Final Report

Self-Supervised Learning for Medical Imaging Classification

Within the context of the course:

CSC_4IM06_TP
(Generative methods, Patch based methods and computational photography)

Authors:
Abdennour Kerboua,
Judith Amigo,
Daniel Akbarinia

September 24, 2025

Contents

1 Introduction

1.1 Objectives

In this work, we aim to evaluate the effectiveness of self-supervised learning methods for medical image classification. Specifically, we investigate how pretraining with unlabeled data can improve downstream performance in settings with limited annotations. Our study focuses on two recent and very related approaches, SimCLR [1] and Barlow Twins [4], used to learn image representations without labels. We examine the influence of key factors such as batch size, training duration, output dimensionality, and data augmentation strategy. Beyond classification accuracy, we also analyze the structure of the learned feature space using t-SNE to better understand the quality and separability of the resulting latent representations.

1.2 Data Used

We conduct our experiments on two representative datasets from the MedMNIST benchmark [3] : **PathMNIST** and **BloodMNIST**. These datasets were selected for being the largest dataset of images in color of this benchmark, which aligns with standard convolutional architectures (especially ResNet18). PathMNIST consists of histological image patches derived from colorectal cancer tissue slides, while BloodMNIST contains microscopy images of various blood cell types. Both offer clinically meaningful visual diversity and provide a robust testbed for evaluating representation learning approaches in the medical domain.

2 PathMNIST & BloodMNIST

Among the datasets in the MedMNIST collection, **PathMNIST** and **BloodMNIST** are the two largest color image datasets, making them ideal candidates for evaluating our self-supervised learning (SSL) methods. These two datasets were deliberately selected both for their size and their color format, which is essential in our setting: our SSL framework relies heavily on color-based transformations for view generation, and its performance is significantly hindered in the absence of such information. Despite being the second largest dataset of color images, BloodMNIST still has a significantly lower size than PathMNIST, and this will allow us to assess the importance of the availability of numerous unlabeled samples to the usefulness of the SSL techniques.

All images in PathMNIST are fully annotated, with each sample corresponding to a tissue patch labeled according to one of **9 distinct tissue types** (e.g., tumor, stroma, lymphocytes, etc.). The dataset contains a total of **107,180 color images**, and its relatively large size enables SSL methods to learn rich and robust representations. Furthermore, the availability of multiple image resolutions (64×64, 128×128, 224×224) enables a practical trade-off between training quality and computational cost, which was a key consideration throughout this project.

PathMNIST is derived from a histological dataset of colorectal cancer. The training and validation sets come from the *Radboud University Medical Center (Radboud UMC)*, while the test set originates from a different clinical center: the *Klinikum Rechts der Isar, Technical University of Munich*. This split introduces a true *domain shift*, which has a notable impact on generalization performance. In practice, we observed that selecting the best model based on validation set accuracy can lead to overly optimistic results, with a performance gap of up to **5 to 7 percentage points** when compared to test set performance. As a result, it is critical to use the test set as the reference for evaluating generalization.

BloodMNIST, while being the second largest dataset in the MedMNIST collection, is still relatively limited in scale with only around **17,092 annotated images**. It consists of blood smear images where each sample is assigned to one of **8 white blood cell classes** (e.g., neutrophil, eosinophil, monocyte). Although BloodMNIST does not introduce a clinical domain shift like PathMNIST, its smaller size makes it well suited for highlighting the limitations of our SSL approach when trained with fewer examples. The reduced volume means that learned representations are more sensitive to underfitting and less generalizable. This makes BloodMNIST a valuable complementary dataset for evaluating how well our methods scale with data availability and for exposing scenarios where performance begins to degrade due to limited supervision.

3 SimCLR

3.1 Model presentation

3.1.1 Overall Architecture

SimCLR (**S**imple **C**ontrastive **L**earning of visual **R**epresentations) comprises three main components:

1. *Data augmentation module*, which generates two correlated views $\mathbf{x}_i$, $\mathbf{x}_j$ of each image $\mathbf{x}$ via various random transformation:

   - Cropping
   - Color jitter
   - Gaussian blur
   - Grayscale conversion
   - Horizontal flip

   We employ the transformations proposed by the authors of the SimCLR method *"A Simple Framework for Contrastive Learning of Visual Representations"* Chen et al. [1]. In this report, we will evaluate the importance of their combination and intensity.

2. *Base encoder* $f(\cdot)$, typically a RestNet-18 (or 50) backbone, that maps each view to a feature vector

   $$\mathbf{h} = f(\mathbf{x}) \in \mathbb{R}^d,$$

   where typically $d = 512$. We opted here for a ResNet-18 given the size of our training set. The experimations presented in the article *"A Simple Framework for Contrastive Learning of Visual Representations"* Chen et al. [1] were carried out on ImageNet dataset ($\sim$ 1M samples) with a ResNet-50. Then, it was reasonable to suppose that a ResNet-18 will suffice with a dataset of around 90k images.

3. *Projection head* $g(\cdot)$, a small multilayer perceptron (usually two layers with ReLU), that maps $\mathbf{h}$ to a projection

   $$\mathbf{z} = g(\mathbf{h}) \in \mathbb{R}^p,$$

   where commonly $p = 128$. In our implementation, it will be constituted adding a `nn.ReLU()` activation and a `nn.Linear(512,128))` layer after the fully connected layer of the `torchvision` model of ResNet-18. The contrastive loss is applied on the projections $\mathbf{z}$. At inference only $f(\cdot)$ is retained.

