Exponential Moving Average Normalization for Self-supervised and Semi-supervised Learning

Zhaowei Cai, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Zhuowen Tu, Stefano Soatto
Amazon Web Services
{zhaoweic,ravinash,smmaji,fowlkec,ztu,soattos}@amazon.com

Abstract

We present a plug-in replacement for batch normalization (BN) called exponential moving average normalization (EMAN), which improves the performance of existing student-teacher based self- and semi-supervised learning techniques. Unlike the standard BN, where the statistics are computed within each batch, EMAN, used in the teacher, updates its statistics by exponential moving average from the BN statistics of the student. This design reduces the intrinsic cross-sample dependency of BN and enhances the generalization of the teacher. EMAN improves strong baselines for self-supervised learning by 4-6/1-2 points and semi-supervised learning by about 7/2 points, when 1%/10% supervised labels are available on ImageNet. These improvements are consistent across methods, network architectures, training duration, and datasets, demonstrating the general effectiveness of this technique. The code will be made available online.

1. Introduction

Supervised learning has achieved remarkable success on a variety of visual tasks, benefiting from the availability of large-scale annotated datasets such as ImageNet [37], MS-COCO [31], and ShapeNet [6]. However, in some domains such as medical imaging, large amounts of annotations are expensive or time-consuming to collect. Learning effective representations with small amounts (semi-supervised) or no (unsupervised or self-supervised) manual annotation is thus an important problem in computer vision [3, 8, 9, 17, 19, 28, 29, 39, 41, 45].

Although many choices exist for semi- and self-supervised learning [3, 15, 29, 34, 50], an effective approach is the family of student-teacher models [9, 17, 19, 22, 28, 41, 47], where the outputs of the teacher are used to guide the learning of the student on the unlabeled data. Within this family, a common approach is to update the teacher using exponential moving average (EMA) of the student parameters over its training trajectory [41], which we call EMA-teacher, as shown in Figure 1 (left). As discussed in [1, 24, 26], the temporally averaged teacher, as interpreted as the temporal ensembling of the student checkpoints, can improve generalization. Due to this property, it has been adopted in recent self-supervised learning methods [17, 19].

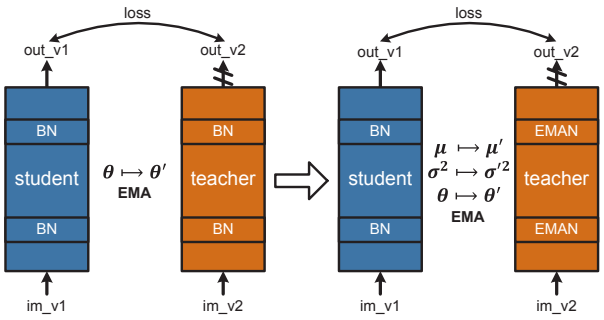


Figure 1. The EMA-teacher framework with standard BN (left) and the proposed EMAN (right). θ are the model parameters, and μ and σ^2 BN statistics. EMA denotes exponential moving average updates. `im_v1` and `im_v2` are two different views of the same image. No gradient is backpropagated through the teacher model.

While the objective and the update mechanisms are different for the student and the teacher, both networks use the standard batch normalization (BN) [25], as in the early EMA-teacher frameworks [41]. However, this can lead to two potential problems:

1. *Cross-sample dependency*. This is an intrinsic property of BN where the output of a sample is dependent on all other samples in the same batch. This cross-sample information leakage may allow the model to "cheat" in semi- or self-supervised learning. To avoid this, some special designs on normalization were applied in [8, 17, 19, 21]. For example, [21] switched to layer normalization [2]; MoCo [19] designed ShuffleBN where a mini-batch uses BN statistics from other randomly sampled mini-batch; and SimCLR [8] and BYOL [17] used Synchronized BN (SyncBN).
2. *Model parameter mismatch*. In the teacher network, its parameters are averaged from the student parameters

of previous iterations, but the batch-wise BN statistics are instantly collected at current iteration. This could lead to potential mismatch between the model parameters and the BN statistics in the parameter space.

We present a simple replacement for standard BN used in the EMA-teacher framework, called exponential moving average normalization (EMAN). As shown in Figure 1 (right), the EMAN statistics (mean μ' and variance σ'^2) in the teacher are exponentially moving averaged from the student BN statistics, similar to the other parameters. The EMAN is simply a linear transform, without batch-wise statistics computation, and thus has removed cross-sample dependency presented in BN in the teacher. Since the normalization statistics and model parameters are both updated using EMA, we expect this to improve stability of training by reducing the potential model parameter mismatches when using BN. This simple design requires only a few lines of code, and can replace other complex normalization schemes (e.g. ShuffleBN, SyncBN, etc.) within various semi- and self-supervised learning techniques.

We have evaluated EMAN within various EMA-teacher frameworks, including recent state-of-the-art semi-supervised learning (FixMatch [39]) and self-supervised learning (MoCo [19] and BYOL [17]) techniques. On self-supervised learning, EMAN improves the performance of MoCo/BYOL by 4-6/1-2 points when 1%/10% labels are available on ImageNet [37]. On semi-supervised learning, EMAN improves the performance of FixMatch by about 7/2 points for 1%/10% labels, leading to the new state-of-the-art performances of 63.0/74.0 top-1 accuracy for 1%/10% labels on ImageNet. These improvements are consistent across methods, network architectures, training duration, and datasets, demonstrating the effectiveness of EMAN as a general technique. In addition, EMAN is just as efficient as standard BN, and does not require cross-GPU communication or synchronization of ShuffleBN or SyncBN. We thus believe that EMAN can be of interest for other future student-teacher variants.

