on SSL methods, whether on different backbone architectures, dataset sizes, or image resolutions. The experiments are more related to understanding the behaviour of these methods under different conditions rather than directly contributing to the results of this study.

Appendix.A CIFAR Pretraining

Appendix A provides a different perspective on the pretraining process, this time using the CIFAR10 dataset, another data with a smaller image resolution but more diversity of 6000 images per class. The works of SimSiam and SimCLR had experimented on CIFAR10 also, so the hyperparameters were kept the same here.

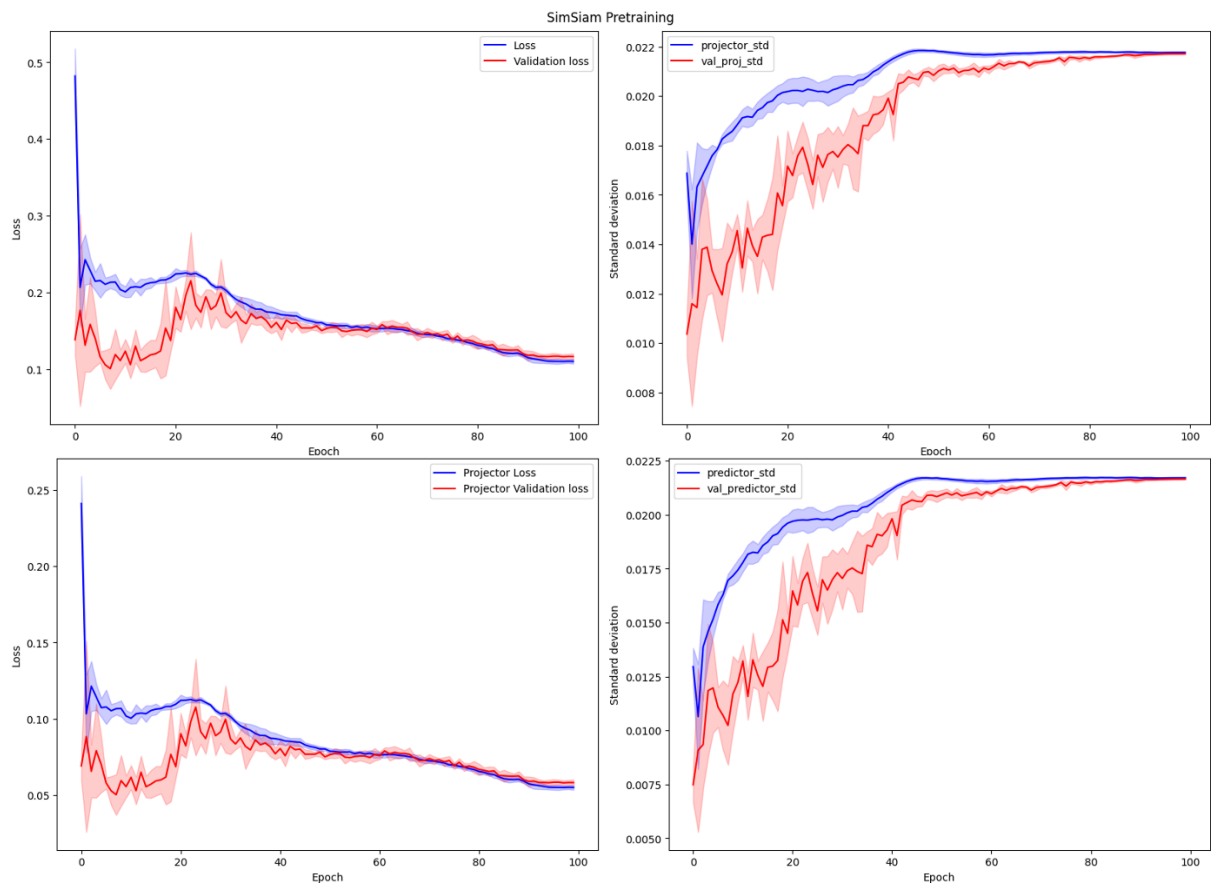


Fig A.1 SimSiam CIFAR pretraining

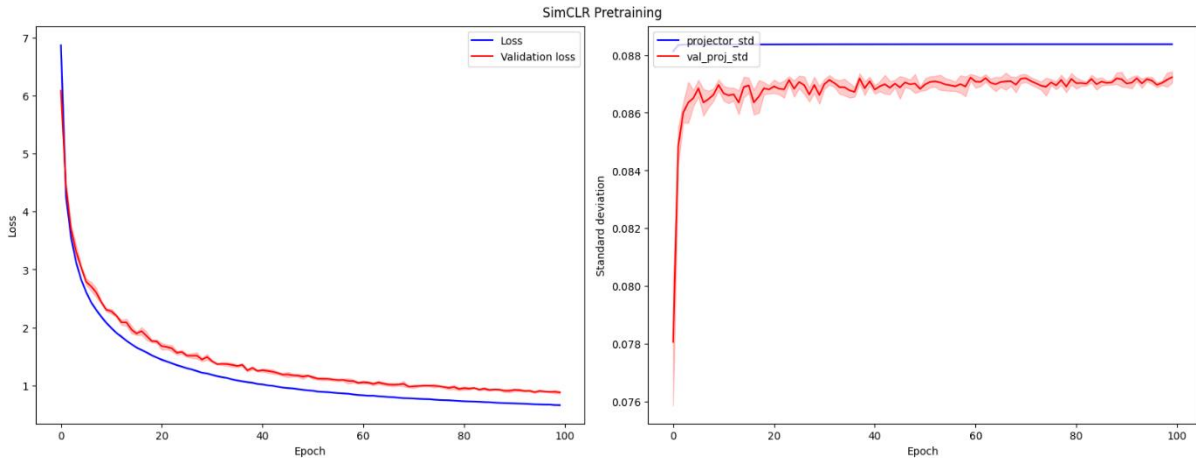


Fig A.2 SimCLR CIFAR pretraining

In all the presented graphs, the shaded regions depict the range of values derived from five cross-validation trials, while the actual curve signifies the mean value. This graphical representation provides a visual understanding of the variability in the models' performance across different trials.

As seen in Figure A.1, the loss curve of the SimSiam model exhibits an initial period of instability during the first 40 epochs. This fluctuation could be ascribed to the model's adaptation to the data and the learning of initial representations. However, post this initial phase, the loss curve of the SimSiam model stabilizes and eventually achieves a lower loss than the SimCLR model, as shown in Figure A.2. This suggests that despite the early instability, the SimSiam model successfully learns effective representations that minimize the loss.

The projector loss curve for the SimSiam model mirrors the trend of the model loss curve, except that losses were nearly halved. This suggests that the projector, which translates the representations into a lower-dimensional space, is capable of preserving critical information while reducing dimensionality. The graphs depicting the standard deviation of the activations of the final layer in the projector and predictor models offer insights into the diversity of the learned representations. A higher standard deviation signifies more diverse representations, which could potentially enhance performance on downstream tasks.

It may seem from here that one model outperforms the other, but this doesn't necessarily indicate the superior quality of its representations. Further insights into each model's performance can be gleaned from linear evaluation.

Appendix.B CIFAR Linear Evaluation

The assessment of the models incorporated several trials to guarantee the reliability of the outcomes. A bar graph illustrating the mean accuracy across five trials for each model is presented below, with the error bar indicating the standard deviation. This graphical representation provides a straightforward comparison of the overall effectiveness of the Supervised Learning, SimSiam, and SimCLR methodologies.

