# Comparative review of linear probing in Self-Supervised Learning model under different domain settings

By

Abijith Ashok

Honors Thesis

Department of Computer Science
University of British Columbia -Okanagan

April 2024

# Abstract

This thesis conducts a comprehensive evaluation of self-supervised learning (SSL) models through linear probing across varied domain-specific datasets—Pneumonia, STL10, and CIFAR-10—within the constraints of single GPU training environments. The study benchmarks the performance of notable SSL models including MoCoV2, SimCLR, DINO, MAE, and SwaV, to assess their capability in learning robust and transferable features without labeled data. Each model was initially trained on the ImageNet1k dataset, followed by linear probing to measure their adaptability and generalization across different visual domains. Our findings indicate that DINO consistently outperforms other models, demonstrating superior capability to distill and generalize knowledge effectively in limited computational settings. SimCLR also shows strong performance, particularly in handling complex image features, suggesting that efficient feature extraction is feasible without extensive resources. In contrast, SwaV and MoCoV2 exhibit variable results, potentially due to their dependence on configurations less suited to single GPU constraints. This research highlights the potential of SSL models to advance real-world applications in various domains, particularly where access to extensive computational resources is restricted. The insights derived from this study contribute to the broader discourse in machine learning, advocating for further optimization of SSL techniques to enhance their practicality and efficacy.

# Chapter 1

# Introduction

In recent years, self-supervised learning (SSL) has emerged as a promising approach for learning representations from unlabeled data[13], offering significant advantages in various machine learning tasks. By leveraging the inherent structure or relationships within the data itself, SSL methods aim to extract meaningful and informative representations, thereby circumventing the need for labeled datasets which are often scarce and expensive to obtain. One common evaluation protocol used to assess the quality of learned representations in SSL is linear probing. Linear probing involves training linear classifiers on top of SSL-trained representations and evaluating their performance on downstream tasks, providing insights into the effectiveness of SSL models.

While SSL has demonstrated remarkable success in many domains, its performance can vary significantly depending on the characteristics of the data distribution. In particular, SSL models often face challenges when confronted with out-of-domain settings, where the test distribution differs from the distribution of the training data. In such scenarios, the ability of SSL models to generalize across domains becomes crucial for achieving robust performance in real-world applications. Linear probing serves as a benchmark for evaluating SSL generalization across different domain settings, offering valuable insights into the transferability of learned representations.

The purpose of this comparative review is to examine the effectiveness of linear probing in SSL models under various domain settings. We aim to provide a comprehensive analysis of existing literature, comparing the performance of SSL models across different domains and identifying key factors that influence SSL generalization. By synthesizing findings from multiple studies, we seek to elucidate the strengths and limitations of linear probing in assessing SSL performance and to offer insights into potential strategies for improving model robustness across diverse domains.

In the following sections, we will first provide an overview of SSL and linear probing, discussing their importance in representation learning and evaluation.

We will then delve into the challenges posed by different domain settings and the significance of domain generalization in SSL. Finally, we will outline the scope and methodology of our comparative review, setting the stage for a comprehensive analysis of linear probing in SSL across various domain settings.

## 1.1 Research Motivation

The impetus for this research stems from my immense interest in self-supervised learning (SSL) as a paradigm capable of overcoming the limitations posed by supervised learning, particularly the dependency on large labeled datasets, which are often costly and labor-intensive to produce. SSL techniques, by leveraging unlabeled data, promise to reduce these dependencies significantly. The versatility and robustness of feature representations learned through SSL methods are crucial for their applicability across a diverse array of domains—from standard image recognition tasks in datasets like ImageNet and CIFAR-10 to more specialized applications such as medical imaging in the Pneumonia dataset. This study is motivated by the need to critically evaluate how well these SSL models, specifically those utilizing techniques such as contrastive learning and distillation, perform when their learned representations are subjected to linear probing across different datasets. The ultimate goal is to ascertain the transferability and effectiveness of SSL representations, which could significantly influence future research directions and practical applications in fields where annotated data are scarce or where domain adaptation is necessary.

## 1.2 Research Objectives

The primary research objective of this study is to systematically evaluate the effectiveness of self-supervised learning (SSL) models in generating robust and transferable feature representations. Specifically, the study aims to:

1. Assess the Generalization of SSL Models: Determine the capability of different SSL models—such as MoCoV2, DINO, MAE, SimCLR, and SwaV—to generalize their learned representations to various datasets that differ significantly in terms of complexity, resolution, and domain specificity (ImageNet1k, STL10, CIFAR-10, and Pneumonia).
2. Quantify the Effectiveness of Linear Probing as a Benchmarking Tool: Utilize linear probing to measure the quality of features extracted by SSL models, providing a comparative analysis across models to identify which architectures or

training methodologies yield the most versatile and powerful representations.
3. Explore the Impact of Model Architectures and Training Techniques: Examine how different architectures (e.g., Vision Transformers vs. ResNet) and specific SSL techniques (e.g., contrastive loss, distillation) influence the performance and adaptability of the models to new and unseen data.
4. Identify Strengths and Limitations in SSL Approaches: Pinpoint the strengths and potential limitations of SSL models in handling domain-specific challenges, such as medical imaging, compared to more conventional image recognition tasks.
5. Provide Insights for Future SSL Method Developments: Based on the findings, suggest improvements and future research directions for SSL methodologies to enhance their applicability and effectiveness in diverse real-world applications.

## 1.3 Contribution summary

The key contributions of this study are summarized as follows:

1. Comparative Evaluation of SSL Models: This study provides a comprehensive comparison of several leading SSL models, including MoCoV2, DINO, MAE, SimCLR, and SwaV. By systematically evaluating these models across diverse datasets such as ImageNet1k, STL10, CIFAR-10, and the Pneumonia dataset, the research highlights the strengths and weaknesses of each model in various settings, contributing to a nuanced understanding of which features and model architectures are most effective for specific types of visual data.
2. Benchmarking SSL Effectiveness through Linear Probing: The use of linear probing to benchmark the quality of learned features across different models serves as a rigorous methodological contribution. This approach not only quantifies the transferability and effectiveness of the learned representations but also establishes a standardized metric for comparing SSL models in future studies.
3. Insights into SSL Architectures and Training Techniques: By analyzing how different architectures and training techniques impact model performance, this research sheds light on the underlying mechanisms that contribute to the success or failure of SSL approaches. These insights can guide the development of more robust and efficient SSL models.
4. Domain-Specific Adaptation and Performance: The study extends the application of SSL models to the domain of medical imaging, specifically through the evaluation on the Pneumonia dataset. This contributes to the growing interest in applying advanced machine learning techniques in healthcare, providing a pathway for using SSL in medical diagnostic processes where labeled data are scarce.

# Chapter 2

# Methodology

The primary objective of this study was to evaluate the effectiveness of linear probing as a technique for assessing the quality of features learned by self-supervised learning (SSL) models under different domain conditions. This evaluation was specifically structured around a comparative analysis where each model, except for the MAE linear probing model, was configured with a similar size of approximately 94 million parameters. This consistency in model size aimed to ensure a fair comparison across models in terms of computational load and memory requirements. The MAE model was an exception due to its architecture, necessitating roughly 113 million parameters based on recommendations from the original paper [1].