2. Related Work

Semi-supervised learning leverages unlabeled data to improve the model performance, and has a long history in machine learning [7, 51]. We primarily focus on recent deep-learning based approaches. Pseudo-Labeling [29] generates synthetic labels from the confident predictions to learn on the unlabeled data. Temporal ensembling of predictions was proposed to improve robustness in [28]. Consistency regularization based methods [28, 33, 39, 41] learn by requiring the predictions to be consistent after perturbations on inputs and/or model parameters. For example, Π-model [28] perturbs the model weights, uses dropout [40], and enforces that the clean and noisy predictions be consistent. Mean-teacher [41] proposed the EMA-teacher framework, and learns by enforcing consistency between the student and teacher models. FixMatch [39] assumes consistency between the weakly and strongly augmented inputs. A broader survey of semi-supervised learning techniques can be found in [7, 51].

Unsupervised or self-supervised learning aims to learn representations from data without annotations. It has been particularly effective in natural language processing [12, 36]. Early self-supervised learning approaches in computer vision were based on proxy tasks, e.g. solving jigsaw puzzles [34], colorization [50] and rotation prediction [15]. Recently, the contrastive learning [18] using instance discrimination has achieved promising results [8, 9, 19, 32, 42, 45]. For example, MoCo [19] and SimCLR [8, 9] have narrowed the gap between supervised and unsupervised learning in some domains. BYOL [17] found that, instead of a contrastive loss, optimizing a feature regression loss can achieve better results than prior work [8,9,19]. An extensive survey of self-supervised learning can be found in [27].

The **student-teacher framework** was first introduced in [4] and developed in [22] to distill knowledge from the pretrained teacher model to the new student model. While in [9,22], the teacher is a pretrained and frozen model, other variants are available for different purposes. For example, in [39] the teacher and the student are identical; in [38] the teacher is an ensemble of multiple networks; in [4, 9, 22] the teacher is a more complex network than the student for model compression; in [28] the teacher is a temporal ensemble of student checkpoints with the step of one epoch; in [17, 19, 41], the teacher is a more smoothly temporal ensemble than [28] by exponential moving average.

Normalization is a critical component to enable faster convergence and reduce the dependency on initialization for modern deep networks. While BN [25] is widely used, it introduces some issues, such as requiring large batch sizes for accurate statistics, and mismatch between how BN is used during training and inference. To address these, other normalization techniques have been proposed. Layer Normalization (LN) [2] normalizes along the channel and spatial dimension, Instance Normalization (IN) [43] along only the spatial dimension, and Group Normalization (GN) [44] operates similar to LN but divides the channels into groups. MABN [48] shares some similarities with our EMAN, but mainly focuses on the stability of small batch size training and updates its statistics inside a single network. In self-supervised learning, to avoid the possible information leakage via BN, [21] used LN, SimCLR [8] and BYOL [17] use SyncBN, and MoCo [19] uses ShuffleBN where a mini-batch uses BN statistics from other randomly sampled mini-batch. Although these normalization schemes work well in some specific cases, our experiments will show that they do not generalize well across various semi- and self-supervised learning methods.

3. Preliminaries

3.1. EMA-Teacher Framework

The EMA-teacher framework, with architecture shown in Figure 1 (left), was first introduced in the Mean Teacher [41], to improve the non-smooth temporal ensembling of [28]. The teacher parameters θ' are updated by exponential moving average (EMA) from the student parameters θ,

$$\theta' := m\theta' + (1-m)\theta, \tag{1}$$

where the momentum m is a number close to 1, e.g. 0.999. The student network is exactly the same as the standard supervised network, where the parameters θ are learned by standard SGD. In general, there is no gradient backpropagation through the teacher model, and the teacher model is discarded once training finished.

This EMA teacher can be interpreted as a smooth temporal ensembling of the student checkpoints along the training trajectories. As discussed in [1,24,26], this temporal weight averaging mechanism can stabilize training trajectories and present better performances than the standard SGD update. In consistency based semi- and self-supervised learning, training could be less stable [1], where the EMA-teacher framework with improved generalization can help. Due to its good performance, this EMA-teacher has derived different variants for different tasks [17, 19].

While the EMA-teacher has the special update rule for the learnable parameters, it does not for its normalization operators. Instead, the standard BN is used in both student and teacher models as in [41].

3.2. Batch Normalization

BN [25] can stabilize the learning and enable faster convergence, and thus has been widely adopted. It has different training and inference modes. During training, BN first computes the mean and the variance of the layer inputs for the current batch $\{x_i\}_{i=1}^n$,

$$\begin{aligned} \mu_{\mathcal{B}} &= \frac{1}{n}\sum_{i=1}^{n} x_i, \\ \sigma^2_{\mathcal{B}} &= \frac{1}{n}\sum_{i=1}^{n} (x_i - \mu_{\mathcal{B}})^2, \end{aligned} \tag{2}$$

where n is batch size. Next, every sample x in the current batch is normalized using the batch-wise statistics $\mu_{\mathcal{B}}$ and $\sigma^2_{\mathcal{B}}$, and then an affine transformation with learnable parameters γ and β is applied,

$$\hat{x} = BN(x) = \gamma \frac{x - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} + \beta, \tag{3}$$

where ϵ is a small constant for numerical stability.

