Reducing Data for Vision Foundation Models

Data-Efficiency of Self-Supervised Learning with Momentum Contrast

Makar Kuleshov[1]

Supervisor(s): Jan van Gemert[1]**, Alex Manolache**[1]**, Petter Reijalt**[1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 21, 2026

Name of the student: Makar Kuleshov
Final project course: CSE3000 Research Project
Thesis committee: Jan van Gemert, Alex Manolache, Petter Reijalt, Mitchell Olsthoorn

Abstract

Self-supervised contrastive learning is a popular way to pre-train vision foundation models. So far, it has mostly been studied with large pre-training datasets, and it is most accessible to organizations with massive computational resources. In this work we evaluate the data-efficiency of one such method, Momentum Contrast (MoCo), and investigate how to make it work better when less data is available. We pre-train a Vision Transformer with MoCo on subsets of Tiny-ImageNet ranging from 1,000 to 100,000 images, and evaluate the learned representations on a diverse set of downstream tasks using linear probing. We investigate how the training parameters of MoCo should be chosen for a given amount of data, how the downstream accuracy scales with the amount of pre-training data, and how this scaling differs across types of downstream tasks. We find that the best parameters depend on the amount of data: the optimal number of negatives used for the contrastive objective grows with the size of the dataset, while the momentum coefficient has no single best value. We also find that pre-training is beneficial even with very little data, the downstream accuracy grows approximately log-linearly with the size of the pre-training set, and the data-efficiency growth rate is larger for tasks that are similar to the pre-training data.

1 Introduction

Deep neural networks are at the core of modern computer vision. They are being actively developed and drive much of the progress in automatic image understanding tasks. Traditionally, these networks are trained on carefully curated datasets designed for a specific task, such as object detection or per-pixel prediction. The current dominant approach moves beyond this by first pre-training a foundation model [1] on large-scale broad data. This foundation model is then used as a backbone in a second step, where it is adapted on task-specific data.

A lot of the current research on foundation models focuses on improving computational efficiency [15; 16], expanding the range of tasks they can handle [8; 14], and scaling them up by training on even more data [3; 5; 17]. However, relatively little attention has been given to how these models can be trained with less data. As a result, developing such models is mostly limited to a small number of, often commercial, organizations that have more resources to work with massive datasets. This creates a fragile dependency, where access to important models for downstream tasks may be restricted by commercial interests. At the same time, the lack of transparency about the data used to train these models raises concerns about bias, copyright, and privacy.

A popular approach for training foundation models is self-supervised learning (SSL) [12]. It learns representations directly from the images themselves, without using external labels. This is well suited to foundation models because it captures general features that transfer well to a wide range of downstream tasks.

Contrastive learning is one of the main categories of SSL methods [10], and Momentum Contrast (MoCo) [9] is one of its representatives. At the time of publication, its latest version, MoCo v3 [2], achieved some of the best results among SSL methods when trained on ImageNet, a large dataset of general images. However, the performance of MoCo has not been studied in settings where only a small dataset is available.

The goal of this project is to close this gap and study the data-efficiency of MoCo. We train it on subsets of Tiny-ImageNet (a scaled-down version of ImageNet) of varying sizes, and evaluate the resulting models on the Visual Task Adaptation Benchmark (VTAB) [20]. More specifically, we focus on the following research questions:

1. How should the parameters specific to training a foundation model using MoCo be chosen to obtain the best accuracy for different sizes of pre-training data?
2. How does the downstream accuracy of MoCo scale with the amount of data used for pre-training?
3. How does this scaling vary across the different types of downstream tasks?

We make the following contributions. First, we show that the optimal parameters for MoCo depend on the amount of training data, and differ from the values recommended for the large-data setting in the original work. Second, we find that pre-training is beneficial even with very little data, and that the downstream accuracy grows approximately log-linearly with the size of the pre-training set. Finally, we demonstrate that the data-efficiency of pre-training varies across the different types of downstream tasks.

