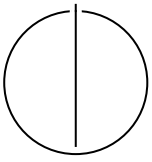
SCHOOL OF COMPUTATION,
INFORMATION AND TECHNOLOGY —
INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Self-Supervised Learning for Efficient 3D Object Detection in Autonomous Driving

Philip Hierhager



Acknowledgments

First and foremost, I thank my Lord and Savior Jesus Christ, who died on the cross and rose again on the third day, so that whoever believes in Him shall not perish but have eternal life. He is the Truth, the Way, and the Life, and by His wounds, I have been healed. Nothing would be possible without Him.

I am deeply grateful to Driveblocks for the opportunity to conduct this thesis and for providing access to the necessary resources. I sincerely thank my industry supervisor, Felix Nobis, for his crucial support and guidance throughout this project.

I also express my sincere gratitude to Esteban Rivera, my academic supervisor, whose expertise, supervision, and encouragement were critical to the success of this thesis. I thank the TUM School of Computation, Information and Technology for their academic support and inspiring environment.

Additionally, I am deeply thankful to my family and friends for their enduring support and encouragement.

Finally, I gratefully acknowledge the scientific support and resources of the AI service infrastructure LRZ AI Systems provided by the Leibniz Supercomputing Centre (LRZ) of the Bavarian Academy of Sciences and Humanities (BAdW), funded by Bayerisches Staatsministerium für Wissenschaft und Kunst (StMWK).

Abstract

This thesis examines how Self-Supervised Learning (SSL) can enable efficient perception in autonomous driving. Existing SSL methods underperform with lightweight backbones which are critical for real-time applications. To address this challenge, two novel SSL methods are proposed, *DiSiam* and *DiSiam+*, which use siamese networks and combine the simplicity of SimSiam-style weight sharing with the effectiveness of knowledge distillation and the robustness of multi-view augmentation. Thorough experiments on image classification and object detection tasks evaluate their performance across varying backbones, batch sizes, learning rates, and datasets. The outcomes demonstrate that DiSiam-based methods achieve superior performance compared to all SSL method baselines in image classification. Despite these improvements, the outcomes of this thesis also show that negative-sample-free methods such as SimSiam, DiSiam and DiSiam+ exhibit limited performance on dense perception tasks like object detection. Furthermore, a novel chaining strategy is introduced to further stabilize training and prevent representational collapse, and extensive studies demonstrate that randomly initialized neural networks can preserve input structure in the latent space which leads to performance substantially above chance in image classification. This thesis advances the field of SSL by introducing efficient frameworks and providing valuable insight into method behavior.

Contents

1 An Introduction to Self-Supervised Learning for Efficient Perception

1.1 Introduction

The rapid development of deep learning has led to major breakthroughs in computer vision and substantially advanced the capabilities of intelligent systems. Autonomous driving is one of the most important applications where vehicles are required to perceive, interpret, and interact with the environment. A critical component of such systems is the perception module, which is necessary for understanding the surrounding area through tasks such as object detection, tracking, and segmentation. These tasks typically depend on large amounts of manually annotated data to achieve high accuracy, and thus posing substantial challenges in terms of scalability, cost, and domain adaptation.

To address the dependency on labeled data, SSL has developed as a promising solution. SSL allows models to learn useful feature representations from unlabeled data, and has demonstrated outstanding results across various downstream computer vision tasks, such as image classification, object detection, and segmentation. However, many state-of-the-art SSL methods have not been evaluated with lightweight backbones, which are critical for real-time applications like autonomous driving. This makes them unsuitable for deployment in resource-constrained environments, where efficiency is crucial.

This thesis addresses the challenge of combining SSL with the need for computational efficiency in the context of perception for autonomous driving. Specifically, it explores the question: *How can self-supervised learning be combined with the need for resource-efficient models in autonomous driving?*

1.2 Objectives and Contributions

The main objectives of this thesis are twofold. First, it aims to evaluate how the proposed SSL frameworks perform in image classification tasks compared to other SSL approaches and baselines. Second, it focuses on assessing its effectiveness in fine-tuning heads for 2D object detection tasks. Additionally, it aims to implement a

3D object detection model from scratch for later evaluation of SSL methods also on this downstream task.

The contributions of this work are as follows. Two novel SSL methods, *Distillation Siamese (DiSiam)* and *Multi-View Distillation Siamese (DiSiam+)*, are introduced, designed for efficiency and simplicity. A series of ablation and robustness studies are done to compare these methods with other SSL frameworks. These give insight into the strengths and weaknesses of the methods. Furthermore, the performance of randomly initialized backbones is analyzed in detail, as it offers a stronger baseline than assumed. Finally, a chaining method is introduced which improves training stability of several SSL methods.

1.3 Thesis Structure

This thesis is organized into the following chapters. First, Chapter 2 presents a review of relevant literature in SSL and 3D object detection. Chapter 3 introduces the design of the proposed SSL methods, while Chapter 4 details the experimental methodology. Chapter 5 then presents the results of the experiments, and chapter 6 discusses the findings and its implications and limitations. Lastly, Chapter 7 concludes the thesis and describes potential directions for future work.

2 Literature Review: Self-Supervised Learning and Monocular 3D Object Detection

2.1 Introduction to the Literature Review

This chapter provides a concise review of the literature relevant to this thesis. It first starts with an overview of the domain of representation learning, with a particular focus on SSL. It explores its principles, key methods, and recent research directions. Furthermore, state-of-the-art methods in 3D object detection are discussed. The literature review forms the foundation for understanding the research problem and motivates the approach taken in the thesis. Efficient perception in autonomous driving requires effective unsupervised pretraining of lightweight backbones, as label annotation is expensive and real-time performance is critical.

2.2 Representation Learning and Self-Supervised Learning: An Overview

2.2.1 Fundamentals of Representation Learning

Representation learning is a subfield of machine learning that focuses on learning effective feature representations automatically from data, rather than relying on handcrafted features [10, 38]. The aim is to extract the underlying structure and manifolds within the data. However, for images, this task is particularly challenging, as inferring the correct latent structure is generally intractable [29], and exact posterior inference is typically infeasible for real-world probabilistic models [11]. Despite this, images are suitable for representation learning, as they are high-dimensional but often lie on low-dimensional manifolds that capture the semantic information [11]. Learning robust representations of images can significantly improve performance on downstream tasks and improve generalization in domain adaptation scenarios [48, 89].

2.2.2 Unsupervised and Self-Supervised Learning Paradigms

While supervised learning relies on costly human-annotated labels, unsupervised learning aims to find patterns and structure in raw data without labels. Unsupervised representation learning is a subfield of unsupervised learning that focuses on learning effective feature representations from data.

SSL is a paradigm of high traction within unsupervised representation learning where the supervision training signal is automatically generated from the data itself. It uses the inherent properties of the data [58] and instead of predicting manually annotated labels, SSL methods are trained on *pretext tasks* where the labels are inferred from the data's structure. An example of such pretext task is the prediction of missing parts of the input, relative positions of patches, or the augmentation applied to the data. Success in those pretext tasks require the model to learn meaningful representations [125] and can be effectively transferred to downstream tasks like classification, detection, or segmentation with less or no human-annotated labels [24, 58, 48].

While SSL is the primary focus of this thesis, other unsupervised learning approaches also exist. These include *dimensionality reduction* methods like PCA [35], TriMap [3], and Factor Analysis [11]; *manifold learning* methods such as t-SNE [78], Locally Linear Embedding [96], Uniform Manifold Approximation and Projection [81], and Multi-dimensional Scaling [12]; *generative* models like Variational Autoencoders [61]; and *clustering* techniques such as K-means [11].

2.2.3 Key Paradigms in Self-Supervised Learning

SSL methods can be categorized based on their pretext task design and underlying methods. The most prominent categories in computer vision include context-based, contrastive, and masked image modeling methods. The next sections provide an overview of these paradigms and highlight their principles and advancements.

Context-based Self-Supervised Learning

Different pretext tasks for context-based SSL have been used to effectively learn the latent features of images. It uses the inherent contextual relationship of the augmented images and thus encourages the model to learn to recognize both fine-grained and overall patterns in the data [43]. Gidaris et al. used random multiples of rotations of images and made backbones with Convolutional Neural Network (CNN) design predict the rotation of the image [40]. Zhang et al. trained a CNN network to map from grayscale to color images [122]. To accomplish this, they needed to operate in the CIE Lab color space as distances there correspond to perceptual distances. Furthermore, to make the loss more robust against desaturated results, they used a multinomial

classification on the discretized color space instead of the Euclidean loss. Noroozi et al. [85] used a jigsaw puzzle pretext task where their context-free siamese network learned to predict the correct permutation of patches of an image. Doersch et al. cut out two patches from an image and asked the model to predict the relative position of those [29]. They take inspiration from the text domain where context-based pretext tasks for SSL are already used since 2005 [29] and argue that the ImageNet paper [64] introduced a new era of deep neural networks that learn rich feature representations. Mundhenk et al. then builds upon the work of Doersch et al. [29] and adds several hand-crafted methods like random rotations or random zooms to the image augmentations to make the SSL method more robust [84]. Another approach was taken by Caron et al. with their DeepCluster method [16]. They use clustering as pseudo-labels for the classification training of the backbone, which is similar to the expectation and maximization steps in the Expectation-Maximization algorithm [82]. Context-based SSL has become a popular approach due to its simplicity and effectiveness [40]. However, it often gives limited performance compared to newer approaches and can be sensitive to the choice of pretext task, leading to alternative approaches like contrastive SSL and masked image modeling becoming dominant [43].

Contrastive Self-Supervised Learning: Principles and Research Directions

Introduction and Core Principle Contrastive SSL is an important paradigm within SSL that contrasts positive and potentially negative pairs of data instances to learn robust representations. It typically uses siamese networks to compare augmented views of the same image against each other (positive pairs) or against views of other images (negative pairs) using some (dis-)similarity or decorrelation metrics. The core paradigm is based upon the discovery of Wu et al. [112, 43] that real visual similarity is crucial for grouping instances, rather than just semantic labeling that result from supervised representation learning by classification. They pushed this to the extreme and asked if a meaningful metric reflecting apparent similarity could be learned using purely discriminative instance-level learning [112]. Their successful answer opened the way for contrastive SSL, which has achieved remarkable success. It outperforms even supervised pretraining when fine-tuning on downstream tasks such as the Pascal Visual Object Classes Challenge object detection task [34, 23] and even ImageNet classification [64, 18]. Foundational work predates this peak in interest, with Hadsell et al. [45] presenting a contrastive loss in 2006 for dimensionality reduction that laid the groundwork for the contrastive SSL paradigm.

Motivation and Learned Representations The success of contrastive SSL is often attributed to the nature of the representations they learn. Zhao et al. demonstrates that contrastive SSL methods learn robust low- and mid-level representations [125]. This

contrasts with supervised pretraining, where the task (e.g. classification) encourages invariance within classes and thus focuses primarily on high-level semantic features, potentially overlooking low-level features that are required for detailed visual information. By modeling those low-level features effectively, SSL enables learning a meaningful latent structure of the entire image. The authors of the paper present evidence for this hypothesis by using the learned representations from both self-supervised and supervised pretraining to reconstruct the original images. The images reconstructed from the latent encodings of SSL far exceeds the quality of those learned from supervised pretraining [125].

Contrastive Objective Formulation The foundation of SSL is the goal to pull representations of positive pairs (different augmented views of the same image) closer together in the latent space while pushing apart representations of negative pairs (views of different images). This is often formalized using objectives related to mutual information maximization. As Tian et al. describes [104], contrastive SSL with negative samples for example learns a discriminative function $h_\theta(\cdot)$ that yields a high score for positive pairs and a low score for negative pairs. Given a positive pair (v_1^1, v_2^1) and k negative samples $\{v_2^2, \ldots, v_2^{k+1}\}$, the contrastive loss encourages distinguishing the positive pair from the k negatives. A common formulation, related to maximizing a lower bound on mutual information [80], is given by:

$$\mathcal{L}_{\text{contrast}}^{V_1,V_2} = - \mathop{\mathbb{E}}_{\{v_1^1, v_2^1, \ldots, v_2^{k+1}\}} \left[\log \frac{h_\theta(\{v_1^1, v_2^1\})}{\sum_{j=1}^{k+1} h_\theta(\{v_1^1, v_2^j\})} \right] . \tag{2.1}$$

Minimizing this loss essentially maximizes a lower bound on the mutual information between the two views V_1 and V_2 [104]. This encourages the model to learn representations that are invariant to the applied augmentations but discriminative across different instances. However, approximating the mutual information accurately, especially with a large number of negative samples, can be challenging, and there are stringent formal limitations on the approximation accuracy [80]. The methods presented in the following sections use different variations of this loss to learn the representations. Even the cosine similarity loss used in negative-sample-free methods can be interpreted as a special case of the contrastive loss [42].

Methods Utilizing Negative Samples A influential line of research in contrastive SSL centers on effective strategies for handling negative samples.

Memory Bank Approaches: Memory bank approaches were among the first to approach the challenge of a large number of negative samples. Those methods utilize a memory bank or lookup table to store feature representations of past image views, a technique first observed in person re-identification [113]. The Instance Discrimination framework [112] applies this to SSL, employing a memory bank to approximate the

contrastive loss and learn to discriminate between views of different images. The pretext task involves matching random crops of the same image. This strategy relates to deep metric learning approaches that similarly use non-parametric approximations, such as the triplet loss [98] or the multi-class N-pair loss [100].

Mutual Information Estimation Based Methods: Another research avenue explores various approximations to mutual information [111, 19]. The Deep InfoMax method [4], built upon Mutual Information Neural Estimation [9] that is trained to estimate mutual information, uses only the discriminator of a Generative Adversarial Network [41] to perform contrastive SSL. The work compares different mutual information lower bounds, identifying InfoNCE [88] which was derived from Noise-Contrastive Estimation [44] as the most effective. It also illustrates its distinctions from Jensen-Shannon Divergence [86] and Donsker-Varadhan representation [30] across datasets of varying sizes. Building upon Deep InfoMax, the Augmented Multiscale Deep InfoMax method [5] maximizes mutual information across independently augmented images and multiple feature scales.

Efficient Negative Sampling with Batches and Queues: Closely linked to Deep InfoMax is the Contrastive Predictive Coding method [88], which also employs mutual information estimation but was initially limited by its fixed autoregressive nature, making it less adaptable for image tasks. Tian et al.'s Contrastive Multiview Coding framework [104] generalizes approaches like Contrastive Predictive Coding, Deep InfoMax, and Instance Discrimination to multiple image views and shows that the SSL performance correlates with the number of views. A major breakthrough for SSL arrived when Simple Framework for Contrastive Learning of Visual Representations (SimCLR) [21] was introduced. It uses a simpler design for negative instance samples by using augmented views of all of the images in one mini-batch as negative examples to the image view at hand. In addition, it introduces a simpler design using only a lightweight projection head and emphasizes the importance of robust data augmentations, especially random cropping and color distortion. Despite its simplicity, the SimCLR is highly dependent on large batch sizes and sensitive to augmentation choices [21, 42]. To mitigate the batch size dependency, the MoCo framework [49] proposes a dynamic dictionary for negative sampling. This dictionary, built with a queue and a momentum-updated encoder, provides a large and consistent source of negative samples. Further innovations include the SwAV framework [17], which approximates the contrastive task using online clustering with discrete prototype vectors as pseudo-labels and enforces consistency between the pseudo-labels of different augmentations. While the conceptually related DeepCluster method updates its clustering every n epochs, SwAV modifies its prototype vectors during the batches.

Methods Utilizing Decorrelation Objectives Instead of explicitly contrasting positive and negative pairs, some methods focus on *decorrelating features*. Inspired by the

redundancy-reduction principle [8], the Barlow Twins method [120] aims to make the cross-correlation matrix between the outputs of a siamese receiving two augmented views of the same image as close to the identity matrix as possible. This design ensures that learned features are invariant to augmentations (along the diagonal) and independent of each other (off-diagonal), conceptually similar to the Information Bottleneck objective [106]. A notable advantage of Barlow Twins is that it does not require large batch sizes [120]. Similarly, VICReg [7] enhances Barlow Twins by adding explicit variance and covariance regularization terms to the loss, which helps to ensure that features are spread out and prevents representational collapse.

Negative-Sample-Free Methods Another influential development in the field of SSL is the introduction of negative-sample-free methods. Those methods aim to achieve the same performance as contrastive SSL methods without relying on negative sample comparisons. Build Your Own Latent (BYOL) [42] uses a siamese architecture with an online and target network. The target network's weights are a momentum average of the online's and the gradient does not flow through the target network. The core idea is to train the online network to predict the output of the target network for augmented views of the same image. This self-distillation process, combined with the architectural details like an additional predictor head and batch normalization, allows BYOL to mostly avoid collapsing to trivial solutions without using negative sample pairs. This representational collapse, where the model produces constant feature representations [56] is a key challenge for negative-sample-free methods. Dimensional collapse is a more general challenge where the model learns to produce a low-dimensional representation of the data, effectively collapsing the feature space into a lower-dimensional manifold [56]. Other negative-sample-free methods have since emerged, building on similar self-distillation or momentum update principles. Simple Siamese (SimSiam) [24] shares weights between the target and online backbone and projection head and points out that the stopgradient operation on the target is crucial for the method's robustness against collapse. Resource-Efficient Self-supervised Learning (FastSiam) [92] uses more than two views to stabilize the target, improving robustness and convergence speed. Distill Your Own Latent (DYOL) [73] improves upon BYOL by using an extra assistant network, which helps to improve the performance for smaller backbones. Instead of using a static distillation teacher like [52], it simultaneously distills knowledge from the target to the online network and trains the assistant network. The behavior is thus termed *dynamic knowledge distillation*.

The rapid research progress in contrastive SSL methods highlight the potential of leveraging instance-level relationships for representation learning. Those methods have shown impressive progress in learning effective representations and have become a crucial part of modern SSL.

Masked Image Modeling: A Novel Approach

Introduction and Core Principle Masked image modeling is a new SSL method that has gained significant traction in recent years. It is inspired by the success of masked language modeling in Natural Language Processing where models like BERT [27] have shown that masking parts of the input and training the model to predict the masked content can lead to rich feature representations. The core idea of masked image modeling is to mask out or change parts of the input image and train the model to reconstruct the original content. This pretext task forces the model to learn meaningful representations by understanding the relationships between the visible and hidden parts of the image.

Key Methods Different masked image modeling methods primarily vary in what and how they mask, and what they predict.

Masked Autoencoders: An influential method is the Masked Autoencoder (MAE) [48]. The approach masks large portions of the image patches (e.g. 75%) and trains a Vision Transformer (ViT) network to encode only the visible patches. A separate decoder then attempts to reconstruct the original pixel values for all patches, including the masked ones. The decoder uses the encoder's output and the positional embeddings of the masked patches for reconstruction. The asymmetric design with a small decoder and large masking ratio is key to MAE's efficiency and performance.

Predicting Discrete Tokens: Instead of reconstructing raw pixels of the original image, some methods predict discrete visual tokens. BEIT [6] trains a ViT encoder to predict visual tokens generated by a discrete Variational Autoencoder [94] for the masked patches.

Predicting Pixels: SimMIM [115] simplifies the method by masking random patches and training a ViT to predict the original pixel values of the masked patches directly using a simple regression loss. It does not require a separate decoder as MAE or BEIT does.

Effectiveness and Relationship to Other Paradigms Masked image modeling has shown outstanding success, particularly when applied to ViTs [31]. The patch-based nature of ViTs aligns well with the masking operation. It has been shown that masked image modeling pretraining yields representations that effectively transfer to various downstream tasks, often surpassing contrastive methods, especially on tasks requiring dense predictions like segmentation [48]. While contrastive methods focus on learning the invariance to augmentations and discrimination of instances, masked image modeling focuses on learning to understand the internal structure by predicting the mask's content.

2.2.4 A Critical Discussion on Self-Supervised Learning

SSL has evolved significantly over recent years. It started from simple context prediction tasks to sophisticated contrastive, generative and predictive approaches, that demonstrate success in learning high-quality feature representations, sometimes even beating the supervised baseline without any human-annotated labels [48]. Early SSL approaches were simple but limited in their ability to capture complex relationships in data. SSL greatly advanced with contrastive frameworks by using negative sampling and complex augmentation strategies to discriminate instances. The development of negative-sample-free methods like SimSiam or BYOL showed that competitive performance is possible without negative pair sampling and even without complex updating schemes like exponential moving averages between the siamese networks, simplifying training. More recently, masked image modeling has become a powerful approach for ViTs. The choice of a self-supervised paradigm and method strongly depends upon factors like the available computational resources, the preferred backbone architecture (CNN vs. ViT), and the specific characteristics of the data and downstream task. For instance, contrastive methods often require careful augmentation strategies, while masked image modeling methods are more suitable for ViTs. Negative-sample-free methods like BYOL and SimSiam deliver strong performance even with small batch sizes, making them suitable for resource-constrained environments. Contrastive methods however that do use negative samples like SimCLR and MoCo are more sensitive to the batch size, but they are more robust against collapse [42].

Complex tasks in autonomous driving, such as 2D or 3D object detection, require a large amount of data to achieve reliable predictions. To overcome data scarcity, SSL is a promising path to utilize large amounts of unlabeled data to improve performance. Perception tasks in autonomous driving have to operate in real-time in resource-constrained environments. As most visual transformers are computationally expensive, CNN-based SSL methods like SimCLR or SimSiam are preferred. However, approaches like SimCLR and BYOL show limited performance with small backbones such as ResNet-18 [50, 73]. This challenge is mitigated by DYOL [73], which uses an extra assistant network. However, the use of an exponential moving average adds further complexity in the hyperparameter tuning of the method, making it less straightforward compared to SimSiam [24] or SimCLR [21]. SimCLR however is sensitive against small batch sizes [42] and typically requires GPU clusters for training, which can be restrictive. Therefore, there is a need for a novel SSL method that performs well with small backbones and batch sizes, while remaining simple to tune.

As mentioned in Section 1.2, this thesis not only thoroughly explores SSL for efficient perception, but also aims to implement a monocular 3D object detection pipeline following industry-grade implementation standards. This enables future research to

evaluate the proposed SSL methods on this downstream task. The following section provides an overview of state-of-the-art models in monocular 3D object detection, and motivates the selection of the model implemented as part of this work.

2.3 Monocular 3D Object Detection: An Overview

Monocular 3D object detection is a challenging task in perception for autonomous driving. The section introduces the formal problem definition, the challenges, and the evolution of models in the field.

2.3.1 Problem Formulation

Given a RGB image $I \in \mathbb{R}^{W \times H \times 3}$ where W and H are the width and height of the image, the goal is to detect all objects in the image, find their class label C and estimate their 3D bounding boxes. Each object is represented by a 3D bounding box $B = (h, w, l, x, y, z, \theta)$, where (h, w, l) are the height, width, and length of the object in meters, and (x, y, z) are the coordinates of the object center in the camera coordinate frame. The variable θ represents the yaw angle, which defines how a cubic bounding box is oriented in the horizontal plane (i.e., how it is rotated around the vertical axis). The camera intrinsic matrix K is known for both training and inference.

2.3.2 Key Challenges

Monocular 3D object detection faces significant challenges primarily due to the ill-posed nature of inferring 3D structure from a single 2D image. The loss of depth information during projection makes accurately estimating object size, position, and orientation in 3D space difficult. Further challenges come from perspective projection, occlusions, and varying object appearances. Despite these challenges, recent research in deep learning has made impressive progress in this field.

2.3.3 Monocular 3D Object Detection: Model Taxonomy and Evolution

Early Approaches

Early methods, such as [22] rely on handcrafted 3D proposals with energy minimization and spatial priors. This is extended by [83] and [116] which improve the 2D detectors with orientation, disparity estimation, and feature fusion.

Depth-Aware CNN Models

Subsequent efforts focused on incorporating depth-awareness into the CNN-based model pipeline. M3D-RPN [13] and D4LCN [28] introduced depth-aware layers to share features across 2D and 3D space. DD3D [90] improved this approach with depth pretraining.

Center-Based Models

Center-based models like SMOKE [74], MonoDLE [77], and MonoCon [72] eliminated 2D box estimation branches of earlier models and emphasized robust localization, with MonoDLE removing distant objects out of training data and MonoCon adding auxiliary learning tasks around the center.

One-Stage Detectors

FCOS3D [109] proposed a one-stage method that directly regresses the 3D bounding boxes directly extending the FCOS [105] detector and simplifying the pipeline. PGD [108] extended FCOS3D by appending a secondary depth estimation stage and using a consistency loss between 2D bounding box predictions and the projected 3D bounding boxes. This helps specifically with few data points and does not add any computational cost to inference.

Transformer-Based Models

Transformer-based approaches like MonoDTR [55] and MonoDETR [121] use auxiliary depth supervision to guide object detection with global depth priors while MonoATT [126] improves detection performance by using adaptive tokens that represent a cluster of pixels.

Depth Estimation-Centric Models

Methods focused on depth estimation form another major class. MonoFlex [123] treats cut off objects differently than others to regress depth more efficiently, MonoGround [93] introduces the ground plane as prior to depth prediction, and DEVIANT [65] uses convolutions equivariant to depth translations. MonoCD [117] introduced a complementary depth branch to reduce ambiguity, while MonoLSS [68] uses learnable sample selection to only regress from pixels that are important to the detection.

2.3.4 Discussion on Monocular 3D Object Detection

The progress shows a clear trend from proposal-based heuristics to end-to-end learning, with increasing focus on depth estimation, transformer-based architectures and auxiliary supervision for more robust 3D object detection. Due to its straightforward design, solid performance, and real-time efficiency, FCOS3D is chosen as a detection model to be implemented in this thesis.

2.4 FCOS3D: A One-Stage Monocular 3D Object Detection Model

FCOS3D [109] is a fully convolutional, anchor-free, one-stage framework for monocular 3D object detection. It is built upon the 2D detector FCOS [105] by reformulating the 3D detection task as a center-based regression problem and features a classification branch and a regression branch. The classification branch predicts class labels and attributes, while the regression branch outputs the centerness score, 2D center offset, 3D center depth, 3D bounding box size (width, length, height), object rotation angle, discretized orientation direction of the 3D bounding box, and the velocity of the object. The predicted 2D center offsets are used to determine the projected 3D center on the image plane. Combined with the predicted depth and the camera intrinsics, this point is then backprojected to recover the 3D object center.

To mitigate challenges like scale variations and occlusions, FCOS3D uses a Feature Pyramid Network [69] for multi-scale feature maps. During training, targets are assigned to feature map levels based on the size of the 2D bounding box that encloses the projected 3D bounding box. Within each level, targets are assigned to feature map locations based on the distance of their receptive field centers to the projected 3D center. An overview of the model architecture is shown in Figure 2.1.

FCOS3D does not rely on Light Detection and Ranging (LiDAR), stereo vision, or depth priors, which makes it suitable for cost-efficiency and real-time applications. It achieves competitive results on the nuScenes [14] dataset.