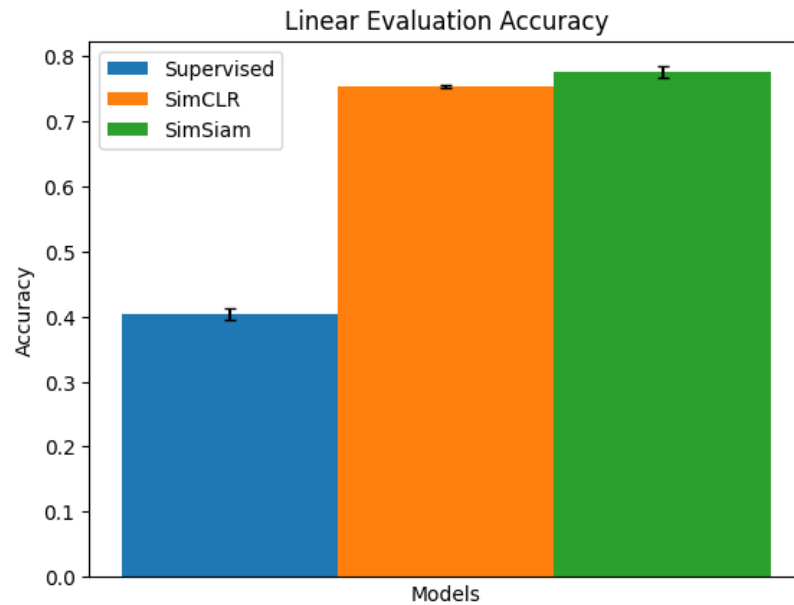


Fig B.1 CIFAR mean accuracy Linear Evaluation

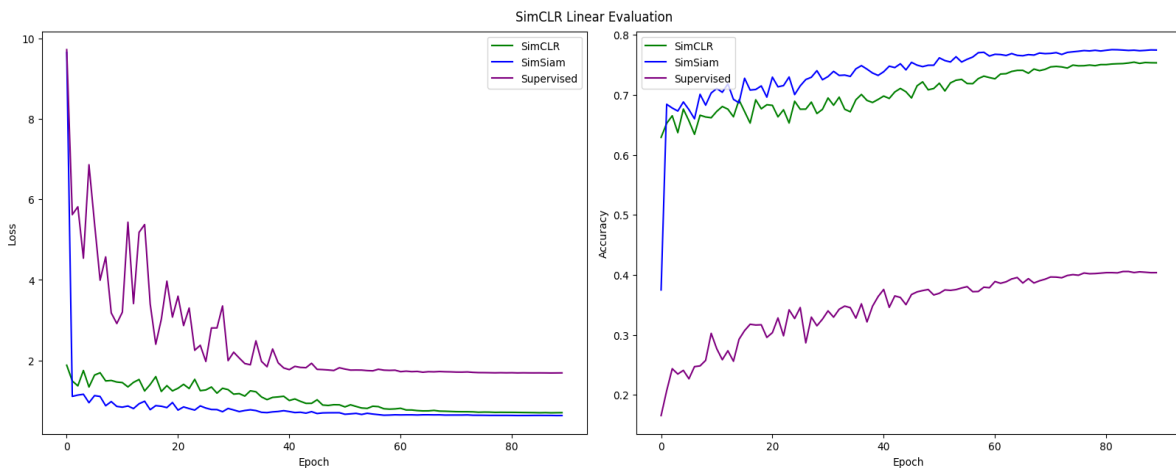


Fig B.2 CIFAR Linear Evaluation graph comparison

The Supervised Learning method – using Resnet18 – showed again low accuracy among the three methods. During the linear evaluation stage, SimSiam this time outperformed SimCLR as expected based on its lower pretraining loss. This could mean that

representations might have generalized as well, opposite to what was observed in the linear evaluation of ImageNet_subset_I.

