Accelerating Self-Supervised Learning via Efficient Training Strategies

Mustafa Taha Koçyiğit   Timothy M. Hospedales   Hakan Bilen

University of Edinburgh, UK

{taha.kocyigit,t.hospedales,h.bilen}@ed.ac.uk

Abstract

Recently the focus of the computer vision community has shifted from expensive supervised learning towards self-supervised learning of visual representations. While the performance gap between supervised and self-supervised has been narrowing, the time for training self-supervised deep networks remains an order of magnitude larger than its supervised counterparts, which hinders progress, imposes carbon cost, and limits societal benefits to institutions with substantial resources. Motivated by these issues, this paper investigates reducing the training time of recent self-supervised methods by various model-agnostic strategies that have not been used for this problem. In particular, we study three strategies: an extendable cyclic learning rate schedule, a matching progressive augmentation magnitude and image resolutions schedule, and a hard positive mining strategy based on augmentation difficulty. We show that all three methods combined lead up to 2.7 times speed-up in the training time of several self-supervised methods while retaining comparable performance to the standard self-supervised learning setting.

1. Introduction

Learning representations without manual human annotations that can be successfully transferred to various downstream tasks has been a long standing goal in machine learning [17, 48]. Self-supervised learning (SSL) aims at learning such representations discriminatively through pretext tasks such as identifying the relative position of image patches [14] and solving jigsaw puzzles [36]. The recent success of SSL methods [23, 6, 8] builds on contrastive learning where the representations are in a latent space invariant to various image transformations such as cropping, blurring and colour jittering. Contrastive learned representations have been shown to obtain on par performance with their supervised counterparts when transferred to various vision tasks including image classification, object detection, semantic segmentation [7, 22], and extended to medical imaging [1] as well as
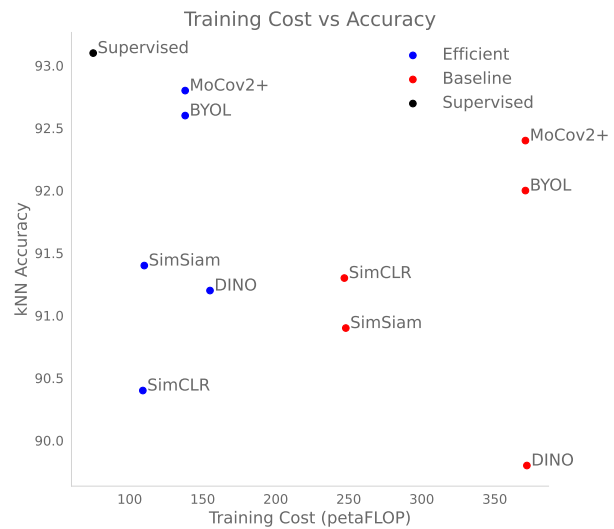

Figure 1. Classification accuracy *v.s.* training cost of various SSL methods, which are trained on a subset of ImageNet by using ResNet50 (see Sec. 4.2 for experimental details). Our method (drawn in blue) successfully significantly accelerates the SSL methods (drawn in red) without any significant drop in their performance.

multi-view [45] and multi-modal learning [37].

Despite the remarkable progress, an important downside of SSL methods is their high training cost which hampers the development and adoption of these promising techniques. Even the most efficient SSL methods require at least an order of magnitude more computation to reach the performance of supervised methods, *e.g.* training BYOL [22] requires 23 times more computation resulting from 8.8 times more iterations and 2.6 times more computation at each iteration than its supervised counterpart to reach similar accuracy. The large training cost is largely due to the challenging task of learning invariant representations over a large set of images and various augmentation transforms. In this paper, we focus on developing algorithms that can speed up the training of SSL while maintaining their performance.

Prior work in efficient supervised training reduced the

cost of training by gradually increasing the training resolution together with the augmentations magnitude using Progressive Learning [44]. While augmentation is a regularization mechanism in supervised learning it is the main source of supervision in SSL and therefore plays a much more important role. Additionally, Super Convergence [41] is introduced which uses a cyclic learning rate and anti-phased momentum schedule that is used to accelerate convergence in supervised learning tasks. The longer duration of training makes the application of Super Convergence more difficult and using it together with Progressive Learning causes instabilities in the early stage of training.

In this work, we aim to increase the training efficiency of self-supervised training methods by optimizing the learning rate, resolution, and augmentation schedules to allow reaching the same level of performance with a smaller computational budget. Inspired from [41] and [44] that were used in supervised learning we propose a combined learning rate and resolution schedule for self-supervised learning. Additionally, we modify the augmentation strategy used in self-supervised learning for faster training.

Our contributions are three folds. Fixed 1-cycle Learning Rate schedules (see Sec. 3.1) allows Super-Convergence [41] to take place for longer training times by fixing the duration of its warm-up phase while extending the decay phase and comparing it against alternative learning rate schedules used in self-supervised learning. Super Progressive Learning (see Sec. 3.2) proposes a suitable Progressive Learning schedule [44] for SSL by adding a warm-up stage with full resolution at the beginning of the resolution schedule while increasing the augmentation magnitude gradually during training so that it works well with our proposed learning rate schedule. Hard Augment (see Sec. 3.3) selects a hard augmentation pair dynamically from multiple low resolution augmentations in order to boost the learning signal and regularize the training.