At inference, however, it is not desirable to use the batch-wise statistics, $\mu_{\mathcal{B}}$ and $\sigma^2_{\mathcal{B}}$, since the output of an input should be deterministic and not dependent on other inputs in the same batch. The population statistics, $E[\mu]$ and $E[\sigma^2]$, should be used instead. But this requires an additional stage of statistics gathering on a large sample population, which could be undesirable. In many implementations, a more practical and efficient strategy is widely used, collecting the proxy statistics μ and σ^2 by exponential moving average during training,

$$\begin{aligned} \mu &:= \alpha\mu + (1-\alpha)\mu_{\mathcal{B}}, \\ \sigma^2 &:= \alpha\sigma^2 + (1-\alpha)\sigma^2_{\mathcal{B}}, \end{aligned} \tag{4}$$

where the momentum α here is usually 0.9. With the proxy statistics μ and σ^2, the BN at inference becomes

$$\hat{x} = BN(x) = \gamma \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta, \tag{5}$$

which differs from its training mode of (3). This practical strategy is very common in many implementations, e.g. as default in PyTorch and TensorFlow.

4. Exponential Moving Average Normalization

In the EMA-teacher framework, as introduced in Section 3.1, both the student and the teacher use the standard BN during training,

$$\begin{aligned} y &= f(BN(x), \theta), \\ y' &= f(BN(x), \theta'). \end{aligned} \tag{6}$$

where f is the intermediate layers of `relu-conv`, which takes the output of normalization as input. The standard BN is well aligned with the model parameters for a typical network (e.g. the student) which is updated by SGD, since the parameters are optimized with those batch-wise statistics. However, it is no longer the case for the teacher that is updated by EMA. Two reasons suggested that. First, the teacher is used to generate pseudo ground-truth to guide the learning of the student. With batch-wise BN, these generated pseudo labels will be cross-sample dependent, which is not desirable. For example, the pseudo label of x_1 is dependent on x_2 if x_1 and x_2 are in the same training batch. Second, there is a possible mismatch between the model parameters θ' and batch-wise BN statistics ($\mu_{\mathcal{B}}$ and $\sigma^2_{\mathcal{B}}$) in the teacher model. The former is averaged from the student parameters of previous iterations, but the latter is instantly collected at current iteration, and the former is not optimized for the latter. This mismatch could lead to non-smoothness in the parameter space.

To resolve these issues, we propose using exponential moving average normalization (EMAN) for the teacher during training (student still uses BN),

$$y' = f(EMAN(x), \theta'), \tag{7}$$

Algorithm 1 PyTorch-like Pseudocode of EMAN Update

```
# f_s, f_t: encoder networks for student and teacher

params_s = f_s.parameters() # learnable parameters
params_t = f_t.parameters() # learnable parameters
for s, t in zip(params_s, params_t):
    t = momentum*t + (1-momentum)*s

buffers_s = f_s.buffers() # BatchNorm proxy statistics
buffers_t = f_t.buffers() # BatchNorm proxy statistics
for s, t in zip(buffers_s, buffers_t):
    t = momentum*t + (1-momentum)*s
```

where

$$\hat{x} = EMAN(x) = \gamma \frac{x - \mu'}{\sqrt{\sigma'^2 + \epsilon}} + \beta, \tag{8}$$

where μ' and σ'^2 are also exponentially moving averaged from the student μ and σ^2, in the same way of (1),

$$\begin{aligned} \mu' &:= m\mu' + (1-m)\mu, \\ \sigma'^2 &:= m\sigma'^2 + (1-m)\sigma^2. \end{aligned} \tag{9}$$

The key difference between (3) and (8) is the normalization factors. They are batch-wise $\mu_\mathcal{B}$ and $\sigma^2_\mathcal{B}$ in (3), but EMA updated μ' and σ'^2 in (8). This new normalization technique for the teacher is simply a linear transform which is no longer dependent on batch statistics. EMAN eliminates cross-sample dependence in the teacher, and there is no mismatch between the model parameters (θ') and its normalization factors (μ' and σ'^2). Note that although the student is still cross-sample dependent, this is a less serious issue than the cross-sample dependency in the teacher. EMAN is better aligned with the EMA-teacher framework than the standard BN (and probably other normalization), and as we show next, it is generally applicable in different EMA-teacher variants for different tasks [17, 19, 39, 41].

4.1. Applications

We have applied EMAN to recent state-of-the-art semi-supervised learning (FixMatch [39]) and self-supervised learning (MoCo [19] and BYOL [17]) methods. Applying EMAN to these three techniques is simple, requiring a few lines of code change, as shown in Algorithm 1, where the learnable parameter update is adopted as in [17, 19, 41].

FixMatch [39] uses identical teacher and student models, with architecture shown in Figure 2 (left). The teacher generates pseudo labels after thresholding, which are then used to guide the learning of the student with standard cross-entropy loss. A tricky mechanism in FixMatch is to concatenate the strongly and weakly augmented images first and then forward them to the model together. In this case, the teacher and the student are using exactly the same BN statistics. We first reframe FixMatch in the EMA-teacher framework (with standard BN), motivated by its success. However, this change leads to much worse performance, with possible reasons discussed above in this section.