2.5 Connecting Self-Supervised Learning and Perception Tasks

Monocular 3D object detection is a challenging perception task due to the difficulty of predicting 3D information, such as depth, dimensions, and orientation, from a single 2D image. Similarly, 2D object detection and classification also benefit significantly from robust feature representations, especially with limited data or domain shifts. The efficiency of perception systems executing such tasks strongly relies on the quality of

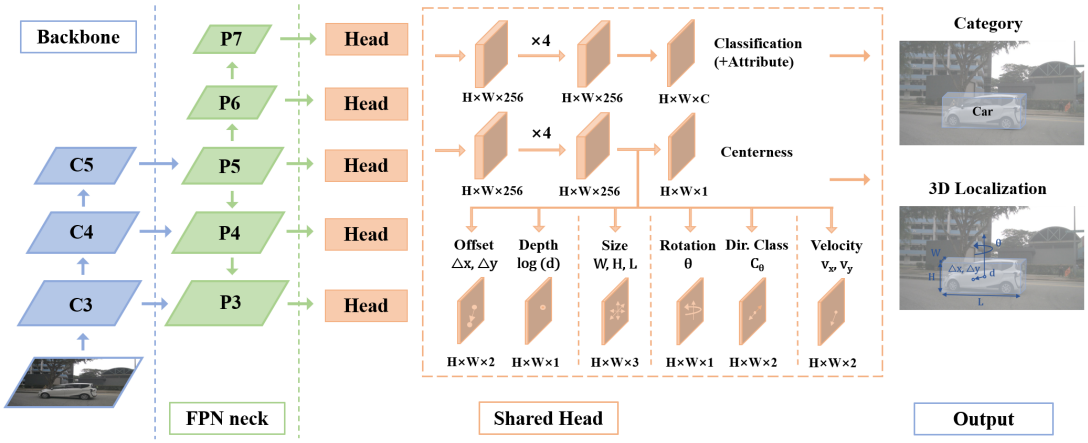


Figure 2.1: An overview of FCOS3D [109], as presented in the original work. The model uses a backbone network extended with Feature Pyramid Networks [69] to extract multi-scale feature maps. The features of one level are then processed by two separate convolutional branches, a classification branch and a regression branch. The classification branch features two heads, predicting class labels and attributes. The regression branch uses seven heads responsible for predicting the centerness, the 2D center offset, the 3D center depth, the 3D bounding box size, the rotation angle, the direction of the 3D bounding box, and the velocity of the object.

the feature extraction capabilities of the backbone network. SSL offers a promising approach to learn such robust features from vast amounts of unlabeled data to improve the performance and robustness of downstream tasks.

The use of SSL to directly improve monocular 3D object detection and evaluation thereof is an active research area. While some studies have explored SSL for 3D perception tasks using LiDAR point clouds [99, 119, 124, 71], or have incorporated auxiliary losses for tasks like monocular depth estimation [60, 20, 110] or improving classifier robustness for rare classes in 3D detection [60], the direct impact of image-based SSL pretraining to monocular 3D object detection performance is less explored. This highlights a research gap and an opportunity where advancements in SSL pretraining could bring significant benefits.

This thesis addresses the need for more powerful feature representations by focusing on the development and extensive evaluation of novel SSL approaches. The primary aim is to produce more effective and robust feature representations, particularly for lightweight backbone architectures operating under resource constraints often required in autonomous driving. The difficult challenges in tasks such as monocular 3D object detection serves as a key motivation for this research on better SSL frameworks. The core contributions and empirical results of this work are the advancement within the SSL domain itself, with the proposed methods showing significant performance improvements in image classification. The thesis delivers foundational SSL advancements that open the way for more data-efficient and robust perception systems across various tasks.

2.6 Summary of Literature Review and Thesis Contribution

This chapter provides a comprehensive overview of representation learning, with a focus on SSL paradigms and their ability to learn from unlabeled data. It also reviews the state-of-the-art in Monocular 3D object detection. The potential of SSL methods to address key perception challenges such as efficiency and feature robustness presents a promising research direction for autonomous driving perception.

Building upon this foundation, the following chapters details the proposed SSL methods *DiSiam* and *DiSiam+*, which combine the simplicity of SimSiam's weight sharing mechanism with DYOL's dynamic knowledge distillation capabilities and FastSiam's multiple view approach to improve feature learning. The following sections describe the method design, the methodology, present the results of its application to classification and 2D object detection, and discuss the implications and contributions of this thesis.

3 The DiSiam Framework: Combining Simplicity with Dynamic Knowledge Distillation

This section details the design of the proposed contrastive SSL methods, *DiSiam* and *DiSiam+*. These methods use the simplicity of SimSiam while using the dynamic knowledge distillation capabilities of DYOL to improve performance on small backbones. First, the core concepts and design choices of related methods are reviewed. Subsequently, the *DiSiam* method and its extended variant *DiSiam+* are introduced.

3.1 Review of Prior Self-Supervised Learning Methods

Before describing the proposed methods, this section gives a rigorous overview of existing key methods in self-supervised learning.

3.1.1 SimCLR: A Pioneer in Contrastive Learning

SimCLR [21] samples a minibatch of N images and generates two augmented views per image, forming $2N$ samples. For a positive pair (i, j), the remaining $2(N-1)$ samples are treated as negatives. The loss is computed over all positive pairs in the batch. In addition to this effective negative sampling strategy, the authors propose a simplified method design, adding a single fully connected projection head after the backbone, which empirically improves performance. The model is trained using the InfoNCE (sometimes also called NT-Xent) loss to maximize the *mutual information* between augmented views. Given the cosine similarity function $\text{sim}(\boldsymbol{u}, \boldsymbol{v}) = \frac{\langle \boldsymbol{u}, \boldsymbol{v} \rangle}{\|\boldsymbol{u}\| \|\boldsymbol{v}\|}$ between two latent image features, the InfoNCE loss for a positive image pair (i, j) is defined by

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1, k\neq i}^{2N} \exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)}\,, \tag{3.1}$$

where τ is the temperature parameter [21]. Algorithm 1 describes the SimCLR learning procedure. A limitation of SimCLR is its reliance on high batch sizes [21, 42] as it

needs a large number of negative samples for effective learning. In contrast, negative-sample-free methods like BYOL [42] and SimSiam [24] do not even require negative samples.

Algorithm 1: SimCLR training loop

Inputs :

$\mathcal{D}, \mathcal{T}$	set of images and transformation distribution
$\theta, f_\theta, g_\theta$	parameters, encoder, projection head
optimizer	optimizer updating parameters
K, N	total steps and batch size
τ	temperature parameter

for $k \leftarrow 1$ **to** K **do**
  $\mathcal{B} \leftarrow \{x_i \sim \mathcal{D}\}_{i=1}^{N}$ `// sample batch`
  for $x_i \in \mathcal{B}$ **do**
    Sample $t \sim \mathcal{T}$, $t' \sim \mathcal{T}$ `// sample transformations`
    $z_{2i-1} \leftarrow g_\theta(f_\theta(t(x_i)))$, $z_{2i} \leftarrow g_\theta(f_\theta(t'(x_i)))$ `// projections`
  end
  for $i \leftarrow 1$ **to** $2N$ **do**
    for $j \leftarrow 1$ **to** $2N$ **do**
      $s_{i,j} \leftarrow \frac{\langle z_i, z_j \rangle}{\|z_i\| \|z_j\|}$ `// cosine similarity`
    end
  end
  for $i \leftarrow 1$ **to** N **do**
    $\ell_i \leftarrow -\log\left(\frac{\exp(s_{2i-1,2i}/\tau)}{\sum_{k=1,k\neq 2i-1}^{2N}\exp(s_{2i-1,k}/\tau)} \cdot \frac{\exp(s_{2i,2i-1}/\tau)}{\sum_{k=1,k\neq 2i}^{2N}\exp(s_{2i,k}/\tau)}\right)$ `// loss for` x_i
  end
  $\Delta\theta \leftarrow \frac{1}{2N}\sum_{i=1}^{N}\partial_\theta \ell_i$ `// loss gradient`
  $\theta \leftarrow \text{optimizer}(\theta, \Delta\theta)$ `// update encoder and projector params`
end

Output: encoder f_θ

3.1.2 BYOL: A Self-Supervised Learning Framework without Negative Samples

BYOL [42] is a negative-sample-free contrastive SSL framework. It only relies on positive pairs and employs an online network and a target network. Both include a projection head g, such as SimCLR, but the online network also introduces an

extra prediction head q. The target network is updated via an exponential moving average of the online network rather than gradient descent. The method is trained by minimization of the mean squared error between the normalized outputs of the online and target branches, which is equivalent to maximizing the cosine similarity between their unnormalized representations [42]. Algorithm 2 summarizes the learning procedure. A critical challenge in BYOL is maintaining high performance for small backbones [73] and preventing representational collapse. The latter is mitigated by the exponential moving average update, which however introduces additional complexity through the momentum coefficient.

Algorithm 2: BYOL training loop

Inputs :

$\mathcal{D}, \mathcal{T}, \mathcal{T}'$	set of images and transformation distributions
$\theta, f_\theta, g_\theta, q_\theta$	online parameters, encoder, projector, predictor
ξ, f_ξ, g_ξ	target parameters, encoder, projector
optimizer	optimizer updating online parameters
K, N	total steps and batch size
$\{\tau_k\}_{k=1}^K, \{\eta_k\}_{k=1}^K$	target update and learning rate schedules

for $k \leftarrow 1$ **to** K **do**
- $\mathcal{B} \leftarrow \{x_i \sim \mathcal{D}\}_{i=1}^N$ `// sample batch`
- **for** $x_i \in \mathcal{B}$ **do**
  - Sample $t \sim \mathcal{T}$ and $t' \sim \mathcal{T}'$ `// sample transformations`
  - $z_1 \leftarrow g_\theta(f_\theta(t(x_i))), \quad z_2 \leftarrow g_\theta(f_\theta(t'(x_i)))$ `// online projections`
  - $z'_1 \leftarrow g_\xi(f_\xi(t'(x_i))), \quad z'_2 \leftarrow g_\xi(f_\xi(t(x_i)))$ `// target projections`
  - $l_i \leftarrow -2 \cdot \left(\frac{\langle q_\theta(z_1), z'_1 \rangle}{\|q_\theta(z_1)\| \|z'_1\|} + \frac{\langle q_\theta(z_2), z'_2 \rangle}{\|q_\theta(z_2)\| \|z'_2\|} \right)$ `// loss for` x_i
- **end**
- $\Delta\theta \leftarrow \frac{1}{N} \sum_{i=1}^N \partial_\theta l_i$ `// loss gradient`
- $\theta \leftarrow \text{optimizer}(\theta, \Delta\theta, \eta_k)$ `// update online params`
- $\xi \leftarrow \tau_k \xi + (1 - \tau_k)\theta$ `// update target params`

end

Output: encoder f_θ

3.1.3 SimSiam: A Simplified BYOL without Exponential Moving Average

SimSiam [24] builds upon BYOL [42], but instead of updating the target network using an exponential moving average, it shares the backbone and projection head between the target and online branches. The online network is trained to predict the output of

the target branch, which differs only by receiving a differently augmented image. The training objective is a cosine similarity loss between the outputs of the two branches. The primary challenges in SimSiam are avoiding representational collapse [24] and maintaining performance with small backbones [73]. The SimSiam learning procedure is summarized in Algorithm 3.

Algorithm 3: SimSiam training loop

Inputs :

$\mathcal{D}, \mathcal{T}, \mathcal{T}'$	set of images and transformation distributions
$\theta, f_\theta, g_\theta, q_\theta$	parameters, encoder, projector, online predictor
optimizer	optimizer updating online parameters
K, N	total steps and batch size
$\{\eta_k\}_{k=1}^{K}$	learning rate schedules

for $k \leftarrow 1$ **to** K **do**
  $\mathcal{B} \leftarrow \{x_i \sim \mathcal{D}\}_{i=1}^{N}$ `// sample batch`
  for $x_i \in \mathcal{B}$ **do**
    Sample $t \sim \mathcal{T}$ and $t' \sim \mathcal{T}'$ `// sample transformations`
    $z_1 \leftarrow g_\theta(f_\theta(t(x_i))), \quad z_2 \leftarrow g_\theta(f_\theta(t'(x_i)))$ `// online projections`
    $z_1' \leftarrow \text{stopgrad}\,(z_2), \quad z_2' \leftarrow \text{stopgrad}\,(z_1)$ `// target projections`
    $l_i \leftarrow -\left(\frac{\langle q_\theta(z_1), z_1' \rangle}{\|q_\theta(z_1)\| \|z_1'\|} + \frac{\langle q_\theta(z_2), z_2' \rangle}{\|q_\theta(z_2)\| \|z_2'\|}\right)$ `// loss for` x_i
  end
  $\Delta\theta \leftarrow \frac{1}{N} \sum_{i=1}^{N} \partial_\theta l_i$ `// loss gradient`
  $\theta \leftarrow \text{optimizer}(\theta, \Delta\theta, \eta_k)$ `// update online params`
end

Output: encoder f_θ

3.1.4 FastSiam: An Improved SimSiam for Small Batch Sizes

FastSiam [92] improves the convergence speed and robustness of SimSiam, particularly under small batch sizes. It stabilizes the training by averaging the loss over multiple augmented views of the same image. The learning procedure is detailed in Algorithm 4.

3.1.5 DYOL: Dynamic Knowledge Distillation for Lightweight Backbones

DYOL [73] extends BYOL [42] by introducing an additional assistant network. Both the target and assistant networks share the same architecture, while the online network is smaller. The assistant and target networks operate as in standard BYOL, with the

Algorithm 4: FastSiam training loop

Inputs :

$\mathcal{D}, \mathcal{T}$	set of images and transformation distribution
$\theta, f_\theta, g_\theta, q_\theta$	parameters, encoder, projector, online predictor
optimizer	optimizer
K, N, M	training steps, batch size, number of augmentations
$\{\eta_k\}_{k=1}^{K}$	learning rate schedule

for $k \leftarrow 1$ **to** K **do**
  $\mathcal{B} \leftarrow \{x_i \sim \mathcal{D}\}_{i=1}^{N}$ `// sample batch`
  for $x_i \in \mathcal{B}$ **do**
    Sample $\{t^{(j)} \sim \mathcal{T}\}_{j=1}^{M}$ `// sample` M `transformations`
    for $j \leftarrow 1$ **to** M **do**
      $z_j \leftarrow g_\theta(f_\theta(t^{(j)}(x_i)))$ `// online projections`
      $z'_j \leftarrow \text{stopgrad}\left(z_j\right)$ `// target projections`
    end
    $l_i \leftarrow -\frac{1}{M(M-1)} \sum_{j,k=1, j\neq k}^{M} \frac{\langle q_\theta(z_j), z'_k \rangle}{\|q_\theta(z_j)\| \|z'_k\|}$ `// loss for` x_i
  end
  $\Delta\theta \leftarrow \frac{1}{N} \sum_{i=1}^{N} \partial_\theta l_i$ `// loss gradient`
  $\theta \leftarrow \text{optimizer}(\theta, \Delta\theta, \eta_k)$ `// update online params`
end

Output: encoder f_θ

target network's weights updated via an exponential moving average of the assistant network. Even though the online network has a different architecture, it is trained to predict the output of the target network. A key limitation of DYOL is the added complexity introduced by the moving average update. The learning procedure is detailed in Algorithm 5.

Building upon the insights and challenges of these prior works, *DiSiam* and *DiSiam+* are proposed, novel self-supervised learning frameworks designed to combine the simplicity of SimSiam with the dynamic knowledge distillation capabilities of DYOL for improved performance, particularly for small backbones.

3.2 Motivation Behind DiSiam: Addressing Current Limitations

The motivation for introduction of the new methods comes from the trade-offs observed in existing methods. SimSiam and BYOL demonstrate performance limitations when

Algorithm 5: DYOL training loop

Inputs :

$\mathcal{D}, \mathcal{T}, \mathcal{T}'$	set of images and transformation distributions
$\theta, f_\theta^o, g_\theta^o, q_\theta^o$	online parameters, backbone, projector, and predictor
ξ, f_ξ, g_ξ	target parameters, backbone, and projector
$\psi, f_\psi, g_\psi, q_\psi^a$	assistant parameters, backbone, projector, and predictor
optimizer	optimizer updating online and assistant parameters
K, N	total steps and batch size
$\{\tau_k\}_{k=1}^K, \{\eta_k\}_{k=1}^K$	target update and learning rate schedules

for $k \leftarrow 1$ **to** K **do**
  $\mathcal{B} \leftarrow \{x_i \sim \mathcal{D}\}_{i=1}^N$ `// sample batch`
  for $x_i \in \mathcal{B}$ **do**
    Sample $t \sim \mathcal{T}$ and $t' \sim \mathcal{T}'$ `// sample transformations`
    $z_1^o \leftarrow g_\theta(f_\theta(t(x_i))), \quad z_2^o \leftarrow g_\theta(f_\theta(t'(x_i)))$ `// online projections`
    $z_1^a \leftarrow g_\psi(f_\psi(t(x_i))), \quad z_2^a \leftarrow g_\psi(f_\psi(t'(x_i)))$ `// assistant projections`
    $z_1^t \leftarrow \text{stopgrad}\left(g_\xi(f_\xi(t'(x_i)))\right)$
    $z_2^t \leftarrow \text{stopgrad}\left(g_\xi(f_\xi(t(x_i)))\right)$ `// target projections`
    $l_o \leftarrow -\left(\frac{\langle q_\theta(z_1^o), z_1^t\rangle}{\|q_\theta(z_1^o)\|\|z_1^t\|} + \frac{\langle q_\theta(z_2^o), z_2^t\rangle}{\|q_\theta(z_2^o)\|\|z_2^t\|}\right)$ `// online loss for` x_i
    $l_a \leftarrow -\left(\frac{\langle q_\psi^a(z_1^a), z_1^t\rangle}{\|q_\psi^a(z_1^a)\|\|z_1^t\|} + \frac{\langle q_\psi^a(z_2^a), z_2^t\rangle}{\|q_\psi^a(z_2^a)\|\|z_2^t\|}\right)$ `// assistant loss for` x_i
    $l_i \leftarrow 0.5 \cdot (l_a + l_o)$ `// average over assistant and online loss`
  end
  $\Delta\theta \leftarrow \frac{1}{N}\sum_{i=1}^N \partial_\theta l_i$ `// loss gradient`
  $\theta \leftarrow \text{optimizer}(\theta, \Delta\theta, \eta_k)$ `// update online params`
  $\xi \leftarrow \tau_k \xi + (1 - \tau_k)\theta$ `// update target params`
end

Output: online encoder f_θ^o

using smaller backbone models [73]. DYOL [73] addresses this challenge through its dynamic knowledge distillation mechanism but introduces additional complexity due to the exponential moving average update for the target network, which makes the tuning of hyperparameters more difficult.

The proposed method *DiSiam* aims to bridge this gap, possessing the simplicity of SimSiam by relying only on weight sharing between the target and assistant networks, while also being effective by incorporating the dynamic knowledge distillation capability of DYOL. It achieves strong performance even with smaller backbones.

To further improve performance, a second framework named *DiSiam+* is introduced.

This method builds upon DiSiam by incorporating the stabilizing measure introduced in FastSiam [92], which utilizes multiple image views for the target network. By stabilizing the target network, this approach leads to improved performance and convergence speed.

Furthermore, to counter potential representational collapse often observed by self-distillation methods [24, 42] like DiSiam, DiSiam+, SimSiam, and BYOL, a novel *chaining mechanism* is introduced. Pre-training the backbone with a collapse-resistant method such as SimCLR [21], followed by self-distillation, is expected to produce superior results, as SimCLR provides a more stable initialization for subsequent training.

3.3 DiSiam: Method Design and Implementation

3.3.1 Architecture Components

Following the architecture of DYOL [73], a siamese network with *online*, *target*, and *assistant* branches is introduced. The online network has a separate architecture from the target and assistant networks, and is updated using gradient descent. The target and assistant networks share the same architecture and weights. This design choice eliminates the need for exponential moving averages, and thus it strongly simplifies the model.

The online and assistant networks compute the similarity loss between their predictions and the projections of the target network. Each branch receives two independently augmented views of the image x, x_1 and x_2. They pass through all networks, and the loss is symmetrized.

Each network consists of a backbone f and a projection head g. Moreover, the online and assistant networks include a prediction head q to introduce asymmetry [42]. All heads are implemented as multi-layer perceptrons. A stop-gradient operation is applied after the projection head in the target branch, which has been experimentally shown to mitigate representational collapse [24].

The design is equivalent to SimSiam [24] when focusing only on the target and assistant networks. The main novelty is the decoupled online network with a smaller architecture. The output of the training algorithm is the encoder of the online network.

3.3.2 Formulation of Loss Functions

Contrastive negative-sample-free SSL methods aim to maximize the feature similarity between outputs of siamese networks receiving independently augmented views of the same image. In the proposed framework, the similarity loss is computed both between the predicted outputs of the online predictor and the target projector - termed

online loss, and between the outputs of the online predictor and the assistant projector - termed assistant loss. The final contrastive loss is defined as the average of both losses.

The similarity loss function is the negative cosine similarity described by

$$D(\boldsymbol{p}, \boldsymbol{z}) = -\frac{\langle \boldsymbol{p}, \boldsymbol{z} \rangle}{\|\boldsymbol{p}\| \|\boldsymbol{z}\|}. \tag{3.2}$$

Following [24, 42, 73] the similarity loss is symmetrized. The *online loss* is given by

$$D_{online} = D(\boldsymbol{p}_1^{online}, \text{sg}(\boldsymbol{z}_2^{target})) + D(\boldsymbol{p}_2^{online}, \text{sg}(\boldsymbol{z}_1^{target})) \tag{3.3}$$

where $\boldsymbol{p}_i^{online}$ are the outputs of the online predictor, and $\boldsymbol{z}_j^{target}$ are the outputs of the target projector for the two augmented views. The stop-gradient operator $\text{stopgrad}(\cdot)$ prevents gradients from flowing back to the target network during training.

Similarly, the *assistant loss* is defined as

$$D_{assistant} = D(\boldsymbol{p}_1^{assistant}, \text{stopgrad}(\boldsymbol{z}_2^{target})) + D(\boldsymbol{p}_2^{assistant}, \text{stopgrad}(\boldsymbol{z}_1^{target})). \tag{3.4}$$

The final contrastive loss used in *DiSiam* is

$$L_{contrastive} = \frac{D_{online} + D_{assistant}}{2}. \tag{3.5}$$

3.3.3 DiSiam Training Algorithm

Based on the method design details and loss functions described in Subsection 3.3.1 and Subsection 3.3.2, the learning algorithm of *DiSiam* is summarized in Algorithm 6.

3.4 DiSiam+: Improving Robustness with Multi-View Learning

To improve performance and robustness, an extended framework *DiSiam+* is proposed. It uses multiple views of the same image to stabilize learning, rather than relying on only two views, similar to the approach used in FastSiam [92]. The learning procedure is summarized in Algorithm 7.

3.5 The Chaining Method: A Strategy to Prevent Representational Collapse

To alleviate representational collapse risks in *DiSiam*, *DiSiam+*, and potentially in *SimSiam* and *BYOL*, a chaining strategy is introduced. The approach involves initially training the backbone with a collapse-resistant method like SimCLR [21], followed by

Algorithm 6: DiSiam training loop

Inputs :

$\mathcal{D}, \mathcal{T}, \mathcal{T}'$	set of images and transformation distributions
$\theta, f_\theta^o, g_\theta^o, q_\theta^o$	online parameters, backbone, projector, predictor
ξ, f_ξ, g_ξ	target and assistant parameters, backbone, projector
q_ξ^a	assistant predictor
optimizer	optimizer updating online and assistant parameters
K, N	total steps and batch size
$\{\eta_k\}_{k=1}^K$	learning rate schedules

for $k \leftarrow 1$ **to** K **do**
- $\mathcal{B} \leftarrow \{x_i \sim \mathcal{D}\}_{i=1}^N$ `// sample batch`
- **for** $x_i \in \mathcal{B}$ **do**
  - Sample $t \sim \mathcal{T}$ and $t' \sim \mathcal{T}'$ `// sample transformations`
  - $z_1^o \leftarrow g_\theta(f_\theta(t(x_i))), \quad z_2^o \leftarrow g_\theta(f_\theta(t'(x_i)))$ `// online projections`
  - $z_1^a \leftarrow g_\xi(f_\xi(t(x_i))), \quad z_2^a \leftarrow g_\xi(f_\xi(t'(x_i)))$ `// assistant projections`
  - $z_1^t \leftarrow \text{stopgrad}\,(z_2^a), \quad z_2^t \leftarrow \text{stopgrad}\,(z_1^a)$ `// target projections`
  - $l_o \leftarrow -\left(\frac{\langle q_\theta(z_1^o), z_1^t\rangle}{\|q_\theta(z_1^o)\|\|z_1^t\|} + \frac{\langle q_\theta(z_2^o), z_2^t\rangle}{\|q_\theta(z_2^o)\|\|z_2^t\|}\right)$ `// online loss for` x_i
  - $l_a \leftarrow -\left(\frac{\langle q_\theta(z_1^a), z_1^t\rangle}{\|q_\theta(z_1^a)\|\|z_1^t\|} + \frac{\langle q_\theta(z_2^a), z_2^t\rangle}{\|q_\theta(z_2^a)\|\|z_2^t\|}\right)$ `// assistant loss for` x_i
  - $l_i \leftarrow 0.5 \cdot (l_a + l_o)$ `// average over assistant and online loss`
- **end**
- $\Delta\theta \leftarrow \frac{1}{N}\sum_{i=1}^N \partial_\theta l_i$ `// loss gradient`
- $\theta \leftarrow \text{optimizer}(\theta, \Delta\theta, \eta_k)$ `// update online params`

end

Output: online encoder f_θ^o

pretraining with a self-distillation framework such as SimSiam, DiSiam or DiSiam+. The transition from SimCLR to the self-distillation method should occur once SimCLR approaches convergence, which can be determined by monitoring the loss and online k-Nearest Neighbors (kNN) accuracy of the encoder. The rationale behind this transition strategy is based on the hypothesis that SimCLR's robustness against representational collapse becomes effective in its convergence phase.