The study was designed to be accessible to individuals with limited computational resources, specifically targeting those with a single GPU possessing under 5GB of GPU RAM. This consideration influenced the choice and configuration of SSL models, ensuring that the experiments could be replicated easily using the models and training regimes available from the publicly accessible repository. This approach not only enhances the practical relevance of the study's findings but also allows researchers and practitioners with limited hardware capabilities to participate in advanced SSL evaluations without requiring high-end computational infrastructure.

Five Self-Supervised Learning (SSL) models were selected for this study based on their prominence and diverse approaches to learning representations: SimCLR [2], MoCoV2[3], DINO[4], MAE, and SwaV[5]. Each model offers unique mechanisms for handling unlabeled data, making them ideal candidates for exploring the robustness of learned features across different datasets.

Each of the five SSL models were then Pre-Trained on ImageNet1k. The ImageNet1k dataset, comprising over a million labeled images across 1,000 classes, served as the primary training ground for each model. This dataset was chosen for its complexity and size, which challenge the models to develop robust and generalizable features. Training on ImageNet1k also provides a common ground for comparing the performance of different SSL models.

After training on ImageNet1k, checkpoints of each model were saved. These checkpoints represent the state of the model at epoch 100. and contain all the information about the model weights and training state. Saving checkpoints is crucial as it allows for the transfer of learned features to different tasks without the need to retrain the models from scratch.

The saved checkpoints were then used as starting points for training on three other datasets: STL10, CIFAR-10, and Pneumonia. For the Pneumonia and STL10 datasets, a rebalancing step was introduced to ensure a fair train-test-validation(70-15-15) split, aiding in a more balanced evaluation of the linear model performance across classes.

STL10, CIFAR-10, and Pneumonia were selected as evaluation datasets allowing the study to cover a range of domain conditions—from closely related to the training domain (STL10 and CIFAR-10) to significantly different (Pneumonia). This selection is designed to test the hypothesis that pre-trained SSL models can adapt to both similar and distinctly different domain conditions through effective transfer learning strategies.

By detailing and ensuring transparency in the methodologies used to train, save, and apply these models, this study aims to provide insights into the adaptability and efficiency of SSL models when faced with varying domain-specific challenges. The outcomes are expected to contribute to the broader understanding of how self-supervised learning can be effectively deployed in real-world scenarios, where labeled data is scarce or costly to obtain.

## 2.1 Detailed Model Descriptions

In this section, we delve into the specifics of each self-supervised

learning model used in our study. These models were selected based on their unique approaches to learning representations from unlabeled data, their proven efficacy in prior studies, and their compatibility with our research goals.

### 2.1.1 SimCLR (Simple Contrastive Learning of Representations)

SimCLR[2] is a contrastive learning framework that learns visual representations by maximizing agreement between differently augmented views of the same data sample in a latent space.

**Architecture:** It utilizes a standard architecture that includes a base encoder (in our case a resnet50 [6][14]) followed by a projection head.

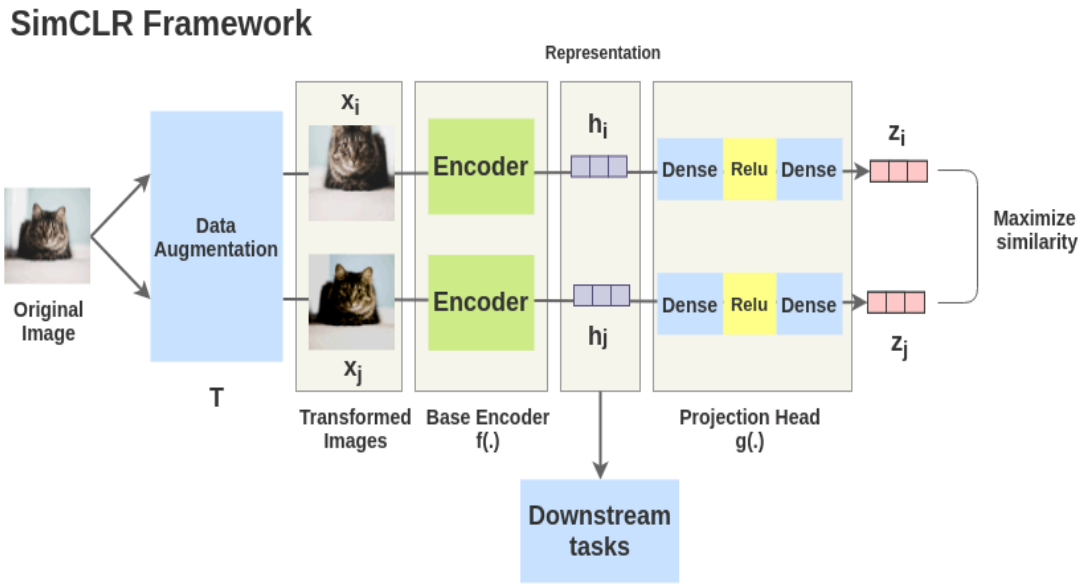


**Figure 2.1.1**: Figure obtained from "The Illustrated SimCLR Framework." [15]

**Training Mechanism:** The model is trained using a contrastive loss function, which forces the representations of augmented versions of the same image to be similar, while pushing representations of different images apart.

The original paper defines a random transformation function T that takes an image and applies a combination of `random (crop + flip + color`

`jitter + grayscale)`. This random transformation function is applied to get a pair of 2 images. Thus, for our batch size of 128, we get 2*N = 256 total images. These transformed images are prepared in similar pairs within every batch before being passed through an encoder to get image representations. In the original paper and in our setup we use the ResNet50[6] architecture as the ConvNet [7] encoder. The output is a 2048 dimensional vector h.

The similarity between two augmented versions of an image is calculated using cosine similarity [8]. For two augmented images $xi$ and $xj$, the cosine similarity is calculated on its projected representations $zi$ and $zj$.

Cosine similarity between image embeddings

$$S_{i,j} = \frac{z_i^T z_j}{(\tau ||z_i|| . ||z_j||)}$$

where,

- $\tau$ is the adjustable temperature parameter. It can scale the inputs and widen the range `[-1, 1]` of cosine similarity
- $\|zi\|$ is the norm of the vector.

**Loss calculation:** SimCLR uses a contrastive loss called “NT-Xent loss” (Normalized Temperature-Scaled Cross-Entropy Loss [9]). First, the augmented pairs in the batch are taken one by one. Next, we apply the softmax function to get the probability of these two images being similar. This softmax calculation is equivalent to getting the probability of the second augmented image being the most similar to the first image in the pair. Here, all remaining images in the batch are sampled as a dissimilar image (negative pair). Thus, we don’t need specialized architecture, memory banks or queues as needed by other approaches like InstDisc [10] , MoCo [3] or PIRL [11].

$$Softmax = \frac{Similarity(each\ pair)}{\sum_{Sum\ of\ all\ pairs\ in\ a\ batch} Similarity(first\ image\ in\ pair, all\ images\ in\ batch)}$$

Then, the loss is calculated for a pair by taking the negative of the log of the above calculation. This formulation is the Noise Contrastive Estimation(NCE) Loss.

$$l(i,j) = -log\left(\frac{exp(s_{i,j})}{\sum_{k=1}^{2N} l_{[k!=i]} exp(s_{i,k})}\right)$$

We calculate the loss for the same pair a second time as well where the positions of the images are interchanged. Finally, we compute loss over all the pairs in the batch of size N=2 and take an average.

$$L = \frac{1}{2N}\sum_{k=1}^{N}[l(2k-1,\ 2k) + l(2k,\ 2k-1)]$$

Based on the loss, the encoder and projection head representations improve over time and the representations obtained place similar images closer in the space.