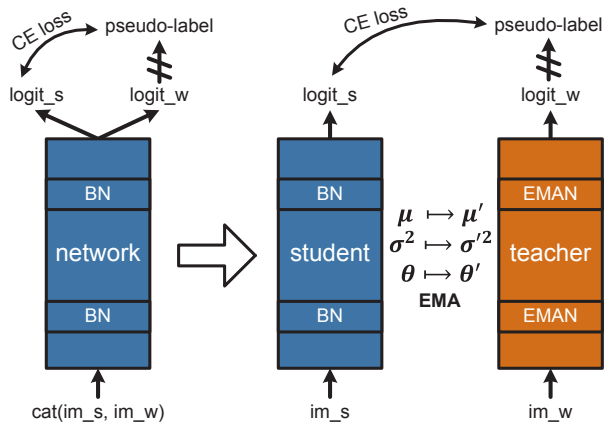
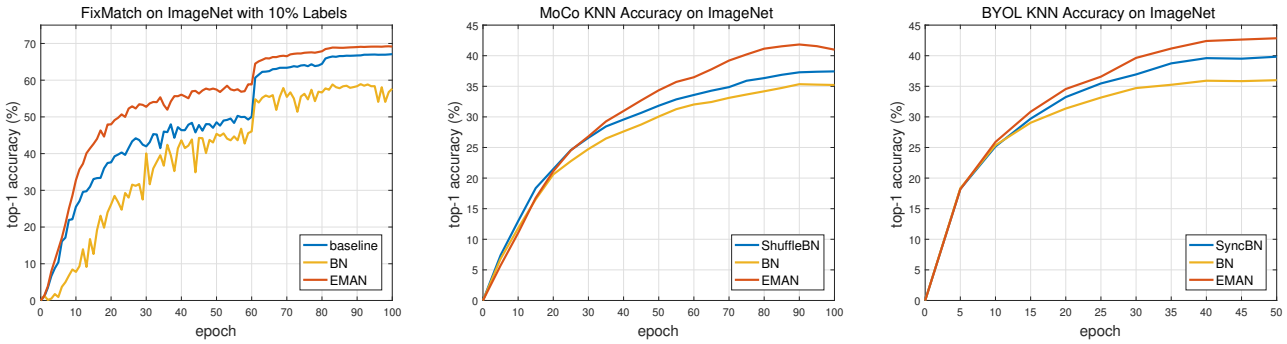


Figure 2. The architecture change of FixMatch using EMAN. `im_s`/`im_w` is the strongly/weakly augmented view of an image, `cat` concatenation. The other symbols are similar as Figure 1.

MoCo [19] has bridged the gap between supervised and unsupervised learning in multiple visual tasks. It can be interpreted as a variant of EMA-teacher, where the key (teacher) model is EMA updated from the query (student) model, and a contrastive loss is constructed between their outputs. MoCo also found it problematic to use BN in both student and teacher, due to possible information leakage. The model would probably "cheat" with local BN statistics to find a low-loss trivial solution rather than learning good representations. Instead, MoCo uses ShuffleBN in the teacher, in which the batch-wise BN statistics are computed inside a randomly shuffled mini-batch samples across distributed GPUs. This ensures that the batch statistics used to compute the query and the key come from two different subsets, avoiding the cheating issue to some extent.

BYOL [17] can also be interpreted as a EMA-teacher variant similar to MoCo, although the student/teacher is named as online/target network. It differs from the other contrast based self-supervised learning [8, 19, 45] by formulating the self-supervised learning problem as a regression task, bridging the student and teacher outputs with a simple L2 loss. [14] hypothesizes that the reason why BYOL does not need contrastive loss is BN also plays an role of implicit contrast term, not just normalization. To have stronger implicit contrast and avoid knowledge leakage, SyncBN is adopted in both student and teacher models, in which the BN statistics are collected globally across GPU cards and machines. This requires efficient synchronization technique and leads to slower training speed.

Our experiments show that using standard BN in both teacher and student models results in poor performances in all these three techniques. Although different solutions have been proposed to avoid that, e.g. Shuffle BN in MoCo and SycnBN in BYOL, they do not generalize well in other techniques as will be shown in our experiments. To have a general and simpler solution, we apply EMAN in all



(a) (b) (c)

Figure 3. The training accuracy curves of FixMatch, MoCo and BYOL on ImageNet, by using different normalization schemes.

three techniques, as in Figure 1 and 2. EMAN can improve over the standard BN by a large margin, and even surpass the ShuffleBN/SyncBN counterparts, universally in FixMatch/MoCo/BYOL. In addition, the training will be simpler and more efficient since EMAN requires no cross-GPU communication or synchronization as needed in ShuffleBN/SyncBN. We expect EMAN to be applicable to other student-teacher variants.