3.1.2 Contrastive Loss (NT-Xent)

Given a minibatch of N images, SimCLR constructs $2N$ augmented samples : a positive pair of 2 images $(\mathbf{x}_{2k}, \mathbf{x}_{2k-1})$ for x_k an image of the batch. For each positive pair $(\mathbf{x}_{2k}, \mathbf{x}_{2k-1})$ (two views of the same image), the *normalized temperature-scaled cross-entropy* loss is:

$$\ell_{2k,2k-1} = -\log \frac{\exp\big(\text{sim}(\mathbf{z}_{2k}, \mathbf{z}_{2k-1})/\tau\big)}{\sum_{i=1}^{2N} \mathbf{1}_{[2k\neq i]} \exp\big(\text{sim}(\mathbf{z}_{2k}, \mathbf{z}_i)/\tau\big)},$$

where

$$\text{sim}(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u}^\top \mathbf{v}}{\|\mathbf{u}\| \, \|\mathbf{v}\|}, \quad \tau > 0 \text{ (temperature).}$$

The total loss to minimize is averaged over all N positive pairs:

$$\mathscr{L} = \frac{1}{2N} \sum_{k=1}^{N} \big[\ell_{2k-1,2k} + \ell_{2k,2k-1}\big].$$

3.1.3 Training Procedure

The training is organized as follows :

- At each iteration, sample a batch of N images, generate two augmentations per image, and compute $\{\mathbf{z}_i\}_{i=1}^{2N}$.
- Compute the NT-Xent loss and backpropagate through both g and f. For that, we compute a similarity matrix after normalizing the features $(\mathbf{z}_i)_{1\leq i\leq 2N} \to \mathbf{S} = [\tilde{\mathbf{z}}_i^\top \tilde{\mathbf{z}}_j]_{1\leq i,j\leq 2N}$. Then for each images $\mathbf{x}_i$ we

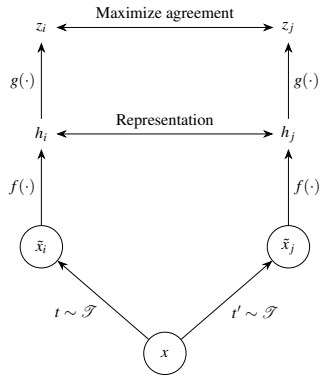


Figure 1: Schematic of SimCLR framework. Source : *"A Simple Framework for Contrastive Learning of Visual Representations"* Chen et al. [1]

compute a $2N-1$ logit vector containing in the first place the similarity between $\mathbf{x}_i$ and his positive correspondent $\mathbf{x}_j$ and in the rest of the vector the similarity with the other images of the batch :

$$\textbf{logits}(2k) = [\mathbf{S}(2k, 2k-1), \mathbf{S}(2k, i)]_{i \neq 2k}$$

We then make it a multi-class classification task where the ground-truth label is always **0** the position of the positive correspondent in the **logits** vector. And compute the **NT-Xent** loss using the Cross Enropy Loss.

- Use the largest minibatch size possible to provide many negative examples, which improves representation quality and preserve independance of the batches.
- After convergence, discard the projection head g and retain f for downstream tasks.

3.1.4 Inference and Downstream Use

At inference time, only the encoder f is used. For a new image x, one computes

$$\mathbf{h} = f(\mathbf{x}) \in \mathbb{R}^d$$

and then either

- Attach a *linear classifier* on top of h and fine-tune on labeled data;
- Use h directly for retrieval or clustering.

3.2 Method evaluation

3.2.1 Model Finetuning

We pretrain the model following the procedure explained in 3.1.3 for 100 epochs on PathMNIST, which takes approximately 7 hours on our reference GPU setup (Nvidia Tesla P100 GPU provided by Kaggle platform).

Then, the first part of our analysis consisted of assessing the performance of the models in medical imaging classification. To this end, we fine-tune the pre-trained encoder by removing its original projection head $g(\cdot)$ and replacing it with a single fully connected layer of size $512 \times \texttt{num\_class}$, preceded by a ReLU activation. Fine-tuning is performed on a randomly selected portion p of the labeled training set, with

$$p \in \{0.01,\ 0.05,\ 0.1,\ 0.2,\ 0.5,\ 1\}.$$

For each p, we compare the fine-tuned model against a fully supervised ResNet-18 trained under identical data proportions.

To stabilize the fine-tuning process, we adopt a two-stage schedule:

1. **Backbone freeze.** Keep all pretrained encoder weights fixed and train only the new MLP head for the first 10 epochs.

2. **Full fine-tuning.** Unfreeze the entire network and continue training for an additional 30 epochs.

We obtain the following results. We focused on Top-1 accuracy in contrary to the article experiment since the number of classes is drastically lower than in ImageNet (100 classes for ImageNet, only 10 for PathMNIST):

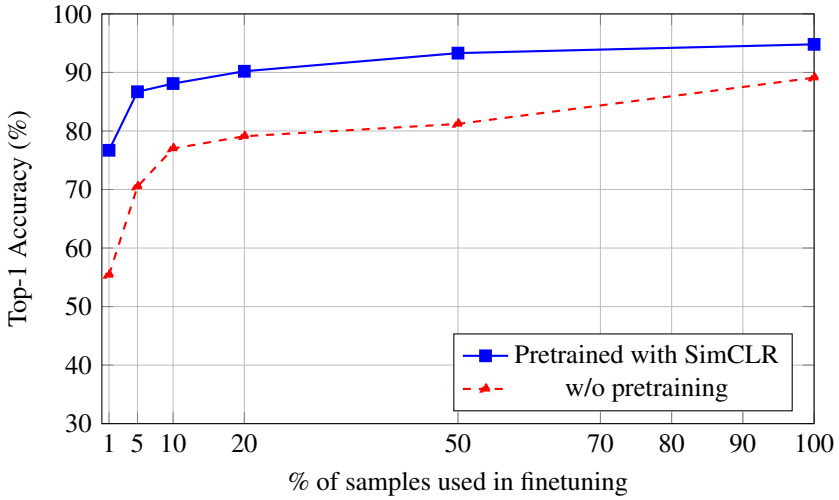


Figure 2: Finetuned Pretrained Model v. Not pretrained model on PathMNIST

The pretrained model consistently outperforms the from-scratch model across all proportions p of labeled data. This consistency confirms that the self-supervised representations learned by SimCLR provide highly transferable features for medical image classification, even when large amounts of annotations are available.

The largest benefit is observed in the regime $p < 0.5$, which is our primary focus for reducing manual annotation effort. For instance, at $p = 0.01$ and $p = 0.05$, the pretrained model exceeds the non-pretrained baseline by 15 to 20 percentage points in Top-1 Accuracy, highlighting the effectiveness of pretraining under low-supervision conditions.