**Relevance to Study:** SimCLR was included due to its ability to effectively learn discriminative features from complex image datasets like ImageNet1k, making it suitable for testing on diverse downstream tasks.

### 2.1.2 MocoV2 (Momentum Contrast Version 2)

MoCo v2 [3] is an extension of the MoCo framework, which is designed for unsupervised learning of visual representations by creating a dynamic dictionary of encoded data samples with a momentum-updated encoder. This approach allows for a consistent representation of the dictionary keys over time, improving the quality of the contrastive learning process.

**Architecture**: MoCo v2 uses a base encoder and a momentum encoder, both employing the ResNet50 architecture. Similar to SimCLR, it outputs a 2048-dimensional feature vector. The primary difference in architecture from SimCLR is the addition of a momentum encoder that gradually updates its parameters using a moving average of the base encoder's parameters, ensuring slow evolution of the dictionary representations.

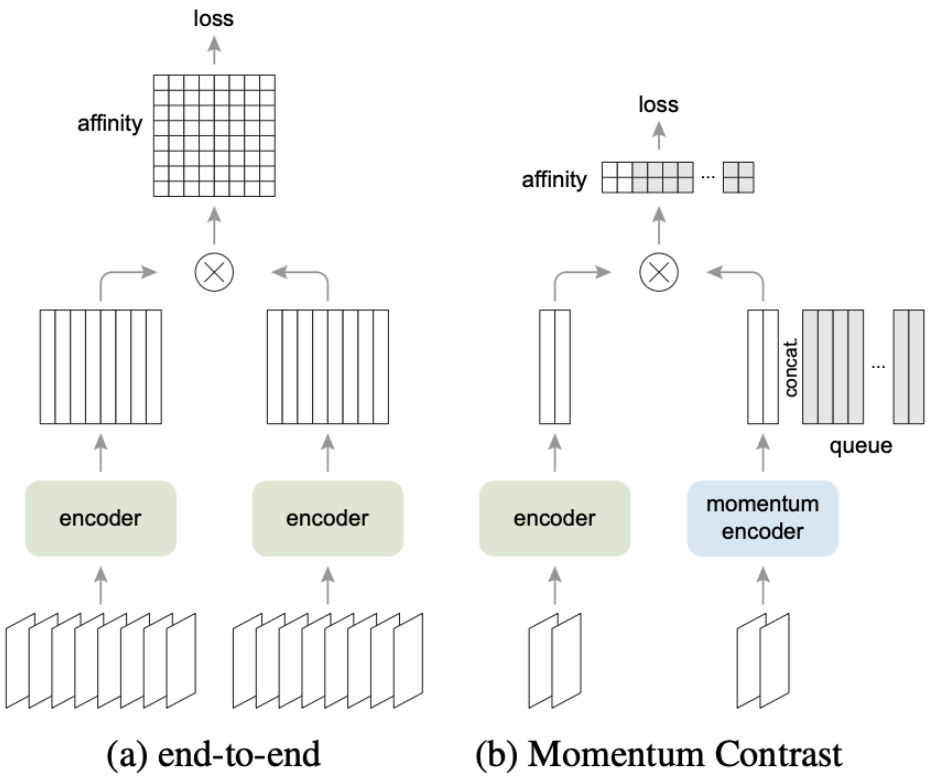


Figure 1. A **batching** perspective of two optimization mechanisms for contrastive learning. Images are encoded into a representation space, in which pairwise affinities are computed.

**Figure 2.1.2:** Figure 1 from the original paper

**Training Mechanism:** MoCo v2 enhances the original MoCo by incorporating a queue that holds a set of previously computed representations from the momentum encoder. These representations serve as negative samples in the contrastive learning loss calculation, making it possible to have a large and consistent set of negatives without the need for very large batch sizes. The model processes a batch of images where each image is subjected to random transformations (crop, flip, color jitter, and grayscale) to generate a query and a key. The query is encoded by the base encoder, while the key is encoded by the momentum encoder.

The core idea is to maximize the similarity of the query to its corresponding key while minimizing its similarity to other keys (negative samples) in the queue. The similarity measurement is performed using cosine similarity, similar to SimCLR.

**Loss calculation:** The contrastive loss used in MoCo v2 is the InfoNCE loss, which is a variant of the NT-Xent loss used by SimCLR. The loss is computed by considering a positive pair (a query and its corresponding key) and multiple negative pairs (the query and the other keys in the queue). The softmax function computes the probability that a positive pair is more similar than the negative pairs:

$$Softmax = \frac{exp(similarity(query, key)/\tau)}{\sum_{all\ keys\ in\ the\ queue} exp(similarity(query, other\ keys)/\tau}$$

where
$\tau$ is a temperature scaling parameter that adjusts the sensitivity of the softmax distribution.

**Queue and Momentum Update**: The queue is updated by enqueuing the current batch's keys and dequeuing the oldest keys to maintain a fixed size. The momentum encoder's parameters are updated as a weighted sum of its own parameters and the base encoder's parameters, controlled by a momentum coefficient. This slow update helps in maintaining consistency of the dictionary keys over time.

**Relevance to the study**: Moco was chosen because of its ability to handle representations over large datasets without needing large batch sizes makes it particularly useful for environments with computational constraints.

### 2.1.3 DINO (Self-Distillation with No Labels)

DINO is a self-supervised learning framework that employs the concept of knowledge distillation in a teacher-student setup to learn visual representations without the need for labeled data. The key idea is to create consistency between differently augmented views of the same image, processed by the teacher and student networks, which evolve during training.

**Architecture:** DINO utilizes two networks, a student and a teacher, both typically based on the Vision Transformer (ViT) architecture, although it can also be implemented using CNNs like ResNet50. Both networks output high-dimensional feature vectors, but the teacher network's weights are a moving average of the student's, updated slowly to provide

stable targets for the student.

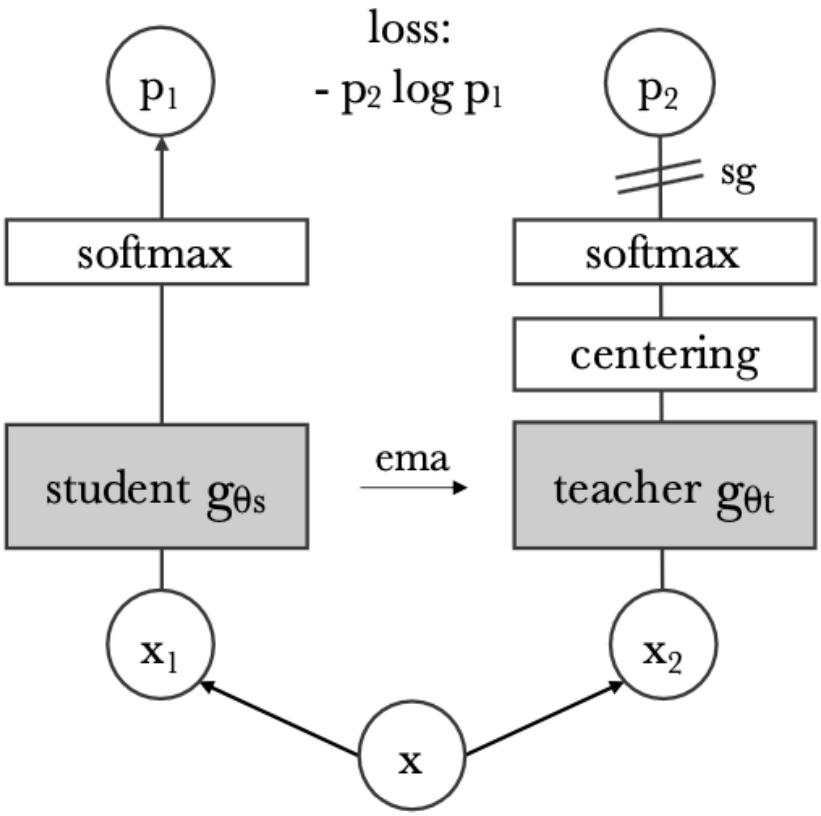


**Fig: 2.1.3**: Model Diagram from the original paper

**Training Mechanism:** During training, two differently augmented views of the same image are generated. One view is processed by the student network, and the other by the teacher network. The augmentations include random crops, color jitter, Gaussian blur, and solarization, which are more aggressive compared to typical contrastive learning methods, promoting robustness in the learned features.