2 Background

2.1 Contrastive Learning

Contrastive learning is a self-supervised approach that learns useful image representations without labels. The main idea is to learn an embedding space in which different views of the same image lie close together, while views of different images are pushed apart [18]. The two views of an image are produced by applying random data augmentations, such as cropping, color jittering, and blurring, to the same source image. The two augmentations of one image form a positive pair, while augmentations of all other images act as negatives. By repeatedly pulling positives together and pushing negatives apart, the model is forced to capture the semantic content that is invariant to the augmentations rather than low-level pixel statistics. The resulting backbone can then be transferred to downstream tasks. Well-known contrastive learning methods include SimCLR, InstDisc, MoCo [10]. MoCo is the latest method that builds upon the previous work and improves their results, which is why we chose it for our work.

2.2 Vision Transformer

The Vision Transformer (ViT) [6] adapts the transformer architecture, originally developed for natural language processing, to images, and has become one of the most prominent

architectures in computer vision. ViTs also show strong results in self-supervised learning: in particular, the authors of MoCo v3 [2] show that a ViT backbone gives stronger results than a convolutional network. For this reason we use a ViT as the backbone in this paper.

2.3 Visual Task Adaptation Benchmark

Foundation models trained with self-supervised learning are usually evaluated on a diverse set of benchmarks, since a good representation should transfer well to many different tasks rather than to a single one [7]. An example of such a benchmark is the Visual Task Adaptation Benchmark (VTAB) [20], which collects a large number of image classification tasks and groups them into three categories. The *natural* tasks consist of natural images captured with standard cameras, such as everyday objects, animals, and scenes. The *specialized* tasks use images captured with specialized equipment, for example medical images and remote-sensing imagery. The *structured* tasks require an understanding of the structure of a scene, such as counting objects or estimating distances. By covering these different categories, VTAB gives a broad evaluation of the learned representations which we find important for our research.

3 Methodology

3.1 Momentum Contrast

Momentum Contrast (MoCo) is a contrastive learning method. We base our work on its latest version, MoCo v3 [2]. It uses two encoders, each built from a backbone that maps an augmented image to an embedding. The query encoder f_q processes the first view to produce a query embedding q, and the key encoder f_k processes the second view to produce the matching positive key k^+. The augmented views are passed through projector and predictor heads. The key encoder is not trained by backpropagation but is updated as an exponential moving average of the query encoder, where the rate of the updates is controlled by the momentum coefficient. This keeps the keys consistent across training steps and stabilizes learning.

As negative keys $\{k^-\}$, the embeddings that the query is pushed away from, we use the other images in the current batch.

Training minimizes the InfoNCE loss [18], which for a single query is

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k^+/\tau)}{\exp(q \cdot k^+/\tau) + \sum_{k^-} \exp(q \cdot k^-/\tau)}, \quad (1)$$

where τ is a temperature hyper-parameter. Minimizing the loss increases the similarity of the positive pair while decreasing the similarity to the negatives, and it is applied symmetrically by swapping the roles of the two views.

3.2 Approach

The focus of this project is to analyze the quality of the representations learned by MoCo when the training data is limited to different degrees, and to find out how to adapt it in such cases.

The two parameters of MoCo that most strongly affect the results are the number of negative keys and the momentum coefficient. The number of negatives matters because it determines how many other images each image is contrasted against in the loss, since we use in-batch negatives, it is set by the batch size. The momentum coefficient is important because it controls how quickly the key encoder follows the query encoder, and therefore how fast the learned representations are allowed to change during training.

To answer the research questions we conduct the following experiments:

1. We analyze how the number of negative keys used in the contrastive objective affects the results, and find the optimal value for different sizes of available data.
2. We analyze which momentum coefficient works best for different sizes of the training data.
3. Using the best parameters found in the previous experiments, we evaluate the model on the VTAB [20] benchmark of downstream tasks and analyze how the results change with the amount of available pre-training data.

4 Experimental Setup

In this section we describe our experimental setup to make the experiments reproducible. We first present the datasets, then the model configuration, the training and early-stopping protocol, and finally the linear-probing evaluation. The full source code is available in our public repository [13].

4.1 Datasets

The original MoCo v3 paper [2] pre-trained on the full ImageNet dataset. Because of the limited computational resources available for this project, pre-training on full ImageNet is infeasible, so we switched to Tiny-ImageNet. It is a downscaled subset of ImageNet that contains broad natural images, such as animals, vehicles, and everyday objects, grouped into 200 classes. The training set has 100,000 images, 500 per class, each of resolution 64×64 pixels. Despite its smaller scale, Tiny-ImageNet still covers a diverse set of general image categories.