3.6 Comparative Analysis of Self-Supervised Learning Methods

This section compares different SSL methods. Table 3.1 summarizes the key features of each method, such as the presence of an assistant network, weight sharing mechanisms,

Algorithm 7: DiSiam+ training loop

Inputs :

$\mathcal{D}, \mathcal{T}$	set of images and transformation distribution
$\theta, f_\theta^o, g_\theta^o, q_\theta^o$	online parameters, backbone, projector, predictor
ξ, f_ξ, g_ξ	target and assistant parameters, backbone, projector
q_ξ^a	assistant predictor
optimizer	optimizer updating online and assistant parameters
K, N	total steps and batch size
$\{\eta_k\}_{k=1}^K$	learning rate schedules

for $k \leftarrow 1$ **to** K **do**
  $\mathcal{B} \leftarrow \{x_i \sim \mathcal{D}\}_{i=1}^N$ // sample batch
  for $x_i \in \mathcal{B}$ **do**
    Sample $\{t^{(j)} \sim \mathcal{T}\}_{j=1}^M$ // sample M transformations
    for $j \leftarrow 1$ **to** M **do**
      $z_j^o \leftarrow g_\theta(f_\theta(t(x_i)))$ // online projection
      $z_j^a \leftarrow g_\xi(f_\xi(t(x_i)))$ // assistant projection
      $z_j^t \leftarrow \text{stopgrad}\left(z_j^a\right)$ // target projections
    end
    $l_a \leftarrow -\frac{1}{M(M-1)} \sum_{j,k=1, j\neq k}^M \frac{\langle q_\xi^a(z_j^a), z_k^t \rangle}{\|q_\xi^a(z_j^a)\| \|z_k^t\|}$ // assistant loss
    $l_o \leftarrow -\frac{1}{M(M-1)} \sum_{j,k=1, j\neq k}^M \frac{\langle q_\theta^o(z_j^o), z_k^t \rangle}{\|q_\theta^o(z_j^o)\| \|z_k^t\|}$ // online loss
    $l_i \leftarrow \frac{(l_a + l_o)}{2}$ // average over assistant and online loss
  end
  $\Delta\theta \leftarrow \frac{1}{N} \sum_{i=1}^N \partial_\theta l_i$ // loss gradient
  $\theta \leftarrow \text{optimizer}(\theta, \Delta\theta, \eta_k)$ // update online params
end

Output: online encoder f_θ^o

the use of negative samples as described in Section 2.2.3, multi-view approaches, their training complexity, and their dependency on augmentation strategies. Critically, it also evaluates their performance with lightweight backbone models, as it is a significant challenge in the field. The detailed results of the performance evaluation are presented in Chapter 5.

While methods like SimCLR [21] rely on negative samples and large batch sizes, and demonstrate high dependency on the augmentation strategy and thus moderate complexity, BYOL [42] and SimSiam [24] innovated negative-sample-free learning,

mitigating the dependency on large batch sizes and also making the SSL method more robust against suboptimal augmentation strategies [42]. However, SimCLR, SimSiam, and BYOL all struggle with low performance with small backbones [73]. FastSiam is expected to display the same performance limitations for small backbones, but it is future work to prove this. BYOL introduces additional complexity with an exponential moving average strategy and FastSiam with multiple image views. Therefore, they are classified as medium-complexity methods. SimSiam, however, is categorized as low-complexity due to its simple design.

DYOL [73], building upon BYOL, improved performance for smaller models but introduces high complexity by using an exponential moving average update and assistant networks. DiSiam directly addresses this trade-off by combining the architectural simplicity of SimSiam with the powerful assistant mechanism of DYOL. As DiSiam only uses the assistant network and no exponential moving average updating scheme, it is classified as low-complexity method. DiSiam+ extends the DiSiam method by using multiple instead of only two image views, stabilizing the training like FastSiam [92] does. The augmentation dependency of DiSiam and DiSiam+ is expected to be low, consistent with other negative-sample-free methods. The comparison in Table 3.1 emphasizes how DiSiam and DiSiam+ are designed to reach high performance with lightweight backbones while maintaining simplicity. The detailed evaluation of the SSL methods is presented in Chapter 5.

Table 3.1: Summary of Self-Supervised Learning Methods

Feature	**SimCLR**	**BYOL**	**SimSiam**	**FastSiam**	**DYOL**	**DiSiam**	**DiSiam+**
Method Design							
Assistant Network					✓	✓	✓
Weight Sharing			✓	✓		✓	✓
Uses Negatives	✓						
Multi-view ($\geq$3)				✓			✓
Complexity[1]	Medium	Medium	Low	Medium	High	Low	Medium
Performance							
Augmentation Dependency	High	Low[2]	Low[2]	Low[2]	Low[2]	Low[2]	Low[2]
Small Backbone Performance	Low	Low	Low	Low[3]	High	High	High

[1]Complexity refers to the implementation and training difficulty, including architectural parts such as assistant networks and exponential moving average updates, as well as training requirements like multi-view inputs and large batch sizes. Assistant networks do not add complexity when used with weight sharing, as no separate parameters are introduced. However, they do increase complexity when combined with exponential moving average updates, which requires maintaining an additional set of parameters.

[2]BYOL [42] showed low augmentation dependency. As all tested negative-sample-free methods can be seen as variants of BYOL the same is hypothesized for SimSiam, DiSiam, DiSiam+, and DYOL.

[3]FastSiam [92] is expected to struggle with small backbones, as does BYOL [42]. This remains future work to prove.

4 Methodology

4.1 Experimental Design: Objectives and Scope

The primary goal of these experiments is to evaluate and compare the performance of various contrastive SSL methods across different perception tasks. This includes the *DiSiam* and *DiSiam+* frameworks detailed in Chapter 3. The focus lies on negative-free-sample methods SimSiam, DiSiam, and DiSiam+. For enhanced comparability, SimCLR is also included. All experiments ensure statistical reproducibility by using fixed random seeds and deterministic CUDA operations using PyTorch [91], NumPy [47], and PyTorch Lightning [36] settings.

A key aspect of this research is to assess the robustness of the methods against varying conditions. This includes backbone sizes and architectures, learning rates, batch sizes, and pretraining datasets.

The experiments aim to answer the following research questions:

- How do the proposed methods *DiSiam* and *DiSiam+* perform compared to existing methods such as SimSiam and SimCLR?
- Do *DiSiam* and *DiSiam+* improve performance on lightweight backbones?
- What is the impact of the chaining strategy?
- How well do the learned feature representations generalize across different datasets and downstream tasks including classification and object detection?
- How robust are the SSL methods against hyperparameter variations?

4.2 Datasets: Selection and Characteristics

The experiments utilize several datasets for pretraining and evaluations on downstream tasks. This section provides an overview of those datasets, including their characteristics.

4.2.1 Pretraining Datasets

The following datasets are used for pretraining the models:

- **CIFAR-10** [62]: A dataset with 50,000 training images and 10,000 test images, each with a resolution of 32x32 pixels. It contains 10 classes.
- **COCO 2017** [70]: A large-scale dataset with 118,000 images used for pretraining. The images are resized to a resolution of 224x224 pixels.

4.2.2 Evaluation Datasets

The following datasets are used for evaluating the pretrained models:

- **CIFAR-10** [62]: The same dataset used for pretraining, with 50,000 training images and 10,000 test images for image classification.
- **ImageNet** [63]: A subset of ImageNet with 10 classes[1] and 13,394 images are used. It is divided into 9,469 training and 3,925 validation samples and used for image classification.
- **nuScenes** [14]: A dataset with 1,000 scenes and 204,894 images used for 3D object detection tasks. The dataset is split into 700 scenes for training and 150 scenes for validation. The images are resized to a resolution of 360x640 pixels.
- **OpenImages** [66]: A subset of the OpenImages dataset with 50,000 randomly selected images used for 2D object detection tasks. The images are resized to a resolution of 360x640 pixels.
- **Proprietary**: A proprietary dataset of 50,000 images recorded by various cameras in diverse environments used for 2D object detection. The data samples are resized to a resolution of 360x640 pixels.

4.3 Self-Supervised Learning: Experimental Setup

The training procedure for the the contrastive SSL methods involve the following steps:

- **Augmentation Pipeline**: The images are augmented using random resized crop, random horizontal flip, color jitter, random gray scale, and gaussian blur. The augmentation strategies for SimSiam, DiSiam, and DiSiam+ follow those described

[1]The exact classes are: *tench, English springer, cassette player, chain saw, church, French horn, garbage truck, gas pump, golf ball, parachute*

in the SimSiam paper [24], while SimCLR uses the pipeline defined in the original SimCLR paper [21].

- **Batch Size**: According to various papers, the batch size is crucial for the performance of the models [42, 21]. Thus, it is varied as part of the hyperparameter sensitivity analysis in Section 4.6. If the batch sizes are not reported alongside the results, the batch size is set to 1024 by default. For the COCO 2017 dataset, gradient accumulation is used to simulate larger batch sizes, as the resolution of the images and resulting memory requirements are substantially larger than those of the CIFAR-10 dataset

- **Learning Rate**: The base learning rate for SimSiam and the DiSiam methods is set to 0.03. This is then scaled linearly with the batch size taking 256 as the base batch size. The final learning rate is calculated as base_lr $\times \frac{\text{batch\_size}}{256}$. For SimCLR, the base learning rate is set to 0.075 and scaled with a square root scaling. The resulting learning rate is calculated as base_lr $\times \sqrt{\text{batch\_size}}$. A robustness analysis is conducted using various learning rates in Subsection 5.3.2.

- **Optimizer**: The negative-sample-free frameworks SimSiam, DiSiam, and DiSiam+ are trained using Stochastic Gradient Descent [95] with a momentum of 0.9 and a weight decay of 10^{-4}. SimCLR is trained using the LARS [118] optimizer with a momentum of 0.9 and a weight decay of 10^{-6}.

- **Training Epochs**: For experiments with the CIFAR-10 dataset, the models are trained for 800 epochs, while for the COCO 2017 dataset, the training is conducted for 250 epochs.

- **Learning Rate Scheduler**: For the SimCLR method, the learning rate follows a linear warmup for the first 10 epochs, and then decays with a cosine decay schedule [76] as indicated in the original paper [21]. For SimSiam, DiSiam, and DiSiam+, a cosine decay schedule without warmup is used as described in the SimSiam paper [24, 76], decaying to zero over the course of the training.

4.4 Evaluation Protocol

The evaluation of the quality of the learned representations follows those protocols:

- **k-Nearest Neighbors**: For assessing the semantic information captured by the backbones, the learned representations are evaluated using a k-Nearest Neighbors classifier with $k = 200$, and $\tau = 0.1$. The Top-1 and Top-5 accuracy indicates the discriminative power of the learned features. To avoid cluttering the results

section, only the Top-1 kNN accuracy is reported leaving out the Top-5 accuracy, as both show similar behavior across different studies.

- **Object Detection Head Finetuning**: For evaluating the transfer learning capabilities of the learned features to object detection tasks, the backbone weights are used to initialize the backbone of the object detection models. The object detection models including the backbone are finetuned on the training split of the dataset. The number of epochs and the learning rate schedule is indicated in the results section. For 2D object detection the performance is measured with mean Average Precision (mAP) at an IoU threshold of 0.5 (mAP@0.5). For 3D object detection, the performance can be measured with mAP at different IoU thresholds (mAP@0.5, mAP@0.25, mAP@0.75) and the nuScenes detection score (NDS) [15]. The mAP is calculated for the 3D object detection task on the nuScenes dataset and for the 2D object detection task on the OpenImages dataset.

4.5 Self-Supervised Learning: Method Variants

Several methods are used in the experiments using different backbones. The following frameworks are evaluated:

- **SimCLR**: The contrastive SimCLR framework [21] is used as a baseline for comparison.
- **SimSiam**: The negative-sample-free framework SimSiam [24] is used as a baseline for comparison.
- **DiSiam**: The proposed negative-sample-free framework DiSiam using dynamic knowledge distillation.
- **DiSiam+**: The proposed negative-sample-free framework DiSiam+ using dynamic knowledge distillation and multiple image augmentations.

For ablation studies detailed in Section 4.6 the models in Table 4.1 are used as backbones and potentially assistant backbones of the SSL models.

Even though the original ResNet paper [50] uses a distinct architecture for CIFAR-10 where the first convolution layer is a convolutional layer with kernel size 3x3 instead of a 7x7 layer and the first max pooling layer is removed [50], the normal ResNet architecture is used even for the CIFAR-10 dataset to ensure comparability with the models pretrained on COCO 2017.

Backbone	Number Parameters
EfficientNet-B0	5,288,548
EfficientNet-B1	7,794,184
EfficientNet-B2	9,109,994
EfficientNet-B3	12,233,232
EfficientNet-B4	19,341,616
EfficientNet-B5	30,389,784
EfficientNet-B6	43,040,704
EfficientNet-B7	66,347,960
ResNet-18	11,689,512
ResNet-34	21,797,672
ResNet-50	25,557,032
ResNet-101	44,549,160
ResNet-152	60,192,808

Table 4.1: The backbone architectures and their number of parameters. The EfficientNet and ResNet architectures were introduced by Tan and Le [102] and He et al. [50] respectively.

4.6 Ablation Study Design

A series of ablation studies are made to isolate the impact of different design choices on the performance of the models. The following aspects are assessed:

- **Backbone**: The impact of different backbone architectures and sizes on the performance of the models.
- **Assistant Backbone**: The impact of different assistant backbones on the performance of the models.
- **Learning Rate**: The impact of different learning rates on the performance of the models.
- **Chaining Method**: The impact of the chaining method on the performance of the models.
- **Projection Layer**: The impact of the number of projection layers on the performance of the models.
- **Batch Size**: The impact of different batch sizes on the performance of the models.

- **Learning Rate Scheduler**: The impact of different learning rates on the performance of the models.

The ablation studies are conducted using the CIFAR-10 dataset for classification tasks and the COCO 2017 dataset for object detection tasks. The results of the ablation studies are presented in Section 5.3 and Section 5.4.

4.7 Establishing Baselines for Comparison

Clear baselines are essential for evaluating performance. For image classification tasks, random guessing is used as the first baseline, followed by random initialization, which can be empirically significantly higher than random guessing accuracy. To demonstrate this result, involved experiments were done with the CIFAR-10 dataset. The upper baseline consists of ImageNet-pretrained weights provided by the PyTorch library [91]. For EfficientNet-B0, those weights were produced by supervised pretraining followed by self-supervised pretraining, as described in [114]. For ResNet-18, the features are learned solely through supervised pretraining on the ImageNet dataset [63]. Furthermore, the upper baseline for CIFAR-10 is extended by upscaling the input images to 224x224 pixels, which is the input resolution the PyTorch models were initially pretrained on. This ensures a fair comparison to the SSL methods, which are trained on the CIFAR-10 resolution.

For object detection tasks, the upper and lower baselines is the performance of the fine-tuned object detection models when initializing them with the ImageNet-pretrained PyTorch weights and the default Kaiming He random initialization scheme [51] respectively.

4.8 Kaiming He Initialization: Details and Explanation

As the Kaiming He initialization [51] forms the basis of the random initialization baseline experiments, this section revisits important concepts and introduces new perspectives.

The Kaiming He method initializes weights from a Gaussian distribution $\mathcal{N}(0, std^2)$, where its standard deviation is defined as

$$\text{std} = \frac{\text{gain}}{\sqrt{\text{fan\_in}}} \quad \text{or} \quad \text{std} = \frac{\text{gain}}{\sqrt{\text{fan\_out}}}$$

for *Fan In* and *Fan Out* mode respectively. fan_in and fan_out describes the number of input and output dimensions of the weight matrix, assuming the weights are applied

via transposed multiplication (i.e. $x@w^\top$). This means, that the variance differs across layers. The gain scales the standard deviation and is typically set to $\sqrt{2}$ for ReLU activations to preserve variance across layers [51].

This thesis introduces the *gain ratio*, a hyperparameter that multiplies the base gain and allows for scaling the initialization variance. The final standard deviation is computed as

$$\text{std} = \text{gain\_ratio} \cdot \frac{\text{gain}}{\sqrt{\text{fan\_in}}} \quad \text{or} \quad \text{std} = \text{gain\_ratio} \cdot \frac{\text{gain}}{\sqrt{\text{fan\_out}}}$$

depending on the fan mode.

4.9 Object Detection Model Architectures

4.9.1 3D Object Detection Model Architecture

As described in the literature review in Chapter 2, the FCOS3D [109] is used for the 3D object detection task on the nuScenes dataset. It is a fully convolutional, anchor-free, single-stage detector that directly regresses 3D object properties - such as depth, orientation, dimensions, and 2D center - from single images. FCOS3D extends the 2D FCOS [105] detector by adding 3D-specific heads. The model is trained using multiple loss functions for the different prediction heads. As detailed in Section 2.5, the thesis focuses on advancing SSL methods for perception tasks in autonomous driving, rather than evaluating SSL methods on the 3D object detection model.

4.9.2 2D Object Detection Model Architecture

The 2D object detection model used in the thesis is a proprietary model that is based on an end-to-end CNN architecture [107].

4.10 Hardware and Computational Resources

4.10.1 Hardware Setup

For the pretraining tasks and classification evaluations, single H100 or A100 GPUs from the clusters of Leibniz-Rechenzentrum [67] are utilized. Evaluation on the object detection tasks is performed on single T4 GPUs.

4.10.2 Software Stack

The experiments are implemented using the PyTorch framework [91] and the torchvision library [79]. The SSL methods are implemented using the PyTorch Lightning [36] and lightly library [101]. GPU acceleration is enabled through the CUDA toolkit [87] and the cuDNN library [25]. The codebase is written in Python 3.12 [37]. Experiment logging and visualization is done using the TensorBoard library [1]. On top of that, to visualize the learned features TriMap [3] is used.

5 Results: Analyzing Model Performance and Transferability

This chapter presents the results of the thesis, comparing the proposed method *DiSiam* and *DiSiam+* with the SSL methods SimSiam and SimCLR across various scenarios. The results are organized into several sections. Starting with demonstrating a lower baseline using randomly initialized models on the CIFAR-10 dataset in Section 5.1, context for the following analyses is provided, followed by a detailed comparison of the SSL methods on CIFAR-10, including ablation and robustness studies on crucial hyperparameters and architectural choices in Section 5.2 and Section 5.3. Following this, the transferability of the learned feature representations is assessed by pretraining on yet another dataset in Section 5.4 and extending the evaluation to 2D object detection in Section 5.6. Finally, the key findings of the experiments are summarized in Section 5.7.

5.1 Beyond Random Guessing: Establishing the Random Baseline on CIFAR-10

To provide a basis on which to meaningfully interpret the following results, a baseline is first established by using a kNN classifier on features extracted from randomly initialized backbones. As discussed in Section 4.7, the upper performance baseline is given by the weights provided by the PyTorch library [91].

Contrary to the common expectation that random initialization on a 10-class dataset like CIFAR-10 should yield approximately 10% accuracy (like random guessing), the conducted experiments reveal markedly higher performance for specific widths of the Gaussian initialization distribution. This particularly suggests that even with millions of randomly initialized parameters, the input space characteristics are surprisingly well preserved in the latent lower-dimensional feature space. Various combinations of gain ratios, fan modes, and backbone models are evaluated. Each experiment is repeated using six different random seeds (42, 1337, 2025, 31415, 271828, 123456789) to ensure statistical robustness.

Figure 5.1 and Table 5.1 show the results of feature representations for different randomly initialized backbones and architectures. The backbone size correlates positively

with the kNN Top-1 accuracy, particularly for EfficientNet architectures, showing that larger models tend to preserve the input structure better than smaller ones. The same is true if a randomly initialized classifier is added to the model, as shown in Figure 5.2 and Table 5.2. Remarkably, ResNet architectures consistently outperform EfficientNet architectures, even though EfficientNet models have shown superior performance in supervised learning [102]. The maximum Top-1 kNN accuracy for feature vectors is 39.83% and is observed for the ResNet-152 backbone using the *Fan Out* Mode, with a *gain ratio* of 1.5 and thus the final gain value of 2.12. The achieved accuracy is four times higher than that of random guessing on CIFAR-10.

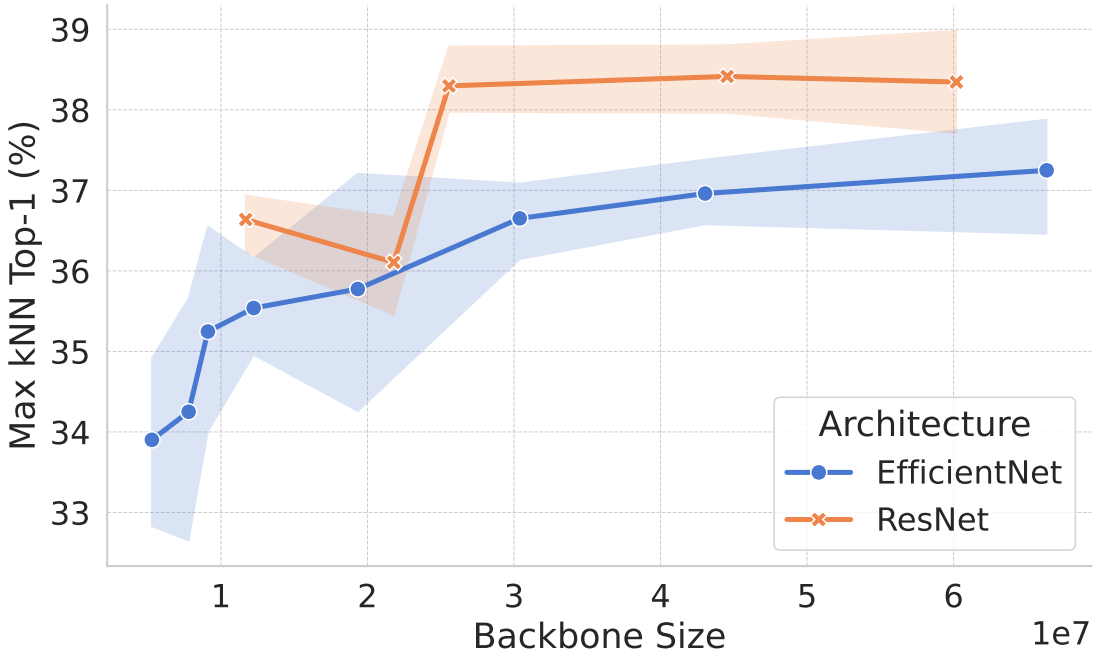


Figure 5.1: kNN Top-1 accuracy on CIFAR-10 for the best hyperparameter settings of randomly initialized models without their classifier layers. Models are initialized using the Kaiming He schemes [51] with varying gain values. Accuracy is plotted against model size. Results are averaged over six random seeds and the shaded area signifies the 95% confidence interval. ResNet architectures outperform EfficientNet ones, despite EfficientNet's superior performance in supervised learning [102]. Proper initialization improves accuracy up to four-fold compared to random guessing.

ResNet				
Backbone	*Fan Mode*	*Gain Ratio*	*Top-1 Accuracy (%)*	*Model Size (M)*
ResNet-18	Fan In	0.90	36.64 ± 0.48	11.7
ResNet-34	Fan In	0.70	36.11 ± 0.88	21.8
ResNet-50	Fan Out	0.80	38.30 ± 0.58	25.6
ResNet-101	Fan Out	0.70	**38.41 ± 0.58**	44.5
ResNet-152	Fan Out	0.70	**38.34 ± 0.89**	60.2
EfficientNet				
Backbone	*Fan Mode*	*Gain Ratio*	*Top-1 Accuracy (%)*	*Model Size (M)*
EfficientNet-B0	Fan Out	1.20	33.90 ± 1.51	5.3
EfficientNet-B1	Fan Out	1.20	34.25 ± 2.07	7.8
EfficientNet-B2	Fan Out	1.20	35.25 ± 1.71	9.1
EfficientNet-B3	Fan Out	1.20	35.54 ± 0.87	12.2
EfficientNet-B4	Fan Out	1.20	35.77 ± 2.02	19.3
EfficientNet-B5	Fan Out	1.30	36.65 ± 0.66	30.4
EfficientNet-B6	Fan Out	1.90	**36.96 ± 0.54**	43.0
EfficientNet-B7	Fan Out	2.30	**37.25 ± 0.99**	66.3

Table 5.1: Numerical kNN Top-1 accuracy values on CIFAR-10 corresponding to the models without a classifier in Figure 5.1. Model size is shown in millions of parameters. The best two results are highlighted in bold. Reported values are computed as the mean ± standard deviation over six random seeds.

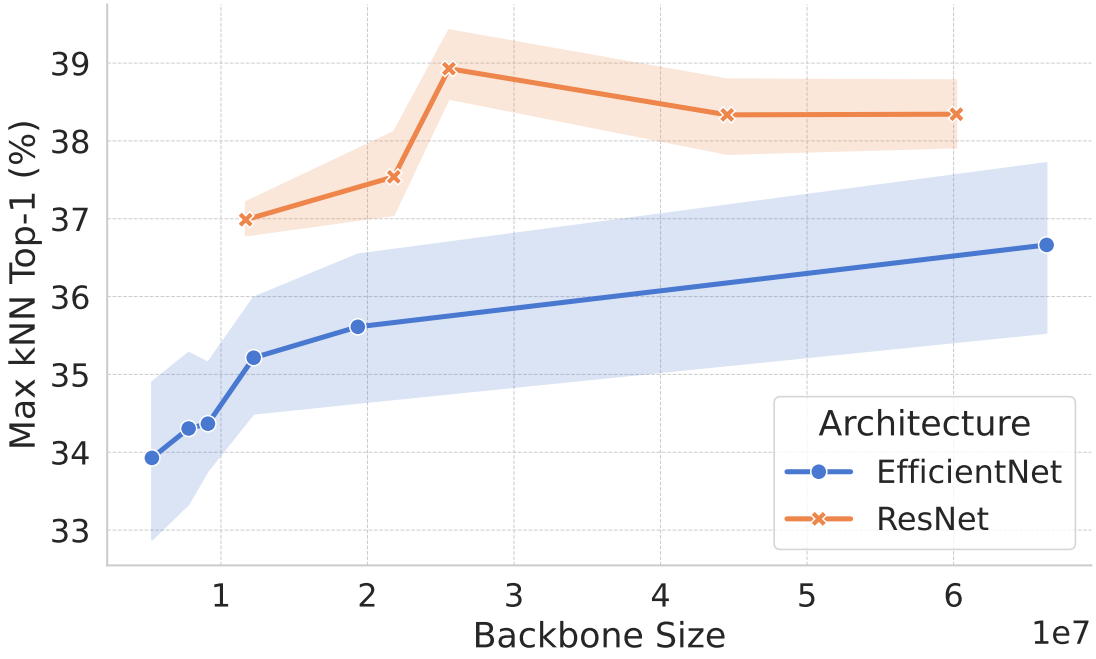
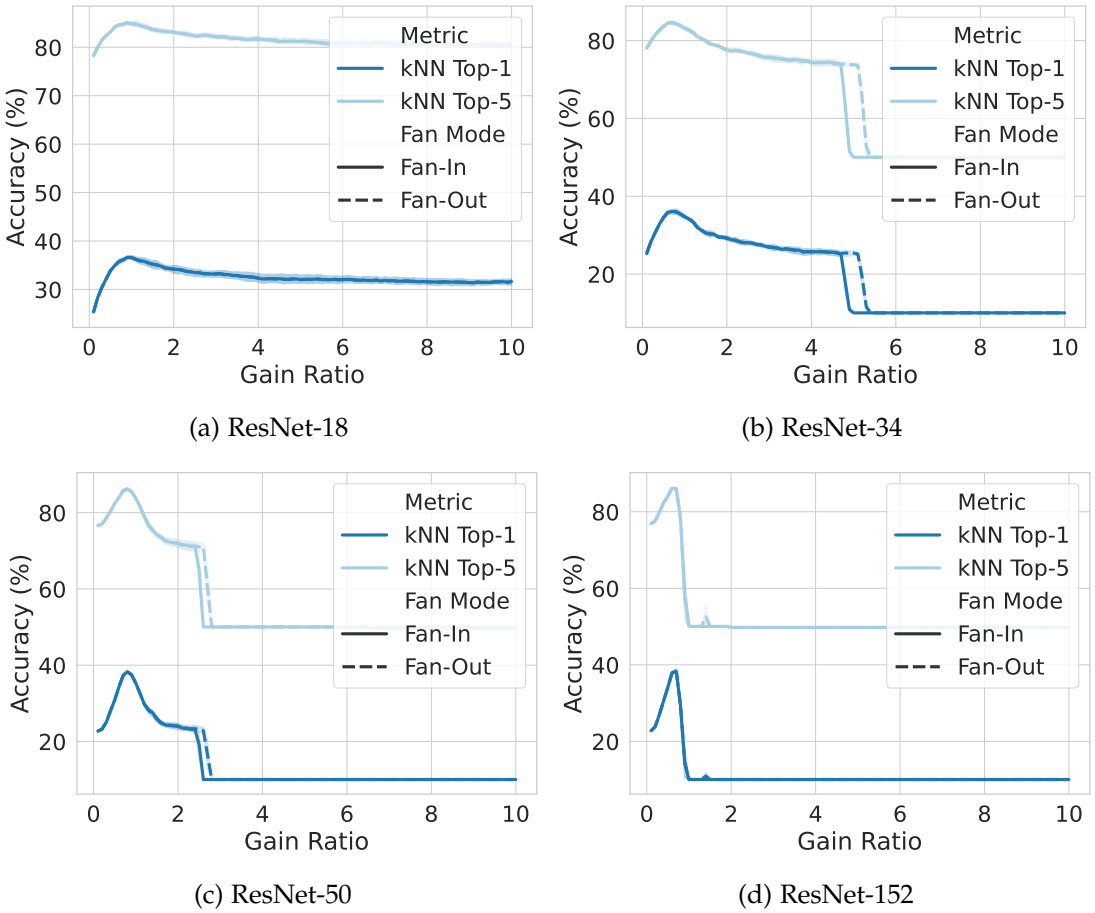


Figure 5.2: kNN Top-1 accuracy on CIFAR-10 for the best hyperparameter settings of randomly initialized models using also their classifier. Models are initialized using the Kaiming He schemes [51] with varying gain values. Accuracy is plotted against model size. Results are averaged over six random seeds and the shaded area signifies the 95% confidence interval. ResNet architectures outperform EfficientNet ones, despite EfficientNet's superior performance in supervised learning [102]. Proper initialization improves accuracy up to four-fold compared to random guessing.