A particularly striking result is that the pretrained model fine-tuned on only 5% of the labeled data outperforms the non-pretrained model fine-tuned on 50% of the data. It indicates a tenfold reduction in required annotations to achieve comparable performance.

Our two-stage fine-tuning protocol produced stable learning curves without abrupt accuracy oscillations. This stability was crucial for making fair comparisons across all values of p.

In practical terms, these findings imply that leveraging SimCLR pretraining in medical imaging tasks can drastically reduce annotation budgets while maintaining or exceeding the performance of a fully supervised ResNet-18.

3.2.2 Exploration of the latent representation

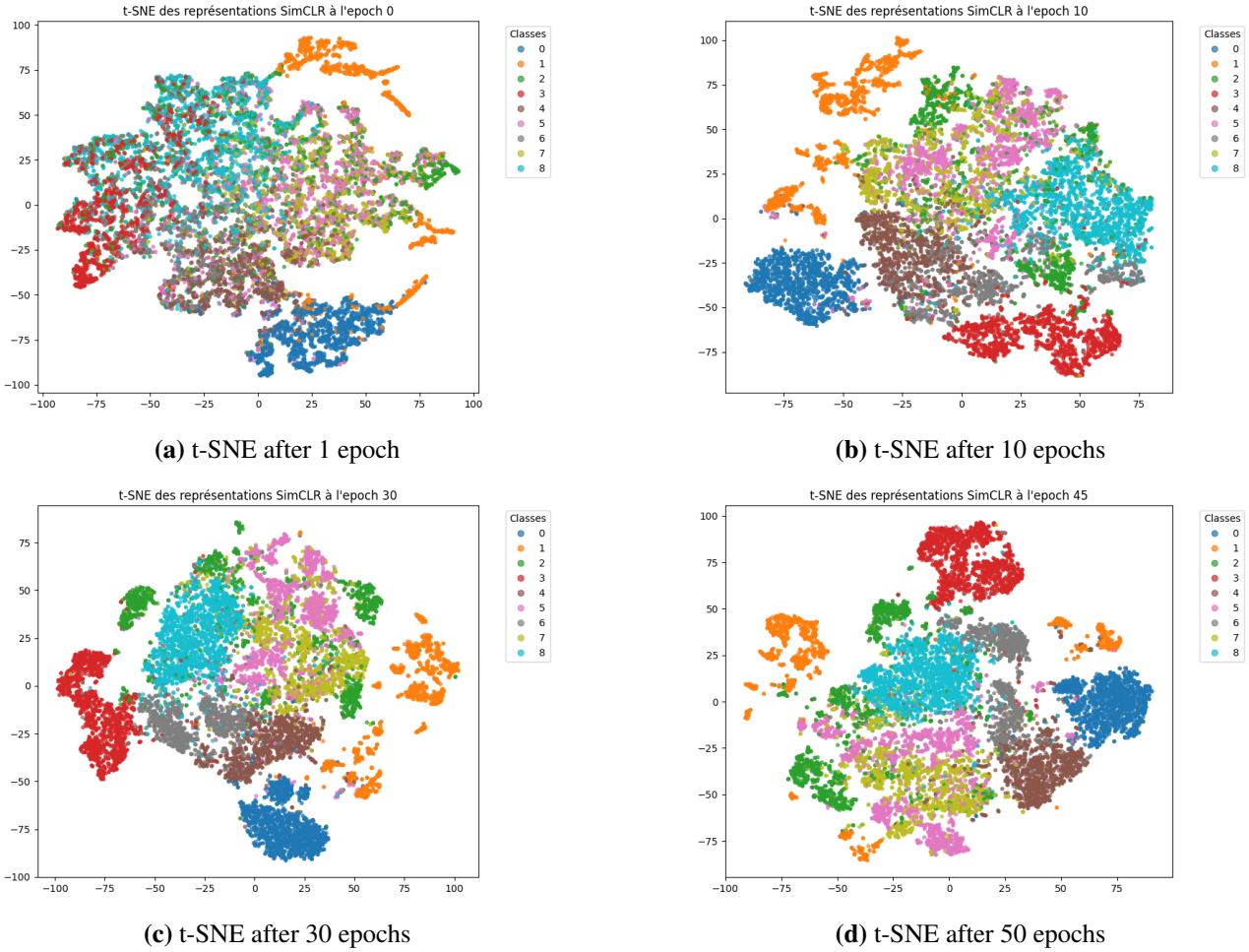
To investigate how pre-training duration affects the structure of learned features, we applied t-SNE to the latent vectors extracted from the encoder after 1, 10, 30, and 50 epochs of contrastive pre-training. As shown in Figure 3, the cluster boundaries become progressively sharper with longer training: after just 1 epoch the classes are heavily superimposed, at 10 epochs some separation appears, and by 30–50 epochs the major cell-type clusters are well defined.

3.3 Influence of certain hyper-parameters

3.3.1 Data augmentations, image transformations

Combination of transformations. To analyze the individual and combined effects of our data augmentations, we constructed an evaluation matrix $A \in \mathbb{R}^{n \times n}$, where each entry on the frozen encoder. Here, i and j index the set of n candidate augmentations (e.g. random cropping, color jittering, Gaussian blur, etc.). The matrix is shown in Figure 4.

From this matrix, it is clear that random cropping is the most critical augmentation for downstream linear



(a) t-SNE after 1 epoch

(b) t-SNE after 10 epochs

(c) t-SNE after 30 epochs

(d) t-SNE after 50 epochs

Figure 3: Evolution of t-SNE embeddings for PathMNIST latent features at different pre-training epochs. Clusters sharpen as training progresses.