We show that our method accelerates the training of different self-supervised learning methods and architectures while maintaining comparable performance to standard training. We provide ablation studies for analysing the contribution of different methods and determining the optimum hyperparameters of our methods and add theoretical justifications.

2. Related Works

Generative models like DBN [25], VAE [32] and GANs [20] have been used for unsupervised representation learning however discriminative SSL have proven to be more effective. While early work on SSL focused on designing various pretext tasks such as solving jigsaw puzzles [36], colouring [51, 33, 52] and predicting rotation [18], more recent techniques focused on learning augmentation invariant representation [49] [46] to replace the loss functions that require manual supervision. Recently, Contrastive learning methods[23, 6, 8] have shown promising results for various

target tasks and caused renewed interest in this area. However, at least an order of magnitude, more computation is required when training these methods in order to get comparable performance to supervised learning. In fact, we demonstrate our method in these recent models and show that SimSiam [9], BYOL [22], MoCov2 [8], SimCLR [6] and DINO [4] training can significantly be sped up.

By adaptation gradient noise during training large scale data parallel training has been used for increasing training speed in large deep learning methods. Using many accelerators together and increasing the total batch size training speed can be scaled up almost linearly with the number of accelerators [35]. This way, supervised ImageNet training can be done in minutes [42] or hours [21] instead of days even though the total training cost remains roughly the same. This required adapting the hyperparameters so that gradient noise was managed properly [43, 28, 5] by using the linear relationship between the batch size and learning rate [21, 42]. It was shown that training with larger learning rates had also a regularization effect [28, 5, 34] and prevented convergence to sharper minima [31]. We use the relationship between gradient noise and learning rate schedules to optimize for training efficiency instead of speed where we synchronize the learning rate schedule with our resolution schedule to reduce the total cost of training. We use a cyclic learning rate schedule [41] which combines cyclic learning rate and cyclic momentum adaptation together as our learning rate schedule due to its efficient training performance and adapt it to longer training regimes common in SSL training.

Curriculum learning [2] has been used to accelerate convergence for deep learning methods and when applying it to self-supervised learning, augmentation strength and image resolution are the most relevant parameters. Image resolution directly affects accuracy [26, 47] while reducing run-time quadratically. Previous work [27] on efficient supervised learning gradually increased the image resolution in the DAWN benchmark [11] to accelerate training while having slightly lower performance. Recently [44] proposed gradually increasing the training image resolution while increasing augmentations strength to allow for both faster and accurate training. While augmentations are used for regularization in supervised learning they are the main source of supervision for recent self-supervised methods and determine task difficulty. We build on this intuition and adapt Progressive Learning for efficient self-supervised training since in order to reduce the training cost especially in the initial phase while increasing the difficulty of the task gradually. To the best of our knowledge having a good schedule for augmentation magnitude and its relation to resolution hasn't been examined for self-supervised learning.

Hard positive mining is another technique that can be used to accelerate training. Importance sampling based on sample loss has been shown to accelerate supervised learn-

ing [30, 29]. In object detection [38] selects regions with the highest loss in order to select useful positives and balance positive to negative regions effectively. [19] applied multiple augmentations and back-propagated only the augmented sample with the maximum loss in order to improve the adversarial robustness of their method while increasing supervised performance.[3] have introduced a multi-crop strategy for self-supervised learning that combines low resolution and high resolution crops in order to show increased performance while trying to keep the computational cost limited. We mine hard augmentations during training by utilizing the loss based importance sampling technique to dynamically select the most useful augmentations. Since our focus is on efficient training, we used down sampled versions of the augmented images in the selection pass to minimize the overhead.

3. Method

In this section we introduce the techniques used to accelerate self-supervised training. Each technique can be used to accelerate training on its own but they can be combined together in a synergistic way to enable faster training. We introduce Fixed 1-cycle Learning Rate Schedule in Sec. 3.1, Super Progressive Learning in Sec. 3.2 and Hard Augment in Sec. 3.3.

We formulate a contrastive loss function that measures difference between pairs of images and optimizing this loss allows us to learn parameters θ for a deep neural network f_θ that produces similar representations for augmentations of the same image and have representation that is going to be useful for various target tasks. Let D be an unlabeled dataset consisting of $|D|$ images with resolution r. We randomly sample two image transformations τ and $\bar{\tau}$ from the transformation space T for each training image $\boldsymbol{x}$, apply them to $\boldsymbol{x}$ to obtain two views $\boldsymbol{v}$ and $\bar{\boldsymbol{v}}$, extract their features through the deep neural network $\boldsymbol{z} = f_\theta(\boldsymbol{v})$ and $\bar{\boldsymbol{z}} = f_\theta(\bar{\boldsymbol{v}})$ respectively. To learn the network weights θ, we minimize the loss function L that represents the mismatch between two representations $\boldsymbol{z}$ and $\bar{\boldsymbol{z}}$ over the dataset and transformation space:

$$\min_\theta \mathbb{E}_{x \sim D, (\tau, \bar{\tau}) \sim T} L(\boldsymbol{z}, \bar{\boldsymbol{z}}) \qquad (1)$$

We use minibatch Stochastic Gradient Descent (SGD) optimizer with momentum:

$$L^{(B)}(\theta) = \frac{1}{|B|} \sum_{\boldsymbol{x} \sim B, (\tau, \bar{\tau}) \sim T} L(\boldsymbol{z}, \bar{\boldsymbol{z}}), \qquad (2)$$

where the minibatch B consists of $|B|$ randomly sampled images from D and the loss is averaged over the samples to obtain a noisy yet unbiased estimate of the true gradient.