**Knowledge Distillation:** The core of DINO lies in using the outputs (i.e., the logits from the last layer before softmax) of the teacher network as soft labels to train the student network. The student learns to predict the same class distribution as the teacher for corresponding augmented views of the same image. This distillation process does not require

external labels, relying solely on the consistency between the teacher's and student's predictions.

**Centering and Sharpening:** The outputs from the teacher are sharpened using a temperature parameter to make the distribution more peaky, which helps in emphasizing more confident predictions. Simultaneously, a centering mechanism is applied where a center vector (a moving average of the teacher's outputs) is subtracted from the teacher's logits before the softmax application. This helps in stabilizing the training by reducing the dominance of any particular feature in the teacher's output.

**Loss Calculation:** The student's predictions are matched against the sharpened and centered teacher's outputs using a cross-entropy loss. This drives the student to produce similar predictions to the teacher for the same augmented view of an image, thereby learning robust and generalizable visual features.

**Relevance to the study:** DINO was chosen for its innovative approach to self-supervised learning that mimics supervised techniques, ideal for exploring the potential of Self-supervised learning in more structured applications.

### 2.1.4 MAE (Masked Autoencoder for Self-supervised learning)

MAE is a self-supervised learning framework that employs an autoencoder architecture, specifically designed to learn high-quality visual representations by reconstructing images from partial inputs. This method is based on the principle of masking a significant portion of the input image and predicting the missing parts.

**Architecture:** The MAE framework consists of two main components: an encoder and a decoder. The encoder we have used is the vit-base-path16-224 that processes only the visible parts of the input image, while the decoder is typically a smaller and lighter network compared to the encoder. The encoder extracts features from the unmasked portions of the image, and these features are then used by the decoder to reconstruct the original image.

**Training Mechanism:** During training, 75% of the input image is randomly masked, and the encoder only sees the remaining visible

parts. This selective viewing forces the encoder to learn robust and comprehensive representations from limited data. The decoder then attempts to reconstruct the full image, including the masked parts, using only the features provided by the encoder.

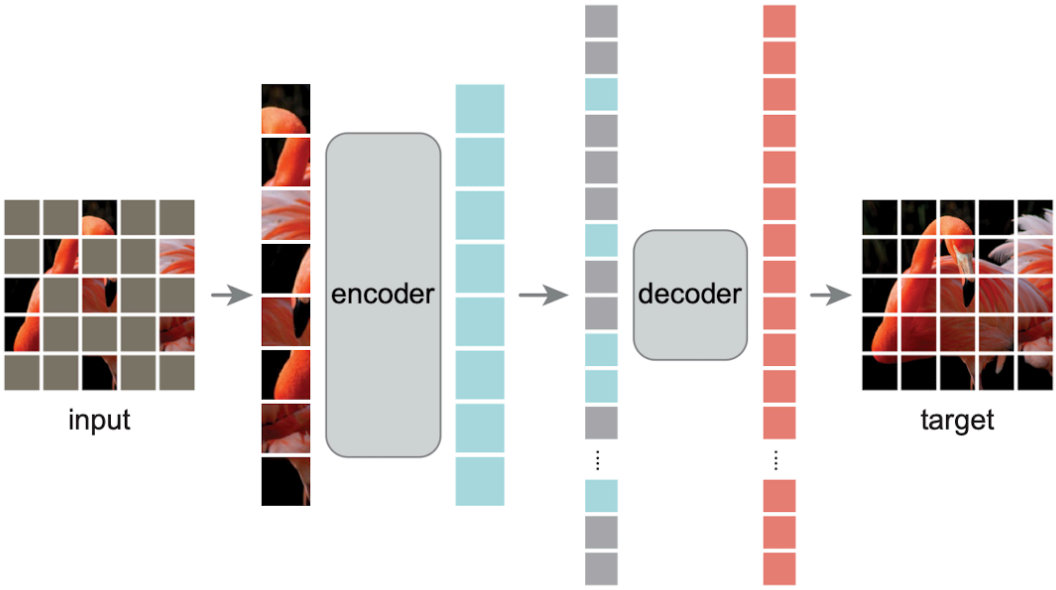


**Figure 2.1.4**: Model diagram from the original paper explaining the encoder decoder setup

In MAE, the image is divided into patches and only the unmasked patches are processed by the encoder, significantly reducing the computational load as compared to processing the entire image.

The strategy of masking a significant portion of the image encourages the model to infer contextual information and understand the relationships between different parts of the visual data. This helps in learning more generalizable and robust features compared to methods that process the entire image directly.

**Loss calculation:** The primary training objective is the *reconstruction loss*, calculated between the original full image and the reconstructed image output by the decoder. This loss is a pixel-wise mean squared error (MSE) or L1 loss, emphasizing accurate reconstruction of the masked regions. The focus on reconstructing only the masked parts,

rather than the entire image, enhances the efficiency and effectiveness of learning.

**Relevance to the study**: MAE was chosen for its ability to understand contextual relationships in datasets

### 2.1.5 SwaV (Swapping Assignments between Views)

SwaV is a self-supervised learning framework that introduces a novel approach to learning visual representations by swapping cluster assignments between multiple views of the same image. This method leverages contrastive learning principles but avoids directly comparing feature vectors, focusing instead on predicting cluster assignments across different augmentations.

**Architecture**: SwaV uses a ResNet50 to extract features from images. The key components are the base encoder that generates feature maps from input images, and a subsequent multi-layer perceptron (MLP) head that outputs representations used for clustering.