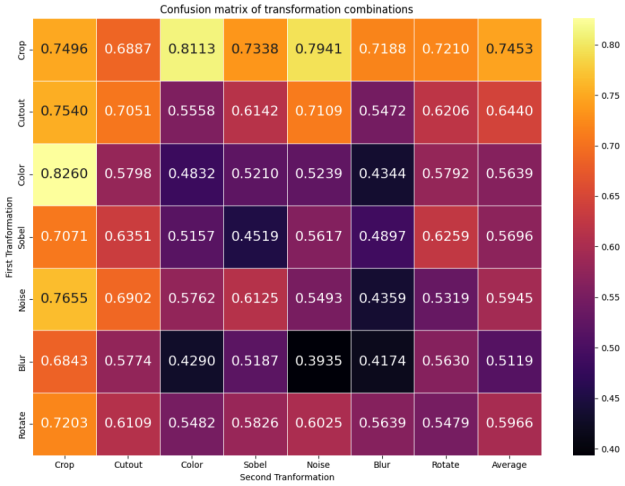


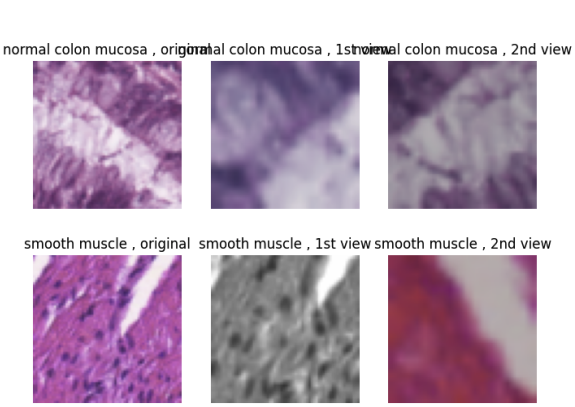
Figure 4: A_{ij} = Top-1 Accuracy of a linear probe when applying transformation i followed by j

separability: whenever it appears in either the row or column index, A_{ij} is substantially high. Color jittering is the second most effective transform, providing additional gains particularly when combined with random cropping. In contrast, other augmentations (such as Gaussian blur or horizontal flipping) yield smaller and less consistent improvements in linear-probing accuracy.

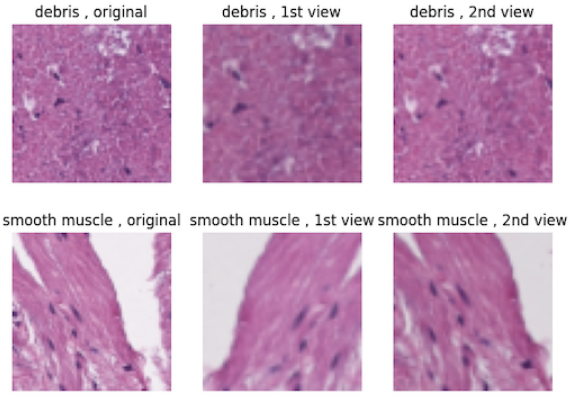
Intensity of the transformations. When we reduce the severity of our two key augmentations (random crop

p (% labels)	Original (max crop 20%, full-strength jitter)	Light (max crop 80%, 0.1× jitter)
1	76.7	52.9
5	86.7	74.7
10	88.2	80.4
20	90.2	89.1
50	93.3	89.6
100	94.8	94.8

Table 1: Comparison of Top-1 Accuracy (%) for original vs. lighter augmentation intensities



(a) Transformation recommended by the article



(b) Light transformations

Figure 5: Different transformation settings.

retaining at least 80 % of image size, and color-jitter coefficients divided by 10), the fine-tuning performance of the pretrained model drops markedly. This suggests that the contrastive pretraining failed to learn sufficiently general features under weaker augmentations, leading to poorer downstream classification. Table 1 summarizes the Top-1 Accuracy of the pretrained model across different label proportions p for the two transformation settings. Examples of different image transformations are presented in Figure 5.

3.3.2 Influence of training set size

To further probe the robustness of the SSL models, we repeated the fine-tuning experiments on BloodMNIST, which contains significantly fewer samples than PathMNIST. Although the pretrained model still outperforms the from-scratch baseline at $p = 0.01$, its relative advantage diminishes far more rapidly as p increases. This suggests that on smaller datasets, contrastive pretraining may capture less generalizable features, and the performance gap closes quickly once even a modest amount of labeled data is available. We show the results in Figure 6.

3.3.3 Influence of training duration

As reported in the original SimCLR study [1], longer self-supervised pre-training consistently improves the quality of learned representations. We observe a similar trend on our dataset: linear evaluation accuracy rises sharply in the first 20 epochs, then continues to climb more gradually up to 50 epochs. This indicates that extended contrastive training allows the model to discover increasingly robust features, though with diminishing returns beyond around 40-50 epochs. We show the results in Figure 7.

3.3.4 Features dimension

Next, we examine how the size of the representation vector f_{dim} affects linear evaluation accuracy. As shown in Figure 8, accuracy improves markedly when increasing from very low dimensions ($32 \rightarrow 64$), peaks around 512, and then exhibits diminishing or even negative returns at larger sizes. This suggests an optimal "sweet spot" around $f_{\text{dim}} = 512$, where the model balances expressivity and generalization without overparameterizing the projection space. Those findings have however to be nuanced since the great computational time cost makes

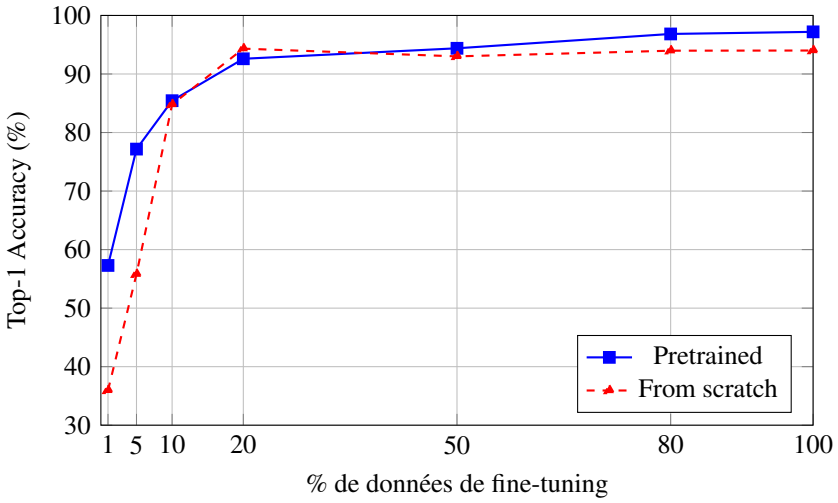


Figure 6: Top-1 accuracy as a function of fine-tuning data proportion on BloodMNIST, showing the rapid narrowing of the pretrained model's advantage beyond extremely low p.