To gain qualitative insights into the impact of initialization, the performance of different architectures and model sizes are evaluated as a function of different gain ratios. Figure 5.3 shows the kNN Top-1 accuracies of the feature representations using ResNet models, and Figure 5.4 and Figure 5.5 using EfficientNet models. Furthermore, Figure 5.6 and Figure 5.7 demonstrates the results for models with the randomly initialized default classifier.

ResNet				
Backbone	*Fan Mode*	*Gain Ratio*	*Top-1 Accuracy (%)*	*Model Size (M)*
ResNet-18	Fan Out	0.90	36.99 ± 0.31	11.7
ResNet-34	Fan Out	0.80	37.54 ± 0.70	21.8
ResNet-50	Fan Out	0.80	**38.93 ± 0.61**	25.6
ResNet-101	Fan Out	0.80	38.33 ± 0.63	44.5
ResNet-152	Fan Out	0.70	**38.34 ± 0.59**	60.2
EfficientNet				
Backbone	*Fan Mode*	*Gain Ratio*	*Top-1 Accuracy (%)*	*Model Size (M)*
EfficientNet-B0	Fan Out	1.20	33.93 ± 1.44	5.3
EfficientNet-B1	Fan Out	1.20	34.31 ± 1.38	7.8
EfficientNet-B2	Fan Out	1.20	34.37 ± 0.96	9.1
EfficientNet-B3	Fan Out	1.30	35.21 ± 0.99	12.2
EfficientNet-B4	Fan Out	1.30	**35.61 ± 1.39**	19.3
EfficientNet-B7	Fan In	0.50	**36.66 ± 1.53**	66.3

Table 5.2: Numerical kNN Top-1 accuracy values on CIFAR-10 corresponding to the models with classifier in Figure 5.2. Model size is shown in millions of parameters. The best two results are highlighted in bold. Reported values are computed as the mean ± standard deviation over six random seeds.



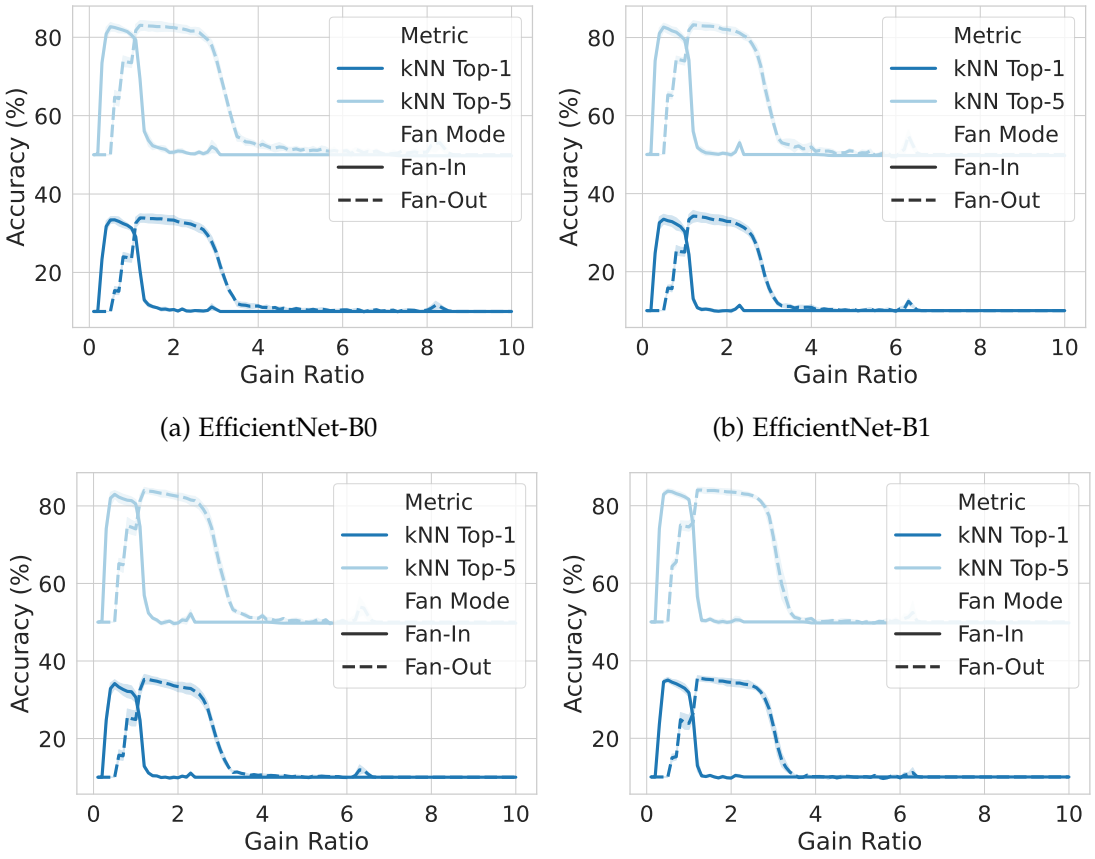
(a) ResNet-18

(b) ResNet-34

(c) ResNet-50

(d) ResNet-152

Figure 5.3: Top-1 kNN accuracy curve of different ResNet models without their classifiers to the initial gain ratio in random Kaiming He initialization on CIFAR-10 corresponding to Figure 5.1. Small ResNet models remain robust to high-variance initializations, larger ones become increasingly sensitive. Results are averaged over six random seeds per hyperparameter setting and the shaded area signifies the 95% confidence interval. The final gain value is computed as the standard Kaiming gain factor for ReLUs ($\sqrt{2}$) multiplied by the gain ratio.



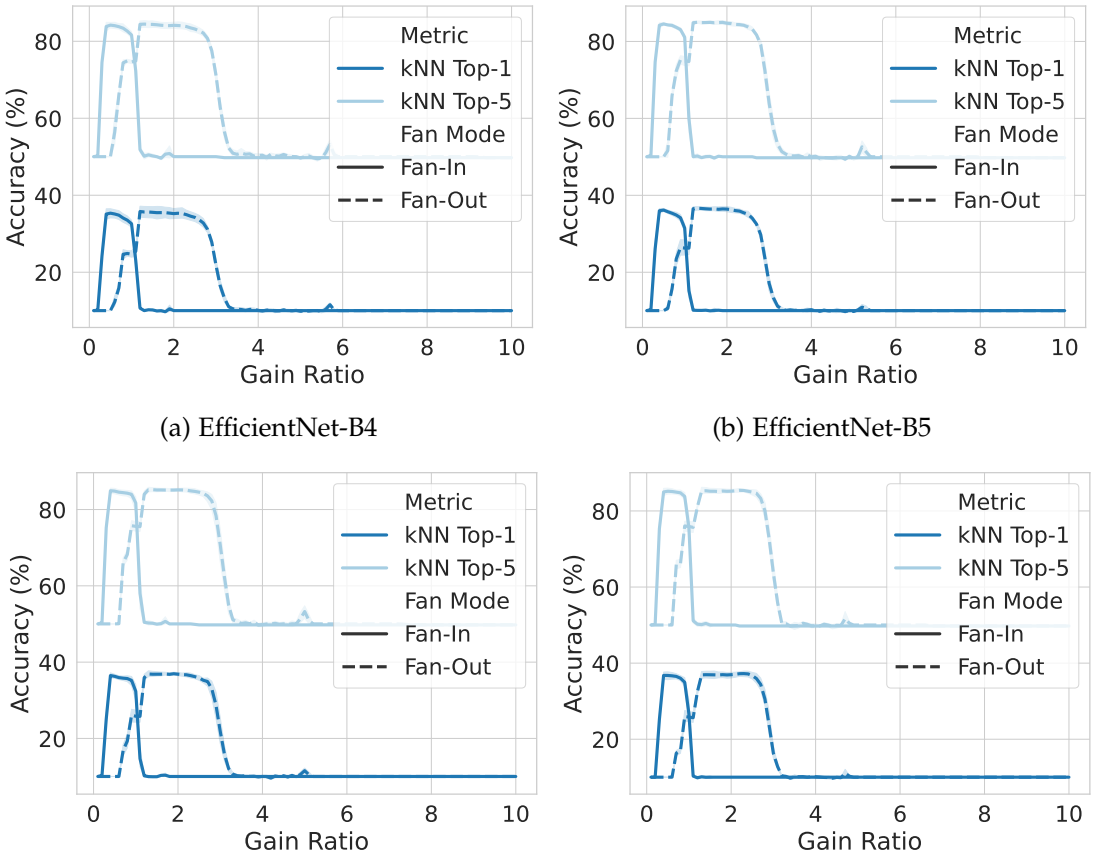
(a) EfficientNet-B0

(b) EfficientNet-B1

(c) EfficientNet-B2

(d) EfficientNet-B3

Figure 5.4: Top-1 kNN accuracy curve of different EfficientNet models without their classifiers to the initial gain ratio in random Kaiming He initialization on CIFAR-10. EfficientNet models show strong scalable robustness, maintaining wide accuracy peaks at even large model sizes. Results are averaged over six random seeds per hyperparameter setting and the shaded area signifies the 95% confidence interval. The gain value is computed as the standard Kaiming factor for ReLUs ($\sqrt{2}$) multiplied by the gain ratio.



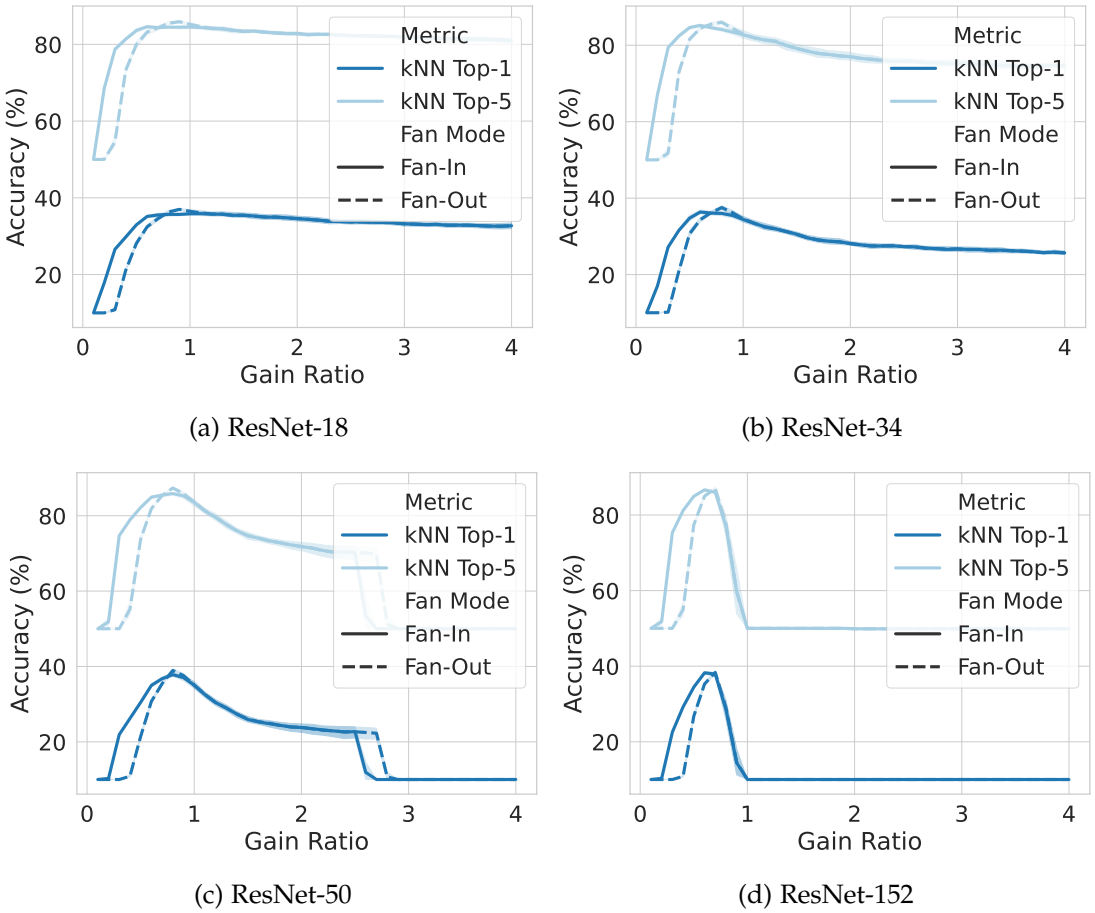
(a) EfficientNet-B4

(b) EfficientNet-B5

(c) EfficientNet-B6

(d) EfficientNet-B7

Figure 5.5: Top-1 kNN accuracy curve of different EfficientNet models without their classifiers to the initial gain ratio in random Kaiming He initialization on CIFAR-10. EfficientNet models show strong scalable robustness, maintaining wide accuracy peaks at even large model sizes. Results are averaged over six random seeds per hyperparameter setting and the shaded area signifies the 95% confidence interval. The gain value is computed as the standard Kaiming factor for ReLUs ($\sqrt{2}$) multiplied by the gain ratio.



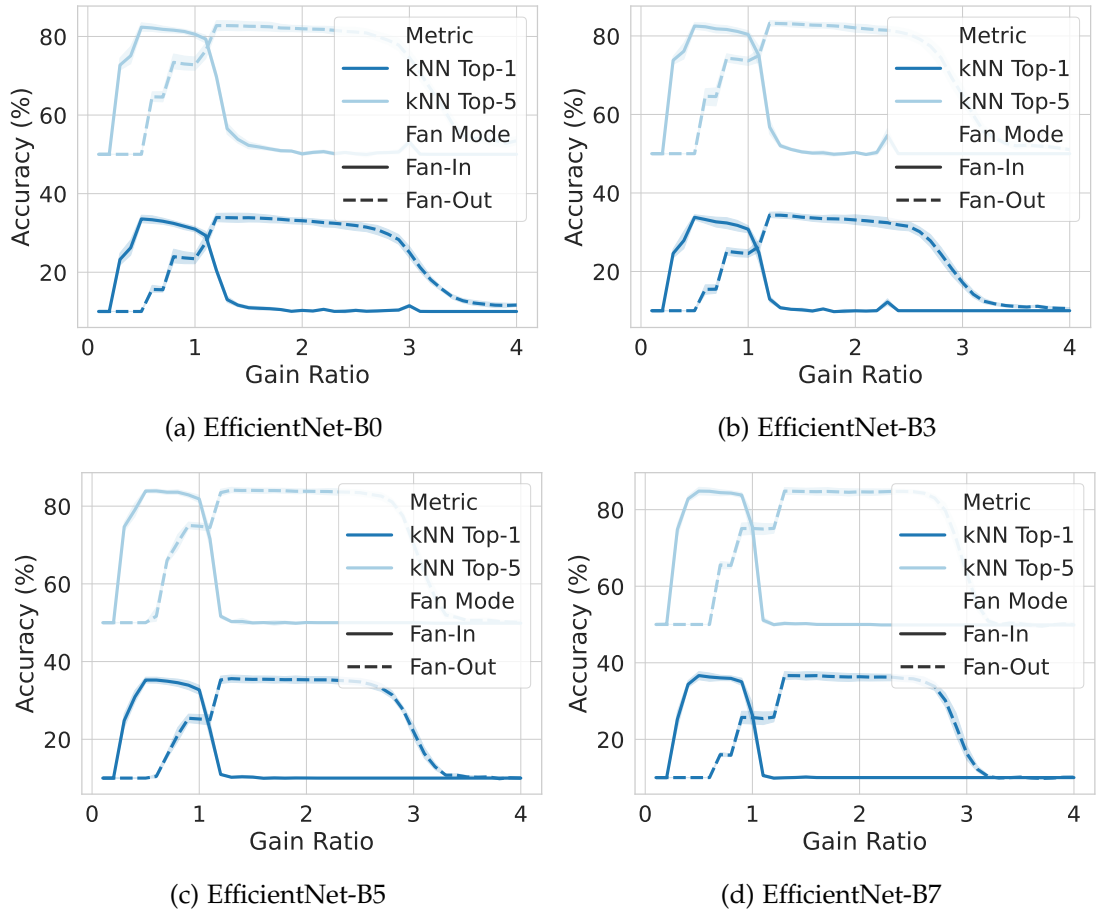
(a) ResNet-18

(b) ResNet-34

(c) ResNet-50

(d) ResNet-152

Figure 5.6: Top-1 kNN accuracy curve of different ResNet models with their classifiers to the initial gain ratio in random Kaiming He initialization on CIFAR-10 corresponding to Figure 5.2. Small ResNet models remain robust to high-variance initializations, larger ones become increasingly sensitive. Results are averaged over six random seeds per hyperparameter setting and the shaded area signifies the 95% confidence interval. The final gain value is computed as the standard Kaiming factor for ReLUs ($\sqrt{2}$) multiplied by the gain ratio.



(a) EfficientNet-B0

(b) EfficientNet-B3

(c) EfficientNet-B5

(d) EfficientNet-B7

Figure 5.7: Top-1 kNN accuracy curve of different EfficientNet models with their classifiers to the initial gain ratio in random Kaiming He initialization on CIFAR-10. EfficientNet models show strong scalable robustness, maintaining wide accuracy peaks at even large model sizes. Results are averaged over six random seeds per hyperparameter setting and the shaded area signifies the 95% confidence interval. The gain value is computed as the standard Kaiming factor for ReLUs ($\sqrt{2}$) multiplied by the gain ratio.

The plots demonstrate a significant sensitivity of performance to the variance of the initialization Gaussian. Both too small and too large gain values lead to substantial performance loss. The low performance decline against high-variance initialization suggests a greater robustness against initialization choices of the ResNet-18 architecture. As the three smallest EfficientNet architectures have fewer parameters than the ResNet-18 model and do not exhibit the same behavior, the reason for the robustness against

high variance does not come from the low number of parameters in the model. This emphasizes the crucial role of network architecture choice in random initialization performance.

To quantitatively analyze the robustness of the models, the full width half maximum (FWHM) of the models with classifiers is calculated and compared. The FWHM is a measure of the width of the curve at half its maximum height, and it provides an indication of the sensitivity of the model to initialization choices. A smaller FWHM indicates that the model is more sensitive to initialization choices, while a larger FWHM indicates that the model is more robust. The FWHM values for each model are shown in Table 5.3 and Figure 5.8.

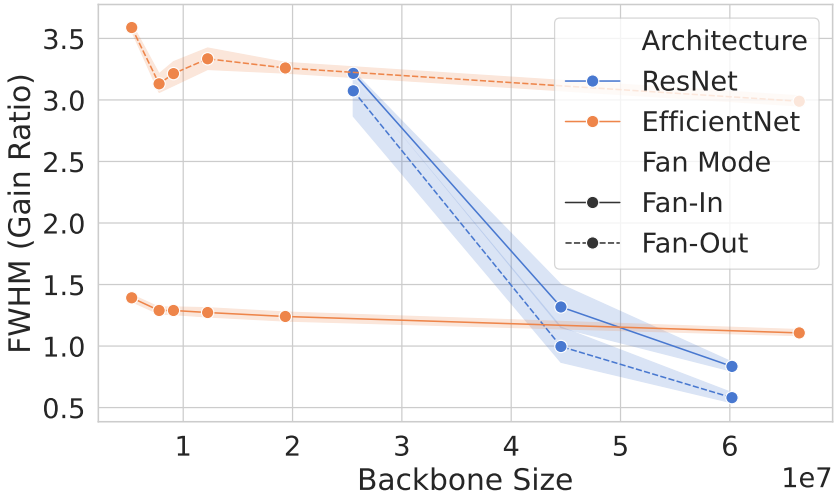


Figure 5.8: Full width at half maximum (FWHM) values for different models with their classifier. FWHM is a measure of the width of the accuracy peak with respect to the gain ratio. Smaller FWHM suggest higher sensitivity to the initialization variance, while larger ones indicate higher robustness. The FWHM is computed as the width of the curve at half its maximum height. The corresponding accuracy curves are shown in Figure 5.7 and Figure 5.6. The ResNet-18 and ResNet-34 models are excluded, as their accuracy did not fall below half the maximum on the right side of the peak. Results are averaged over six random seeds per hyperparameter setting and the shaded area signifies the 95% confidence interval.

For further analysis, the feature representations for a low-performing gain ratio of 2.6 (gain value $\sim$ 3.68 and kNN Top-1 performance of $\sim$ 10%) and a high-performing

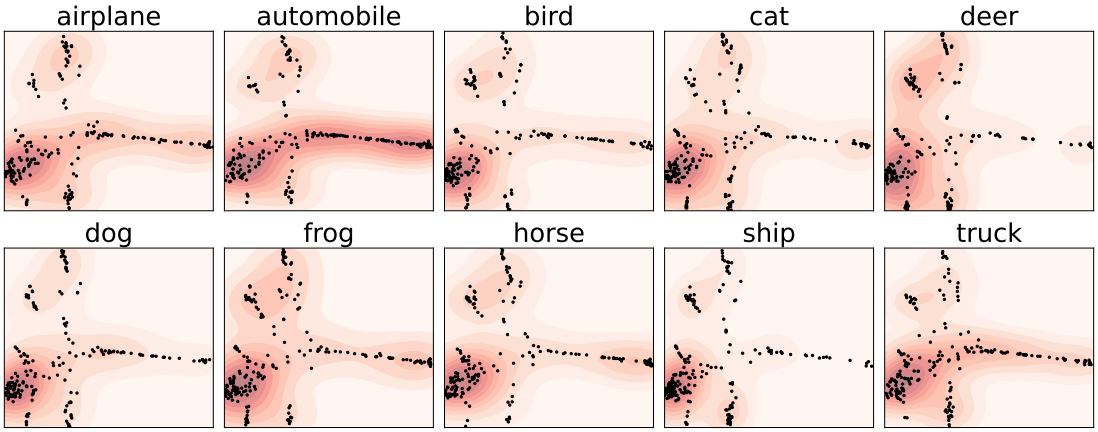
Fan In Mode		
Backbone	*FWHM*	*Model Size (M)*
ResNet-50	**3.22 ± 0.02**	25.6
ResNet-101	1.32 ± 0.23	44.5
ResNet-152	0.83 ± 0.05	60.2
EfficientNet-B0	1.39 ± 0.04	5.3
EfficientNet-B1	1.29 ± 0.04	7.8
EfficientNet-B2	1.29 ± 0.04	9.1
EfficientNet-B3	1.27 ± 0.05	12.2
EfficientNet-B4	1.24 ± 0.05	19.3
EfficientNet-B7	1.11 ± 0.03	66.3
Fan Out Mode		
Backbone	*FWHM*	*Model Size (M)*
ResNet-50	3.22 ± 0.25	25.6
ResNet-101	1.32 ± 0.21	44.5
ResNet-152	0.83 ± 0.05	60.2
EfficientNet-B0	**3.59 ± 0.04**	5.3
EfficientNet-B1	3.13 ± 0.11	7.8
EfficientNet-B2	3.21 ± 0.13	9.1
EfficientNet-B3	3.33 ± 0.12	12.2
EfficientNet-B4	3.26 ± 0.06	19.3
EfficientNet-B7	2.99 ± 0.06	66.3

Table 5.3: Numerical Full width at half maximum (FWHM) values for different randomly initialized models with their classifiers corresponding to Figure 5.8. FWHM describes the width of the accuracy peak with respect to the gain ratio. Smaller values suggest higher sensitivity to initialization, while larger values indicate higher robustness. The ResNet-18 and ResNet-34 models are excluded, as their accuracy did not fall below half the maximum on the right side of the peak. Reported values are computed as the mean ± standard deviation over six random seeds. The best result is highlighted in bold.

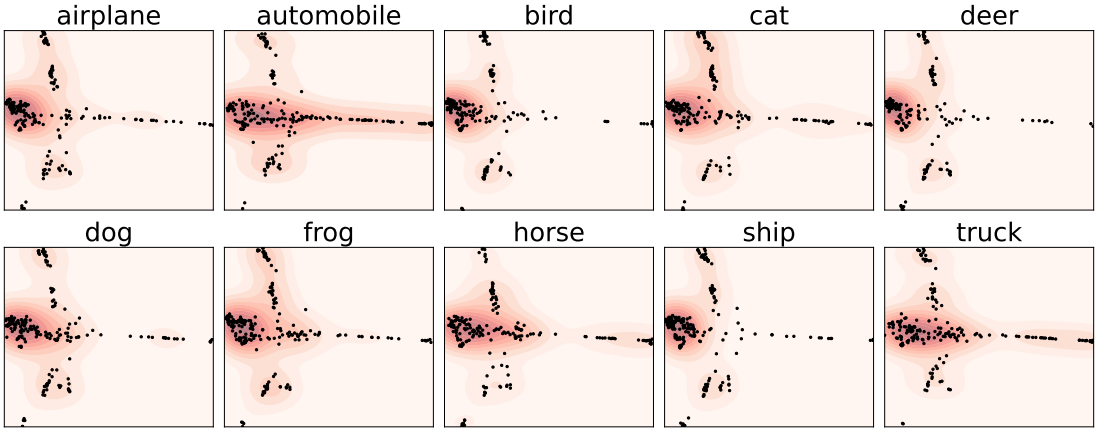
gain ratio of 1.5 (gain value $\sim$ 2.12 and kNN Top-1 performance of $\sim$ 36%) using the EfficientNet-B0 backbone are plotted in Figure 5.9 and Figure 5.10.