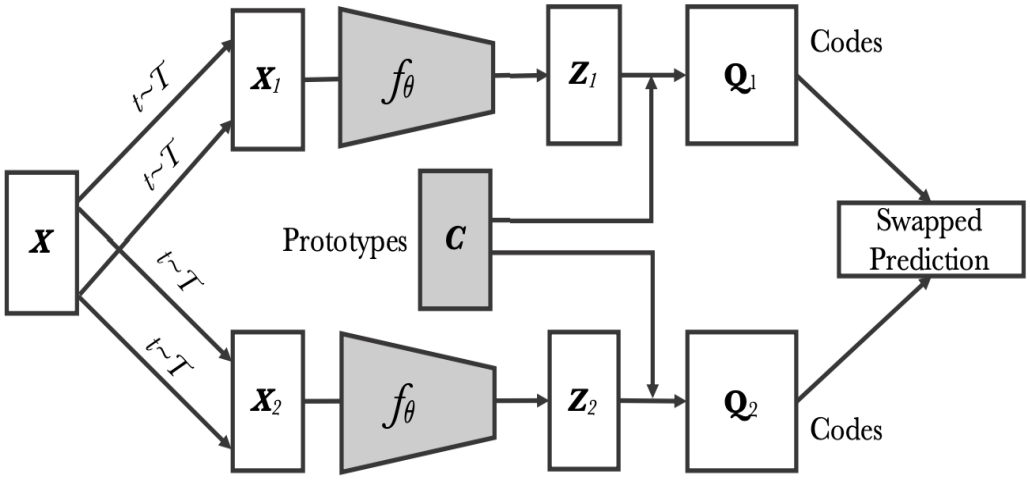


**Figure 2.1.5**: SwaV architecture as described in the original paper

**Training Mechanism:** SwaV operates by generating multiple augmented views of the same image. Each view is processed through the same encoder and MLP head to produce representations, which are

then used to predict the cluster assignments. The unique aspect of SwaV is that these cluster assignments are treated as "prototypes", and each view's feature vector is mapped to these prototypes using a differentiable clustering algorithm.

The central idea of SwaV is to predict the cluster assignment of one view based on the cluster assignment of another view. This is achieved through a "swapping" mechanism, where the model is trained to predict the cluster of one augmented view from another. This method enforces consistency between different views of the same image in terms of their cluster assignments, which indirectly encourages the model to learn invariant and robust features.

SwaV uses a set of learnable prototypes that represent different clusters. The assignment of feature vectors to these prototypes is optimized using the Sinkhorn-Knopp algorithm, which enforces a balanced partition of the feature space. This algorithm ensures that each prototype is used roughly equally, preventing trivial solutions where a few prototypes dominate the clustering process.

**Loss Calculation**: The loss is calculated after comparing the soft assignments between different views (swapped predictions) using a loss function that is conceptually similar to a cross-entropy loss. The model then optimizes the similarity of the assignments across different view to ensure that it learns representations invariant to the applied augmentations.

**LARS optimizer [12]:** A learning rate that is scaled linearly with batch size is chosen, and a cosine warm-up scheduler is employed to adjust the learning rate across training epochs.

# 2.2 Training on ImageNet1k

### 2.2.1 Obtaining Initial Checkpoints

For the foundational phase of our study, we trained a variety of prominent self-supervised learning models on the ImageNet1k dataset. The models selected were SimCLR, MoCoV2, DINO, MAE, and SwaV, each chosen for their unique mechanisms in learning representations from unlabeled data. The

process of training these models to obtain initial checkpoints, crucial for subsequent transfer learning tasks, is visually represented in the figure below.

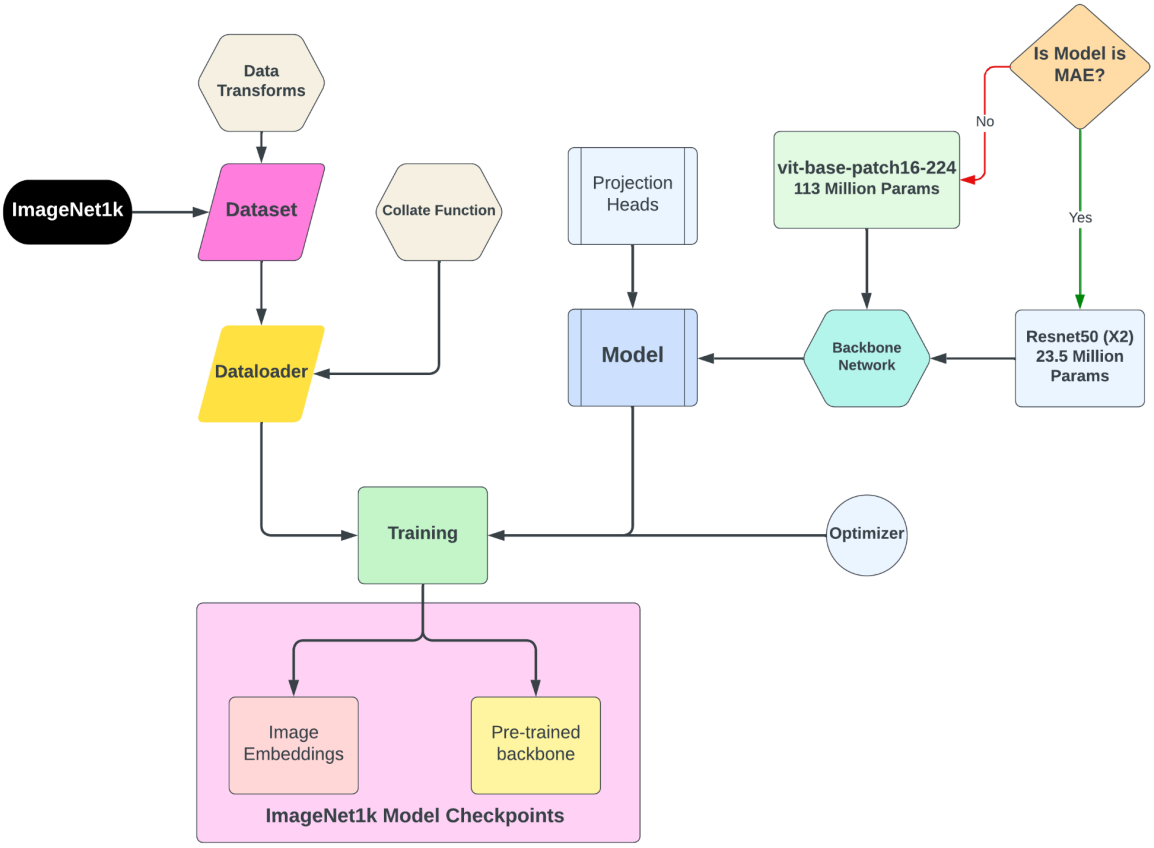


**Figure 2.2.1**: Training workflow of Obtaining Initial Checkpoints for SSL Models on ImageNet1k. The process has been adapted from the Lightly SSL repository. Note: This figure is adapted from figure 3.2.2

**Input Data Preparation:** The models commence with high-quality images from the ImageNet1k dataset. These images are processed using libraries like LightlyDataset, Torchvision, which facilitate various image manipulations to enhance model robustness and simulate diverse visual scenarios.

**Dataset Handling and Transformations:** The prepared images are curated into a dataset, where they are subjected to numerous transformations. These transformations are vital for augmenting the dataset and ensuring the model's ability to generalize across varied visual inputs.

**Data Loading:** Utilizing a dataloader, the transformed images are batched and readied for the training process. The collate function within the dataloader ensures that each batch is uniformly structured, catering to the needs of neural networks.

**Model Architecture:** The core of our training process involves a backbone network—selected from well-known architectures like ResNet50 or Vit-base-patch16-224—paired with specific heads tailored to each SSL technique. This modular approach allows for focused feature extraction and learning.

**Loss Computation**: A loss function specifically chosen for each SSL approach evaluates the discrepancy between the model predictions and the expected outcomes, guiding the learning process.

**Checkpoint testing accuracy:**

| Models\Datasets | ImageNet1k (top1) | ImageNet1k (top5) |
|---|---|---|
| Mocov2 | 61.5 | 84.1 |
| DINO | 68.2 | 87.9 |
| MAE fine_tuning | 81.3 | 95.5 |
| MAE linear_Prob | 46.0 | 70.2 |
| SimCLR | 63.2 | 85.2 |
| SwaV | 67.2 | 88.1 |

**Outputs:**

- I**mage Embeddings:** Post-training, the models generate embeddings for each image, encapsulating the distilled essence of the visual features into a

compact form suitable for downstream tasks.