5. Experiments

ImageNet [37] is mainly used in all experiments, which contains ∼1.28 million images for training and 50K images for validation. The proposed EMAN has been evaluated on the state-of-the-art self-supervised learning (MoCo [19] and BYOL [17]) and semi-supervised learning (FixMatch [39]). For MoCo, the official implementation was used, but FixMatch and BYOL were reimplemented in PyTorch [35]. The default network is ResNet-50 [20] and the default hyperparameters in the corresponding papers were used, unless noted otherwise. For FixMatch, the batch size for labeled (unlabeled) images is 64 (320) with initial learning rate 0.03. For BYOL, the batch size is 512 with initial learning rate 0.9. All experiments were run on a machine with 8 V100 GPU cards. The self-supervised pretrained models were evaluated by 1) linear classification following [8, 9, 17, 19]; and 2) kNN classification with $k = 20$ following [5, 30, 45, 52], on top of the frozen representation. The other settings will be introduced in the following specific experimental sections. More experimental details can be found in the appendix.

5.1. The Effect of EMAN

The effect of the proposed EMAN was evaluated. For FixMatch, only 10% labels were used and the rest data as unlabeled. For MoCo and BYOL, we showed the accuracies of the kNN classifier along the training, since it is too expensive to train additional linear classifier. The kNN classifier used 10% `train` (50% `val`) as training set (query) for efficiency purpose (the observations are consistent with all `train`/`val` data). FixMatch/MoCo/BYOL was trained for 100/100/50 epochs, where FixMatch drops learning rate by 10 times at 60th and 80th epoch, and MoCo/BYOL uses cosine learning schedule. All training uses linear warm-up learning rate for 5 epochs.

student	teacher	FixMatch	MoCo	BYOL
default	default	67.1	54.4	55.4
BN	BN	58.9	52.5	52.0
SyncBN	SyncBN	52.0	53.3	55.4
BN	ShuffleBN	55.8	54.4	52.6
GN	GN	63.3	49.3	failed
IN	IN	61.3	46.5	failed
BN	EMAN	**69.2**	**55.8**	**56.2**

Table 1. Accuracy with different normalization.

FixMatch was reframed to the EMA-teacher framework as in Figure 2, using standard BN, denoted as "BN" in Figure 3 (a). However, this architecture change leads to much worse performance than the baseline FixMatch ("baseline"). Switching to standard BN also leads to worse performance than the baseline MoCo (ShuffleBN) and BYOL (SyncBN), as shown in Figure 3 (b) and (c). By simply changing the standard BN to the proposed EMAN in the teacher model, significant boosts are available in all FixMatch/MoCo/BYOL, e.g. roughly 10/6/7 points. This simple change also surpassed all three very strong baseline FixMatch/MoCo/BYOL by about 2/4/3 points.

To check the generalization, SyncBN and ShuffleBN were also evaluated in the other techniques, as shown in Table 1, where MoCo and BYOL were measured by linear classification on 10% labeled data. Although they work well within their own technique (i.e., ShuffleBN in MoCo and SyncBN in BYOL), they do not generalize very well across techniques. For example, SyncBN is 1.1 points worse than ShuffleBN in MoCo and even 6.9 points worse than BN in FixMatch; and ShuffleBN is 2.8 points worse than SyncBN in BYOL and even 3.1 points worse than BN in FixMatch. In contrast, EMAN generalizes very well in

	Method	1% labels		10% labels		100% labels	
		top-1	top-5	top-1	top-5	top-1	top-5
	Supervised [3, 20]	25.4	48.4	56.4	80.4	76.1	92.9
Linear	MoCo	43.2	71.0	58.8	82.6	67.5	88.1
Linear	MoCo-EMAN	48.9	75.3	60.5	83.5	67.7	88.0
Linear	MoCo (2×)	51.5	77.6	64.2	86.0	72.4	90.9
Linear	MoCo-EMAN (2×)	56.8	80.4	65.7	86.4	72.3	90.6
Linear	MoCo (800)	50.4	76.6	63.0	85.4	70.3	90.0
Linear	MoCo-EMAN (800)	55.4	79.3	64.0	85.3	70.1	89.3
Linear	BYOL	51.3	76.3	64.8	86.2	71.4	90.2
Linear	BYOL-EMAN	55.1	78.9	66.7	87.3	72.2	90.7
Finetune	MoCo	44.8	73.4	63.3	86.1	76.1	92.9
Finetune	MoCo-EMAN	50.4	77.8	64.9	87.1	76.0	93.0
Finetune	MoCo (2×)	53.1	79.9	67.9	88.6	79.2	94.6
Finetune	MoCo-EMAN (2×)	59.2	83.7	69.7	89.8	78.9	94.3
Finetune	MoCo (800)	50.9	78.1	66.3	87.7	77.2	93.6
Finetune	MoCo-EMAN (800)	57.4	82.3	68.1	88.5	77.4	93.6
Finetune	BYOL	52.1	77.3	67.7	88.5	77.0	93.5
Finetune	BYOL-EMAN	54.6	78.6	68.1	88.6	77.1	93.5

Table 2. The linear and the finetuning evaluation on ImageNet. The default model is ResNet-50 trained for 200 epochs. "2×" means ResNet-50 of 2× width and "800" means 800 epochs.

all three techniques and achieved the best results. Other cross-sample independent normalization techniques were also evaluated in Table 1, including Group Normalization (GN) [44] and Instance Normalization (IN) [43]. But they all lead to inferior performances. Also note that EMAN is as simple as BN, unlike ShuffleBN in MoCo and SyncBN in BYOL which rely on cross-GPU communication or synchronization. For example, switching SyncBN to EMAN in BYOL, the training can be speeded up by about 30% with PyTorch implementation on a machine with 8 GPUs.