The update rule for θ is given as:

$$\begin{aligned} \mu_t &= \beta_t \mu_{t-1} - \epsilon_t \nabla_{\theta_t} L^{(B)}(\theta_t) \\ \theta_{t+1} &= \theta_t - \mu_t \end{aligned} \qquad (3)$$

where ϵ_t is the learning rate and β_t is the momentum weighs at step t. The values for the learning rate and momentum is given by a learning rate scheduler $(\epsilon_t, \beta_t) = S(t)$.

Eq. (3) can be interpreted as stochastic differential equation (*e.g.* [43]):

$$\frac{d\theta}{dt} = \mu, \quad \frac{d\mu}{dt} = \beta_t \mu - \frac{dL}{d\theta} + \eta_t \qquad (4)$$

where $\eta(t) \sim \mathcal{N}(0, g\boldsymbol{F}(\theta)/|D|)$ is an additive Gaussian noise originating from the stochasticity and $\boldsymbol{F}(\theta)$ is the covariance matrix for gradients of the samples and g is the "noise scale" which is given by $g \approx \frac{\epsilon_t |D|}{b(1-\beta_t)}$. For our analysis we will utilize the noise in the gradients which is proportional with the learning rate ϵ_t while being inversely proportional with batch size b and $1 - \beta_t$. This relationship plays an important role when adapting hyperparameters to different setups and for giving us a theoretically grounded understanding for learning rate schedules and their relationship with Progressive Learning.

3.1. Fixed 1-cycle Learning Rate Schedule

Cosine annealing learning rate schedule which has been used widely in SSL [23, 8, 9] decays the learning rate starting from a maximum learning rate value ϵ_{max} using a cosine function and can be described as,

$$\epsilon_t = \frac{1}{2}\epsilon_{max}(\cos(\frac{t}{L}\pi) + 1), \qquad (5)$$

where L is the total number of iterations.

Methods that perform better with larger batch sizes train with larger learning rates in order to maintain the gradient "noise scale" $g \propto \frac{\epsilon_t}{b}$ during training. In order to prevent instabilities in the early stages of training due to rapidly changing parameters values [21] a linear warm-up phase for the learning rate is added [6, 22, 10, 50, 3, 4],

$$\epsilon_t = \begin{cases} \frac{t}{t_w}\epsilon_{max}, & t < t_w \\ \frac{1}{2}\epsilon_{max}(\cos(\frac{t-t_w}{L-t_w}\pi) + 1), & \text{otherwise} \end{cases} \qquad (6)$$

where t_w is the number of warm up steps.

On the other hand, Cyclic Learning Rate (CLR) schedule introduced by [39] and later refined as the 1-cycle learning rate (1-CLR) schedule [41, 40] starts with a small learning rate and than increases the learning rate to a very high value and than decays it gradually to a very small value while changing the momentum at the opposite direction as the learning rate. The learning rates and momentum values at each step using a cosine window can be calculated by
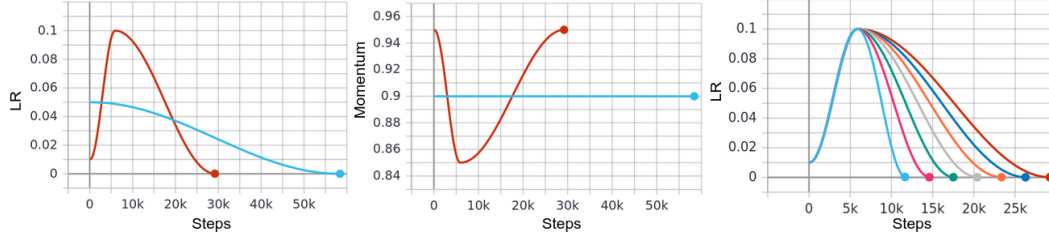
Figure 2. Learning rate (left) and momentum (middle) for standard cosine annealing learning rate schedule in blue and F1-CLR schedule in red are visualized. Learning rate for F1-CLR schedule with different length of training instances is visualized (right)

$$\epsilon_t = \begin{cases} \epsilon_{max} - \frac{1}{2}\epsilon_{max}(\cos(\frac{t}{\rho L}\pi) + 1) & t < \rho L \\ \frac{1}{2}\epsilon_{max}(\cos(\frac{t - \rho L}{L(1-\rho)}\pi) + 1) & \text{otherwise} \end{cases} \quad (7)$$

$$\beta_t = \begin{cases} \beta_l + \frac{1}{2}(\beta_h - \beta_l)(\cos(\frac{t}{\rho L}\pi) + 1) & t < \rho L \\ \beta_h - \frac{1}{2}(\beta_h - \beta_l)(\cos(\frac{t - \rho L}{L(1-\rho)}\pi) + 1) & \text{otherwise} \end{cases}$$

where ρ is the time percentage of time allocated for the first phase, while β_l and β_h are lower and higher limits for momentum respectively. We use the learning rate (LR) range test proposed by [40] to set the maximum learning rate ϵ_{max} please see the details in supplementary material C.