- **Pre-trained Backbone**: The backbone network, having been refined over the course of training, is preserved as a checkpoint. This pre-trained model serves as a versatile feature extractor for subsequent applications on different datasets.

### 2.2.2. Source of Model Training Code - Lightly SSL Repository

To facilitate the training of our self-supervised learning models on ImageNet1k and to ensure the reproducibility and reliability of our initial checkpoints, we utilized pre-existing code from the Lightly SSL repository. Lightly is an open-source library designed to help with the training of self-supervised learning models, providing robust and tested implementations of various SSL techniques.

The training scripts and necessary configurations for models such as SimCLR, MoCoV2, DINO, MAE, and SwaV were directly adopted from the Lightly SSL GitHub repository. This ensured that our training procedures were aligned with state-of-the-art practices in the SSL community.

**Code Adaptation**: While the core architecture and training logic were kept consistent with the repository's implementations, slight modifications were made to adapt the hyperparameters and training schedules to our specific hardware setup and research goals.

**Importance of Using Lightly SSL:**

- **Reproducibility:** By using an established codebase, we ensure that our research can be replicated and verified by other researchers in the field, which is essential for the academic validation of our findings.
- **Reliability:** Lightly SSL is a well-maintained and widely-used framework in the machine learning community, which provides confidence in the stability and performance of the training process.
- **Efficiency:** Leveraging pre-built, optimized code allows us to focus on the experimental aspects of our research without needing to reinvent foundational training mechanisms.

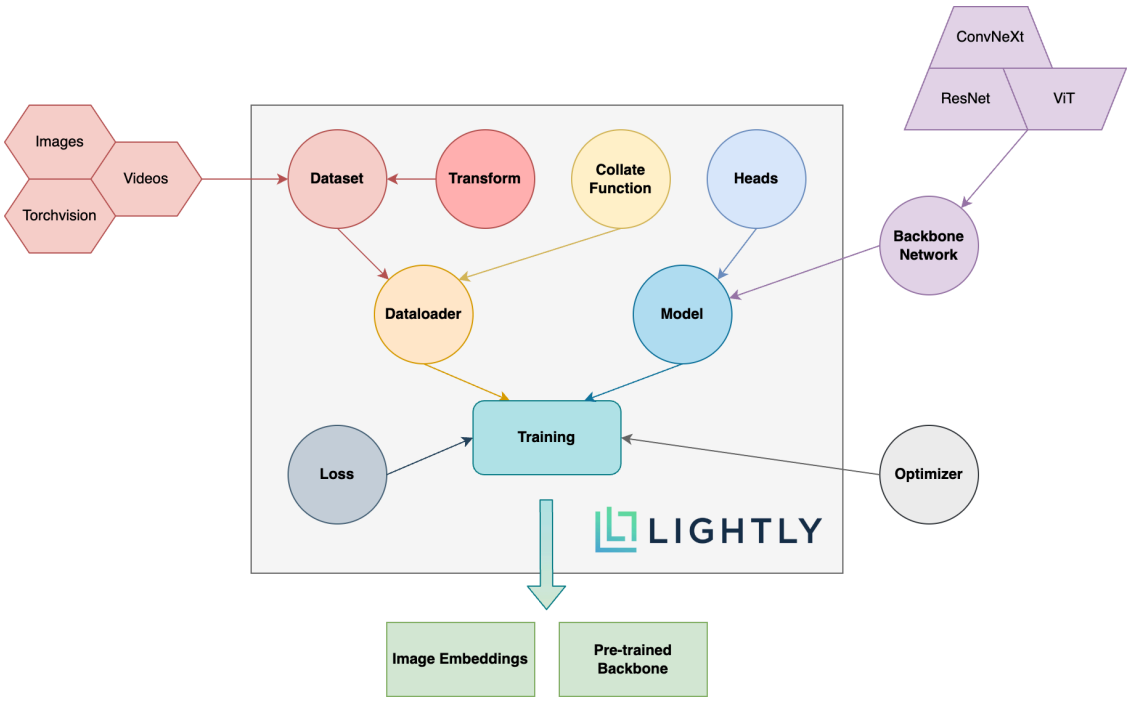


**Figure 2.2.2**: An illustration of how each model was trained on imageNet1k

## 2.3 Data preparation

For the evaluation of linear probing in self-supervised learning models under different domain settings, three distinct datasets were utilized: STL10, CIFAR-10, and Pneumonia. Each dataset presents unique characteristics and challenges, making them suitable for assessing the adaptability and effectiveness of learned representations across varying domains.

### 2.3.1. STL10

The STL10 dataset is designed for the task of unsupervised image classification. It derives inspiration from the CIFAR-10 dataset but with some modifications to better suit unsupervised learning scenarios.

**Content:** STL10 consists of 5000 training images and 8000 test images, divided into 10 classes. Each image is of higher resolution (96x96 pixels) compared to CIFAR-10, providing more detailed visual content for model training and evaluation. It contains 10 classes: airplane, bird, car, cat,

deer, dog, horse, monkey, ship, and truck.

**Usage:** For the purpose of linear probing STL10 was resized to tensors of 224 x 224 resolution to correspond to the input dimension of the models used in this study. The dataset was also balanced and normalized so that the test-train-val split would be 70-15-15 giving us 9090 training images, 1960 testing images and 1950 validation images.

### 2.3.2. CIFAR-10

CIFAR-10 is a well-known dataset used in various machine learning and computer vision research areas. It is particularly favored for evaluating image recognition algorithms.

**Content:** The dataset comprises 60,000 32x32 color images in 10 different classes, with 6,000 images per class. The dataset is split into 50,000 training images and 10,000 test images. The classes are: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck.

**Usage:** CIFAR10 was resized to tensors of 224 x 224 resolution to correspond to the input dimension of the models used in this study. The dataset was also balanced and normalized so that the test-train-val split would be 70-15-15 giving us 40000 training images, 10000 testing images and 10000 validation images.

### 2.3.3. Pneumonia Dataset (Chest X-Ray Images)

The Pneumonia dataset consists of chest X-ray images, which are used for the binary classification task of distinguishing normal conditions from pneumonia.

**Content:** The dataset is organized into 3 folders (train, test, val) and contains subfolders for each image category (Pneumonia/Normal). There are 5,863 X-Ray images (JPEG) and 2 categories (Pneumonia/Normal).

**Usage:** Pneumonia dataset was resized to tensors of 224 x 224 resolution to correspond to the input dimension of the models used in this study. The dataset was also balanced so that the test-train-val split would be 70-15-15 giving us 4099 training images, 880 testing images and 877 validation images.