To study how downstream accuracy scales with the amount of pre-training data, we construct nested splits of Tiny-ImageNet of increasing size: 1,000, 2,000, 4,000, and so on by doubling up to 64,000, plus the full 100,000 images. The images within each class are shuffled once before the splits are formed. Each split is balanced, containing an equal number of images from every class. The splits are nested, meaning that every larger split contains all smaller splits as subsets. This approach ensures that the only variable changing across the splits is the dataset size, so differences in downstream accuracy can be attributed to the amount of pre-training data rather than to which particular images were sampled.

For downstream evaluation we use the Visual Task Adaptation Benchmark (VTAB) [20]. We evaluate on all of its tasks, across all three categories: 7 natural, 4 specialized, and 8 structured. More specifically, we use VTAB-1K, the variant in which each task provides 1,000 training images. Keeping the amount of training data the same for every task makes the comparison between tasks fair and representative.

4.2 Model Configuration

Across all experiments we use the same MoCo configuration, changing only the parameters that a given experiment requires such as the batch size and the momentum coefficient.

As the backbone we use a ViT-Tiny/8. We chose this variant because its patch size of 8 is well suited to the small 64×64 images we work with.

For the remaining settings we follow the recommendations of the original MoCo v3 implementation [2]. We use the AdamW optimizer with a base learning rate of $1.5e\text{-}4$. This learning rate is scaled linearly depending on the batch size with the following rule:

$$\text{lr} = 1.5e\text{-}4 \times \text{BatchSize}/256, \tag{2}$$

At the start of training the learning rate is warmed up linearly over the first 40 epochs, which avoids large, destabilizing updates while the weights are still close to their random initialization.

To generate the two views of each image that the contrastive objective compares, we use the same data augmentations as in the reference code: a random resized crop, color jittering, random grayscaling, Gaussian blur, and a random horizontal flip.

4.3 Training Duration and Early Stopping

Our setting differs from the original paper in two ways: we train on Tiny-ImageNet rather than ImageNet, and we train on subsets of different sizes. Because of this, the number of epochs needed for a run to converge is not known in advance. We therefore adopt a protocol with a dynamic training duration and early stopping, which lets each run train for as long as it keeps improving.

Since the total number of epochs is not known in advance, we keep the learning rate constant after the warm-up.

To track progress we use kNN monitoring [19], a technique that is widely used in self-supervised learning research. Every 5 epochs, we evaluate the query encoder on the Tiny-ImageNet validation set using a kNN classifier built from the embeddings of the training data. Figure 1 shows the resulting monitoring curve for one run.

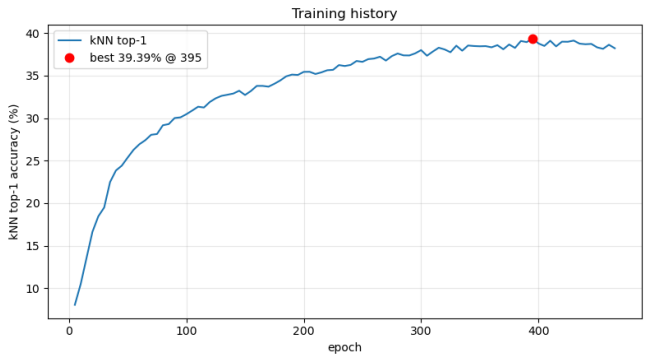

Figure 1: Example of kNN accuracy tracking for pre-training on 64000 images. The training protocol allows the model to achieve the maximum result according to the monitoring metric.

We continue training each model as long as its kNN accuracy keeps improving. To decide when to stop, we keep track of the running maximum of the kNN accuracy and terminate once this maximum has not improved during the last 15% of the epochs. We always let a model train for at least 350 epochs, since the early part of training tends to be noisier and its measurements are less reliable.

We consider this approach fair across the different splits because each model is allowed to train at its own pace until it reaches its best monitoring value. As the patience criterion is expressed in relative terms, the final shapes of the monitoring curves follow the same structure.

Finally, the checkpoint that reaches the highest kNN accuracy is the one we keep and use for the next steps.