Those feature visualizations use the TriMap [3] algorithm to project high-dimensional latent representations into a two-dimensional space. TriMap is a nonlinear dimensionality reduction method that constructs a set of triplets (i, j, k), where point i is closer to point j than to point k in the original space. For each data point i, a fixed number of nearest neighbors j are selected using approximate nearest neighbor search, and distant points k are sampled to form informative triplets. Furthermore, additional random triplets are included. The low-dimensional embedding is created by minimizing a loss function over the sampled triplet constraints. The resulting axes of the two-dimensional embedding space are abstract and do not correspond to any specific feature dimensions - thus, axis labels are omitted.

While the feature representations for the gain ratio 1.5 show more distinct class-wise densities, both projected latent spaces exhibit limited class separation. Similarity metrics for the approximated class-wise densities underscore this observation with an average symmetric Kullback-Leibler divergence of 0.1128 and Jensen-Shannon Divergence of 0.0487 for gain ratio 1.5, and 0.0807 and 0.0353 respectively for gain ratio 2.6, showing only slight differences.

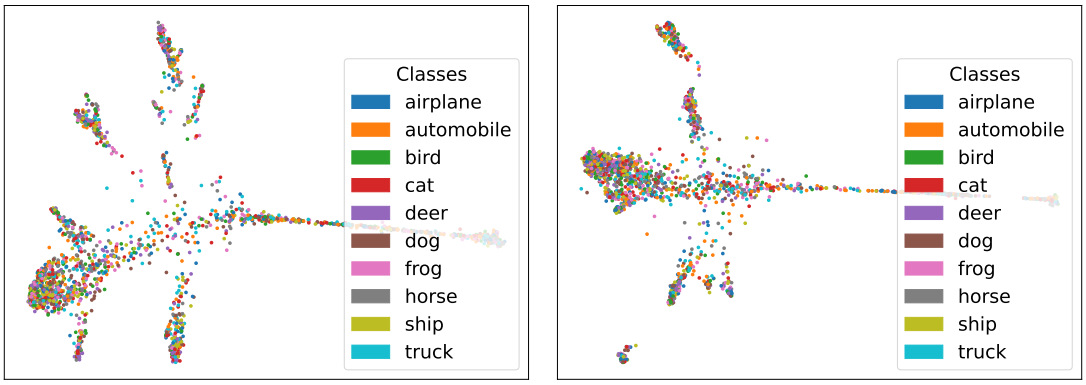


(a) Gain Ratio 2.6



(b) Gain Ratio 1.5

Figure 5.9: Feature representation subplots for each CIFAR-10 class using a randomly initialized EfficientNet-B0 with different gain ratios. Representations are projected to two dimensions using TriMap [3], and a Gaussian kernel is used for density estimation.



(a) Gain Ratio 2.6

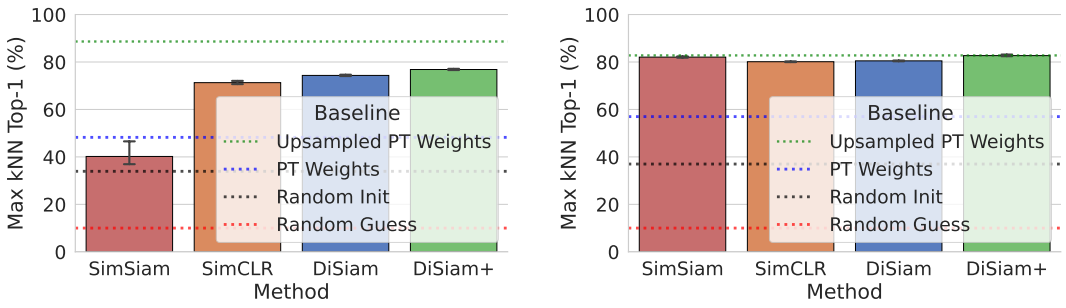
(b) Gain Ratio 1.5

Figure 5.10: Feature representations for each CIFAR-10 class, visualized using a randomly initialized EfficientNet-B0 backbone with different gain ratios. The plots correspond to Figure 5.9

5.2 Self-Supervised Learning on CIFAR-10: DiSiam's Advantage

The CIFAR-10 dataset [62] serves as the primary benchmark for performance evaluation and in-depth ablation studies of different SSL methods. While lightweight models like EfficientNet-B0 are preferred for real-time perception tasks in autonomous driving, results for ResNet-18 are also reported for completeness.

Using a ResNet-34 backbone as the assistant network for the proposed *DiSiam* methods, kNN Top-1 accuracies of 77.1% with an EfficientNet-B0 and 83.38% with a ResNet-18 backbone are achieved. Comparative performance results are presented in Figure 5.11.



(a) EfficientNet-B0

(b) ResNet-18

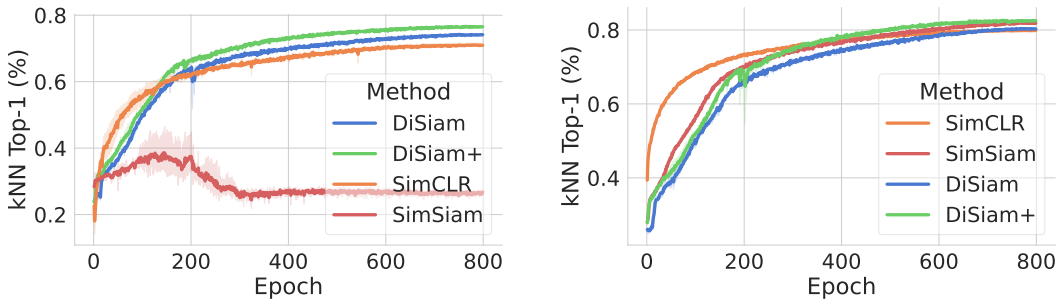
Figure 5.11: Maximum kNN Top-1 accuracy for different SSL methods on EfficientNet-B0 and ResNet-18, using the original hyperparameters from each paper and pretrained and evaluated on the CIFAR-10 dataset. DiSiam methods outperform SimSiam and SimCLR on EfficientNet-B0. DiSiam and DiSiam+ use ResNet-34 as an assistant backbone. Reported results are averaged over three random seeds (7, 41, 923), with error bars showing the 95% confidence interval.

The initial results show a significantly lower performance for SimSiam on EfficientNet-B0, where it can merely achieve 46.55% Top-1 accuracy. The proposed methods *DiSiam* and *DiSiam+* also outperform SimCLR on EfficientNet-B0, which reaches up to 72.04% Top-1 accuracy. Using the ResNet-18 backbone, the proposed method *DiSiam+* achieves up to 83.14% outperforming the upper baseline of the PyTorch weights (82.78%), which are pretrained on ImageNet and applied on the upsampled (224x224) CIFAR-10 images. Although the ResNet architecture is designed for a resolution of 224x224, the SSL methods are able to learn meaningful representations at a much smaller resolution 32x32. The results are even more impressive when considering that the PyTorch weights are pretrained on ImageNet and thus on a much larger dataset. The outcomes are numerically summarized in Table 5.4.

To qualitatively observe the accuracy curves of different SSL methods, Figure 5.12 shows the kNN Top-1 accuracy over training epochs for EfficientNet-B0 and ResNet-18 backbones respectively.

EfficientNet-B0	
Method	*Max kNN Top-1 (%)*
Random Initialization	33.90 ± 1.51
SimSiam	41.65 ± 5.36
SimCLR	71.29 ± 0.69
DiSiam	74.35 ± 0.28
DiSiam+	**76.84 ± 0.26**
PyTorch Weights	48.16
PyTorch Weights (Upsampling)	**88.64**
ResNet-18	
Method	*Max kNN Top-1 (%)*
Random Initialization	36.64 ± 0.48
SimSiam	82.16 ± 0.3
SimCLR	80.12 ± 0.16
DiSiam	80.48 ± 0.16
DiSiam+	**82.69 ± 0.39**
PyTorch Weights	56.98
PyTorch Weights (Upsampling)	**82.78**

Table 5.4: Maximum numerical kNN Top-1 accuracy results corresponding to Figure 5.11. Reported values are computed as the mean ± standard deviation over three random seeds (7, 41, 923). The two best results are highlighted in bold.



(a) EfficientNet-B0

(b) ResNet-18

Figure 5.12: kNN Top-1 accuracy during training for EfficientNet-B0 and ResNet-18, corresponding to Figure 5.11. DiSiam and DiSiam+ show consistently higher accuracy than SimSiam and SimCLR. Results are averaged over three random seeds, with the shaded area representing the 95% confidence interval.

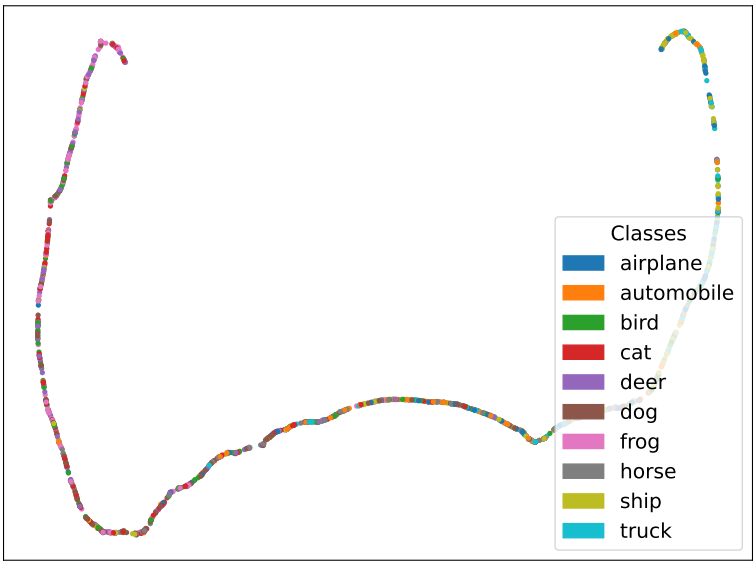
Figure 5.13, Figure 5.15, Figure 5.16 and Figure 5.14 show 2D TriMap [3] visualizations of the feature representations learned by different SSL methods using the EfficientNet-B0 backbone.

In Figure 5.13, feature representations learned by SimSiam lie on a narrow line, suggesting partial representational collapse, also known as dimensional collapse [57]. This indicates that SimSiam maps features to a lower-dimensional manifold, limiting their expressiveness and capacity to produce meaningful latent representations.

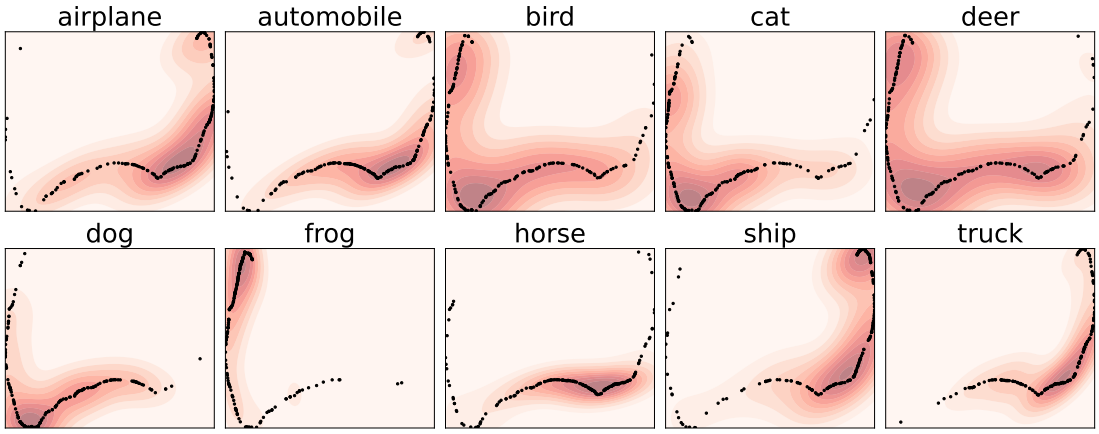
SimCLR (Figure 5.14) demonstrates well-separated class clusters, indicating strong latent preservation of semantic information. The contrastive loss with negative samples encourages not only to pull positive pairs together but also negative ones apart, which results in large inter-class distances. This becomes particularly apparent when observing the large distance between the industrial and natural domain classes of CIFAR-10. Solely the airplane class appears to be a false positive in the animal cluster, likely due to its visual similarity with birds.

DiSiam and DiSiam+ (Figure 5.15, Figure 5.16) also show strong class separability. The proposed methods avoid over-separating semantically unrelated classes, thus making more efficient use of the latent feature space compared to SimCLR.

The results suggest that the proposed methods are more effective in capturing meaningful feature representations for classification tasks. To further validate this hypothesis and gain deeper insights, the following sections present detailed analyses and ablation studies.



(a) Scatter Plot



(b) Subplots with Density Estimation

Figure 5.13: TriMap [3] visualization of SimSiam feature representations of EfficientNet-B0 on CIFAR-10 and a corresponding density estimation using Gaussian kernels. It shows limited class separability consistent with its kNN accuracy in Figure 5.11 and low dimensionality in its feature space, suggesting partial dimensional collapse [57].

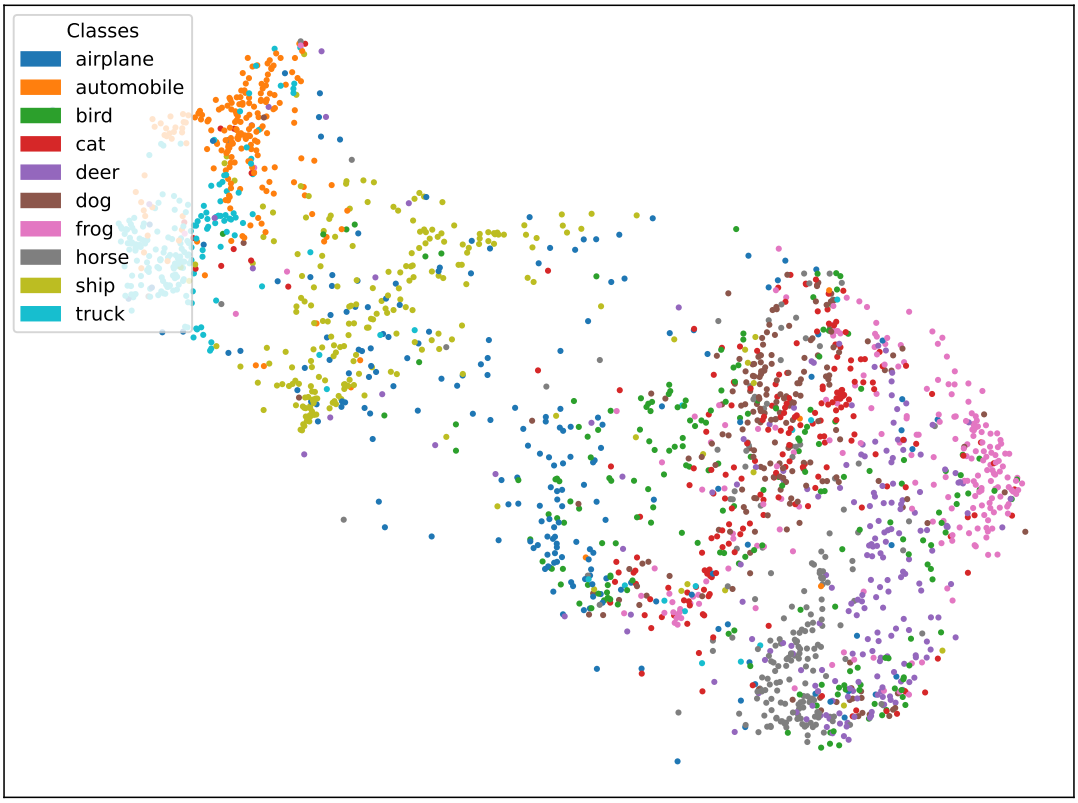


Figure 5.14: TriMap [3] visualization of SimCLR feature representations of EfficientNet-B0 on CIFAR-10. It shows solid class separability consistent with its kNN accuracy in Figure 5.11.

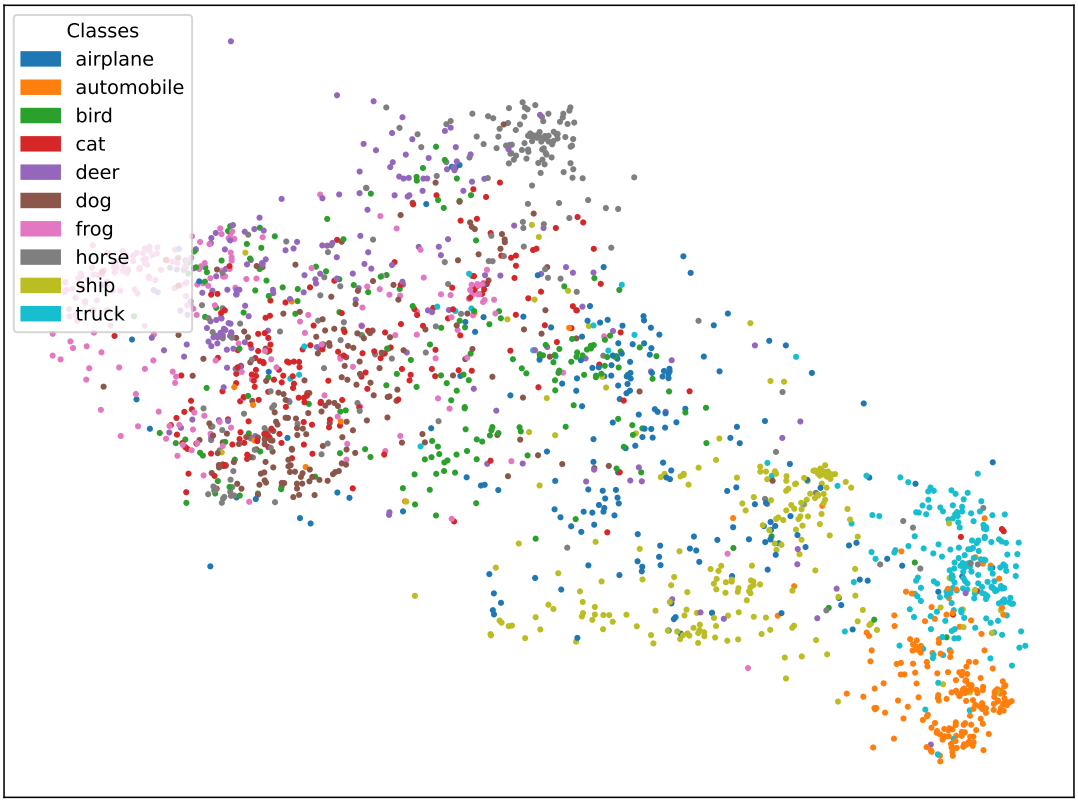


Figure 5.15: TriMap [3] visualization of DiSiam feature representations of EfficientNet-B0 on CIFAR-10. It shows strong class separability consistent with its kNN accuracy in Figure 5.11.

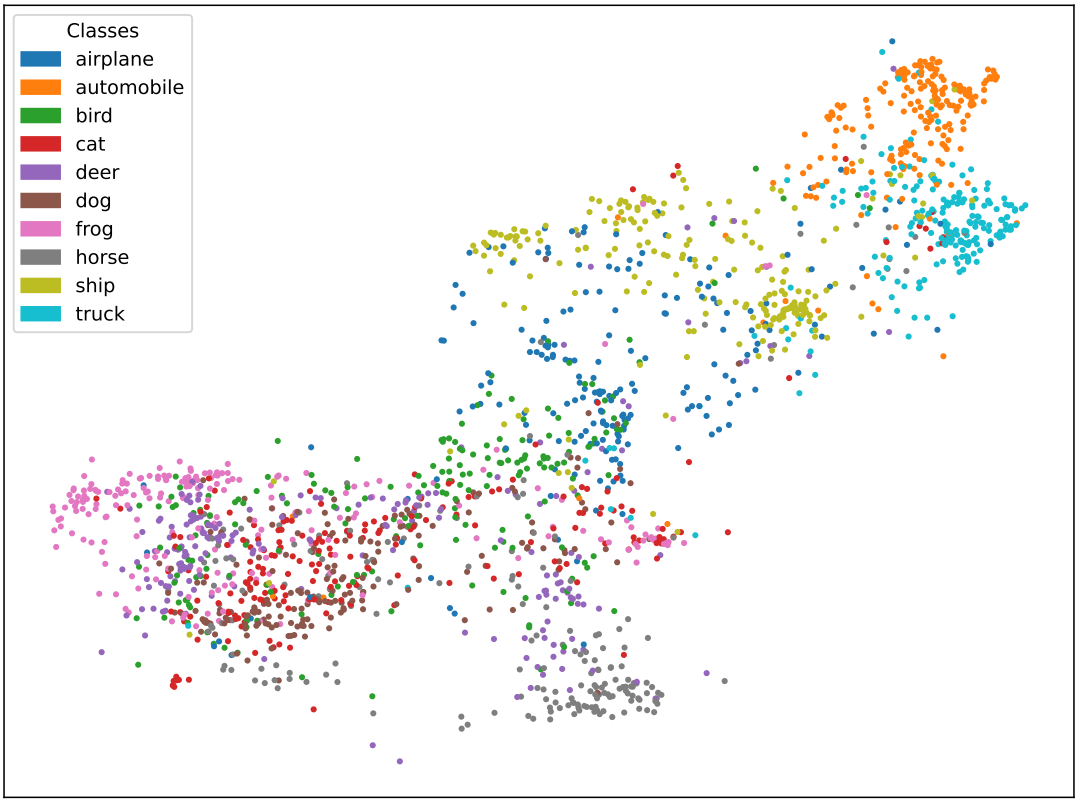
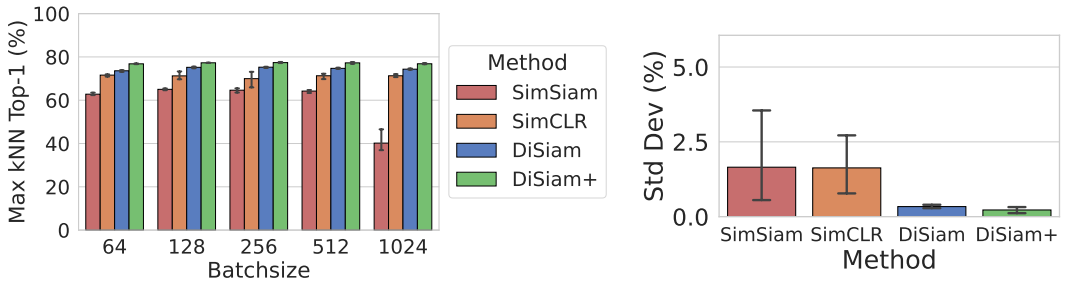


Figure 5.16: TriMap [3] visualization of DiSiam+ feature representations of EfficientNet-B0 on CIFAR-10. It shows strong class separability consistent with its kNN accuracy in Figure 5.11.



(a) Maximum kNN Top-1 Accuracy (b) Standard Deviation of Accuracy

Figure 5.17: (a) Maximum kNN Top-1 accuracy for various batch sizes on CIFAR-10 using EfficientNet-B0 as backbone and ResNet-34 as assistant backbone for DiSiam and DiSiam+ averaged across three random seeds (7, 41, 923). Error bars indicate 95% confidence interval. (b) Standard deviation per method of maximum kNN Top-1 accuracy averaged across different batch sizes with error bars indicating the 95% confidence interval.

5.3 Evaluating Self-Supervised Learning on CIFAR-10: Ablations and Performance Analysis

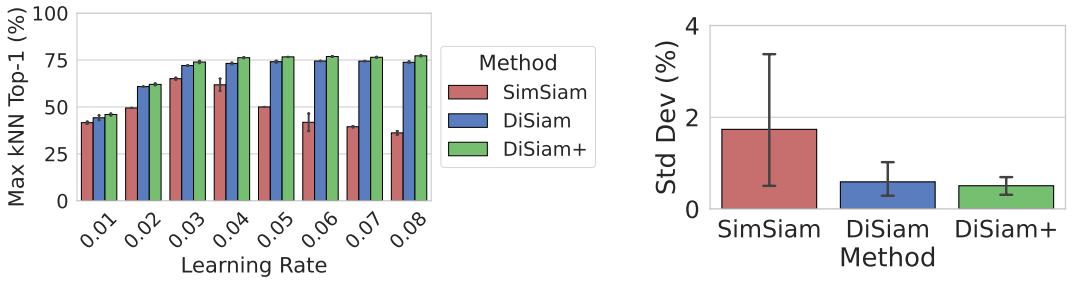
This section investigates the behavior of the baseline SSL methods SimSiam and SimCLR, and the proposed DiSiam methods in different settings and backbone architectures. It also aims to evaluate the robustness and sensitivity of those methods to different hyperparameters.

5.3.1 How Batch Size Impacts Method Performance

The experiments on the influence were done using an EfficientNet-B0 backbone and a ResNet-34 assistant backbone for DiSiam and DiSiam+. For SimSiam and SimCLR, the settings detailed in the original papers are used. The outcomes are shown in Figure 5.17.

The results consistently show that DiSiam and DiSiam+ achieve better performance than SimSiam and SimCLR across all tested batch sizes. Remarkably, SimSiam's performance negatively correlates with increasing batch size, dropping critically at a batch size of 1024.

In contrast, DiSiam and DiSiam+ demonstrate higher robustness to different random initializations as its standard deviation visualized in Figure 5.17b is consistently lower than for SimSiam and SimCLR.



(a) Maximum kNN Top-1 Accuracy

(b) Standard Deviation of Accuracy

Figure 5.18: (a) Maximum kNN Top-1 accuracy for various learning rates on CIFAR-10 using EfficientNet-B0 as backbone and ResNet-34 as assistant backbone for DiSiam and DiSiam+ averaged across two random seeds (7, 41). Error bars show 95% confidence interval. (b) Standard deviation per method of maximum kNN Top-1 accuracy averaged across different learning rates with error bars indicating the 95% confidence interval.