## 2.4 Linear Probing Setup

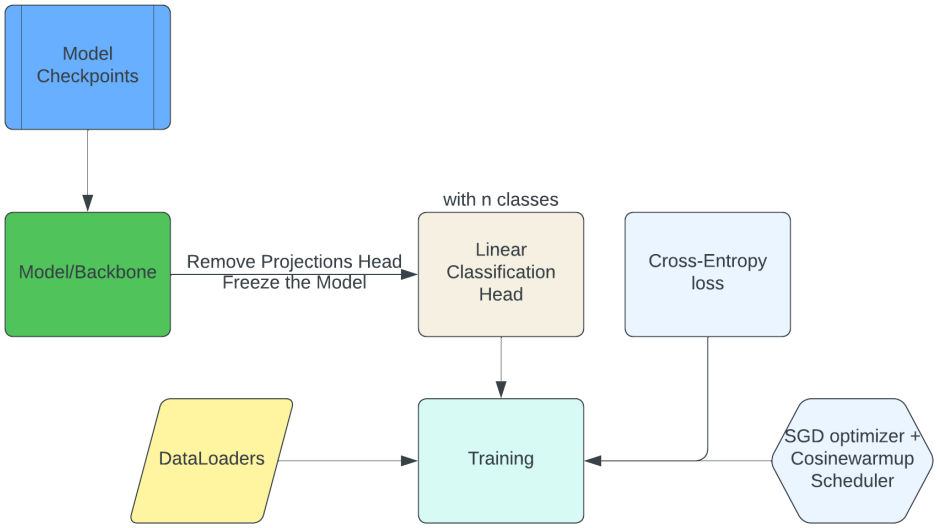


**Figure 2.4** shows the workflow to generate the linear probe for downstream classification task

The backbone or feature extraction part of the pre-trained models obtained from 3.2 are used as the starting point. These models have been trained on ImageNet1k using a self-supervised learning approach. These “backbones” provide the learned feature representations that will be used to evaluate transferability to different tasks.

**Modifications for Probing:** The projection head used during self-supervised training is removed, and the entire model except for the backbone is frozen. This ensures that the linear probing evaluates the quality of the fixed features without further tuning.

A simple linear layer added on top of the frozen backbone which is configured to have as many outputs as there are classes in the probing

task (denoted as 'n classes' in the diagram).

**Configuration for training:**
Only the weights of the linear classification head are trainable. The backbone remains frozen, ensuring that the training focuses on the ability of the fixed features to adapt to new tasks. We train each classifier head for 100 epochs for cifar10 and stl10, and 200 epochs for the pneumonia dataset.

**Loss**: Cross-entropy loss

**Optimizer**: SGD Optimizer + Cosine Warmup Scheduler:

# Chapter 3

# Experimental Results and Analysis

In this section, we detail our comparative analysis and out-of-domain assessment results.

## 3.1 Model Specific Performance

**Performance metrics:**

- **Training Accuracy**: This metric measures the percentage of correct predictions made by the model on the dataset it was trained on.
- **Training Loss**: This represents the model's error rate on the training dataset. A lower training loss generally indicates better learning but needs to be balanced against other metrics to ensure the model is not just memorizing the training data.
- **Validation Accuracy**: This metric measures how well the model performs on a subset of the data that is not used for training (but comes from the same distribution). It helps in tuning the model's hyperparameters without touching the test set. High validation accuracy suggests that the model generalizes well on unseen data from the same distribution as the training set.
- **Testing Accuracy**: The most critical metric, testing accuracy, assesses the model's performance on completely unseen data that ideally comes from a different distribution or domain.

### 3.1.1: MocoV2:

| Dataset | Training Accuracy | Training Loss | Validation Accuracy | Testing Accuracy |
|---|---|---|---|---|
| Pneumonia | 94.90 | 0.1149 | 95.94 | 94.99 |
| Stl10 | 80.42 | 0.3873 | 79.79 | 79.13 |

| | | | | |
|---|---|---|---|---|
| Cifar10 | 81.15 | 0.556 | 77.55 | 77.32 |

### 3.1.2 SimCLR:

| Dataset | Training Accuracy | Training Loss | Validation Accuracy | Testing Accuracy |
|---|---|---|---|---|
| Pneumonia | 96.80 | 0.0725 | 97.23 | 97.39 |
| Stl10 | 79.13 | 0.3383 | 80.12 | 81.25 |
| Cifar10 | 83.27 | 0.5042 | 80.19 | 80.34 |

### 3.1.3 DINO:

| Dataset | Training Accuracy | Training Loss | Validation Accuracy | Testing Accuracy |
|---|---|---|---|---|
| Pneumonia | 97.87 | 0.0881 | 96.19 | 96.7 |
| Stl10 | 90.72 | 0.0603 | 88.67 | 88.06 |
| Cifar10 | 86.44 | 0.4485 | 81.92 | 81.71 |

### 3.1.4 MAE:

Linear Probing @ 50 epochs:

| Dataset | Training Accuracy | Training Loss | Validation Accuracy | Testing Accuracy |
|---|---|---|---|---|
| Pneumonia | 95.14 | 0.1249 | 97.14 | 95.00 |
| Stl10 | 91.89 | 0.313 | 84.92 | 85.71 |
| Cifar10 | 83.72 | 0.5134 | 81.75 | 82.98 |

Fine tuning @ 25 epochs:

| Dataset | Training Accuracy | Training Loss | Validation Accuracy | Testing Accuracy |
|---|---|---|---|---|
| Pneumonia | 97.26 | 0.0941 | 95.48 | 95.56 |
| Stl10 | 88.67 | 0.2049 | 81.64 | 83.06 |
| Cifar10 | 82.35 | 0.675 | 80.23 | 78.02 |

### 3.1.5 SwaV:

| Dataset | Training Accuracy | Training Loss | Validation Accuracy | Testing Accuracy |
|---|---|---|---|---|
| Pneumonia | 96.87 | 0.0774 | 95.84 | 95.91 |
| Stl10 | 88.53 | 0.1576 | 87.62 | 86.58 |
| Cifar10 | 81.18 | 0.753 | 75.01 | 75.08 |

## 3.2 Comparative Analysis

This section provides a detailed comparative analysis of the performance of five self-supervised learning (SSL) models: MoCoV2, SimCLR, DINO, Masked Autoencoder (MAE), and SwaV. The models were evaluated based on their training, validation, and testing accuracy and loss across three different datasets: Pneumonia, STL10, and CIFAR-10. This analysis aims to highlight the strengths and weaknesses of each model in various settings and provide insights into their generalization capabilities.

### 3.2.1. Performance on Pneumonia Dataset

- DINO exhibited the highest training accuracy (97.87%) and impressive testing accuracy (96.70%), suggesting robust feature extraction capabilities in medical images.
- SimCLR followed closely in testing accuracy (97.39%), indicating its

effectiveness in handling complex image features typical in medical datasets.
- MAE (Fine Tuning) and SwaV showed competitive performances, with testing accuracies above 95%, demonstrating their utility in healthcare applications where labeled data are scarce.
- MoCoV2, while slightly lower, still performed admirably with a testing accuracy of 95.00%.

### 3.2.2. Performance on STL10 Dataset

- DINO stood out with the highest testing accuracy (88.06%) and the lowest training loss (0.0603), showcasing its efficiency in learning from complex visual representations.
- MAE (Linear Probe) also showed strong performance, achieving a testing accuracy of 85.71%, highlighting the effectiveness of linear probing in extracting useful features.
- SwaV and SimCLR provided moderate results with testing accuracies of 86.58% and 81.25%, respectively.
- MoCoV2 had the lowest performance among the models, with a testing accuracy of 79.13%, possibly due to less effective feature generalization from ImageNet training to STL10.

### 3.2.3. Performance on CIFAR-10 Dataset

- DINO again led in performance with the highest testing accuracy (81.71%), reinforcing its strong adaptability and robust feature learning across diverse datasets.
- MAE (Linear Probe) and SimCLR demonstrated good generalization with testing accuracies of 82.98% and 80.34%, respectively.
- MoCoV2 and SwaV showed some challenges in adapting to CIFAR-10, with testing accuracies below 78%, indicating potential areas for improvement in handling diverse object categories.