4.4 Linear Probing Evaluation

The contrastive objective trains the backbone without using the labels. To measure how useful the learned representations are, we use linear probing, a standard technique for evaluating a pre-trained backbone. The projector and predictor heads are discarded, and the backbone is frozen so that its weights are not updated. A single linear classification layer is then trained on top of the embeddings produced by the frozen backbone, using a labeled dataset. The only parameters that are learned are those of this linear layer, which maps an embedding to a class.

Because the backbone is fixed, the accuracy of the linear classifier depends only on how linearly separable the frozen representations already are. A backbone that has learned semantically meaningful features lets even a simple linear layer separate the classes well, whereas a poor backbone does not. This makes linear probing an inexpensive and widely used proxy for the quality of the learned representations without the large cost of fine-tuning the whole network.

For the linear probing implementation we follow the protocol of the official MoCo v3 paper [2], and take the hyperparameters, such as the learning rate, the number of epochs, and the optimizer, from there.

To determine the optimal parameters of momentum contrastive learning, we run linear probing on the validation set of Tiny-ImageNet, similarly to how it was done in the original paper [2]. Since these parameters control the quality of the pre-training itself, we find it most logical to evaluate them on the validation set of the pre-training dataset, as this reflects that quality most directly. For the downstream evaluation, we run the same linear probing on VTAB.

5 Experiment Results

5.1 Number of Negative Keys Analysis

As shown in previous work [2; 9], momentum contrastive learning usually benefits from a larger number of negative keys, since contrasting an image against more other images provides deeper training signal. In this experiment we study whether this still holds in the low-data regime. In our setup the number of negatives corresponds to the batch size.

Training on all data splits would have been too computationally expensive, so we used every other split: 1k, 4k, 16k, and 64k. For each of them we trained with batch sizes of 125,

250, 500, 1000, and 2000. Previous work usually compares batch sizes that are powers of two, we followed the same approach, but adjusted the values slightly so that the batch size divides the size of the training data. Two combinations were left out: a batch size of 2000 is larger than the 1k split, and a batch size of 125 would have taken too long to train on the 64k split with our hardware. The results of the linear-probing evaluation on the Tiny-ImageNet validation set are presented in Table 1.

	Number of negatives				
Split	125	250	500	1000	2000
1k	**10.18**	10.08	9.37	9.13	–
4k	**19.71**	18.64	18.13	17.12	17.92
16k	29.05	31.91	**32.70**	30.72	31.21
64k	–	39.42	42.82	**43.42**	43.34

Table 1: Linear-probe top-1 validation accuracy (%) on Tiny-ImageNet at the final epoch, for each pre-training split and number of negatives. The maximum value in each row is shown in bold. The optimal number of negatives increases with the size of the training data.

As the table shows, in our case a larger batch size does not always lead to better results, and for the smaller splits the best accuracy is in fact reached with the smallest batch sizes. In general, the optimal batch size grows with the size of the training data. A possible explanation is that smaller batches produce noisier gradient estimates, which can act as a regularizer and improve generalization [11]. This effect is especially valuable when little data is available and overfitting is the main risk.

When the batch size approaches the size of the whole dataset, each image is contrasted against almost the same set of negatives at every step. This lack of variety in the negatives can lead to suboptimal convergence, which further reduces the benefit of using very large batches on the small splits.

The difference from the original work can also be explained by our use of Tiny-ImageNet instead of the full ImageNet. Tiny-ImageNet has only 200 classes instead of 1000, so for the same batch size a larger fraction of the negatives belong to the same class as the query. These false negatives make large batches more harmful in our setting than in the original one.

5.2 Momentum Coefficient Analysis

The momentum coefficient is the parameter that determines how the key encoder is updated. In previous work the values 0.9, 0.99, and 0.999 are usually compared. The MoCo v3 paper found 0.99 to perform best [2].

We reused the splits and the optimal number of negatives found in the previous subsection, and ran the same linear-probing evaluation on the Tiny-ImageNet validation set for these three momentum values, to see which one gives the best results. The results are presented in Table 2.

In our case there is no single best value across all splits. As the table shows, different momentum coefficients give the best result for different splits, and we do not observe a clear pattern. From this we conclude that there is no stable momentum coefficient that is suitable for all data regimes, it should be tuned carefully when training a MoCo model.