5.2. Self-supervised Evaluation

We self-supervised pre-train MoCo and BYOL models with EMAN on unlabeled data and then evaluate learned representations on multiple downstream classification tasks.

Linear Classification and Finetuning The linear and finetuning evaluation were on different percentages of labeled ImageNet data, including 1%, 10% and 100%. Only the labeled data were used in these experiments. For 1% (10%) labels, five (three) different sets of samples were run and the averaged numbers are shown in Table 2. We searched the best learning rate from {15,30,60} ({0.2,0.4,0.8}) for MoCo (BYOL) linear evaluation, since they are quite sensitive in these experiments. When finetuning, we found it was important to have different learning rates for the pretrained encoder and the randomly initialized top classifier. We thus used learning rate of 1.0 (0.1) for top classifier for 1% (10%) labels, and searched the best learning rate from {0.0001,0.001,0.01} for the pretrained encoder when finetuning. All experiments were trained for 50 epochs, with learning rate dropped by 10 times at 30th and 40th epoch.

Method	Arch	Epochs	1% labels		10% labels	
			top-1	top-5	top-1	top-5
Supervised [3]	res50	100	25.4	48.4	56.4	80.4
InstDisc [45]	res50	-	-	39.2	-	77.4
PIRL [32]	res50	800	-	57.2	-	83.5
CPC v2 [21]	res161	-	52.7	77.9	73.1	91.2
MoCo-v2 [10]	res50	800	50.9	78.1	66.3	87.7
SimCLR [8]	res50	1000	48.3	75.5	65.6	87.8
PCL [30]	res50	200	-	75.6	-	86.2
SwAV [5]	res50	800	53.9	78.5	**70.2**	**89.9**
BYOL [17]	res50	1000	53.2	78.4	68.8	89.0
MoCo-EMAN	res50	800	**57.4**	**82.3**	68.1	88.5
BYOL-EMAN	res50	200	55.1	78.9	68.1	88.6

Table 3. Comparison with other self-supervised models.

For linear evaluation in Table 2, while EMAN models have comparable performances as the baselines for 100% labels, they improve over the baselines by 1-2 points of top-1 accuracy for 10% labels. The gains become bigger (4-5 points) when only 1% labels are available. The observations are consistent across different techniques (MoCo and BYOL), different architectures (ResNet-50 and ResNet-50 of 2× width), and different epochs (200 and 800). Note that the evaluation on 1%/10% labels is more practical than that on 100% labels, since when full dataset is annotated, the advantage of self-supervised pretraining will be reduced. For example, compared with supervised baseline, the self-supervised models are usually worse for 100% labels, but have significant gains (>30/10 points) for 1%/10% labels, indicating the self-supervised pretraining is much more useful when there is insufficient supervision available.

Finetuning usually achieved better results than the linear classification, in Table 2, with increasing gains for more annotations, but they could be worse if the hyperparameters are not carefully tuned as introduced above, especially for fewer labels. The gains by EMAN over those strong baselines are still consistent with the linear classification, and even larger in most of the experiments with 1% labels.

Comparison with the State-of-the-art The EMAN models were compared with the state-of-the-art self-supervised learning methods for 1%/10% labels in Table 3. To have fair comparison, only the results of ResNet-50 was shown where possible. The reported BYOL [17] was pretrained for 1000 epochs, with 53.2/68.8 top-1 accuracy for 1%/10% labels, but our BYOL-EMAN achieved 55.1/68.1, which was pretrained only for 200 epochs. Our MoCo-EMAN achieved the accuracy of 57.4 for 1% labels, which is much higher than the other methods in the table, and 68.1 for 10% labels. Note that, the comparison between these methods is not completely fair. For example, the SwAV [5], with higher accuracy for 10% labels, used much more expensive multi-crop strategy, which could also benefit our EMAN models.

Method	Epochs	kNN	retrieval	
		top-1	mAP	recall
Supervised	90	74.8	57.9	37.0
MoCo	200	54.5	32.4	18.5
MoCo-EMAN	200	58.0	39.8	24.3
MoCo	800	60.0	41.4	25.6
MoCo-EMAN	800	62.8	47.9	30.5
BYOL	200	62.8	37.5	20.1
BYOL-EMAN	200	64.9	39.8	20.4
InstDisc [45]	-	46.5	-	-
LA [52]	-	49.4	-	-
PCL [30]	200	54.5	39.5†	24.2†
SwAV [5]	800	59.2	35.9†	17.5†

Table 4. The kNN and image retrieval evaluation on ImageNet. † indicates numbers run by us from the pretrained model.

kNN Classification and Image Retrieval Although the linear classification is a common strategy to evaluate the self-supervised models in recent years [8, 17, 19], it requires additional training, which is not the most direct way to evaluate the representations. Instead, we also compared the kNN accuracies on full `train`/`val` data in Table 4, following [5, 30, 45, 52]. With this more direct evaluation, the EMAN still has consistent improvements over the MoCo and BYOL baselines. And they also outperform [45, 52] and recent PCL [30] and SwAV [5].

We also evaluate on the task of image retrieval (find the most relevant entries for each query) on ImageNet which also requires no additional training. This task is a practical application of self-supervised pretraining, since the accurate annotations are usually unavailable in many scenarios of image retrieval. We used `train` as the retrieval database and `val` as queries, and followed [49] to use the top 1000 retrievals for the evaluation of mean averaged precision (mAP) and recall. Table 4 shows the EMAN also has consistent and nontrivial improvements over the baselines for this task. The PCL [30] and SwAV [5] are compared, but they have shown much worse results.