### 3.2.4 Overall Insights

- DINO consistently exhibited top performance across all datasets, demonstrating its robust feature extraction capabilities. The effectiveness of DINO in a single GPU setting suggests that its self-distillation technique, despite being potentially computationally intensive, can be managed effectively even with limited hardware resources. This makes DINO an attractive choice for applications requiring high accuracy without extensive parallel computing infrastructure.

- SimCLR also performed impressively, particularly with medical images and high-resolution datasets. The model leverages a simple yet effective contrastive learning mechanism that scales well on a single GPU. Its efficiency and strong performance indicate that SimCLR is a practical option for researchers or practitioners with constrained computational resources who do not wish to compromise on model performance.

- MAE showed notable differences in performance between its linear probing and fine-tuning strategies, with fine-tuning generally yielding better results. This variation underscores the potential computational trade-offs between quicker, less resource-intensive linear probing and more computationally demanding fine-tuning. For single GPU setups, linear probing with MAE can be a viable strategy to achieve decent performance without the overhead of full model fine-tuning.

- SwaV and MoCoV2 demonstrated moderate success but were somewhat inconsistent across the tested datasets. Given their underlying mechanisms that may benefit from larger batch sizes or longer queue lengths—features that are typically constrained in single GPU environments—their full potential might not be completely realized under these conditions. For users with single GPU setups, tuning SwaV and MoCoV2 may require more careful consideration of batch size and memory management to optimize performance.

### 3.2.5 Limitations of the study

While this research provides valuable insights into the performance of SSL models under various conditions, there are several limitations that need to be acknowledged:

1. Computational Resources: The study was conducted under constraints typical of a single GPU setup. While this reflects real-world scenarios for many developers and researchers, it also limits the scalability and sometimes the full potential of certain models, especially those that might achieve better performance with more extensive computational resources.

2. Model Configurations: Due to the diverse nature of SSL models, uniform configuration across all models was challenging. This might have introduced some bias in favor of models that perform better under the specific configurations chosen for this study.

3. Dataset Limitations: Although the datasets used (STL10, CIFAR-10, and Pneumonia) cover a range of complexities and domains, they do not encompass all possible types of data or tasks. The findings might not be directly transferable to datasets with significantly different characteristics, such as those involving more complex or less structured data types.

4. Dependence on Pre-trained Models: The study heavily relies on models pre-trained on ImageNet1k, which may limit insights into the models' performance when trained from scratch or on significantly different foundational datasets.

5. Evaluation Methodology: The sole use of linear probing as an evaluation metric, while valuable, does not capture the full spectrum of a model's capabilities. Other methods, like fine-tuning or using different layers of the model for classification, could provide additional insights

# Chapter 4

# Conclusion

This thesis has undertaken a comprehensive examination of the performance and adaptability of various self-supervised learning (SSL) models across a spectrum of visual datasets, including Pneumonia, STL10, and CIFAR-10, with a particular focus on settings constrained by single GPU training capabilities. The insights derived from this study not only enhance our understanding of the capabilities and limitations of these models but also provide valuable guidance for their practical application in environments where computational resources are limited.

The analysis revealed that DINO consistently excelled across all evaluated datasets, showcasing its ability to effectively distill and transfer knowledge even within the confines of a single GPU setup. This demonstrates DINO's suitability for scenarios requiring high accuracy without extensive computational resources. SimCLR also emerged as a strong contender, especially in handling complex and high-resolution images, proving that effective feature extraction does not necessarily require elaborate or resource-heavy architectures.

Conversely, models like SwaV and MoCoV2, although moderately successful, displayed some inconsistencies that could be attributed to their dependency on larger batch sizes or more complex memory management strategies, which are often less feasible in single GPU environments. The MAE model highlighted the potential trade-offs between the computational demands of fine-tuning versus the more feasible linear probing approach, which offers a viable alternative for achieving respectable performance without substantial computational overhead.

This study emphasizes the importance of carefully selecting and tuning SSL models to align with both their inherent architectural strengths and the specific computational constraints of the training environment. For researchers and practitioners with limited resources, models like DINO and SimCLR provide promising options for achieving robust feature learning without the necessity for multi-GPU setups, thereby making advanced machine learning tools more accessible and effective in a wider range of applications.

# Acknowledgment

I would like to begin by acknowledging that the land on which we gather is the unceded territory of the Syilx (Okanagan) Peoples.

I would like to express my sincere gratitude to Prof. Mohamed Shami Shehata for his invaluable mentorship and for welcoming me into his telepresence research group. I am truly grateful for the numerous research opportunities he provided me during my undergraduate studies, which have greatly enriched my academic journey.

Additionally, I extend my thanks to the University of British Columbia for providing the necessary resources and facilities that supported this research endeavor.

# Bibliography

[1] He, Kaiming & Chen, Xinlei & Xie, Saining & Li, Yanghao & Dollar, Piotr & Girshick, Ross. (2022). Masked Autoencoders Are Scalable Vision Learners. 15979-15988. 10.1109/CVPR52688.2022.01553.

[2] Chen, Ting, et al. "A Simple Framework for Contrastive Learning of Visual Representations." arXiv preprint arXiv:2002.05709 (2020).

[3] Chen, Xinlei, and Kaiming Fan. "Improved Baselines with Momentum Contrastive Learning." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[4] Caron, Mathilde, et al. "Emerging Properties in Self-Supervised Vision Transformers." arXiv preprint arXiv:2104.14294 (2021).

[5] Caron, Mathilde, et al. "Unsupervised Learning of Visual Features by Contrasting Cluster Assignments." arXiv preprint arXiv:2006.09882 (2020).

[6] Z. Liu, H. Mao, C. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), page 11966-11976.* (*2022*)
DOI: 10.1109/CVPR52688.2022.01167

[7] LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE*, vol. 86, no. 11, 1998.

[8] A. R. Lahitani, A. E. Permanasari and N. A. Setiawan, "Cosine similarity to determine similarity measure: Study case in online essay assessment," 2016 4th International Conference on Cyber and IT Service Management,

Bandung, Indonesia, 2016, pp. 1-6, doi: 10.1109/CITSM.2016.7577578. keywords: {Data mining;Feature extraction;Information retrieval;Media;Algorithm design and analysis;Frequency measurement;Context;Automated Essay Scoring (AES);TF-IDF;Cosine Similarity},

[9] Sohn, Kihyuk. "Improved Deep Metric Learning with Multi-class N-pair Loss Objective." Advances in Neural Information Processing Systems, vol. 29, 2016.

[10] Wu, Zhirong, et al. "Unsupervised Feature Learning via Non-parametric Instance Discrimination." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018.

[11] Misra, Ishan, and Laurens van der Maaten. "Self-Supervised Learning of Pretext-Invariant Representations." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2020

[12] You, Yang, et al. "Large Batch Training of Convolutional Networks." arXiv preprint arXiv:1708.03888. 2017.

[13] "Self-Supervised Learning." IBM, IBM Corporation, Accessed 24 April 2024, https://www.ibm.com/topics/self-supervised-learning.

[14] He, Kaiming, et al. "Deep Residual Learning for Image Recognition." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[15] Chaudhary, Amit. 2020. “The Illustrated SimCLR Framework.” March 4, 2020. https://amitness.com/posts/simclr.html.