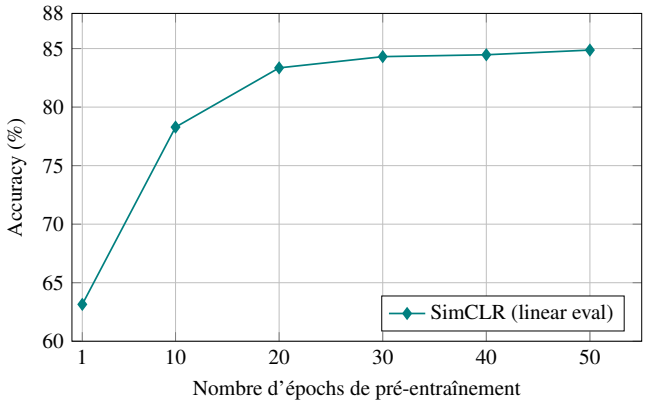


Figure 7: Top-1 accuracy under linear evaluation as a function of pre-training epochs, demonstrating continued gains with longer training.

it difficult to train the model for more than 15 epochs for each value.

4 Barlow Twins

4.1 Model presentation

4.1.1 Overview

Barlow Twins is a self-supervised learning method introduced to address redundancy reduction in deep representations without the need for negative pairs, as typically required in contrastive learning frameworks. The key idea is to maximize the similarity between the representations of two distorted views of the same image, while reducing the redundancy across the components of these representations.

4.1.2 Loss Function

The core of Barlow Twins is its loss function, which is based on the cross-correlation matrix between the embeddings of the two branches of the network. Given a batch of image pairs $(y_b^A, y_b^B)_{1 \leq b \leq B}$ obtained by applying different augmentations to each image, the twins networks encode them into $z_b^A = (z_{b,1}^A, \ldots, z_{b,n}^A)$ and $z_b^B = (z_{b,1}^B, \ldots, z_{b,n}^B)$, with n the size of the latent representation. The cross-correlation matrix $\mathbf{C}$ is computed as:

$$\mathbf{C}_{ij} = \frac{\sum_{b=1}^{B} z_{b,i}^A z_{b,j}^B}{\sqrt{\sum_{b=1}^{B} (z_{b,i}^A)^2} \sqrt{\sum_{b=1}^{B} (z_{b,j}^B)^2}} \quad \forall 1 \leq i \leq n, 1 \leq j \leq n$$

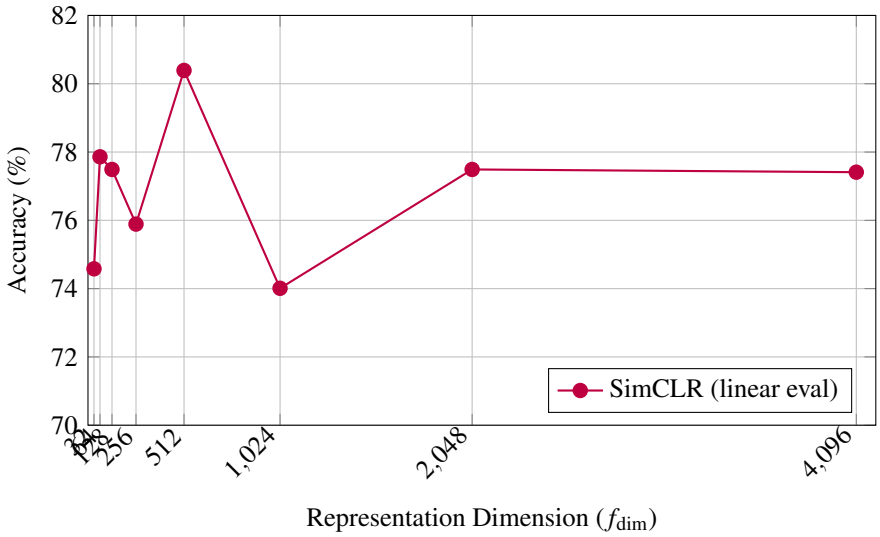


Figure 8: Top-1 accuracy under linear evaluation versus the representation dimension f_{dim}.

The Barlow Twins loss aims to make $\mathbf{C}$ as close as possible to the identity matrix, enforcing invariance (diagonal elements close to 1) and redundancy reduction (off-diagonal elements close to 0).

The loss is then given by :

$$\mathscr{L}_{BT} = \underbrace{\sum_i (1 - C_{ii})^2}_{\textbf{Invariance term}} + \lambda \underbrace{\sum_i \sum_{j \neq i} C_{ij}^2}_{\textbf{Redundancy reduction term}}$$

where λ is a hyperparameter controlling the weight of the redundancy reduction term.

4.1.3 Training and Pipeline

Barlow Twins adopts a Siamese architecture: two identical neural networks (sharing weights) process two independently augmented versions of the same image. The network is trained end-to-end using stochastic gradient descent, minimizing the Barlow Twins loss. Unlike contrastive methods, Barlow Twins does not require large batch sizes or a memory bank to provide negative samples, making it more efficient and scalable in practice.

4.1.4 Major Difference with SimCLR

The major difference between Barlow Twins and SimCLR lies in the way negative samples are used. SimCLR relies on a contrastive loss that compares positive pairs against a large number of negatives to prevent the network from learning trivial representations. In contrast, Barlow Twins entirely removes the need for negative samples, relying instead on the redundancy reduction objective through the cross-correlation matrix. This leads to a more stable and efficient training process, and avoids issues related to sampling or memory constraints.

4.2 Method evaluation

4.2.1 Linear Separability of SSL Representations

To evaluate the quality of the representations learned via Barlow Twins pretraining, we first assessed their linear separability before any supervised fine-tuning. For this, we extracted embeddings from the frozen encoder for both the labeled training and validation sets. We then trained a k-nearest neighbors classifier with cosine similarity on these embeddings, using $k = 200$.

The classifier achieved a top-1 accuracy of 79.7% on the validation set:

```
[k-NN] accuracy top-1 before fine-tuning:  0.797
```

This result is particularly impressive, given that no labels were used during the representation learning phase. It confirms that the learned features are already highly discriminative and well-structured in the embedding space.