	Momentum coefficient		
Split	0.9	0.99	0.999
1k	9.83	**10.18**	8.82
4k	19.12	19.71	**20.54**
16k	30.33	**32.70**	31.27
64k	**43.64**	43.42	35.81

Table 2: Linear-probe top-1 validation accuracy (%) on Tiny-ImageNet at the final epoch, for each pre-training split and momentum coefficient. The best momentum coefficient in each row is shown in bold, there is no clear pattern explaining how to choose the optimal value.

5.3 Pre-Training Across All Splits

After determining the best parameters for MoCo for different training data sizes, we ran the full pre-training on all data splits using these optimal values. For the splits that we did not test directly, we used the parameters found for the closest smaller split (for example, for the 8k split we used the values found for 4k).

To make the evaluation more objective, we performed several training runs with different random seeds. Each run used a unique pair of seeds, one for generating the data subset and one for the random number generators used during training: $(42, 1205)$, $(43, 1206)$, and $(44, 1207)$.

One aspect we would like to highlight is the number of epochs needed to reach the best result under our kNN-monitoring protocol, which is reported in Table 3.

Training data size	Number of epochs
1k	4865
2k	3645
4k	3280
8k	1440
16k	1405
32k	840
64k	395
100k	380

Table 3: Number of pre-training epochs required to reach the early-stopping criterion for each training data size. The optimal number of epochs is decreasing with the growth of the training dataset size.

The original MoCo v3 paper [2] trained for 300 epochs and also tested 600 epochs, which did not give much improvement at their full ImageNet scale. In our setting, the number of epochs is higher for the small splits and generally decreases as the size of the training data grows. A possible explanation is that for smaller splits one epoch corresponds to fewer gradient updates, so more epochs are needed for convergence.

This result can be helpful in a scenario where kNN monitoring is not available: it is important to remember that the

required number of epochs changes with the size of the training data, so it is worth trying to extend the training runs.

5.4 Downstream Evaluation

We ran linear probing on all tasks of the VTAB benchmark [20] for the downstream evaluation. As a baseline, we also evaluated a backbone with randomly initialized weights. Figure 2 shows the accuracy averaged over all tasks, together with this random baseline.

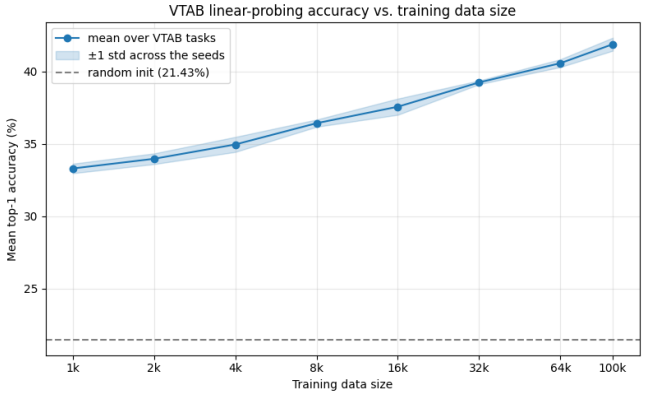

Figure 2: Mean VTAB linear-probing top-1 accuracy as a function of the pre-training data size. The accuracy growth rate is approximately linear on the logarithmic scale.

The results are quite stable: the highest standard deviation across the seeds is 0.56, reached on the 16k split, which is small at our scale, so the results are representative. Pre-training even on the smallest data split already gives a notable improvement over the random initialization, 33.29% (± 0.33) versus 21.43%. From there, the accuracy increases across all data splits at an approximately linear rate on the logarithmic scale. Each doubling of the dataset size improves the accuracy by 1.33% on average.

Figure 3 shows how the accuracy changes for each of the three VTAB task categories. For reference, the random-initialization accuracy of each category is:

- natural: 6.19%,
- specialized: 52.10%,
- structured: 19.44%.

The natural category shows the highest growth, which can be explained by the fact that Tiny-ImageNet, used for pre-training, contains images of a similar kind. The specialized tasks also improve noticeably, but by a smaller amount. The structured tasks, in contrast, stay almost flat: from the random-initialization value of 19.44%, the 1k split reaches only 27.17% (± 0.29) and the 100k split 29.31% (± 0.62). There are several possible explanations for this. First, the structured category contains challenging tasks, such as object counting and distance estimation, for which features learned from general images are less useful. Second, our models are trained on 64×64 images, and this resolution may be too low for such complex tasks.