A phenomenon called Super Convergence has been demonstrated in supervised classification where using larger learning rates and the 1-CLR schedule training time to reach a specified performance has been decreased dramatically [27]. However, unlike supervised learning where longer training times do not generally result in better performance and can sometimes even cause worse performance due to over-fitting, in SSL the quality of the representation typically improves with longer training time [8, 6]. A problem with the current 1-CLR is that the percentage of the first phase is being determined in proportion to the full training duration which causes an extremely long warm-up phase in longer training settings wasting compute time. To address this problem, we propose to extend the annealing phase of 1CLR while keeping the warm up length t_w the same which we call *Fixed 1-cycle Learning Rate* (F1-CLR) schedule,

$$\epsilon_t = \begin{cases} \epsilon_{max} - \frac{1}{2}\epsilon_{max}(\cos(\frac{t}{t_w}\pi) + 1 & t < t_w \\ \frac{1}{2}\epsilon_{max}(\cos(\frac{t - t_w}{L - t_w}\pi) + 1) & \text{otherwise} \end{cases} \quad (8)$$

$$\beta_t = \begin{cases} \beta_l + \frac{1}{2}(\beta_h - \beta_l)(\cos(\frac{t}{t_w}\pi) + 1) & t < t_w \\ \beta_h - \frac{1}{2}(\beta_h - \beta_l)(\cos(\frac{t - t_w}{L - t_w}\pi) + 1). & \text{otherwise} \end{cases}$$

Fig. 2 shows a comparison between cosine schedule and F1-CLR schedule in terms of learning rate (left) and momentum (middle) as well as illustrating the F1-CLR learning rate schedule for different lengths of training (right). The anti-phased movement of momentum in F1-CLR allows larger learning rates to be achieved while keeping gradient noise

in check ($g \propto \epsilon_t/(1 - \beta_t)$ see Eq. (4)). Note that we do not extend the warm up phase, as the warm up is used as a stabilizer, while the learning rate and gradient noise scale are increasing ($g \propto \epsilon_t$ see Eq. (4)).

3.2. Super Progressive Learning

Input resolution is very important factor that effects training time. Typically image resolution is a trade-off hyperparameter between performance and computational load, *i.e.* higher resolution, higher performance but also more computations. In SSL, the standard practice [23, 8, 9, 6, 22, 10, 50, 3, 4] is to train with fixed resolution and fixed augmentation settings.

We hypothesize that higher resolution inputs are only necessary when there is small amount of noise in the gradients, and hence, propose to employ a learning rate aware progressive learning strategy inspired by [44]. However, as starting training with small input resolution as in [44] results in additional noise in the gradients which prevent it from being trivially incorporated to F1-CLR due to the instabilities in the warm-up phase. Hence, we propose train with full resolution images during the warm-up phase before going into the linearly increasing resolution schedule while we gradually increase the augmentation magnitude as shown in Fig. 3 (left, middle). We call this strategy *Super Progressive Learning*. When the learning rate is high there is a large amount of noise in the gradients $g \propto \epsilon_t$ and this allows us to use low resolution images at that point without resulting in bad performance however since the training starts with small learning rates we need to adapt our schedule and use large resolution images.

Note that the speedup of this strategy depends on the input resolution schedule (r_t). Assuming a quadratic relationship between training time and resolution, the speedup M can be calculated as:

$$M = \frac{1}{L}\sum_t (\frac{r_{max}}{r_t})^2, \quad (9)$$

where r_t is the input image resolution at time t, r_{max} is the final resolution. We discretize the resolution steps by 32 due to the fact that most network architectures have 5 pooling/dilation layers. The minimum value for the resolution schedule is determined empirically.
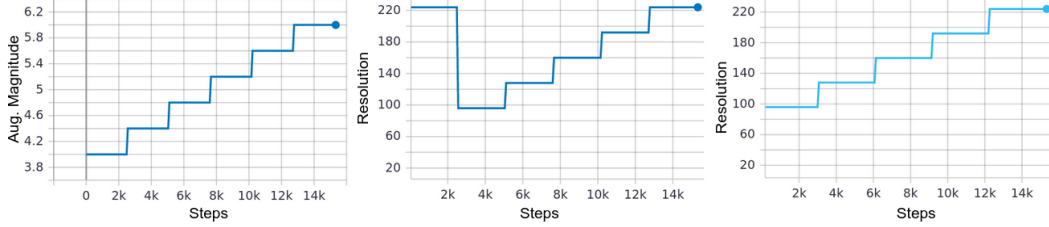
Figure 3. Augmentation magnitude and input resolution schedule where augmentation magnitude is gradually increased while the input size starts large and than gradually increases from its minimum value mimicking the inverse of the F1-CLR schedule. (left) Augmentation Magnitude, (middle) Super Progressive Learning resolution schedule, (right) Progressive Learning resolution schedule