It has also been shown that the unsupervised learning is still lagging behind supervised learning for kNN classification and image retrieval, although SwAV [5] has presented minor gap to supervised learning for linear evaluation. However, the EMAN models can learn better feature representations for these two tasks.

Low-shot Classification Given the superior performances of EMAN in the regimes of few annotations in Table 2, low-shot classification was evaluated, with k samples per class. Following [16, 30], we trained linear SVMs [11] on top of the frozen representations. We searched the best SVM cost parameter $C \in 2^{[-5,5]}$, and averaged the numbers of 5 different sets of samples.

The results in Table 5 have demonstrated that EMAN still improves the MoCo/BYOL baselines in low-shot cases,

	Method	Epochs	k=1	k=2	k=4	k=8	k=16
ImageNet	Supervised	90	46.8	57.2	64.4	68.6	71.0
	PCL [30]†	200	29.5	36.3	42.3	46.9	50.9
	SwAV [5]†	800	23.5	33.6	43.5	51.7	57.8
	MoCo	200	22.8	28.7	34.7	40.7	46.0
	MoCo-EMAN	200	29.3	36.0	41.6	46.9	50.8
	MoCo	800	31.4	38.3	44.1	49.5	53.9
	MoCo-EMAN	800	35.8	43.7	49.8	54.0	57.2
	BYOL	200	25.6	34.2	42.5	49.4	54.7
	BYOL-EMAN	200	27.4	36.8	45.6	52.6	57.5
VOC07	Supervised	90	56.0	69.6	74.9	79.9	82.7
	PCL [30]	200	47.9	59.6	66.2	74.5	78.3
	MoCo	200	47.0	58.9	65.3	72.5	76.3
	MoCo-EMAN	200	50.1	59.7	67.2	74.1	77.9
	BYOL	200	42.8	55.4	63.2	72.8	77.7
	BYOL-EMAN	200	44.6	56.5	65.4	73.9	78.8
iNaturalist	MoCo	1000	21.1	25.4	31.3	36.2	41.8
	MoCo-EMAN	1000	24.0	28.4	33.3	38.0	41.7
	BYOL	200	16.8	22.3	29.0	35.0	40.4
	BYOL-EMAN	200	18.0	23.9	30.5	36.3	41.5

Table 5. The low-shot evaluation. † indicates numbers run by us from the pretrained model.

Method	Pretrained	Schd.	1% labels		10% labels	
			top-1	top-5	top-1	top-5
baseline	None	1×	-	-	67.1	86.7
EMAN	None	1×	-	-	69.2	88.3
baseline	MoCo	1×	51.2	73.5	70.2	89.0
EMAN	MoCo	1×	58.1	80.4	72.0	90.2
EMAN	MoCo-EMAN	1×	60.9	82.5	72.6	90.5
baseline	None	3×	-	-	71.1	88.9
EMAN	None	3×	-	-	72.8	90.3
EMAN	MoCo	3×	61.4	82.1	73.9	91.0
EMAN	MoCo-EMAN	3×	63.0	83.4	74.0	90.9

Table 6. The FixMatch results on ImageNet.

as low as $k = 1$ sample per class. For example, in the experiments of ImageNet, MoCo-EMAN is about 4-6 points better than MoCo. The gains for BYOL are smaller, but still 1-3 points. Note that, our MoCo-EMAN can achieve 35.8% top-1 accuracy for 1000-way 1-shot ImageNet, which is 12.3 points higher than SwAV [5]. Pascal VOC2007 [13] and iNaturalist [23] have also been tested. Since the domain of VOC is similar to ImageNet, we directly used the frozen ImageNet representations for VOC experiments. However, it is not the case for iNaturalist, where ImageNet representations have poor performances, so we train MoCo/BYOL from scratch on iNaturalist for 1000/200 epochs. The improvements by EMAN are still consistent on both datasets.

5.3. Semi-supervised Evaluation

The semi-supervised learning experiments of FixMatch are shown in Table 6, where "1×" means training 50 (100) epochs with learning rate dropped at 30/40th (60/80th)

Method	Arch	1% labels		10% labels	
		top-1	top-5	top-1	top-5
Supervised [3]	res50	25.4	48.4	56.4	80.4
Pseudo-label [3, 29]	res50	-	51.6	-	82.4
S4L Rotation [3]	res50	-	53.4	-	83.8
UDA [46]	res50	-	-	68.8	88.5
FixMatch [39]	res50	-	-	71.5	89.1
SimCLR-v2 [9]	res50	60.0*	-	70.5*	-
FixMatch-EMAN	res50	**63.0**	**83.4**	**74.0**	**90.9**

Table 7. The comparison with other semi-supervised models. * means rough numerical estimates from the plots since no exact numbers for ResNet-50, self-distilled, were reported in [9].

student	teacher	FixMatch	MoCo	BYOL
default	default	67.1	54.4	55.4
BN	BN	58.9	52.5	52.0
BN	EMAN	69.2	55.8	56.2
BN	EMAN (m=0.9)	69.0	51.2	-
BN	EMAN (m=0.99999)	54.6	failed	-
BN	teacher PN (α=0.9)	61.4	54.6	52.4
BN	student PN (α=0.9)	68.6	52.4	failed
BN	teacher PN (α=0.999)	60.9	55.5	54.8
BN	student PN (α=0.999)	69.2	55.7	56.2