5.3.2 The Impact of Learning Rate on Performance

Experiments on the influence of the learning rate reveal that a learning rate of 0.03 instead of the 0.06 that the paper uses for ResNet-18 yields the best performance for SimSiam using an EfficientNet-B0 backbone. For DiSiam and DiSiam+, a learning rate of 0.06 is optimal, which is consistent with the paper settings of SimSiam [24]. SimSiam demonstrates high sensitivity to the choice of learning rate. In contrast, DiSiam and DiSiam+ display high robustness to the learning rate choice. Furthermore, the proposed methods consistently outperform SimSiam across all learning rate settings. On top of that, the results show a consistently higher standard deviation across all learning rates for SimSiam compared to DiSiam and DiSiam+. This indicates that SimSiam is more sensitive to randomness across different learning rate settings. Both results are shown in Figure 5.18.

5.3.3 The Influence of Backbone Size on Performance

The influence of the backbone size on the performance of the SSL methods is investigated in this section. The findings indicate that increasing the parameters of the backbone does not lead to improved performance on the CIFAR-10 dataset. While the performances of SimCLR, DiSiam, and DiSiam+ remain stable across different backbone sizes, SimSiam's performance actually decreases with larger backbones. The result is visualized in Figure 5.19.

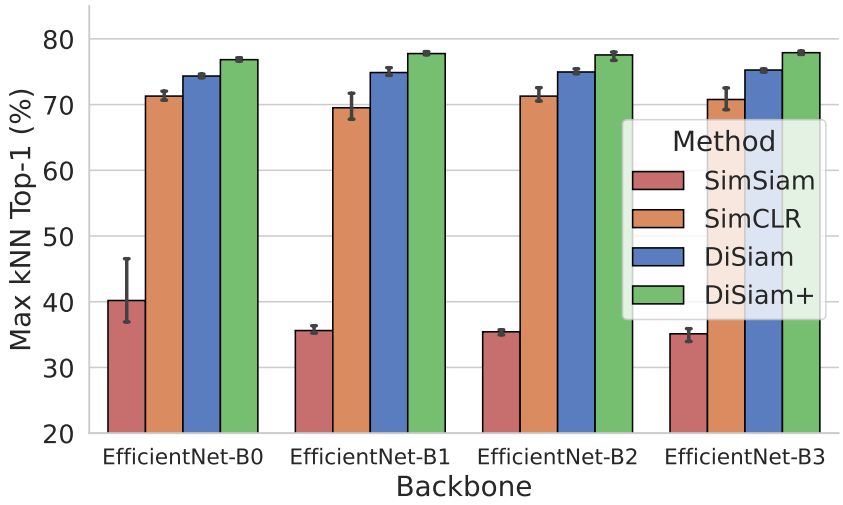


Figure 5.19: Maximum kNN Top-1 accuracy on CIFAR-10 using various EfficientNet backbones and ResNet-34 as assistant backbone for DiSiam and DiSiam+ averaged across three random seeds (7, 41, 923). Error bars indicate 95% confidence interval.

A dedicated ablation study detailed in Figure 5.20 and Figure 5.21 on SimSiam across all EfficientNet and ResNet backbones reveals that its performance remains stable for ResNet architectures, but declines significantly for larger EfficientNet models. Interestingly, the kNN accuracy decreases after a certain training epoch, and even drops to the performance of random guessing of 10% for large enough EfficientNet backbones as detailed in Figure 5.22 which shows the kNN accuracy of the last training epoch 800 per EfficientNet backbone trained with SimSiam. This suggests that SimSiam heavily struggles with representational collapse when using larger EfficientNet backbones.

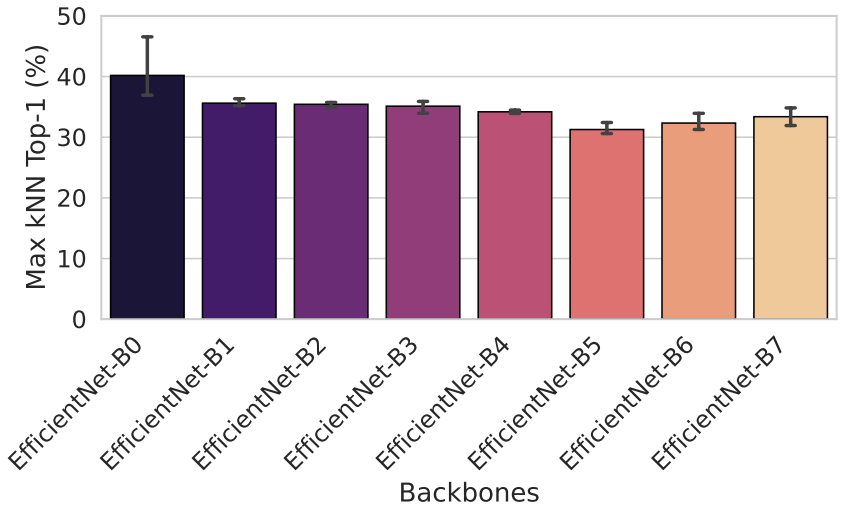


Figure 5.20: Maximum kNN Top-1 accuracy for SimSiam on CIFAR-10 using various EfficientNet backbones averaged across three random seeds (7, 41, 923). Error bars indicate 95% confidence interval.

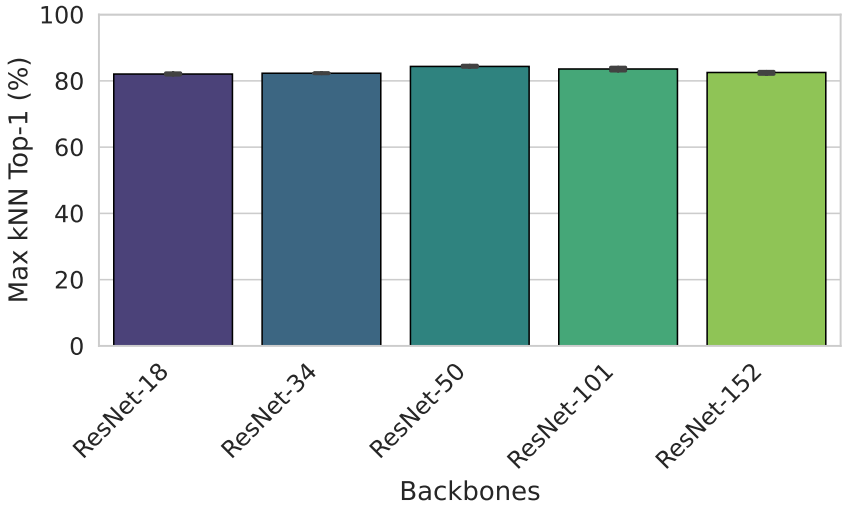


Figure 5.21: Maximum kNN Top-1 accuracy for SimSiam on CIFAR-10 using various ResNet backbones averaged across three random seeds (7, 41, 923). Error bars indicate 95% confidence interval.

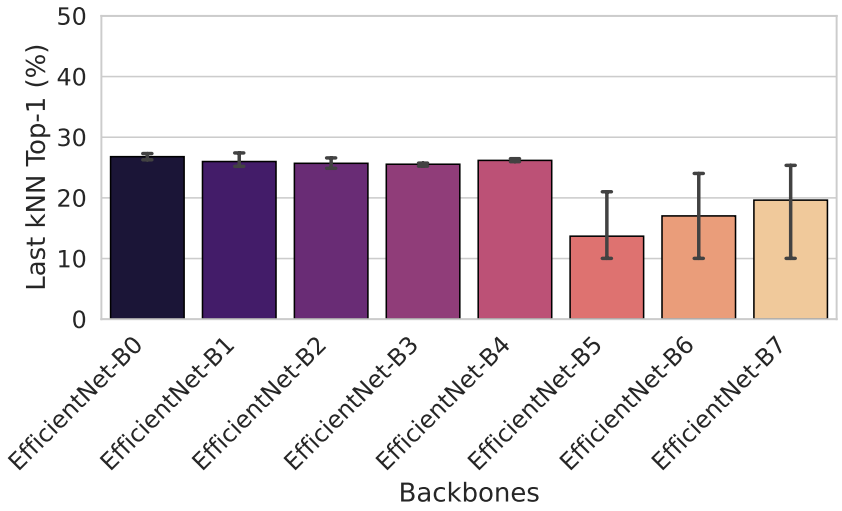


Figure 5.22: kNN Top-1 accuracy of epoch 800 for SimSiam on CIFAR-10 using various EfficientNet backbones averaged across three random seeds (7, 41, 923). Error bars indicate 95% confidence interval.

To gain qualitative insights into the potential representational collapse, the kNN Top-1 accuracy over training epochs for EfficientNet-B5 is shown in Figure 5.23. The corresponding feature representations are shown in Figure 5.24. The 2D TriMap [3] visualization shows constant densities for all classes, strongly indicating representational collapse.

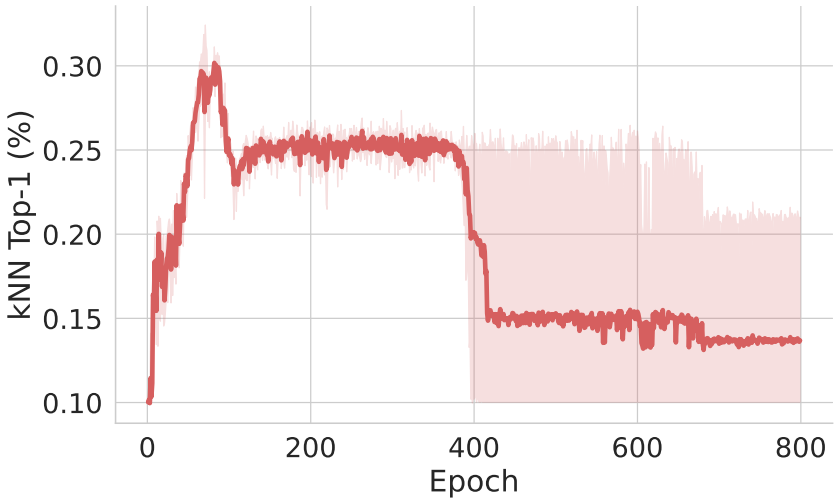
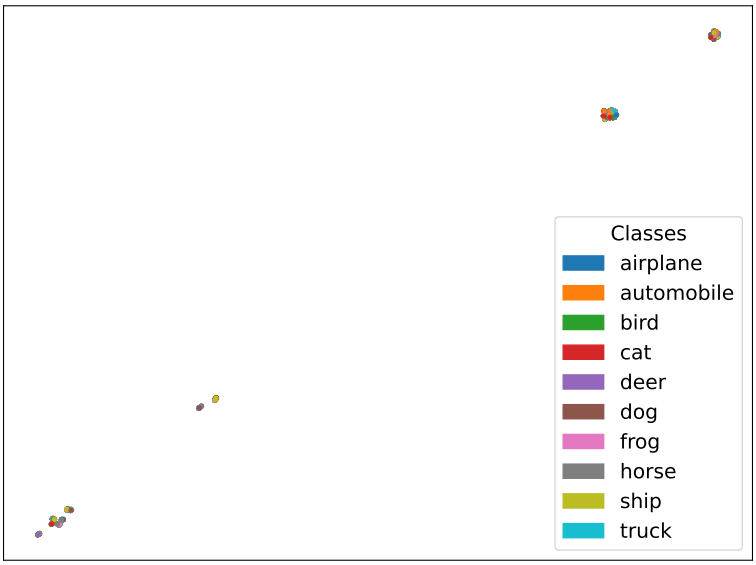


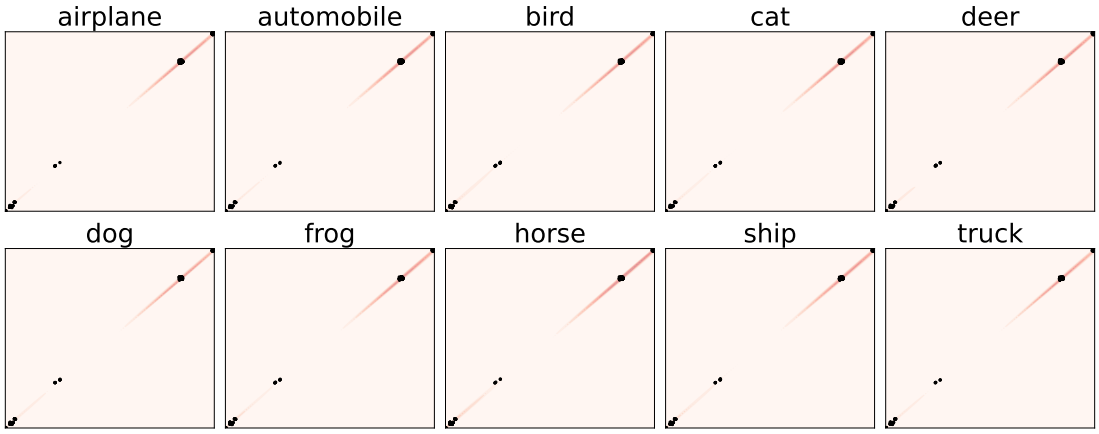
Figure 5.23: kNN Top-1 accuracy during training for EfficientNet-B0, corresponding to Figure 5.11. DiSiam and DiSiam+ show consistently higher accuracy than SimSiam and SimCLR. Results are averaged over three random seeds, with the shaded area representing the 95% confidence interval.

5.3.4 How the Backbone Architecture Influences Performance

As shown in Figure 5.19, the architecture of the backbone plays a crucial role in the performance of the SSL methods. The results reveal that the EfficientNet architecture leads to significantly worse performance than the ResNet architectures for all SSL methods, but particularly SimSiam. This finding is supported by Figure 5.25, which shows the highest achieved kNN Top-1 accuracies for different architectures. Additionally, Table 5.5 shows the best configuration for each architecture, including the backbone and learning rate.



(a) Scatter Plot



(b) Subplots with Density Estimation

Figure 5.24: TriMap [3] visualization of SimSiam feature representations of EfficientNet-B5 on CIFAR-10 and a corresponding density estimation using Gaussian kernels. It shows poor class separability consistent with its kNN accuracy in Figure 5.11 and constant feature density across all classes which strongly indicates representational collapse [56].

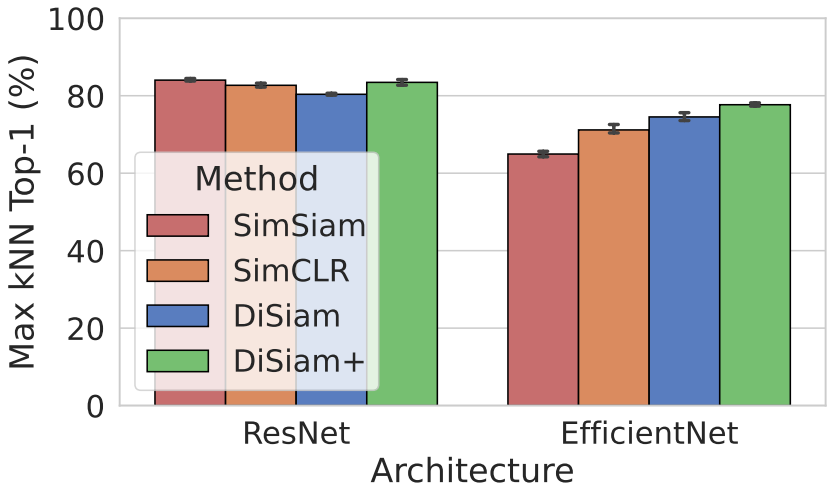


Figure 5.25: Maximum kNN Top-1 accuracy for SimSiam on CIFAR-10 using EfficientNet or ResNet backbones averaged across at least two random seeds. Error bars indicate 95% confidence interval of the best hyperparameter configuration.

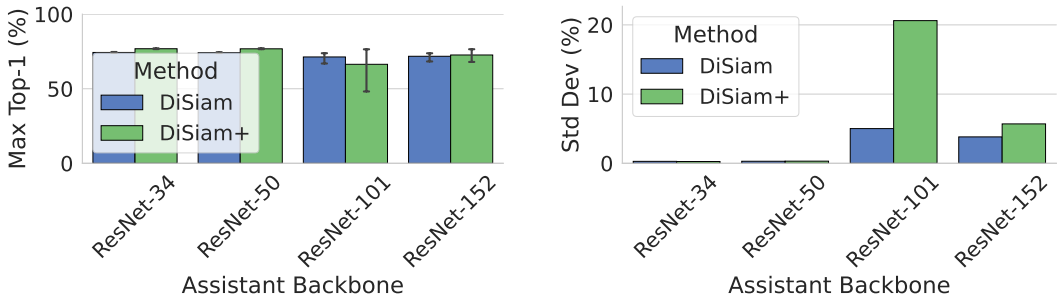
5.3.5 The Role of the Assistant Backbone in DiSiam Methods

For the proposed DiSiam and DiSiam+ methods, the ResNet architecture is leveraged as assistant backbone due to its high observed performance with SimSiam in Subsection 5.3.4. The experiments in this section focus on exploring the influence of the size of the assistant backbone on the performance of the proposed methods. They reveal a slight decline in performance with larger assistant backbones, as shown in Figure 5.26a. More critically however, the variance of the performance strongly increases with larger assistant backbones. This is evident in Figure 5.26b, where the standard deviation of the performance is shown.

To investigate factors that may alleviate the low robustness and performance with large assistant backbones, the impact of using different backbones with a ResNet-152 as assistant backbone is analyzed as a high capacity gap between the main and assistant backbone may contribute to performance limitations. The results shown in Figure 5.27 indicate, that the size of the size of the online backbone positively correlates with the performance of the proposed methods when using a ResNet-152 assistant backbone. As expected, using ResNet-152 as the assistant network with the online backbone EfficientNet-B7 brings a performance drop, since the assistant has less parameters than

		ResNet			
Method	*Backbone*	*Assistant Backbone*	*Learning Rate*	*Epoch*	*kNN Top-1 (%)*
SimSiam	ResNet-50	-	0.06	728	**84.02 ± 0.42**
SimCLR	ResNet-50	-	0.06	686	82.69 ± 0.62
DiSiam	ResNet-18	ResNet-34	0.06	788	80.37 ± 0.25
DiSiam+	ResNet-50	ResNet-152	0.06	771	**83.45 ± 1.03**
		EfficientNet			
Method	*Backbone*	*Assistant Backbone*	*Learning Rate*	*Epoch*	*kNN Top-1 (%)*
SimSiam	EfficientNet-B0	-	0.03	756	64.94 ± 1.0
SimCLR	EfficientNet-B2	-	0.06	734	71.17 ± 1.22
DiSiam	EfficientNet-B1	ResNet-34	0.06	735	**74.52 ± 1.02**
DiSiam+	EfficientNet-B3	ResNet-34	0.06	739	**77.68 ± 0.43**

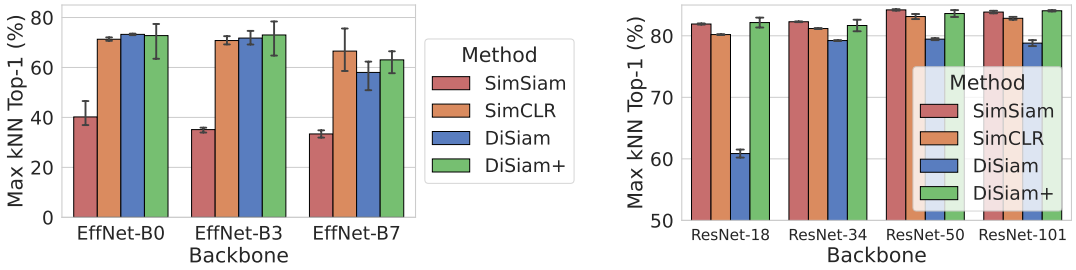
Table 5.5: Numerical kNN Top-1 accuracy values on CIFAR-10 corresponding to the models in Figure 5.25. Not all combinations of methods, learning rates, backbones, and assistant backbones were tested. Reported values are computed as the mean ± standard deviation over at least two random seeds. The best two results for each backbone architecture are highlighted in bold.



(a) Maximum kNN Top-1 Accuracy

(b) Standard Deviation of Accuracy

Figure 5.26: (a) Maximum kNN Top-1 accuracy for DiSiam and DiSiam+ on CIFAR-10 using ResNet assistant backbones with an EfficientNet-B0 backbone averaged across three random seeds (7, 41, 923). Error bars indicate 95% confidence interval of the best hyperparameter configuration. (b) Standard deviation per assistant backbone of maximum kNN Top-1 accuracy with error bars indicating the 95% confidence interval.



(a) EfficientNet backbones

(b) ResNet backbones

Figure 5.27: Maximum kNN Top-1 accuracy for SSL frameworks on CIFAR-10 using a ResNet-152 assistant backbone for DiSiam methods with EfficientNet and ResNet backbones. Three random seeds (7, 41, 923) were used for the EfficientNet experiments, two (7, 41) for the ones with ResNet. Error bars indicate 95% confidence interval.

the online backbone as shown in Table 4.1.

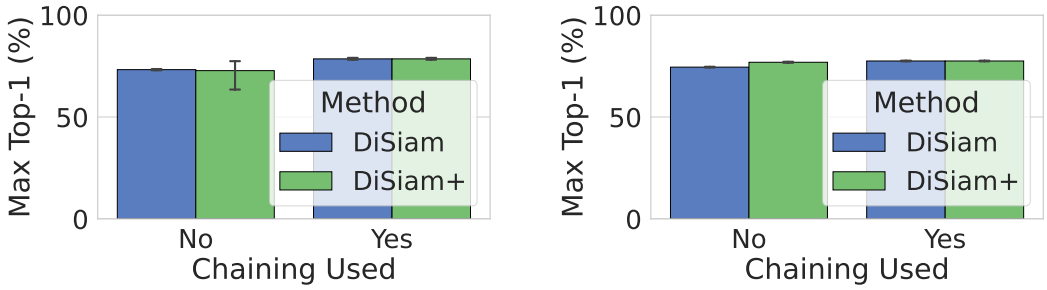
5.3.6 Exploring Chaining of Self-Supervised Learning Methods

The influence of chaining different SSL methods is investigated. The results suggest that chaining can be a reasonable strategy to reduce the performance variance observed with the DiSiam methods when using a ResNet-152 assistant backbone as detailed in Figure 5.28a. When chaining SimCLR with the DiSiam methods with a ResNet-34 assistant backbone as shown in Figure 5.28b, the performance slightly increases compared to the original DiSiam and DiSiam+ methods. As the original DiSiam methods with ResNet-34 assistant backbone already achieve a high performance and robustness, the chaining does not lead to a significant robustness increase.

Chaining SimCLR followed by SimSiam with a EfficientNet-B0 backbone leads to a performance increase compared to SimSiam alone, but it decreases the already achieved performance of SimCLR. However, it does prevent SimSiam from representational collapse which shows that initializing SimSiam with weights that already achieve solid kNN, such as those pretrained with SimCLR, can stabilize its training. The results can be observed in Figure 5.29.

5.3.7 Benchmarking Against PyTorch Weights and Random Baselines

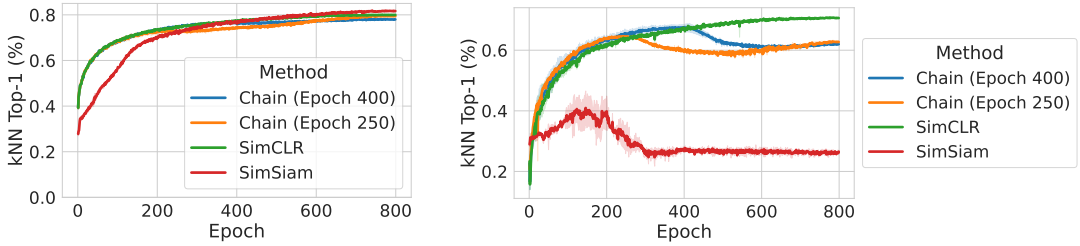
To set the SSL on the CIFAR-10 dataset into perspective, the best performing method on CIFAR-10, *DiSiam+* with assistant backbone ResNet-34 is compared with the pretrained



(a) ResNet-152 assistant backbone

(b) ResNet-34 assistant backbone

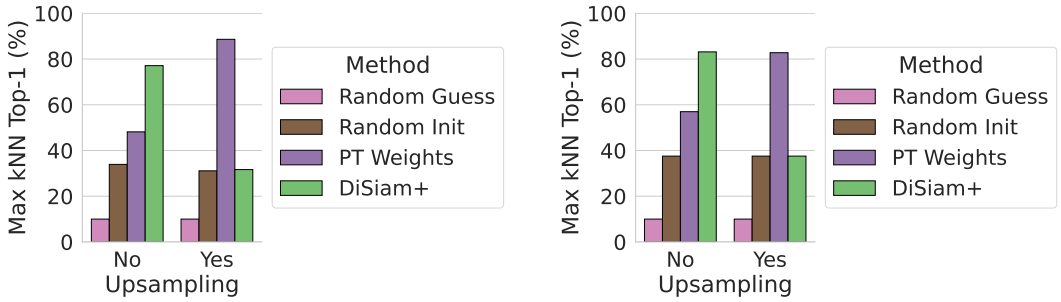
Figure 5.28: Maximum kNN Top-1 accuracy for DiSiam and DiSiam+ on CIFAR-10 using ResNet assistant backbones with EfficientNet-B0 as the main backbone averaged across three random seeds (7, 41, 923). Error bars indicate 95% confidence interval.



(a) ResNet-18 backbone

(b) EfficientNet-B0 backbone

Figure 5.29: kNN Top-1 accuracy during training of SimSiam for EfficientNet-B0 and ResNet-18 backbones using chaining. Results are averaged over three random seeds, with the shaded area representing the 95% confidence interval.



(a) EfficientNet-B0

(b) ResNet-18

Figure 5.30: Maximum kNN Top-1 accuracy for different pretraining and initialization strategies of the EfficientNet-B0 and ResNet-18 backbones on CIFAR-10 resolution (32x32 pixels) and upsampled to 224x224 pixels. The PyTorch weights are pretrained on ImageNet, while the DiSiam+ method is pretrained on the CIFAR-10 training split. The random baseline is established in Section 5.1.

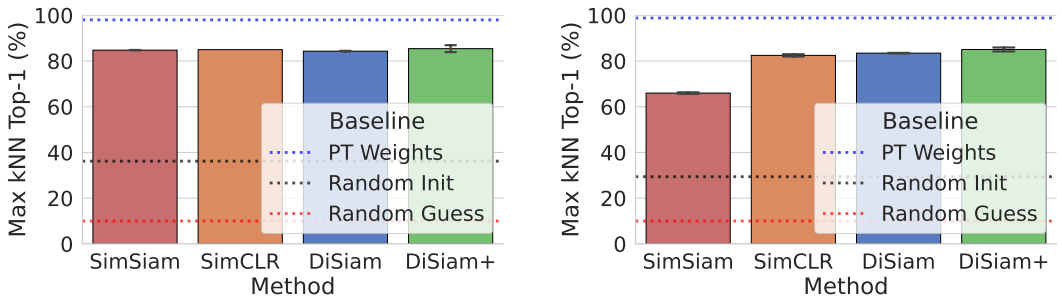
weights provided by the PyTorch library [91] and the random baseline established earlier in Section 5.1. The evaluation is done using an EfficientNet-B0 and a ResNet-18 backbone. Evaluations are once done with the original CIFAR-10 resolution of 32x32 pixels and once with upsampling the input to 224x224 pixels, which is the resolution used for pretraining the PyTorch weights. The results are presented in Figure 5.30.