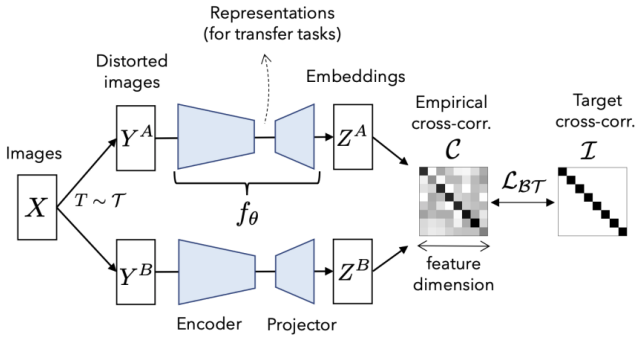


Figure 9: Barlow Twins pipeline: from image, two augmented views are processed by twin networks, and their embeddings are compared using the Barlow Twins loss. Source : *"Barlow Twins: Self-Supervised Learning via Redundancy Reduction"* Zbontar et al. [4]

4.2.2 Model finetuning

To evaluate the effectiveness of the Barlow Twins method, we performed as with SimCLR a linear classifier *finetuning* on increasing percentages of labeled images (1%, 5%, 10%, 20%, 50%, 100%).

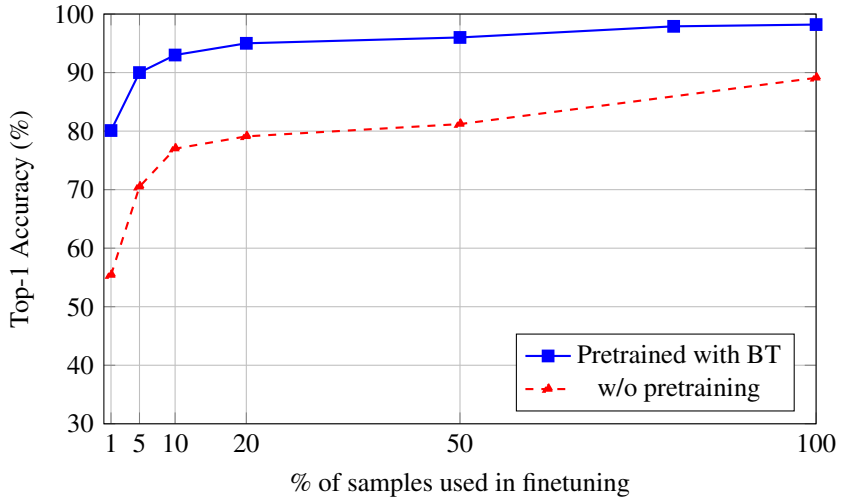


Figure 10: Finetuned Pretrained with Barlow Twins (BT) Model vs. Not pretrained model on PathMNIST.

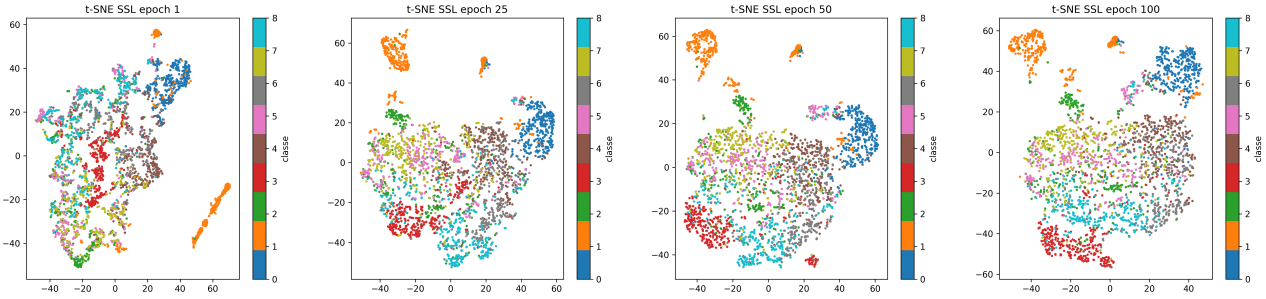
The results are particularly impressive: with only 20% of labeled data, the accuracy already exceeds 90%, and the curve quickly reaches a plateau close to the maximum performance. The macro-AUC also reaches nearly 1.0 from just 20% labeled data. This confirms the strong effectiveness of the representations learned by Barlow Twins, which provide highly discriminative descriptors even under low supervision.

The performance gap with the scratch baseline is especially significant in the low-data regime. With only 1% of annotated samples, the model pretrained with Barlow Twins reaches an accuracy of 80.1%, compared to just 55.4% for the scratch model. At 5% labeled data, the pretrained model achieves 90.0% accuracy, while the scratch model lags behind at 70.5%, confirming a substantial margin of +19.5 points. Even at 10%, the gap remains around 20 points. This underlines the **label efficiency** brought by self-supervised pretraining: high-quality features are learned independently of the annotation process.

Beyond 50%, the performance gap narrows as both models converge toward their respective ceilings, but Barlow Twins still retains an advantage (97.8% vs. 89.1% at full supervision). These results demonstrate that pretraining with Barlow Twins is not only beneficial, but essential for efficient learning in low-resource settings.

4.2.3 Exploring latent representation

Representations after SSL training. Self-supervised training was conducted for 100 epochs. To visualize how the learned representations evolve over time, we projected the extracted latent vectors at different epochs into two dimensions using the t-SNE algorithm. These visualizations allow us to assess the structure of the latent space without using any labels.