3.3. Hard Augment

So far we focused on varying the effective step size and input resolution. Informativeness of training samples is another important factor that can speed up the training by providing more efficient gradients. We propose Hard Augment to boost the learning signal in self-supervised learning. We generate m augmentations $\tau^m = \{\tau_{1..m} : \tau_{1..m} \sim T\}$ for each image and evaluate the loss for all pairs of augmentations $p = \binom{\tau^m}{2}$ using a forward pass to select the augmentation pair that results in the largest amount of loss for back-propagation. By focusing on the samples that produce the most lost and ignoring a large fraction of the augmentations that produce a small loss and do not make a significant contribution to the training we can accelerate training dramatically. The overall objective that we optimize is

$$L_{ha}^{(B)}(\theta) = \frac{1}{|B|} \sum_{\boldsymbol{x} \sim D, \tau_{1..m} \sim T} \left[\max_{\tau_p \in p} L(\boldsymbol{z_i}, \boldsymbol{z_j}) \right]. \quad (10)$$

[19] have shown that using the maximum of multiple augmentations applies a regularisation on the gradient-norm with respect to the input images $\|\nabla_{\boldsymbol{x}} L(\boldsymbol{z_i}, \boldsymbol{z_j})\|_2$ that is in the order of $\sigma\sqrt{\log m}$ where σ is the strength of the augmentation under the assumption that $\tau(x) \sim \mathcal{N}(x, \sigma^2 I)$. The regularization effect in our method can be seen as a corollary of their theorem that uses a pair of augmentations instead with a number of equivalent augmentations as the number of pairs $|p|$.

Crucially in order to decrease the overhead of selection we propose to down sample the images during the selection pass to r_{sel}. The image pairs that have the highest loss will than be used in training in their original resolution r. We do not need to have full resolution for the selection pass as we only need to have sufficient resolution to rank the augmentation pairs with respect to their loss and to find the highest loss pair which requires far less precision than is required for obtaining good quality gradient updates during training.

$$M_{ha} = \frac{r^2 C}{r^2 C + m r_{sel}^2} \quad (11)$$

Algorithm 1 Efficient SSL Training, PyTorch-like

```
# f: backbone + projection layers
# criterion: loss function for SSL method
# aug: augmentation function
# F1CLR: Super Convergence Schedule (Section 3.1)
# SP: Super Progressive Schedule (Section 3.2)

t=0 # iteration
for e in range(epochs):
    for x in loader: # load a minibatch x with |B|
        samples
        eta, beta = F1CLR(t) # Update learning rate
            and momentum weight
        r, m = SP(t) # Update resolution(r) and
            augmentation magnitude(m)
        vs = [aug(x, r, m) for i in range(n)] # n
            random augmentation
        vi, vj = hard_augment(vs) # select hardes
            augmentations
        zi, zj = f(vi), f(vj) # forward-propagation

        L = criterion(zi,zj) # Loss calculation
        L.backward() # back-propagation
        update(f, eta, beta) # SGD update
        t+=1

def hard_augment(vs): # negative cosine
        similarity
    vs_small = interpolate(vs, r_min) # down
        sample views to r_min
    zs = f(vs) # forward prop for all views

    indexes = arg_max_loss_pair(zs) # determine
        pairs with maximum loss
    vi, vj = hard_select(vs, indexes) # select
        pairs by tensor indexing
    return vi, vj
```

We find the minimum resolution r_{sel} that we can use for the selection pass empirically. On a typical setting in ImageNet where the original resolution is $r = 224$ and the reduced selection resolution is $r_{sel} = 64$ with $m = 4$ augmentations and 6 possible pairs and the fraction for full training iteration cost to a forward pass cost is $C = 6$ we only add $1 - M_{ha} = \%5$ overhead on the training while significantly boosting the training speed by focusing on the most valuable augmentations.

We have provided the pseudo code for our method in Algorithm 1. Our method can easily be adopted to different SSL methods without any major difficulty as it operates at the level of augmentations, optimizer parameters and sampling strategies.

Architecture	Baseline		Efficient	
	Acc.(%)	Cost (pF)	Acc.(%)	Speedup
ResNet18	51.1	3756	51.3	2.3x
ResNet50	69.2	8494	69.3	2.3x

Table 1. Our efficient strategies applied to different backbone architectures using SimSiam model on Imagenet dataset. Speed up is reported based on the reduction in training cost which is measured in terms of petaFLOPs

Model	Baseline		Efficient	
	Acc.(%)	Cost (pF)	Acc.(%)	Speedup
SimSiam	90.9	248	91.4	2.3x
BYOL	92.0	371	92.6	2.7x
SimCLR	91.3	247	90.4	2.3x
MoCov2+	92.4	371	92.8	2.7x
DINO	89.8	372	91.2	2.4x

Table 2. Our efficient strategies applied to different SSL models training on ResNet50 backbone architecture on Imagenette dataset. Speed up is reported based on the reduction in training cost which is measured in terms of petaFLOPs

4. Experiments

In this section, we first study evaluate our method on different backbone architectures in Sec. 4.1 and on various existing SSL methods in Sec. 4.2, finally we analyze the effect of each proposed component in our method in Sec. 4.3 and run ablation studies that analyze the hyperparameters and different trade-offs of our method in Sec. 4.4.