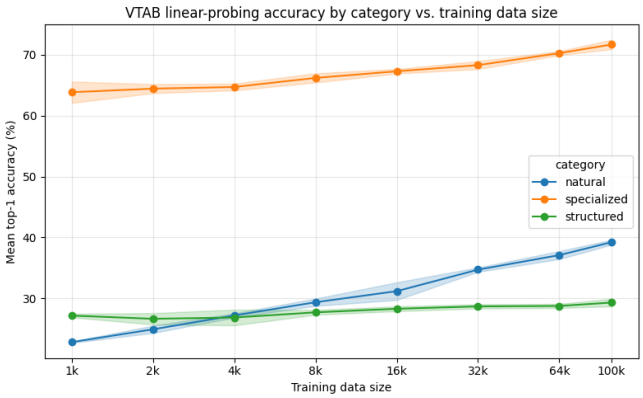

Figure 3: VTAB linear-probing top-1 accuracy for different categories of tasks. The natural tasks have the highest growth, accuracy on the specialized ones increases moderately, structured have almost no improvement.

6 Discussion

6.1 MoCo Parameters for Data-Efficiency

We investigated two parameters of momentum contrastive learning, the number of negative keys and the momentum coefficient, across different amounts of training data, and found that their optimal values differ across dataset sizes. They also differ from those reported in the original work [2], and some of its observations do not hold for small data splits. For example, increasing the number of negatives is not always beneficial when only little data is available. The main takeaway is that, to achieve the best data-efficiency, the parameters have to be adapted: the default values from the original work cannot simply be reused.

6.2 Downstream Accuracy Scaling

After analyzing how the downstream accuracy scales with the size of the training data, we found that even a small amount of pre-training data is already beneficial, and the accuracy grows approximately log-linearly with the dataset size.

This is similar to the findings of "When Does Contrastive Visual Representation Learning Work?" [4], which evaluated an older contrastive method SimCLR with a CNN backbone on larger datasets. Our result therefore extends what was previously known in the field, and possibly indicates a more general relation that holds in contrastive learning.

6.3 Data-Efficiency and Downstream Domains

The rate at which the accuracy grows differs between different categories of downstream tasks. This means that the effectiveness of MoCo for training a foundation model may depend on the domains of the target tasks. In particular, when the expected downstream tasks differ significantly from the pre-training data, the data-efficiency is worse, and more data is needed to reach a comparable level of results.

6.4 Limitations

Our experiments have several limitations, so very general claims should not be drawn from our results. First, we considered only one pre-training dataset, Tiny-ImageNet, and the

results may differ for other datasets. Second, the set of downstream tasks from VTAB that we used is not fully exhaustive, so the behavior may change for other tasks. Third, because of our limited computational resources, we considered only one model variant, ViT-Tiny/8, with images at a resolution of only 64×64 pixels, so the results may change for a different backbone or a higher resolution. Finally, because of the same compute constraints, the first two experiments, on the number of negatives and the momentum coefficient, were each run with a single random seed rather than averaged over several. Repeating them with multiple seeds would make their results more statistically sound.

7 Conclusions and Future Work

In this project we studied the data-efficiency of Momentum Contrast (MoCo) [2], a self-supervised method that learns image representations from data without using labels. Such methods are normally trained on very large datasets, and it is not well understood how they behave when much less data is available. We therefore asked how the training parameters of MoCo should be chosen for different dataset sizes, how its downstream accuracy scales with the size of the pre-training data, and how this scaling differs across types of downstream tasks. To answer these questions, we pre-trained MoCo models with a Vision Transformer [6] backbone on subsets of Tiny-ImageNet of increasing size. We evaluated the learned representations on the downstream tasks of the Visual Task Adaptation Benchmark (VTAB) [20], which covers several different categories. For this we used linear probing, where a simple linear classifier is trained on top of the frozen features.

We found that the parameters of MoCo that work best depend on the amount of available data. The number of negative keys that each image is compared against has an optimum that grows with the size of the dataset. For the momentum coefficient, which controls how quickly the model updates one of its encoders, we did not find a single value that is best for all dataset sizes, and there was no clear pattern in the measurements. We therefore conclude that these parameters cannot be chosen once and reused, but have to be adapted to each size of the available training data.