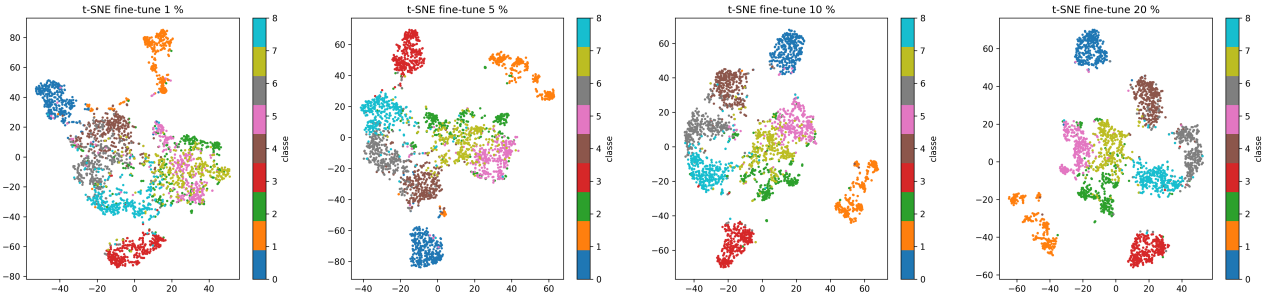


(a) Epoch 1 **(b)** Epoch 25 **(c)** Epoch 50 **(d)** Epoch 100

Figure 11: t-SNE projection of Barlow Twins latent representations at epochs 1, 25, 50, and 100

At the beginning of training, the representations are highly entangled. Gradually, consistent groupings emerge, indicating that the model is structuring the latent space even without supervision. By epoch 100, well-formed clusters appear, reflecting good invariance to augmentations.

Representations after fine-tuning. We then evaluated the impact of supervised fine-tuning on these representations using various amounts of labeled data (1%, 5%, 10%, 20%). The visualizations below illustrate how fine-tuning shapes the latent space.



(a) 1% labels **(b)** 5% labels **(c)** 10% labels **(d)** 20% labels

Figure 12: t-SNE projection after supervised fine-tuning

As early as 5% labeled data, we observe a clear improvement in class separation. From 10%, most classes are already well distinguished. This confirms the effectiveness of fine-tuning in refining the latent space learned by SSL, even in low-supervision settings.

The t-SNE visualizations show that SSL training already enables learning non-trivial and coherent representations. These representations are then effectively refined through supervised fine-tuning, even with a very small amount of labeled data.

4.2.4 Influence of certain hyper-parameters

Data augmentations To isolate the contribution of each transformation in our SSL pipeline, we removed *one* augmentation at a time and re-ran a short SSL pre-training[1] followed by a linear probe with 10 % supervision. Table 2 shows the accuracy obtained by the resulting backbones .

[1] 10 epochs, batch 512 on a 30 % subset to keep runtime reasonable.

Pipeline	Acc. @ 10 %
Full (all transforms)	79.9
One transform removed	
No–Color Jitter	38.0
No–Rot90	36.7
No–Gaussian Blur	73.6

Table 2: Single-factor ablation of data augmentations in the Barlow Twins pre-training pipeline.

Suppressing either *color jitter* or 90° *rotations* cuts performance by more than half, indicating that invariance to colour shifts and orientation is fundamental for histopathology patches. Gaussian blur is also beneficial (–6 pp), but less critical than the two colour/geometry cues. Together with the augmentation-matrix heat map, these results confirm that the full combination of transforms is required to obtain highly transferable representations with Barlow Twins.

Data Augmentation Combinations To better understand how different augmentations interact in the Barlow Twins setting, we constructed a heatmap matrix reporting the linear classification accuracy obtained when pretraining with each pair of transformations. Each cell (i, j) corresponds to the performance obtained when applying transformation i on the first view and j on the second.

Unlike SimCLR, where high diversity between views is desirable, Barlow Twins relies on redundancy reduction. This makes augmentation compatibility even more sensitive: some pairs must preserve structure to allow alignment of latent components. As shown in Figure 13, most combinations involving affine transforms, flips, blur, or color jitter maintain strong performance (above 82%). However, combinations involving random cropping consistently lead to a drop in accuracy, sometimes below 40%.

This result suggests that while random cropping may be beneficial in contrastive frameworks, it disrupts the feature alignment required in redundancy-based methods like Barlow Twins.

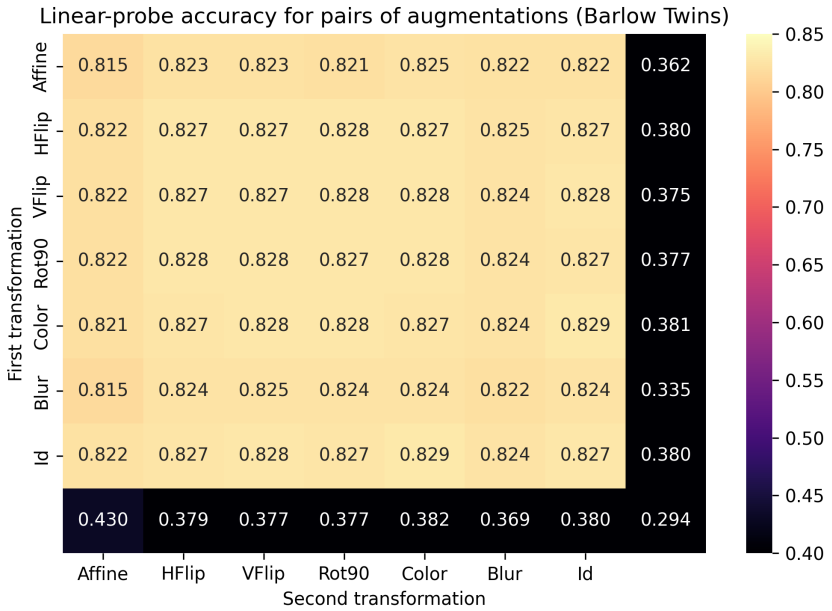


Figure 13: Accuracy of a linear classifier trained on frozen features obtained from Barlow Twins pretrained with various pairs of augmentations. Best performance is obtained with compatible geometric or color-based transformations (e.g., flip + color), while cropping causes a significant drop in accuracy.

5 Influence of the batch size

Given the moderate training set size ($\sim$ 89,000 samples), Figure 14 highlights distinct trends in how Barlow Twins and SimCLR respond to varying batch sizes during linear probing.

SimCLR shows steadily improving performance up to a batch size of 128, peaking around 80% accuracy. This aligns with its reliance on many negative pairs, which become more representative in larger batches. However, as batch size continues to grow beyond 256, SimCLR's accuracy plateaus and then declines—likely due to fewer effective gradient updates per epoch and an increased risk of sampling false negatives in a relatively limited dataset.