We evaluate the experiments on the ImageNet dataset [13] in terms of linear probing accuracy. This is done by freezing the backbone after pre-training and training a linear classification layer in a supervised way following [9]. The evaluation on the Imagenette dataset[1] is done using online kNN accuracy of the feature maps. When calculating speed up we consider the number of steps used in training and multiply that by the number of floating point calculations made in each step and compare it against the baseline.

Implementation Details Here we provide details on the default parameters that we have used during our experiments where we use the SimSiam [9] model with ResNet50 backbone using the SGD optimizer with .9 momentum. On ImageNet experiments we use a batch size of 512 and weight decay of 0.0001. For the baseline experiments we use the Cosine Annealing learning rate schedule with learning rate of 0.1 (η_{max}) and trained them for 200 epochs (t_{max}) and for our accelerated F1-CLR schedule we use a learning rate of 0.16 (η_{max}) with 10 epoch warm-up (t_w) and train them for 120 epochs (t_{max}). For Imagenette experiments we use a batch size of 128, weight decay of 0.0005, learning rate of 0.1 (η_{max}) with Cosine Annealing scheduler and train them for 800 epochs (t_{max}). For our accelerated setting we use

[1]ImageNet subset that consist of 10 classes (tench, English springer, cassette player, chain saw, church, French horn, garbage truck, gas pump, golf ball, parachute) avalilable at `https://github.com/fastai/imagenette`

the F1-CLR schedule with learning rate of 0.2 (η_{max}) with 80 epochs for warm up (t_w) and train for 480 epochs (t_{max}). For Hard Augment we use 6 augmentations generated using SimCLR[6] transforms and scale the colour jittering linearly where the standard scale is defined as 5. For Super Progressive learning we use 96 as the minimum resolution and use 6 stages each a multiple of 32. Additional details can be seen in Supplementary Sec. B.

We have observed that GPU pre-processing plays an important role in obtaining speedups in real time. Since loading images and making multiple augmentations can become a bottleneck when training with faster GPU cards like the Nvidia V100. We use the Nvidia DALI[2] library for fast loading and augmentation of the images in the accelerators.

4.1. Backbone Architecture Analysis

In order to show the generality of our method to different different backbone architectures we applied our training strategies to SimSiam [9] model on the ImageNet dataset and report linear evaluation results and speed-up in Tab. 1.

Our results show that our method can accelerate both ResNet-18 and ResNet-50 architectures by 2.3 times while maintaining the baseline accuracy.

4.2. Model Experiments

We have evaluated our acceleration strategies on different SSL models in Tab. 2 by pre-taining on a ResNet 50 backbone on the Imagenette dataset. We used the default hyperparameters for SimSiam[9], BYOL[22], MoCov2+ [8], DINO [4] and SimCLR [6] models while we used the same optimizer and augmentation hyper-parameters for all of them which can be seen in the supplementary Sec. B.

We have observed that that our method can accelerate the training of different models. We note that methods with a

[2]available at `https://github.com/NVIDIA/DALI`

LR schedule	Sup. Prog.	Hard Aug.	kNN-Acc.(%)	Speedup
CA	-	-	90.9	1.0x
F1-CLR	-	-	90.2 (-.7)	1.7x
CA	-	✓	89.8 (-1.1)	1.8x
F1-CLR	✓	-	90.1 (-.8)	2.7x
F1-CLR	-	✓	90.3 (-.6)	1.5x
F1-CLR	✓	✓	91.4 (+.5)	2.4x

Table 3. Comparing the contribution of each component in our experimental setup. Speed up is reported based on the reduction in training cost. Using SimSiam model with ResNet50 backbone architecture on Imagenette dataset. CA stands for the baseline Cosine Annealing learning rate schedule.

momentum encoder typically have a higher accuracy and training cost even when trained with the same number of steps due to the additional forward passes made with the momentum encoder which helps to stabilize the training. Our method is especially effective for models with a momentum encoder where we can accelerate the training up to 2.7 times.

4.3. Experimental Setup

Here we compare the contribution of each component in our experimental setup in Tab. 3 training with ResNet50 backbone on the Imagenette dataset using default hyperparameters whenever possible. We have observed that our improvements are generally compatible with each other and allow us to train close to the same accuracy while reducing the training cost by 2.4 times.

While Super Progressive Learning is the main driver in reducing computational cost, Hard Augment allows us to improve performance dramatically by prioritizing useful augmentations. We note that Super Progressive Learning and Hard Augment also work well energetically since Hard Augment can select from a harder set of alternatives towards the end of the training which further boost performance.

4.4. Ablations

In the ablation studies use the Imagenette dataset with the parameters described in the previous section. We aim to identify various parameters of our model and examine its behaviour in different circumstances.

Learning Rate Schedule Comparison We compared different learning rate schedulers in isolation in order to see what percentage of improvement is attributable to the different learning rate schedulers on the ImageNet dataset in Tab. 4. We have seen that our Extended Super Convergence method still gives a reliable performance increase in this setup compared to Cosine Annealing and Cosine Annealing with linear warm up.