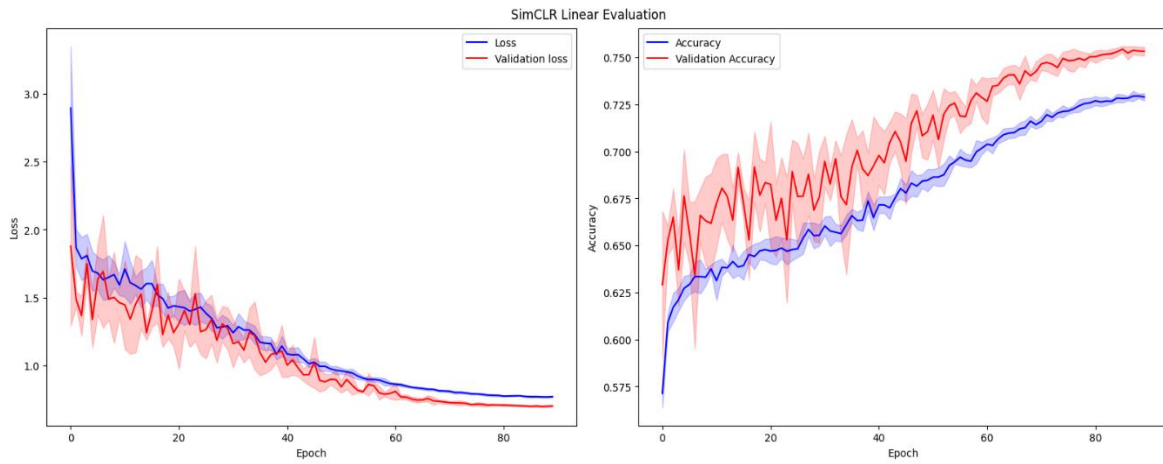


Fig B.3 CIFAR SimCLR linear evaluation

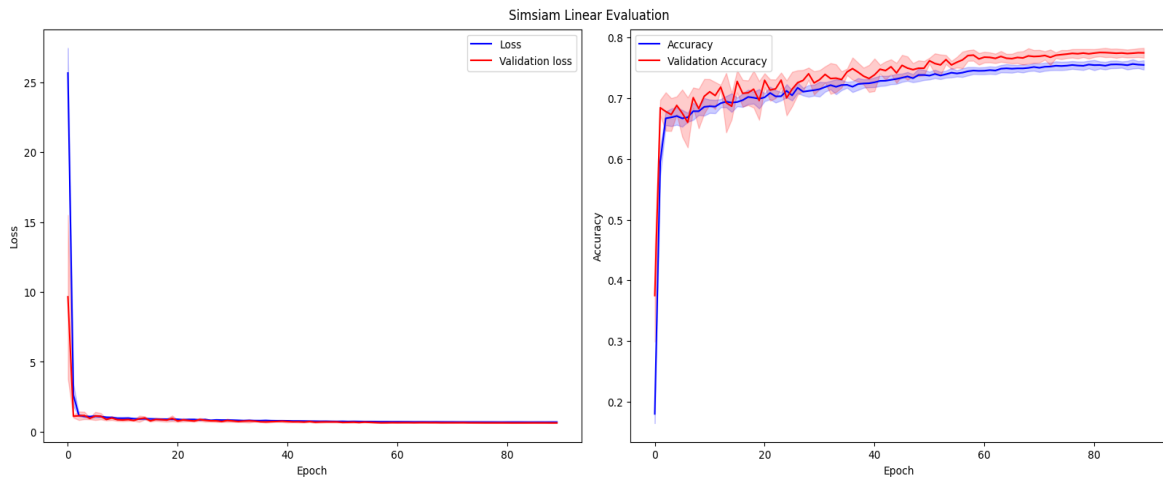


Fig B.4 CIFAR SimSiam linear evaluation

Regarding graph fluctuation in linear evaluation, SimSiam and SimCLR behave oppositely in CIFAR10 compared to what has been observed in ImageNet_subset_I. On the other hand, SimSiam loss remains better in all scenarios, and this time, its superiority was dominant in accuracy as well.

Appendix.C Further ImageNet Experiments

More experiments have been implemented on the default backbone (ResNet50) architecture of most of the current SSL methods. The default backbone was used on the large ImageNet dataset exclusively, but this might not necessarily be the best choice for datasets with smaller resolution or size.

The evaluation of the models included multiple trials to ensure the robustness of the results. A bar plot summarizing the average accuracy across five trials for each model is demonstrated below, the error bar denotes the standard deviation. This visualization offers a clear comparison of the overall performance of the Supervised Learning, SimSiam, and SimCLR methods.