In contrast, Barlow Twins, which does not rely on negative pairs, performs well at smaller batch sizes and reaches its peak accuracy around 256. Beyond that point, its performance drops more sharply than SimCLR's (the same observation was made by the author of the method [4]). This may be due to over-averaging across large batches, which can dilute the training signal in redundancy-reduction objectives when data diversity is limited.

Overall, Barlow Twins offers better performance in the small-to-medium batch regime, making it more robust in data- or memory-constrained scenarios. SimCLR can surpass it at mid-sized batches, but only up to a point—beyond which both methods suffer from diminishing returns due to dataset size limitations.

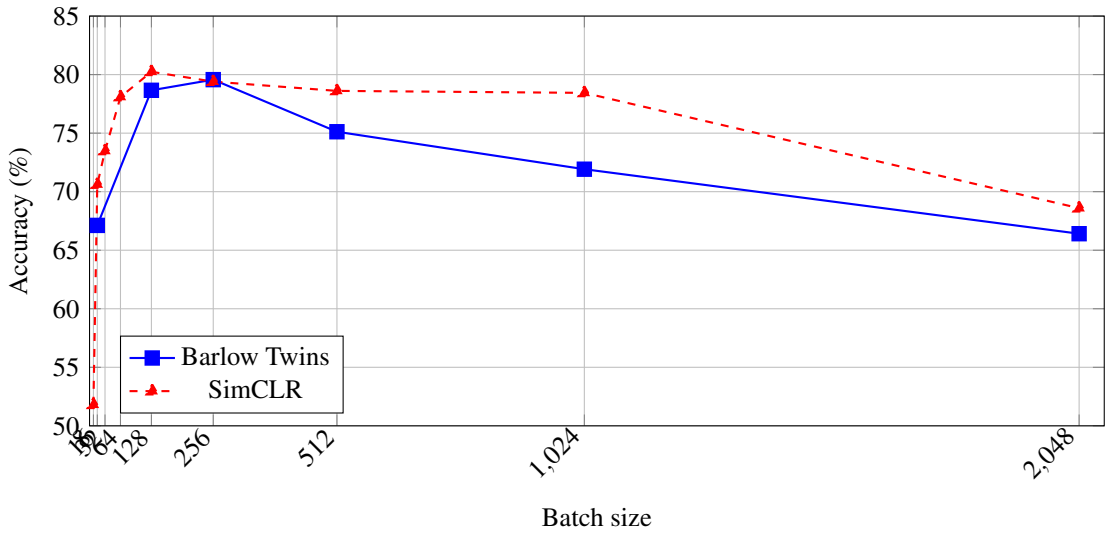


Figure 14: Top-1 Accuracy under linear probing v. the batch size

6 SimCLR v. Barlow Twins

Barlow Twins and SimCLR are both recent self-supervised learning methods designed to learn visual representations from unlabeled data. While they share the core idea of using augmented views of the same images to learn invariance, their approaches to learn image semantic differ. SimCLR relies on a contrastive loss, which requires both positive pairs and a large number of negative pairs. This approach typically necessitates large batch sizes to be effective, but it has the advantage of being conceptually simple and empirically robust. However, the reliance on many negatives can make SimCLR computationally expensive and sensitive to batch size, particularly on datasets with low visual diversity as medical image datasets.

In contrast, Barlow Twins introduces a redundancy-reduction objective based on the cross-correlation matrix between embeddings from positive pairs, without requiring any negative pairs. This makes Barlow Twins more memory-efficient, less sensitive to batch size, and generally more stable during training. Its design also makes it more robust on datasets where inter-class differences are subtle or where visually similar but semantically different samples may otherwise hinder contrastive learning. Although its loss formulation is more complex and may require careful balancing of objectives, in practice Barlow Twins often shows a slight advantage over SimCLR in scenarios with limited data diversity or computational resources as our, as illustrated in Figure 15 below.

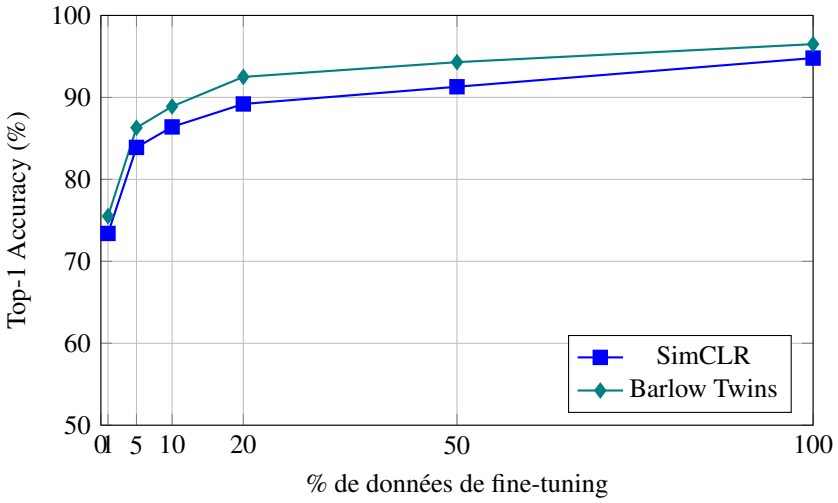


Figure 15: Performance comparison between Barlow Twins and SimCLR on a standard benchmark. Both methods achieve comparable results.

References

[1] Ting Chen et al. “A Simple Framework for Contrastive Learning of Visual Representations”. In: *Proceedings of the 37th International Conference on Machine Learning (ICML)* (2020). URL: https://arxiv.org/abs/2002.05709.

[2] Pietro Gori and Loïc Le Folgoc. *Representation Learning for Computer Vision and Medical Imaging*. https://pietrogori.github.io/teaching/RepLearnMVA. Master MVA, Télécom Paris / IP Paris. 2024.

[3] Jiancheng Yang et al. “MedMNIST v2-A large-scale lightweight benchmark for 2D and 3D biomedical image classification”. In: *Scientific Data* 10.1 (2023), p. 41.

[4] Jure Zbontar et al. “Barlow Twins: Self-Supervised Learning via Redundancy Reduction”. In: *Proceedings of the 38th International Conference on Machine Learning (ICML)* (2021). URL: https://arxiv.org/abs/2103.03230.

List of Figures