Predicting in the original CIFAR-10 resolution, *DiSiam+* outperforms the PyTorch weights using the 224x224 resolution, even though the used ResNet architecture is specifically designed for ImageNet and thus a pixel space of 224x224. The standard version is suboptimal for a pixel space of 32x32 [50]. Furthermore, PyTorch weights are trained on ImageNet with 1.2 million images, while the CIFAR-10 dataset only contains 60,000 images. On the other side, the DiSiam+ model was pretrained on the training split of the CIFAR-10 dataset which might be beneficial for the resulting kNN performance on the CIFAR-10 dataset.

5.4 Exploring Generalizability: COCO Pretraining and ImageNet evaluation

To validate the generalizability of the results, the SSL methods are trained on the COCO 2017 dataset [70] with a resolution of 224x224 pixels. The feature representations of the pretrained backbones are then evaluated on a subset of the ImageNet dataset [63].

Consistent with the findings on CIFAR-10, *DiSiam* and *DiSiam+* outperform SimSiam and SimCLR, particularly when using EfficientNet architectures as the backbone. The results are depicted in Figure 5.31.



(a) ResNet-18

(b) EfficientNet-B0

Figure 5.31: Maximum kNN Top-1 accuracy for different SSL methods on EfficientNet-B0 and ResNet-18, using the original hyperparameters from each paper pretrained on the COCO 2017 dataset and evaluated on a subset of ImageNet with 10 classes. DiSiam and DiSiam+ use ResNet-34 as an assistant backbone. All results are averaged over two random seeds (7, 41), with error bars showing the 95% confidence interval. SimClR was tested with one seed in experiment (a).

To assess the impact of the number of projection layers, the performance of SimSiam, DiSiam and DiSiam+ is compared using two and three projection layers. As also demonstrated in the BYOL paper [42], Figure 5.32 shows that using two projection layers consistently yields better performance than using three.

5.5 Implementing 3D Object Detection

SSL methods have not yet been evaluated on the 3D object detection task. To make this possible, a 3D object detection model was implemented following test-driven development principles and industry best practices. The model was initially validated on the mini nuScenes dataset [14], which includes 10 scenes and is small enough for the model overfit to it. Since 3D object detection is complex, using multiple prediction heads and estimating depth and size form monocular images, extensive hyperparameter tuning is required. Further engineering work is necessary to tune the model successfully for the full nuScenes dataset.

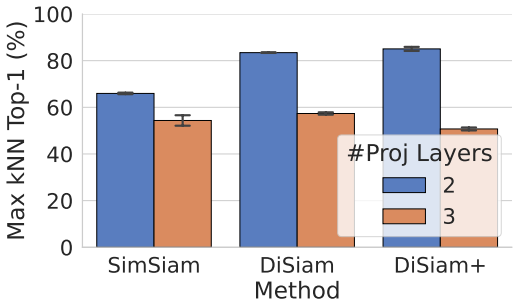


Figure 5.32: Maximum kNN Top-1 accuracy pretrained on COCO 2017 and evaluated on a subset of ImageNet with 10 classes using a different number of projection layers averaged across two random seeds (7, 41). Error bars indicate 95% confidence interval.

Method	*mAP (%)*
Random Initialization	**46.37**
SimSiam	37.19
SimCLR	**50.72**
DiSiam	32.95
DiSiam+	34.22
PyTorch Weights	**55.15**

Table 5.6: mAP in % for different SSL methods fine-tuned and evaluated on a proprietary dataset for autonomous driving. All frameworks are pretrained on the COCO 2017 dataset. The best three methods are highlighted in bold.

5.6 Evaluating the Transfer of Pretrained Models to 2D Object Detection

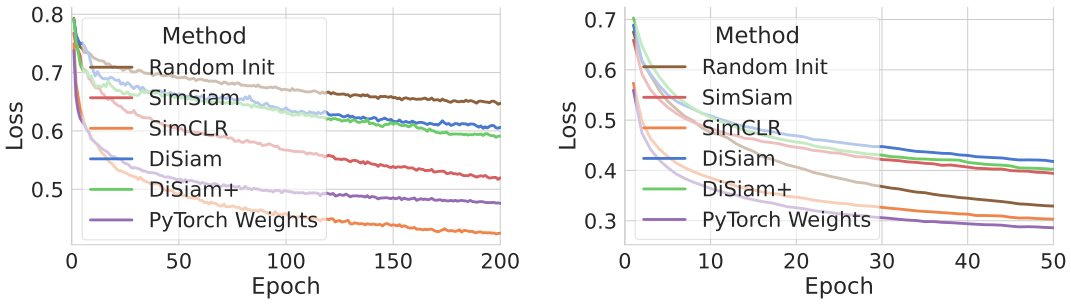
Even though the DiSiam and DiSiam+ methods demonstrate better performance in image classification tasks, the results for 2D object detection are remarkably different. The DiSiam methods partially perform worse than a randomly initialized backbone, while SimSiam shows better performance than random initialization, and SimCLR achieves the best results, nearly matching the accuracy of the PyTorch weights.

For generalizability, the pretrained backbones are evaluated once on a subset of OpenImages [66], detailed in Table 5.7 and once on an proprietary dataset of an autonomous driving company detailed in Table 5.6.

Method	*mAP (%)*
Random Initialization	2.94
SimSiam	**11.97**
SimCLR	**23.18**
DiSiam	3.84
DiSiam+	4.73
PyTorch Weights	**20.73**

Table 5.7: mAP in % for different SSL methods fine-tuned and evaluated on a subset of the OpenImages dataset. All frameworks are pretrained on the COCO 2017 dataset. The best methods are highlighted.

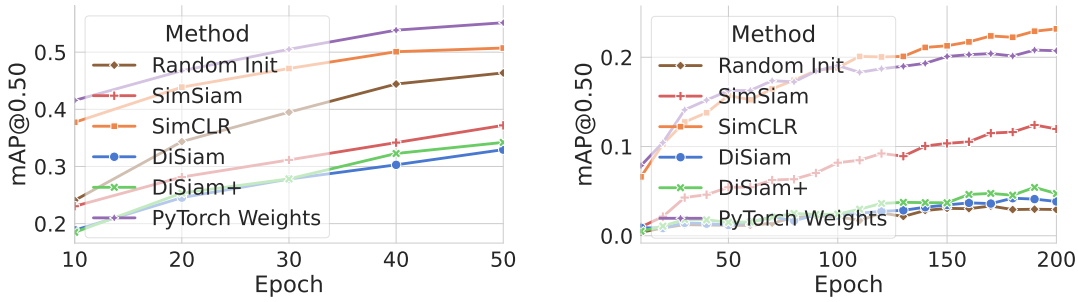
The loss curves for the proprietary and OpenImages datasets are shown in Figure 5.33. The corresponding mAP curves are summarized in Figure 5.34.



(a) Proprietary Dataset

(b) OpenImages Dataset

Figure 5.33: Training loss curves for a 2D object detection model using different self-supervised and pretrained initialization methods for the backbone. (a) shows performance on the proprietary autonomous driving dataset, (b) on the OpenImages dataset. The curves correspond to the numerical mAP results reported in Table 5.6 and Table 5.7.



(a) Proprietary Dataset

(b) OpenImages Dataset

Figure 5.34: mAP values in % for a 2D object detection model using different self-supervised and pretrained initialization methods for the backbone. (a) shows performance on the proprietary autonomous driving dataset, (b) on the OpenImages dataset. The curves correspond to the numerical mAP results reported in Table 5.6 and Table 5.7.

5.7 Key Outcomes and Insights

The experiments reveal several findings. First, SimSiam performs poorly with the EfficientNet architecture due to representational collapse. In contrast, DiSiam and DiSiam+ consistently outperform SimSiam and SimCLR in classification tasks, although this improvement does not extend to 2D object detection. The choice of the backbone architecture is crucial, with ResNet-based methods generally achieving superior results than those based on EfficientNet. Interestingly, the random baseline shows surprising strength, indicating that random initialization can yield performance four times as high as random guessing. Finally, the results demonstrate that strong classification performance does not necessarily imply high transferability to other downstream perception tasks in autonomous driving.

6 Discussion

6.1 Interpretation of Experimental Results

The results presented in Chapter 5 provide insight into the behavior of SSL methods. In the following sections, the main outcomes are analyzed and discussed in detail.

6.1.1 Analysis of Representational Collapse in SimSiam with EfficientNet Architectures

Observation of Collapse

The results of SSL methods with CIFAR-10 in Section 5.2 and Section 5.3 and with COCO 2017 in Section 5.4 show that the SimSiam method performs poorly with EfficientNet architectures. The learned feature representation in Figure 5.13 and Figure 5.24 demonstrate that the network experiences dimensional and even total representational collapse. This means that the network maps to a lower-dimensional space than the available latent space, or even to a constant value [56]. This directly negatively influences the classification performance of the network as shown in Figure 5.23 and Figure 5.22.

Potential Explanations for Collapse

As explained by the SimSiam paper authors [24], non-collapsing behavior was only an empirical observation and never proven theoretically. The simple design of SimSiam with no negative samples makes it prone to representational collapse if not optimized and tuned carefully, as the global optimum is to output always the same value. It does not collapse, however, with the ResNet architecture, thus it is crucial to understand how the EfficientNet architecture could contribute to the collapse. There are various possible explanations for this observation as EfficientNet employs several innovations such as depthwise separable convolutions, inverted residuals, squeeze-and-excitation blocks, and compound scaling.

Squeeze-and-excitation blocks are a kind of self-attention mechanism introduced in [54] which aim to model channel interdependencies. This attention mechanism benefits the model in the supervised case, but it could offer too much model capacity,

changing the loss landscape so that the network is more prone to find the global minimum and thus collapse.

Depthwise separable convolutions [26, 53, 46] factorizes the standard convolution into a depthwise convolution and a pointwise convolution to reduce the computational cost. However, it reduces the interaction between the channels, which makes it harder for the model to form meaningful inter-channel dependencies. This could lead to a situation where it is easier for the model to converge to the global minimum and collapse than to find a robust local minimum that would require complex inter-channel dependencies. The high dependency on color distortion as augmentation in contrastive SSL methods [21, 42] supports this hypothesis, as it means that to model the inter-channel dependencies is crucial to the success of the methods.

Linear bottlenecks [97] are a design choice that compress the channel dimension of the activation space and thus forces diverse features onto a low-dimensional manifold. As SimSiam benefits greatly from a high-dimensional projection head [24], restraining the latent space to low-dimensional manifolds could hinder learning and potentially lead to representational collapse.

Besides architectural incompatibilities, the EfficientNet architecture could hinder the SimSiam algorithm in other ways.

Learning Rate Sensitivity of SimSiam on EfficientNet-B0 was observed in Figure 5.18. It is hypothesized that robustness to learning rate variations lacks also in larger EfficientNet models. This heavily influences the training dynamics of SimSiam, as a learning rate too small leads to no significant learning and a too large learning rate leads to representational or dimensional collapse as it can be seen in Figure 5.24 or Figure 5.13. This suggests that the model might more easily find the global minimum and collapses than it diverges when the learning rate is too high. Intuitively, this suggests that the global minimum is broad and easy to optimize, assuming the model can reach its basin. As the EfficientNet architecture is more sensitive to learning rate variations, it suggests also that the local minima are sharper than those of the ResNet architecture.

Dataset Dependency could be yet another factor why SimSiam collapses on EfficientNet architecture. Figure 5.31b shows that SimSiam does not collapse on the COCO 2017 dataset using the EfficientNet-B0 architecture with the hyperparameter settings of the paper [24]. This is a stark contrast to the CIFAR-10 dataset, where SimSiam experiences dimensional collapse using the same hyperparameter (see Figure 5.13). This suggests that the EfficientNet architecture at least in combination with the negative-sample-free SSL method SimSiam is more prone to collapse on datasets either with low resolution or low number of training data. To confirm this hypothesis further experiments with other datasets like ImageNet [63] are needed.

6.1.2 DiSiam Methods: Improved Performance with EfficientNet Architectures

Observation of Performance Enhancement

As shown in Section 5.2 and Section 5.4, the *DiSiam* and *DiSiam+* methods outperform the SimSiam baseline in classification tasks when using EfficientNet backbones. The improvement is particularly substantial on the CIFAR-10 dataset.

Possible Explanations for Improved Performance

The gain in performance can be attributed to the introduction of the assistant network in DiSiam and DiSiam+. It enables dynamic knowledge distillation from a stronger ResNet backbone to a lightweight EfficientNet backbone. Unlike traditional static knowledge distillation techniques that use a static teacher network (e.g. [52]), DiSiam uses simultaneous training of both the student and teacher networks.

This setting allows the assistant network to even guide the training dynamics of the online backbone. The dynamic knowledge distillation effectively transfers robustness and inductive biases of ResNet architectures to EfficientNet. The result is an SSL method that combines the robustness of ResNet to hyperparameters with the parameter efficiency of EfficientNet.

Empirical results (e.g. Figure 5.18) confirm this hypothesis. DiSiam and DiSiam+ show greater robustness to learning rate and batch size compared to SimSiam. Furthermore, *DiSiam+* even surpasses models initialized with ImageNet-pretrained weights when upsampling CIFAR-10 images (see Figure 5.30a), showing strong representation learning on small-scale datasets.

The results suggest that combining a robust teacher (the target network) with an efficient student (the online network) through dynamic self-distillation improves both classification accuracy and training robustness. Both could prove valuable for deployment in perception for autonomous driving.

6.1.3 Challenges in Transfer Learning with Negative-Sample-Free Methods

Discrepancy in Transfer Performance

The results show a substantial discrepancy between the classification and object detection performance across all evaluated methods. While SimSiam, DiSiam, and DiSiam+ reach strong performance on classification, their transferability to object detection is limited. In contrast, SimCLR shows both solid classification results and excellent transfer to object detection tasks, performing on par with models initialized using ImageNet-pretrained PyTorch weights (see Figure 5.34).

Hypothesized Explanations for Limited Transferability

The difference may be attributed to the combination of SSL method and the backbone architecture. Prior work suggests a strong correlation between the ImageNet classification performance and the downstream task transferability when using ResNet architectures [33]. Both SimSiam [24] and SimCLR [21] have shown strong transfer results with ResNets, but those findings might not generalize to other backbones.

In particular, EfficientNet uses squeeze-and-excitation (SE) blocks, which may push the model toward learning compact, discriminative features, well suited for classification, but not diverse enough for dense prediction tasks such as object detection. While contrastive methods like SimCLR and MoCo [49] have been shown to produce detailed and well reconstructible latent representations [125], this has been studied with ResNet architectures and not using negative-sample-free frameworks.

It is thus hypothesized that the combination of EfficientNet with negative-sample-free methods (SimSiam, DiSiam) may hinder the learning of rich and diverse features required for object detection, leading to poor transferability even with high classification performance. The backbone architecture appears therefore to play a crucial role in the transfer capabilities of the learned representations of negative-sample-free SSL methods.

6.1.4 Impact of Assistant Backbone Size on DiSiam Performance

Observed Effects of Assistant Backbone Size

Experimental findings indicate that DiSiam methods using EfficientNet online backbones perform better and more reliably when using lightweight assistant backbones (e.g. ResNet-18 or ResNet-34) as shown in Figure 5.26. This trend is counterintuitive, as larger assistant networks have greater representational capacity and are expected to improve the self-distillation quality.

Possible Explanations for Performance Decline and Increased Variance

Two main hypotheses may be the reason for this behavior. First, the larger network may be prone to overfitting due to their high capacity. This is especially relevant in scenarios with limited task complexity, as with the CIFAR-10 dataset. Supporting this, Figure 5.21 shows that SimSiam does not benefit from larger backbones, indicating that the learning task may not be complex enough.

Second, the increased variance and reduced performance with larger assistants suggests an incompatability in the dynamic distillation dynamics. A large assistant network may learn feature representations that are too complex or abstract for the

smaller target backbone to copy effectively. A high capacity gap between online and assistant networks potentially leads to the online network then finding suboptimal local minima and can decrease both accuracy and training stability. This hypothesis can be confirmed empirically. Figure 5.27b demonstrates, that a lower capacity gap between the online ResNet-based encoder and the target network (in this case ResNet-152) leads to improved performance. For EfficientNet-based online backbones, the improvement is far smaller than for ResNet ones as shown in Figure 5.27a. Future work could investigate further, if reducing the capacity gap improves the performance of DiSiam methods.

6.1.5 Stabilizing Effect of Chaining SSL Methods

Evidence of Chaining Effectiveness

The outcomes in Subsection 5.3.6 show that pretraining with SimCLR before applying SimSiam markedly improves the training stability and prevents representational collapse. As seen in Figure 5.29b, SimSiam without chaining collapses during training, while chaining it after SimCLR avoids this issue. Although it does not necessarily lead to improved performance over SimCLR, it does ensure stable convergence for SimSiam.

Furthermore, chaining SimCLR before DiSiam and DiSiam+ methods reduces performance variance substantially and slightly improves performance, particularly when using large assistant networks, as shown in Figure 5.28a.

Theoretical Hypotheses for Improved Stability

The observations indicate that chaining allows negative-sample-free methods like SimSiam to benefit from the latent feature representations already learned by contrastive methods such as SimCLR. The initial SimCLR stage may provide a well-formed representation space that prevent early collapse during SimSiam training by already starting from a good optimization basin in the loss landscape. This effect is beneficial especially if SimSiam alone is instable.

For DiSiam and DiSiam+, chaining appears to mitigate the effects of using high-capacity assistant networks. It is probable that the SimCLR pretraining initializes the assistant network in a more stable feature space, so that the small online network can more easily learn from the assistant. It might also reduce the risk of divergent learning trajectories of the siamese network during training.

Overall, chaining seems to be a simple yet effective strategy to improve robustness and training stability in negative-free-sample SSL frameworks.

6.1.6 Analysis of Randomly Initialized Models

Unexpected Performance of Untrained Models

"The parameters of the network are initialized randomly, and the output is no better than random guess" [75] is a common assumption in the literature.

Contrary to the expectation that randomly initialized models would perform no better than random guessing (i.e., approximately 10% on CIFAR-10), the experimental results of the thesis demonstrate substantially higher kNN Top-1 accuracies, reaching up to 39.83%. This four-fold enhancement suggests that even without any training, the randomly initialized parameters somehow preserve characteristics of the input data in the latent feature space. This is true for both feature representations alone and when a randomly initialized linear classifier is included.

Discussion on Inherent Structure in Random Projections

Inherent Structure in Random Projections The surprising finding that randomly initialized networks exhibit non-random performance could be explained by the Johnson-Lindenstrauss lemma [59]. It states that a set of points in a high-dimensional space can be embedded into a low-dimension space such that distances between the points are nearly preserved. Even with millions of random parameters, the network is effectively a non-linear projection, mapping input data to a lower-dimensional feature space where some structure of the input space is maintained. Despite the Johnson-Lindenstrauss lemma being intially proposed for linear projections, it can be extended also to non-linear projections [39]. The vast amount of parameters allows to capture some intrinsic properties of the data even if the parameters are random.

Architectural Performance Differences An outstanding observation is the consistent outperformance of ResNet architectures over EfficientNet architectures in the randomly initialization setting, even though EfficientNet showing better performance in supervised learning tasks [102]. For example, the ResNet-152 backbone achieved the highest average accuracy of 38.41%, while EfficientNet architectures reach only 37.25%. That suggests that the design of ResNets might be inherently better to preserve meaningful data structures in their untrained state. The skip connections in ResNets, which ease the information flow, could play a role here to maintain a more structured feature space even with random weights. These connections mitigate the degradation of the input structure so that the raw input features can propagate more easily through the network without being distorted by too many random transformations. Moreover, the uniformity and modularity of ResNet blocks might contribute to more consistent feature extraction, even when the weights are randomly initialized.

Impact of Initialization Variance The Kaiming He initialization scheme [51], particularly the gain value, proved to be a vital factor. Both too small and large gain values led to substantial performance degradation. This highlights the importance of the initialization of neural networks; the variance of the gaussian initialization distribution directly affects the ability of the model's capability to preserve a structured latent feature space. Interestingly, ResNet-18 demonstrates remarkable robustness to high variance initializations. It performs well even with the highest gain values which were ten-fold the default values of the Kaiming He initialization scheme.

Qualitative and Quantitative Discrepancies Figure 5.9 shows no apparent difference between the 2D TriMap [3] visualizations of a well-performing and low-performing gain ratio. Various reasons could lead to this discrepancy of quantitative (Figure 5.4) and qualitative outcomes. As the kNN metric is itself a robust image classification metric, the reason for the non-intuitive qualitative result has to be searched in TriMap. First, TriMap emphasizes triplet constraints but does not guarantee exact preservation of kNN. Close points in high-dimensional space may appear far in 2D, and the other way round. Second, clusters may appear less or more dense than they actually are. Finally, kNN heavily relies on local neighborhood accuracy, which TriMap may sacrifice to gain consistent global accuracy. Therefore, qualitative results presented in this thesis should be carefully evaluated.

Robustness Measured by Full Width Half Maximum (FWHM) The Full Width Half Maximum (FWHM) analysis quantitatively supports the architectural differences in robustness. Generally, smaller FWHM values indicate higher sensitivity, while larger ones suggest robustness. A clear trend of increasing sensitivity was observed within the ResNet architecture. EfficientNet models however maintained relative consistent FWM values across various sizes. Pure depth scaling is hypothesized to make the ResNet architecture more sensitive to high variance in the initialization distribution when using larger models, as the input structure becomes increasingly distorted by the random transformations. This is supported by the FWHM values in Figure 5.8, which show a clear trend of decreasing FWHM values with increasing depth of the ResNet architecture. In contrast, EfficientNet's compound scaling approach alleviates this issue by scaling depth, width, and resolution simultaneously. Width and resolution scaling amplify the input feature signal while deepening increases the model's capacity. This results in consistent input feature signal propagation through the network, which may allow the EfficientNet architecture to maintain a stable FWHM value across all model sizes, as shown in Figure 5.8, Figure 5.4 and Figure 5.5.

6.2 Efficiency Considerations for Perception in Autonomous Driving

The findings of the thesis indicate that the proposed methods *DiSiam* and *DiSiam+* offer a good balance between performance and efficiency for image classification. They are able to achieve high performance on various datasets using small backbones like EfficientNet-B0, which is particularly relevant for efficient perception tasks in autonomous driving. However, as pointed out in Subsection 6.1.3 the strong classification accuracy does not directly translate to comparable performance on object detection tasks. Thus, future work might investigate on how to improve the transferability of the features learned by negative-sample-free SSL, particularly DiSiam and DiSiam+. This could involve exploring different loss functions, regularization techniques, or architectures design changes that encourage the learning of more generalizable representations that are suitable for dense prediction tasks.

6.3 Relation to Prior Work

As in previous works [21, 24, 42], SimCLR provides solid results in both EfficientNet and ResNet architectures. A key distinction of this work is the observed poor performance and frequent collapse of SimSiam when combined with EfficientNet architectures. This behavior was not reported in previous papers, which primarily focused on ResNet models [24]. It highlights a previously unaddressed architectural incompatibility. Vitally, the outcomes on the surprisingly high performance of feature representations solely from randomly initialized models appear to be a novel discovery in the literature. While there are papers using the outputs of a randomly initialized backbone as an input for a linear classifier [2], or work that shows that randomly initialized networks are not truly random functions [103], the outcomes of this thesis extend beyond that by showing inherent structure in the raw latent feature space. The methods *DiSiam* and *DiSiam+* methods extend existing work, drawing inspiration from the dynamic distillation principles in DYOL [73] and the negative-sample-free method SimSiam [24]. The proposed methods integrate a weight-sharing with an assistant network for dynamic knowledge distillation. This provides an effective SSL solution for efficient perception in autonomous driving.

6.4 Implications for Future Self-Supervised Learning Research

Various implications can be inferred from the results of the thesis. The most crucial for using SSL in autonomous driving is to always evaluate the performance of the SSL

methods directly on the final downstream task. A method that generalizes well to various downstream tasks might not exist yet [32], so it is important to choose the right method carefully. Furthermore, the results suggest that the pretraining dataset and the evaluation dataset are decisive for the final performance. This implies that the choice of pretraining and evaluation datasets has a substantial impact on the performance, introducing uncertainty into the generalizability of reported evaluations. The outcome of the thesis suggests however, that SimCLR generally performs very robust and has no danger of collapsing when used with EfficientNet architectures. Additionally, it performs far better than the negative-sample-free methods on the object detection task, potentially indicating better performance on other dense prediction tasks such as 3D object detection, and segmentation. Given its stable performance with EfficientNet architectures and strong transferability to critical dense prediction tasks like object detection and segmentation, SimCLR stands out as a reliable SSL pretraining methods for efficient perception in autonomous driving.

6.5 Limitations of this Thesis

The thesis is extensive, but only explores a small subset of possible combinations of datasets, backbone architectures, backbone sizes, transfer tasks, and parameter variations. It primarily focused on the EfficientNet architecture for efficient perception in autonomous driving. Additionally, the results rely on only three random seeds for most experiments. That could potentially limit the statistical certainty of results, particularly for outcomes with high variance. Future work could employ a more extensive seed strategy to improve the statistical robustness of the findings.

7 Conclusion

This thesis investigated the application and performance of various SSL methods when combined with efficient backbone architectures, particularly EfficientNet. The experimental outcomes and their interpretation provided crucial insight to understand the potential and limitations of SSL in efficient perception for autonomous driving better.

7.1 Summary of Contributions

Several key contributions were made to the field of SSL for efficient perception in autonomous driving. Crucially, a substantial limitation of SimSiam was demonstrated when used with EfficientNet architecture. The result demonstrated its bias for instability, and dimensional and representational collapse. The finding is critical because it highlights that the effectiveness of SSL is not only dependent on the method but also its compatibility with backbone architectures.

To overcome this challenge, *DiSiam* and its extended variant *DiSiam+* are proposed and evaluated. The novel negative-sample-free methods use dynamic knowledge distillation, where robust ResNet assistant networks guide learning of a lightweight EfficientNet online network. The experiments confirm that both proposed methods consistently outperform SimSiam and SimCLR in classification tasks when combined with EfficientNet backbones, showing improved accuracy and stability. The dynamic distillation process and its simple weight-sharing mechanism makes DiSiam and DiSiam+ promising methods for resource-constrained autonomous driving applications.

Furthermore, the outcomes provide an in-depth analysis on the substantial difference between classification and downstream task accuracy across different SSL methods. This work emphasizes, that high classification accuracy does not imply strong performance on transfer tasks, particularly dense prediction tasks which are crucial for autonomous driving.

The thesis also offers a broad analysis of the robustness of SSL methods to various hyperparameter variations. It shows that smaller assistant networks often give superior results compared to larger ones, and hypothesizes that an optimal capacity gap between the online and assistant backbone is essential for effective dynamic distillation. Furthermore, the stabilizing effect of chaining is demonstrated that improves training stability

and prevents representational collapse. It thus offers a simple, practical strategy for combining the strengths of different SSL paradigms to achieve more robust learning.