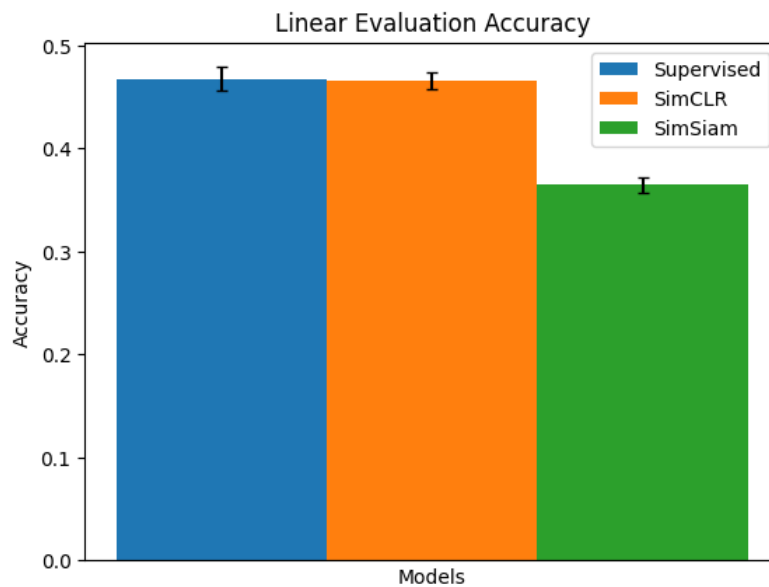


Fig C.1 ResNet50 linear evaluation on ImageNet_subset

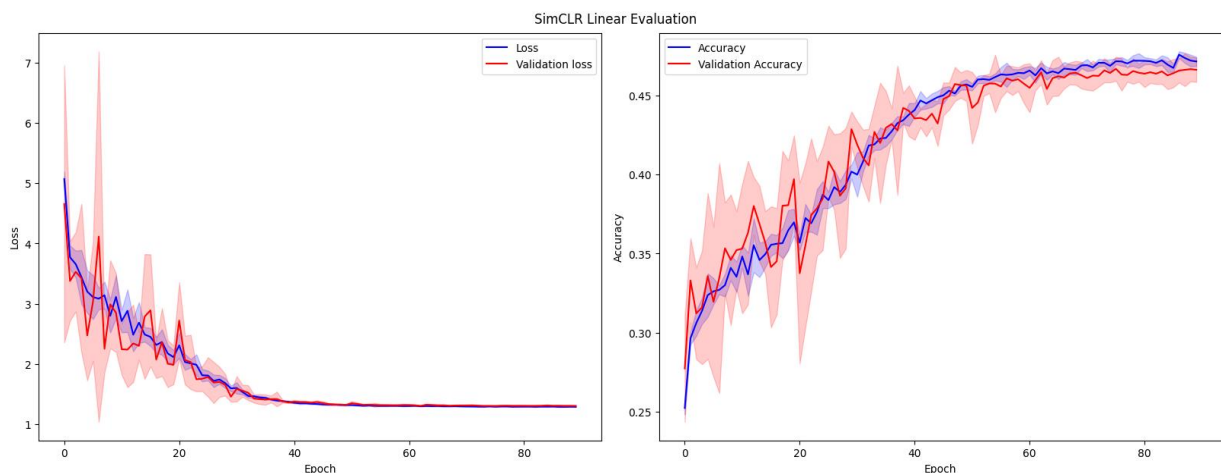


Fig C.2 SimCLR ResNet50 linear evaluation graph on ImageNet_subset_I

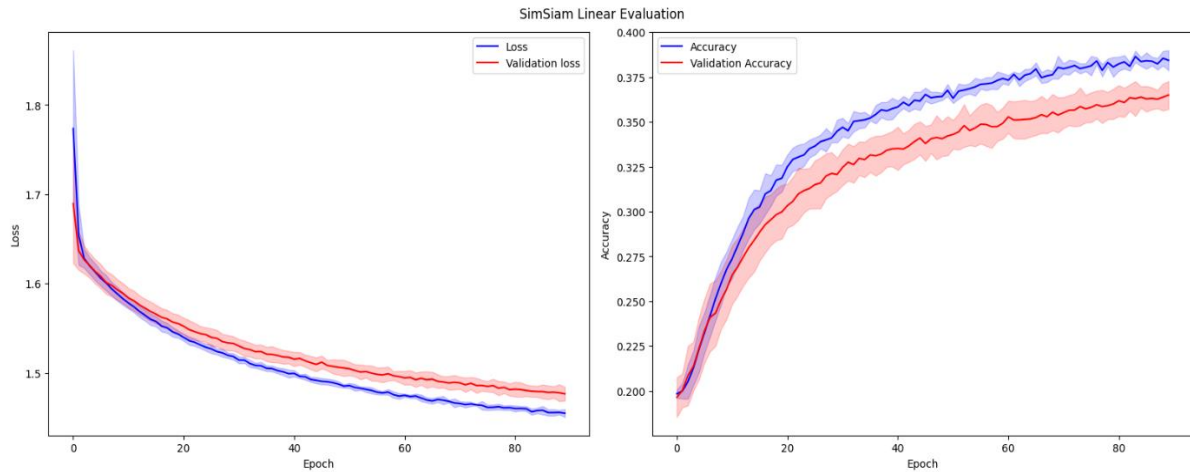


Fig C.3 SimSiam ResNet50 linear evaluation graph on ImageNet_subset_I

In the linear evaluation phase, an interesting phenomenon was observed when comparing the performance of ResNet18 and ResNet50 backbones on 64x64 images. Despite ResNet50's deeper architecture and theoretically superior capacity for feature extraction, it was ResNet18 that delivered better results in experiments.

This counterintuitive outcome can be attributed to the nature of our dataset. The 64x64 images used in our study are relatively low-resolution, which means that the additional layers in ResNet50 may not provide much benefit. In fact, they could potentially over-complicate the model, leading to overfitting and poorer generalization performance. ResNet18, being a shallower model, is less prone to overfitting and may be better suited to the task at hand.

In terms of specific performance metrics, both SimCLR and supervised learning with ResNet50 scored in the mid-forties, with supervised learning having a slight edge. This suggests that both methods were able to learn better representations from the data compared to SimSiam, but neither was able to fully exploit the potential of the ResNet50 backbone.

On the other hand, SimSiam lagged, scoring in the range of the thirties. This is due to the reason that non-contrastive methods like SimSiam are extraordinarily sensitive to model size as reported by Alexander Li et al. (2022). The authors added that the increase or decrease in SimSiam performance is not apparent if we only look at the SimSiam loss, whether it is on the training set or validation set. While this design has been shown to be effective in some contexts, it may not be as well-suited to this scenario of vehicle-type detection with such a dataset size. As a result, it was outperformed by SimCLR in both ResNet50 & ResNet18 backbones.