We also found that pre-training is useful even with a very small amount of data. The smallest pre-training set we used already outperformed a randomly initialized model. The downstream accuracy grows approximately log-linearly with the size of the pre-training set, so that each doubling of the data adds a similar amount of accuracy. This improvement is not the same for every task. Tasks with images that are similar to the pre-training data improve the most as more data is added, whereas tasks that are very different from it improve only slightly. In other words, the data-efficiency of MoCo also depends on how close the downstream task is to the data used for pre-training.

This work can be extended in several directions. One of the possible next steps is to extend the analysis with a different backbone, such as a convolutional neural network like a ResNet, to see whether the same data-efficiency behavior holds beyond transformers. Another promising direction is to compare MoCo with other self-supervised learning methods under the same limited data conditions, to find out whether our conclusions also apply to self-supervised learning more broadly.

8 Responsible Research

8.1 Ethical Considerations

Our research focuses on an important problem that contributes to democratizing the training of foundation models. Today, training such models requires very large datasets, which means that it is mostly accessible to a small number of, often private, organizations. By investigating the behavior of the Momentum Contrast method in the low-data regime, we make it easier for future researchers to train a model when only little data is available. In this way, our work helps make this kind of research more open and accessible.

8.2 Reproducibility

We have described our experimental setup in full detail in the corresponding section, including the datasets, the model configuration, the training and early-stopping protocol, and the evaluation procedure. In addition, we make the source code of our implementation publicly available [13], so that our experiments and results can be reproduced.

References

[1] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models, 2022.

[2] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers, 2021.

[3] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks, 2024.

[4] Elijah Cole, Xuan Yang, Kimberly Wilber, Oisin Mac Aodha, and Serge Belongie. When does contrastive visual representation learning work?, 2022.

[5] Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, et al. Scaling vision transformers to 22 billion parameters, 2023.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[7] Linus Ericsson, Henry Gouk, and Timothy M. Hospedales. How well do self-supervised models transfer?, 2021.

[8] Yutong Feng, Jianwen Jiang, Mingqian Tang, Rong Jin, and Yue Gao. Rethinking supervised pre-training for better downstream transferring, 2022.

[9] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning, 2020.

[10] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning, 2021.

[11] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima, 2017.

[12] Asifullah Khan, Anabia Sohail, Mustansar Fiaz, Mehdi Hassan, Tariq Habib Afridi, Sibghat Ullah Marwat, Farzeen Munir, Safdar Ali, Hannan Naseem, Muhammad Zaigham Zaheer, Kamran Ali, Tangina Sultana, Ziaurrehman Tanoli, and Naeem Akhter. A survey of the self supervised learning mechanisms for vision transformers, 2025.

[13] Makar Kuleshov. Data-efficiency of momentum contrast: source code. https://github.com/kulmak41/moco-data-efficiency, 2026.

[14] Chunyuan Li, Zhe Gan, Zhengyuan Yang, Jianwei Yang, Linjie Li, Lijuan Wang, and Jianfeng Gao. Multimodal foundation models: From specialists to general-purpose assistants, 2023.

[15] Yanyu Li, Geng Yuan, Yang Wen, Ju Hu, Georgios Evangelidis, Sergey Tulyakov, Yanzhi Wang, and Jian Ren. Efficientformer: Vision transformers at mobilenet speed, 2022.

[16] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification, 2021.

[17] Mannat Singh, Quentin Duval, Kalyan Vasudev Alwala, Haoqi Fan, Vaibhav Aggarwal, Aaron Adcock, Armand Joulin, Piotr Dollár, Christoph Feichtenhofer, Ross Girshick, Rohit Girdhar, and Ishan Misra. The effectiveness of mae pre-pretraining for billion-scale pretraining, 2024.

[18] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2019.

[19] Zhirong Wu, Yuanjun Xiong, Stella Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance-level discrimination, 2018.

[20] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, Lucas Beyer, Olivier Bachem, Michael Tschannen, Marcin Michalski, Olivier Bousquet, Sylvain Gelly, and Neil Houlsby. A large-scale study of representation learning with the visual task adaptation benchmark, 2020.