Finally, an outstanding discovery was the surprisingly input structure preservation of randomly initialized neural networks. The networks possess an instant ability to extract robust features from input data even without training reaching a performance four times as high as random guessing. This provides new insights into the inherent inductive biases of neural networks.

7.2 Future Directions and Research Opportunities

The results obtained in the thesis create room for several promising research directions. Firstly, while the thesis used CIFAR-10 and COCO 2017, future work should apply DiSiam and DiSiam+ to larger datasets such as ImageNet [63] to confirm scalability and generalizability. Secondly, the chaining method can be refined further. This could involve more in-depth chaining strategies, such as combining more than two SSL methods, or changing methods when a certain slope of the kNN accuracy is reached. Moreover, one could implement soft chaining by introducing a new loss combining contrastive and negative-sample-free objectives. While training, the loss gets increasing weight on the negative-sample-free loss, which might increase performance and still retains the robustness against representational collapse of SimCLR. Thirdly, a deeper theoretical analysis of assistant backbone dynamics is desirable. This could involve exploring why small assistant backbones perform optimally in dynamic knowledge distillation, analyzing representation complexity, and loss landscape characteristics to predict the ideal capacity gap between the online and assistant networks. Furthermore, the models could be applied to additional downstream tasks vital for autonomous driving, such as semantic segmentation, instance segmentation, 3D object detection, and depth estimation. This might lead to a more thorough understanding of the utility of the learned features for efficient perception in autonomous driving. Future work could also investigate the performance of FastSiam on lightweight backbones, as Table 3.1 lacks performance data for FastSiam using small backbones. Theoretically uncovering the reasons and limitations of the performance of randomly initialized networks could also be a promising research direction. This could involve analyzing the inductive biases of the networks, and the role of random initialization in feature extraction and how these biases can be used to improve SSL methods. Additionally, given the sequential nature of autonomous driving data, extending negative-sample-free methods and DiSiam methods in particular to video-based SSL is a promising research direction. One could for example incorporate temporal consistency objectives to learn robust spatio-temporal features. Finally, it would be valuable, to investigate if different initialization strategies

using the optimal values of the random initialization experiments lead to better SSL results. This could be promising as initializing weights within a robust basin in the loss landscape and could thus converge quicker and reach higher performance.

Abbreviations

BYOL Build Your Own Latent

CNN Convolutional Neural Network

DiSiam Distillation Siamese

DiSiam+ Multi-View Distillation Siamese

DYOL Distill Your Own Latent

FastSiam Resource-Efficient Self-supervised Learning

kNN k-Nearest Neighbors

LiDAR Light Detection and Ranging

mAP mean Average Precision

SimCLR Simple Framework for Contrastive Learning of Visual Representations

SimSiam Simple Siamese

SSL Self-Supervised Learning

ViT Vision Transformer

List of Figures

List of Tables

Bibliography

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng. "TensorFlow: a system for large-scale machine learning." In: *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*. OSDI'16. Savannah, GA, USA: USENIX Association, 2016, pp. 265–283. ISBN: 9781931971331.

[2] E. Amid, R. Anil, W. Kotłowski, and M. K. Warmuth. *Learning from Randomly Initialized Neural Network Features*. 2022. arXiv: 2202.06438 [cs.LG].

[3] E. Amid and M. K. Warmuth. "TriMap: Large-scale Dimensionality Reduction Using Triplets." In: *CoRR* abs/1910.00204 (2019).

[4] P. Bachman, R. D. Hjelm, and W. Buchwalter. "Learning representations by maximizing mutual information across views." In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[5] P. Bachman, R. D. Hjelm, and W. Buchwalter. "Learning representations by maximizing mutual information across views." In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[6] H. Bao, L. Dong, and F. Wei. "BEiT: BERT Pre-Training of Image Transformers." In: *CoRR* abs/2106.08254 (2021). arXiv: 2106.08254.

[7] A. Bardes, J. Ponce, and Y. LeCun. *VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning*. 2022. arXiv: 2105.04906 [cs.CV].

[8] H. Barlow. "Possible Principles Underlying the Transformations of Sensory Messages." In: *Sensory Communication* 1 (Jan. 1961). DOI: 10.7551/mitpress/9780262518420.003.0013.

[9] M. I. Belghazi, A. Baratin, S. Rajeshwar, S. Ozair, Y. Bengio, A. Courville, and D. Hjelm. "Mutual Information Neural Estimation." In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by J. Dy and A. Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 531–540. URL: https://proceedings.mlr.press/v80/belghazi18a.html.

[10] Y. Bengio, A. C. Courville, and P. Vincent. "Representation Learning: A Review and New Perspectives." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (2012), pp. 1798–1828.

[11] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, Jan. 2006. URL: https://www.microsoft.com/en-us/research/publication/pattern-recognition-machine-learning/.

[12] I. Borg and P. J. F. Groenen. "Modern Multidimensional Scaling: Theory and Applications." In: *Journal of Educational Measurement* 40 (1999), pp. 277–280.

[13] G. Brazil and X. Liu. "M3d-rpn: Monocular 3d region proposal network for object detection." In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 9287–9296.

[14] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. "nuscenes: A multimodal dataset for autonomous driving." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11621–11631.

[15] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. "nuscenes: A multimodal dataset for autonomous driving." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11621–11631.

[16] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. "Deep Clustering for Unsupervised Learning of Visual Features." In: *Computer Vision – ECCV 2018*. Ed. by V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss. Cham: Springer International Publishing, 2018, pp. 139–156. ISBN: 978-3-030-01264-9.

[17] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. "Unsupervised Learning of Visual Features by Contrasting Cluster Assignments." In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 9912–9924. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/70feb62b69f16e0238f741fab228fec2-Paper.pdf.

[18] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. *Emerging Properties in Self-Supervised Vision Transformers*. 2021. arXiv: 2104.14294 [cs.CV].

[19] N. Carrara and J. Ernst. "On the Estimation of Mutual Information." In: *Proceedings* 33.1 (2019). ISSN: 2504-3900. DOI: 10.3390/proceedings2019033031. URL: https://www.mdpi.com/2504-3900/33/1/31.

[20] H. Chen, Y. Huang, W. Tian, Z. Gao, and L. Xiong. "Monorun: Monocular 3d object detection by reconstruction and uncertainty propagation." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 10379–10388.

[21] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. "A Simple Framework for Contrastive Learning of Visual Representations." In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by H. D. III and A. Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, July 2020, pp. 1597–1607. URL: https://proceedings.mlr.press/v119/chen20j.html.

[22] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun. "Monocular 3D Object Detection for Autonomous Driving." In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2147–2156. DOI: 10.1109/CVPR.2016.236.

[23] X. Chen, H. Fan, R. B. Girshick, and K. He. "Improved Baselines with Momentum Contrastive Learning." In: *ArXiv* abs/2003.04297 (2020).

[24] X. Chen and K. He. "Exploring Simple Siamese Representation Learning." In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 15745–15753. DOI: 10.1109/CVPR46437.2021.01549.

[25] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer. "cuDNN: Efficient Primitives for Deep Learning." In: *CoRR* abs/1410.0759 (2014).

[26] F. Chollet. "Xception: Deep Learning with Depthwise Separable Convolutions." In: July 2017, pp. 1800–1807. DOI: 10.1109/CVPR.2017.195.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, pp. 4171–4186.

[28] M. Ding, Y. Huo, H. Yi, Z. Wang, J. Shi, Z. Lu, and P. Luo. "Learning depth-guided convolutions for monocular 3d object detection." In: *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition workshops*. 2020, pp. 1000–1001.

[29] C. Doersch, A. Gupta, and A. A. Efros. "Unsupervised visual representation learning by context prediction." In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1422–1430.

[30] M. D. Donsker and S. R. S. Varadhan. "Asymptotic evaluation of certain Markov process expectations for large time." In: 1975.

[31] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale." In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=YicbFdNTTy.

[32] L. Ericsson, H. Gouk, and T. M. Hospedales. "How well do self-supervised models transfer?" In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 5414–5423.

[33] L. Ericsson, H. Gouk, and T. M. Hospedales. "How Well Do Self-Supervised Models Transfer?" In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 5410–5419. DOI: 10.1109/CVPR46437.2021.00537.

[34] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. "The Pascal Visual Object Classes Challenge: A Retrospective." In: *International Journal of Computer Vision* 111 (2014), pp. 98–136.

[35] K. P. F.R.S. "LIII. On lines and planes of closest fit to systems of points in space." In: *Philosophical Magazine Series 1* 2 (1901), pp. 559–572.

[36] W. Falcon and The PyTorch Lightning team. *PyTorch Lightning*. Version 1.4. Mar. 2019. DOI: 10.5281/zenodo.3828935. URL: https://github.com/Lightning-AI/lightning.

[37] P. S. Foundation. *Python Language Reference, version 3.12*. https://www.python.org. 2025.

[38] Z. Ghahramani. "Unsupervised Learning." In: *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2 - 14, 2003, Tübingen, Germany, August 4 - 16, 2003, Revised Lectures*. Ed. by O. Bousquet, U. von Luxburg, and G. Rätsch. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 72–112. ISBN: 978-3-540-28650-9. DOI: 10.1007/978-3-540-28650-9_5. URL: https://doi.org/10.1007/978-3-540-28650-9_5.

[39] B. Ghojogh, A. Ghodsi, F. Karray, and M. Crowley. *Johnson-Lindenstrauss Lemma, Linear and Nonlinear Random Projections, Random Fourier Features, and Random Kitchen Sinks: Tutorial and Survey*. 2021. arXiv: 2108.04172.

[40] S. Gidaris, P. Singh, and N. Komodakis. "Unsupervised representation learning by predicting image rotations." In: *arXiv preprint arXiv:1803.07728* (2018).

[41] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. "Generative adversarial nets." In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'14. Montreal, Canada: MIT Press, 2014, pp. 2672–2680.

[42] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko. "Bootstrap your own latent a new approach to self-supervised learning." In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS '20. Vancouver, BC, Canada: Curran Associates Inc., 2020. ISBN: 9781713829546.

[43] J. Gui, T. Chen, J. Zhang, Q. Cao, Z. Sun, H. Luo, and D. Tao. "A Survey on Self-Supervised Learning: Algorithms, Applications, and Future Trends." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46.12 (2024), pp. 9052–9071. DOI: 10.1109/TPAMI.2024.3415112.

[44] M. Gutmann and A. Hyvärinen. "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models." In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Y. W. Teh and M. Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, May 2010, pp. 297–304. URL: https://proceedings.mlr.press/v9/gutmann10a.html.

[45] R. Hadsell, S. Chopra, and Y. LeCun. "Dimensionality Reduction by Learning an Invariant Mapping." In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)* 2 (2006), pp. 1735–1742.

[46] D. Han, J. Kim, and J. Kim. "Deep Pyramidal Residual Networks." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[47] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. Fernández del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. "Array programming with NumPy." In: *Nature* 585 (2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2.

[48] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. "Masked Autoencoders Are Scalable Vision Learners." In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 15979–15988. DOI: 10.1109/CVPR52688.2022.01553.

[49] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. "Momentum Contrast for Unsupervised Visual Representation Learning." In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 9726–9735. DOI: 10.1109/CVPR42600.2020.00975.

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

[51] K. He, X. Zhang, S. Ren, and J. Sun. "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification." In: *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*. ICCV '15. USA: IEEE Computer Society, 2015, pp. 1026–1034. ISBN: 9781467383912. DOI: 10.1109/ICCV.2015.123. URL: https://doi.org/10.1109/ICCV.2015.123.

[52] G. Hinton, O. Vinyals, and J. Dean. "Distilling the Knowledge in a Neural Network." In: *NIPS Deep Learning and Representation Learning Workshop*. 2015. eprint: 1503.02531.

[53] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications." In: *CoRR* abs/1704.04861 (2017).

[54] J. Hu, L. Shen, and G. Sun. "Squeeze-and-Excitation Networks." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018.

[55] K.-C. Huang, T.-H. Wu, H.-T. Su, and W. H. Hsu. "Monodtr: Monocular 3d object detection with depth-aware transformer." In: *CVPR*. 2022, pp. 4012–4021.

[56] L. Jing, P. Vincent, Y. LeCun, and Y. Tian. "Understanding Dimensional Collapse in Contrastive Self-supervised Learning." In: *CoRR* abs/2110.09348 (2021). arXiv: 2110.09348.

[57] L. Jing, P. Vincent, Y. LeCun, and Y. Tian. "Understanding Dimensional Collapse in Contrastive Self-supervised Learning." In: *CoRR* abs/2110.09348 (2021).

[58] L. Jing and Y. Tian. "Self-Supervised Visual Feature Learning With Deep Neural Networks: A Survey." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43 (2019), pp. 4037–4058.

[59] W. Johnson and J. Lindenstrauss. "Extensions of Lipschitz mappings into a Hilbert space." In: *Conference in Modern Analysis and Probability* 26 (Jan. 1982), pp. 189–206.

[60] C. Kim, U.-H. Kim, and J.-H. Kim. "Self-supervised 3D object detection from monocular pseudo-LiDAR." In: *2022 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE. 2022, pp. 1–6.

[61] D. P. Kingma and M. Welling. "An Introduction to Variational Autoencoders." In: *Found. Trends Mach. Learn.* 12.4 (Nov. 2019), p. 307. ISSN: 1935-8237. DOI: 10.1561/2200000056. URL: https://doi.org/10.1561/2200000056.

[62] A. Krizhevsky. "Learning Multiple Layers of Features from Tiny Images." In: *University of Toronto* (May 2012).

[63] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks." In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira, C. Burges, L. Bottou, and K. Weinberger. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

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

[65] A. Kumar, G. Brazil, E. Corona, A. Parchami, and X. Liu. "Deviant: Depth equivariant network for monocular 3d object detection." In: *ECCV*. Springer. 2022, pp. 664–683.

[66] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov, T. Duerig, and V. Ferrari. "The Open Images Dataset V4: Unified Image Classification, Object Detection, and Visual Relationship Detection at Scale." In: *International Journal of Computer Vision* 128.7 (Mar. 2020), pp. 1956–1981. ISSN: 1573-1405. DOI: 10.1007/s11263-020-01316-z. URL: http://dx.doi.org/10.1007/s11263-020-01316-z.

[67] Leibniz Supercomputing Centre. *Leibniz Supercomputing Centre (LRZ)*. https://www.lrz.de. 2025.

[68] Z. Li, J. Jia, and Y. Shi. "MonoLSS: Learnable sample selection for monocular 3D detection." In: *2024 International Conference on 3D Vision (3DV)*. IEEE. 2024, pp. 1125–1135.

[69] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. "Feature pyramid networks for object detection." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2117–2125.

[70] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. "Microsoft COCO: Common Objects in Context." In: *Computer Vision – ECCV 2014*. Ed. by D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars. Cham: Springer International Publishing, 2014, pp. 740–755. ISBN: 978-3-319-10602-1.

[71] K. Liu, X. Zheng, C. Wang, K. Tang, M. Liu, and B. Chen. *Generalized 3D Self-supervised Learning Framework via Prompted Foreground-Aware Feature Contrast*. 2023. arXiv: 2303.06388 [cs.CV].

[72] X. Liu, N. Xue, and T. Wu. "Learning auxiliary monocular contexts helps monocular 3D object detection." In: *AAAI*. Vol. 36. 2. 2022, pp. 1810–1818.

[73] X. Liu, Y. Li, and S. Wang. "Representation Distillation for Efficient Self-Supervised Learning." In: *2024 IEEE International Conference on Multimedia and Expo (ICME)* (2024), pp. 1–6.

[74] Z. Liu, Z. Wu, and R. Tóth. "Smoke: Single-stage monocular 3d object detection via keypoint estimation." In: *CVPRW*. 2020, pp. 996–997.

[75] Z. Liu. "Super Convergence Cosine Annealing with Warm-Up Learning Rate." In: *CAIBDA 2022; 2nd International Conference on Artificial Intelligence, Big Data and Algorithms*. 2022, pp. 1–7.

[76] I. Loshchilov and F. Hutter. "SGDR: Stochastic Gradient Descent with Restarts." In: *CoRR* abs/1608.03983 (2016).

[77] X. Ma, Y. Zhang, D. Xu, D. Zhou, S. Yi, H. Li, and W. Ouyang. "Delving into localization errors for monocular 3d object detection." In: *CVPR*. 2021, pp. 4721–4730.

[78] L. van der Maaten and G. Hinton. "Visualizing Data using t-SNE." In: *Journal of Machine Learning Research* 9.86 (2008), pp. 2579–2605.

[79] T. maintainers and contributors. *TorchVision: PyTorch's Computer Vision library*. Nov. 2016. URL: https://github.com/pytorch/vision.

[80] D. McAllester and K. Stratos. "Formal Limitations on the Measurement of Mutual Information." In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by S. Chiappa and R. Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, Aug. 2020, pp. 875–884. URL: https://proceedings.mlr.press/v108/mcallester20a.html.

[81] L. McInnes and J. Healy. "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction." In: *ArXiv* abs/1802.03426 (2018).

[82] T. Moon. "The expectation-maximization algorithm." In: *IEEE Signal Processing Magazine* 13.6 (1996), pp. 47–60. DOI: 10.1109/79.543975.

[83] A. Mousavian, D. Anguelov, J. Flynn, and J. Košecká. "3D Bounding Box Estimation Using Deep Learning and Geometry." In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5632–5640. DOI: 10.1109/CVPR.2017.597.

[84] T. N. Mundhenk, D. Ho, and B. Y. Chen. "Improvements to Context Based Self-Supervised Learning." In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, June 2018, pp. 9339–9348. DOI: 10.1109/cvpr.2018.00973. URL: http://dx.doi.org/10.1109/CVPR.2018.00973.

[85] M. Noroozi and P. Favaro. "Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles." In: *Computer Vision – ECCV 2016*. Ed. by B. Leibe, J. Matas, N. Sebe, and M. Welling. Cham: Springer International Publishing, 2016, pp. 69–84. ISBN: 978-3-319-46466-4.

[86] S. Nowozin, B. Cseke, and R. Tomioka. "f-GAN: training generative neural samplers using variational divergence minimization." In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS'16. Barcelona, Spain: Curran Associates Inc., 2016, pp. 271–279. ISBN: 9781510838819.

[87] NVIDIA, P. Vingelmann, and F. H. Fitzek. *CUDA, release: 10.2.89*. 2020. URL: https://developer.nvidia.com/cuda-toolkit.

[88] A. v. d. Oord, Y. Li, and O. Vinyals. "Representation learning with contrastive predictive coding." In: *arXiv preprint arXiv:1807.03748* (2018).

[89] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, R. Howes, P.-Y. Huang, H. Xu, V. Sharma, S.-W. Li, W. Galuba, M. Rabbat, M. Assran, N. Ballas, G. Synnaeve, I. Misra, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. *DINOv2: Learning Robust Visual Features without Supervision*. 2023.

[90] D. Park, R. Ambrus, V. Guizilini, J. Li, and A. Gaidon. "Is pseudo-lidar needed for monocular 3d object detection?" In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 3142–3152.

[91] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. "PyTorch: an imperative style, high-performance deep learning library." In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[92] D. Pototzky, A. Sultan, and L. Schmidt-Thieme. "FastSiam: Resource-Efficient Self-supervised Learning on Single GPU." In: *Pattern Recognition: 44th DAGM German Conference, DAGM GCPR 2022, Konstanz, Germany, September 27–30, 2022, Proceedings*. Konstanz, Germany: Springer-Verlag, 2022, pp. 53–67. ISBN: 978-3-031-16787-4. DOI: 10.1007/978-3-031-16788-1_4. URL: https://doi.org/10.1007/978-3-031-16788-1_4.

[93] Z. Qin and X. Li. "Monoground: Detecting monocular 3d objects from the ground." In: *CVPR*. 2022, pp. 3793–3802.

[94] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. "Zero-Shot Text-to-Image Generation." In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, July 2021, pp. 8821–8831. URL: https://proceedings.mlr.press/v139/ramesh21a.html.

[95] H. Robbins and S. Monro. "A Stochastic Approximation Method." In: *The Annals of Mathematical Statistics* 22.3 (1951), pp. 400–407.

[96] S. T. Roweis and L. K. Saul. "Nonlinear Dimensionality Reduction by Locally Linear Embedding." In: *Science* 290.5500 (2000), pp. 2323–2326. DOI: 10.1126/science.290.5500.2323. eprint: https://www.science.org/doi/pdf/10.1126/science.290.5500.2323. URL: https://www.science.org/doi/abs/10.1126/science.290.5500.2323.

[97] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. "MobileNetV2: Inverted Residuals and Linear Bottlenecks." In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4510–4520. DOI: 10.1109/CVPR.2018.00474.

[98] F. Schroff, D. Kalenichenko, and J. Philbin. "FaceNet: A unified embedding for face recognition and clustering." In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 815–823. DOI: 10.1109/CVPR.2015.7298682.

[99] O. Shrout, O. Nizan, Y. Ben-Shabat, and A. Tal. "Patchcontrast: Self-supervised pre-training for 3d object detection." In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 2456–2466.

[100] K. Sohn. "Improved deep metric learning with multi-class N-pair loss objective." In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. NIPS'16. Barcelona, Spain: Curran Associates Inc., 2016, pp. 1857–1865. ISBN: 9781510838819.

[101] I. Susmelj, M. Heller, P. Wirth, J. Prescott, M. Ebner, and et al. *Lightly*. URL: https://github.com/lightly-ai/lightly.

[102] M. Tan and Q. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks." In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by K. Chaudhuri and R. Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, June 2019, pp. 6105–6114. URL: https://proceedings.mlr.press/v97/tan19a.html.

[103] D. Teney, A. M. Nicolicioiu, V. Hartmann, and E. Abbasnejad. "Neural Redshift: Random Networks are not Random Functions." In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 4786–4796. DOI: 10.1109/CVPR52733.2024.00458.

[104] Y. Tian, D. Krishnan, and P. Isola. "Contrastive Multiview Coding." In: *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI*. Glasgow, United Kingdom: Springer-Verlag, 2020, pp. 776–794. ISBN: 978-3-030-58620-1. DOI: 10.1007/978-3-030-58621-8_45. URL: https://doi.org/10.1007/978-3-030-58621-8_45.

[105] Z. Tian, C. Shen, H. Chen, and T. He. "FCOS: Fully Convolutional One-Stage Object Detection." In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 9626–9635. DOI: 10.1109/ICCV.2019.00972.

[106] N. Tishby and N. Zaslavsky. "Deep learning and the information bottleneck principle." In: *2015 IEEE Information Theory Workshop (ITW)*. 2015, pp. 1–5. DOI: 10.1109/ITW.2015.7133169.

[107] J. Wang, L. Song, Z. Li, H. Sun, J. Sun, and N. Zheng. "End-to-End Object Detection with Fully Convolutional Network." In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 15844–15853. DOI: 10.1109/CVPR46437.2021.01559.

[108] T. Wang, X. ZHU, J. Pang, and D. Lin. "Probabilistic and Geometric Depth: Detecting Objects in Perspective." In: *Proceedings of the 5th Conference on Robot Learning*. Ed. by A. Faust, D. Hsu, and G. Neumann. Vol. 164. Proceedings of Machine Learning Research. PMLR, Nov. 2022, pp. 1475–1485. URL: https://proceedings.mlr.press/v164/wang22i.html.

[109] T. Wang, X. Zhu, J. Pang, and D. Lin. "Fcos3d: Fully convolutional one-stage monocular 3d object detection." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 913–922.

[110] S. Wei, G. Chen, W. Chi, Z. Wang, and L. Sun. "3D Object Aided Self-Supervised Monocular Depth Estimation." In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 10635–10642.

[111] M. Wu, C. Zhuang, M. Mossé, D. L. K. Yamins, and N. D. Goodman. "On Mutual Information in Contrastive Learning for Visual Representations." In: *ArXiv* abs/2005.13149 (2020).

[112] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin. "Unsupervised Feature Learning via Non-parametric Instance Discrimination." In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3733–3742. DOI: 10.1109/CVPR.2018.00393.

[113] T. Xiao, S. Li, B. Wang, L. Lin, and X. Wang. "Joint Detection and Identification Feature Learning for Person Search." In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 3376–3385.

[114] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le. "Self-Training With Noisy Student Improves ImageNet Classification." In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 10684–10695. DOI: 10.1109/CVPR42600.2020.01070.

[115] Z. Xie, Z. Zhang, Y. Cao, Y. Lin, J. Bao, Z. Yao, Q. Dai, and H. Hu. "SimMIM: a Simple Framework for Masked Image Modeling." In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 9643–9653. DOI: 10.1109/CVPR52688.2022.00943.

[116] B. Xu and Z. Chen. "Multi-level Fusion Based 3D Object Detection from Monocular Images." In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 2345–2353. DOI: 10.1109/CVPR.2018.00249.

[117] L. Yan, P. Yan, S. Xiong, X. Xiang, and Y. Tan. "MonoCD: Monocular 3d object detection with complementary depths." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 10248–10257.

[118] Y. You, I. Gitman, and B. Ginsburg. *Large Batch Training of Convolutional Networks*. 2017. arXiv: 1708.03888 [cs.CV].

[119] E. Yurtsever, E. Erçelik, M. Liu, Z. Yang, H. Zhang, P. Topçam, M. Listl, Y. K. Çaylı, and A. Knoll. *3D Object Detection with a Self-supervised Lidar Scene Flow Backbone*. 2022. arXiv: 2205.00705 [cs.CV].

[120] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny. "Barlow Twins: Self-Supervised Learning via Redundancy Reduction." In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, July 2021, pp. 12310–12320. URL: https://proceedings.mlr.press/v139/zbontar21a.html.

[121] R. Zhang, H. Qiu, T. Wang, Z. Guo, Z. Cui, Y. Qiao, H. Li, and P. Gao. "Monodetr: Depth-guided transformer for monocular 3d object detection." In: *ICCV*. 2023, pp. 9155–9166.

[122] R. Zhang, P. Isola, and A. A. Efros. "Colorful Image Colorization." In: *Computer Vision – ECCV 2016*. Ed. by B. Leibe, J. Matas, N. Sebe, and M. Welling. Cham: Springer International Publishing, 2016, pp. 649–666. ISBN: 978-3-319-46487-9.

[123] Y. Zhang, J. Lu, and J. Zhou. "Objects are different: Flexible monocular 3d object detection." In: *CVPR*. 2021, pp. 3289–3298.

[124] Z. Zhang, R. Girdhar, A. Joulin, and I. Misra. "Self-supervised pretraining of 3d features on any point-cloud." In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 10252–10263.

[125] N. Zhao, Z. Wu, R. W. H. Lau, and S. Lin. "What makes instance discrimination good for transfer learning?" In: *ArXiv* abs/2006.06606 (2020).

[126] Y. Zhou, H. Zhu, Q. Liu, S. Chang, and M. Guo. "MonoATT: Online Monocular 3D Object Detection with Adaptive Token Transformer." In: *CVPR*. 2023, pp. 17493–17503.