F1-CLR Warm Up Here we make an ablation study where we train the SimSiam model for 320 epochs with different number of warm up epoch lengths in Tab. 5. We observe

Schedule	Warm-Up	Epochs	Acc.(%)
Cosine Decay	-	200	67.7
Cosine Decay	-	100	66.1
Cosine Decay	Linear-10	100	66.0
F1-CLR (3.1)	F1-CLR (3.1)	100	66.4

Table 4. Comparing different learning rate schedules. Cosine decay with linear warm-up and Super convergence both have 10 epochs of warm-up in order to make the comparison more direct.

that 80 epochs is the optimal warm up length and we use this setting in our future experiments and ablations that utilize F1-CLR schedule.

| Warm Up	32	48	64	80	96	112
Acc(%)	86.8	86.5	86.9	**87.1**	86.8	86.6

Table 5. Super Convergence training where the number of warm up epochs is changed.

Minimum Resolution An important hyper-parameter that determines how much speed-up can be achieved with Super Progressive Learning is the minimum resolution. We made a ablation study in Tab. 6 to see the maximum speedup that can be achieved with our training strategies by increasing the minimum resolution by 32 increments starting from 64. The smallest resolution that does not result in a drop in performance is 96 which we used in our other experiments while 64 was giving better speedups it wasn't maintaining the level of performance we are seeking.

Min. Res.	Speed up	Acc.(%)
128	1.40x	86.6
96	1.60x	86.6
64	1.82x	86.1
0	3x	N/A

Table 6. Ablation study for the Super Progressive learning speedup and accuracy trade-off with different values for minimum resolution using 224 as the maximum resolution.

F1-CLR Training Length To build a better understanding of 1CLR schedule we change the length of the training
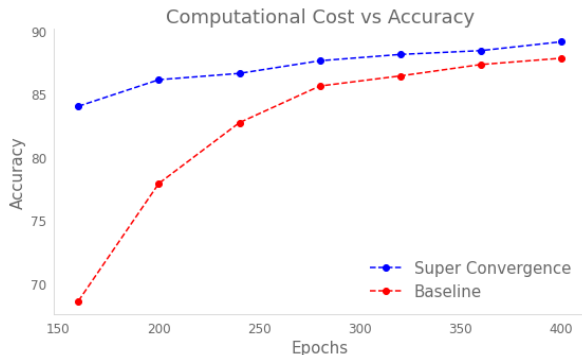
	Baseline		Super Convergence	
Epochs	Acc.(%)		Speedup	Acc.(%)
160	68.8		1.61x	84.0
200	77.9		1.53x	86.1
240	82.7		1.37x	86.6
280	85.6		1.37x	87.6
320	86.4		1.42x	88.1
360	87.3		1.41x	88.4
400	87.8		1.59x	89.1

Table 7. (left):Online validation accuracy of training instances of different lengths that are trained with F1-CLR (right): Total length of training given the fixed duration for warm-up of 80 epochs. Speedups are calculated based on linearly interpolated training length in the baseline given that the baselines has at 800 epochs has %90.0 accuracy

from 160 epochs to 800 epochs using the cosine annealing learning rate schedule and compare against how the proposed extended super convergence schedule performs. Tab. 7 shows that the difference in performance is stark when the training duration is short and gradually becomes smaller. Additionally, our F1-CLR setting leads to between 1.3x-1.6x speedup of the training to reach equivalent performance on the baseline.

Number of Positives In order to understand the effect of number of augmentations we have made an ablation study that examines the trade-off between increased overhead and the effect on accuracy in Tab. 8. We have used the SimSiam [9] model with F1-CLR schedule and our default hyperparameters. We have observed that 6 positives gives good balance between maximizing accuracy while minimizing the overhead so we use this setting in other experiments.

# of Positives	4	5	6	8
Accuracy(%)	89.1	89.0	89.6	89.8
Overhead(%)	5.4	6.8	8.2	10.9

Table 8. Ablation study for the numbers of positives used in Hard Augment. 6 augmentations result in good performance increase while maintaining reasonable overhead.

Multi-Crop Comparison Multi-crop augmentation proposed by SwAV [3] uses multiple augmentations with a combination of high resolution and low resolution crops where only high resolution crops are used for back-propagation and low resolution ones are used only as extra comparison targets. A slight modification to our Hard Augmentation method will produce an intermediate algorithm useful for a comparison where the additional low resolution augmentations that do not have the highest loss can be used as extra targets. We made an ablation study that compares the multi-crop augmentation strategy at the last line of Tab. 9 against our method and our method with extra targets. We train our

method with Extended Super Convergence for 400 epochs and used 4 augmentations for each image.

Min Res.	Selection	Extra Target	Acc(%)
64	Hard		89.1
64	Hard	✓	88.8
96	Hard	✓	89.7
96	Random	✓	88.4

Table 9. Progressive learning speedup with different values for minimum to maximum resolution fraction

We have seen that extra targets are useful when 96 resolution is used for the selection pass and hard augment with extra targets and hard augment outperforms multi-crop augmentation. We have also seen that the standard resolution of 64 that is enough for ranking loss pairs in Hard Augment is not enough for generating accurate targets.

Since increasing selection resolution has a direct effect on the overhead for Hard Augment and the additional improvement that can be obtained from extra targets is small we have decided not to include extra targets in our method.

5. Conclusion

We have demonstrated that self-supervised training can be accelerated by adapting the learning rate, augmentation and resolution schedules for self supervised training and boosting the training signal by hard positive mining on the augmentations. Our method have shows an training speed-up between 2.3 and 2.7 times. which allows a much wider community to reproduce and contribute to the self-supervised learning literature, reduce the financial and carbon cost of training these models. However there is a risk that using efficient training methods the community will adopt larger and computationally more expensive benchmarks which will eliminate some of the intended benefits of our method. As future work we aim to apply our method to recent methods [4, 10] that used Vision Transformer [15] backbones.

References

[1] Shekoofeh Azizi, Basil Mustafa, Fiona Ryan, Zachary Beaver, Jan Freyberg, Jonathan Deaton, Aaron Loh, Alan Karthikesalingam, Simon Kornblith, Ting Chen, et al. Big self-supervised models advance medical image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3478–3488, 2021.

[2] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[3] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020.

[4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, 2021.

[5] Pratik Chaudhari and Stefano Soatto. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–10. IEEE, 2018.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[7] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020.

[8] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[9] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. corr abs/2011.10566 (2020). *arXiv preprint arXiv:2011.10566*, 2020.

[10] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9640–9649, 2021.

[11] Cody Coleman, Deepak Narayanan, Daniel Kang, Tian Zhao, Jian Zhang, Luigi Nardi, Peter Bailis, Kunle Olukotun, Chris Ré, and Matei Zaharia. Dawnbench: An end-to-end deep learning benchmark and competition. *Training*, 100(101):102, 2017.

[12] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. arxiv e-prints, page. *arXiv preprint arXiv:1909.13719*, 2019.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[14] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015.

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[16] William Falcon et al. Pytorch lightning. *GitHub. Note: https://github. com/PyTorchLightning/pytorch-lightning*, 3:6, 2019.

[17] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[18] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[19] Chengyue Gong, Tongzheng Ren, Mao Ye, and Qiang Liu. Maxup: A simple way to improve generalization of neural network training. *arXiv preprint arXiv:2002.09024*, 2020.

[20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[21] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[22] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires,

Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

[23] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. arxiv 2015. *arXiv preprint arXiv:1512.03385*, 2015.

[25] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[26] Elad Hoffer, Berry Weinstein, Itay Hubara, Tal Ben-Nun, Torsten Hoefler, and Daniel Soudry. Mix & match: training convnets with mixed image sizes for improved accuracy, speed and scale resiliency. *arXiv preprint arXiv:1908.08986*, 2019.

[27] Jeremy Howard. Training imagenet in 3 hours for usd 25; and cifar10 for usd 0.26, 2018.

[28] Stanisław Jastrzębski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. Three factors influencing minima in sgd. *arXiv preprint arXiv:1711.04623*, 2017.

[29] Angela H Jiang, Daniel L-K Wong, Giulio Zhou, David G Andersen, Jeffrey Dean, Gregory R Ganger, Gauri Joshi, Michael Kaminksy, Michael Kozuch, Zachary C Lipton, et al. Accelerating deep learning by focusing on the biggest losers. *arXiv preprint arXiv:1910.00762*, 2019.

[30] Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. In *International conference on machine learning*, pages 2525–2534. PMLR, 2018.

[31] 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. *arXiv preprint arXiv:1609.04836*, 2016.

[32] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[33] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6874–6883, 2017.

[34] Aitor Lewkowycz, Yasaman Bahri, Ethan Dyer, Jascha Sohl-Dickstein, and Guy Gur-Ari. The large learning rate phase of deep learning: the catapult mechanism. *arXiv preprint arXiv:2003.02218*, 2020.

[35] Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, et al. Pytorch distributed: Experiences on accelerating data parallel training. *arXiv preprint arXiv:2006.15704*, 2020.

[36] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pages 69–84. Springer, 2016.

[37] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[38] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 761–769, 2016.

[39] LN Smith. Cyclical learning rates for training neural networks. arxiv. *Preprint at https://arxiv.org/abs/1506.01186*, 2015.

[40] Leslie N Smith. A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. *arXiv preprint arXiv:1803.09820*, 2018.

[41] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. arxiv e-prints, page. *arXiv preprint arXiv:1708.07120*, 2017.

[42] Samuel L Smith, Pieter-Jan Kindermans, Chris Ying, and Quoc V Le. Don't decay the learning rate, increase the batch size. *arXiv preprint arXiv:1711.00489*, 2017.

[43] Samuel L Smith and Quoc V Le. A bayesian perspective on generalization and stochastic gradient descent. *arXiv preprint arXiv:1710.06451*, 2017.

[44] Mingxing Tan and Quoc V Le. Efficientnetv2: Smaller models and faster training. *arXiv preprint arXiv:2104.00298*, 2021.

[45] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European conference on computer vision*, pages 776–794. Springer, 2020.

[46] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? *Advances in Neural Information Processing Systems*, 33:6827–6839, 2020.

[47] Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy: Fixefficientnet. *arXiv preprint arXiv:2003.08237*, 2020.

[48] Laurenz Wiskott and Terrence J Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural computation*, 14(4):715–770, 2002.

[49] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

[50] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.

[51] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016.

[52] Richard Zhang, Phillip Isola, and Alexei A Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1058–1067, 2017.