Table 8. The ablation experiments. "PN" means proxy norm, m EMAN momentum of (9) and α BN momentum of (4).

epoch, for 1% (10%) labels. For 10% labels, the top-1 accuracy is improved to 69.2 by EMAN from the baseline of 67.1. No results were reported for 1% labels since the default hyperparameters do not work very well. The default FixMatch is trained from scratch. However, as seen in Section 5.2, the self-supervised pretrained models can be a significant help for semi-supervised scenarios. Therefore, we also trained FixMatch initialized from the self-suprvised pretrained models, with initial learning rate of 0.003. When finetuned from MoCo, the 1× baseline FixMatch has 3.1 points of improvement and FixMatch-EMAN 2.8 points for 10% labels. For 1% labels, the gains by EMAN over the baseline FixMatch become bigger (∼7 points). When finetuned from MoCo-EMAN, additional gains are available, which is consistent with the observations of Section 5.2. Finally, we have trained 3× longer models with cosine learning rate schedule, and the improvements are still consistent.

Comparison with the State-of-the-art The FixMatch-EMAN models are compared with the state-of-the-art semi-supervised methods in Table 7. For 10% labels the proposed FixMatch-EMAN achieves 74.0 top-1 accuracy, beating out the original FixMatch [39] by 2.5 points. Note that this is very close to the fully supervised learning accuracy of 76.1 in Table 2. For 1% labels, the best previously reported results are SimCLR-v2 of roughly 60.0, with knowledge distillation being trained for 300 epochs after self-supervised pretraining and semi-supervised finetuning. Our FixMatch-EMAN achieved 63.0, which is about 3.0 points higher than SimCLR-v2, with simpler pipeline and fewer epochs (150). Finally, we note the specifically designed semi-supervised learning algorithms (in Table 7) outperform self-supervised pre-trainning followed by semi-superivised finetuning (in Table 3) for annotation insufficient scenarios.

5.4. Ablation Studies

The ablation experiments, with results in Table 8, followed the experimental settings of Table 1. MoCo and BYOL were evaluated by linear classification as in Section 5.2 on 10% labeled data.

EMAN Momentum We have tested different EMAN momentums of (9), but the momentum for parameter update of (1) is remained the same 0.999 as in [19, 41]. When $m = 0.9$ of EMAN, the statistics are updated much faster than the parameters, and the accuracy drops for MoCo but remains almost the same for FixMatch. When $m = 0.99999$, the statistics are updated much slower, and both MoCo and FixMatch have much worse performances. These have shown that the normalization statistics should well aligned with the parameters to ensure stable performance.

Other EMAN-similar Designs Two other designs, achieving similar goals of EMAN in Section 4, were also evaluated. Both of them use (8) in the teacher during training, but the difference is what proxy statistics μ' and σ'^2 to use. The first design is to use the collected proxy statistics of the teacher following (4) up to the previous iteration. This is similar to run the inference mode (5) during training in standard BN, but update the proxy statistics using (4) on the fly. The second design is to simply copy the proxy statistics from the student, by setting $m = 0$ in (9). They are denoted as "teacher PN" and "student PN" in Table 8, respectively. When using the default BN momentum $\alpha = 0.9$, both designs usually lead to worse performance than EMAN. By setting $\alpha = 0.999$ to have better aligned statistics with the parameters, better results are available, and the "student PN" achieved very close performances to EMAN.

6. Conclusion

In this paper, we proposed a simple normalization technique, exponential moving average normalization (EMAN), for EMA-teacher based semi- and self-supervised learning. It resolves the issues of cross-sample dependency and parameter mismatch when using the standard BN in EMA-teacher framework. This simple design improves the state of the art in semi- and self-supervised learning. These improvements are consistent across different techniques, network architectures, training duration, and datasets, showing that EMAN is generally applicable.

References

[1] Ben Athiwaratkun, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson. There are many consistent explanations of unlabeled data: Why you should average. In *ICLR*, 2019. 1, 3

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 1, 2

[3] Lucas Beyer, Xiaohua Zhai, Avital Oliver, and Alexander Kolesnikov. S4L: self-supervised semi-supervised learning. In *ICCV*, pages 1476–1485, 2019. 1, 6, 8

[4] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541, 2006. 2

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020. 5, 6, 7

[6] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv:1512.03012*, 2015. 1

[7] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006). *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009. 2

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, volume 119, pages 1597–1607, 2020. 1, 2, 4, 5, 6, 7

[9] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. In *NeurIPS*, 2020. 1, 2, 5, 8

[10] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv:2003.04297*, 2020. 6

[11] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, 1995. 7

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186, 2019. 2

[13] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010. 7

[14] Abe Fetterman and Josh Albrecht. Understanding self-supervised and contrastive learning with bootstrap your own latent (byol). https://untitled-ai.github.io/understanding-self-supervised-contrastive-learning.html, 2020. 4

[15] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018. 1, 2

[16] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In *ICCV*, pages 6390–6399, 2019. 7

[17] 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. In *NeurIPS*, 2020. 1, 2, 3, 4, 5, 6, 7

[18] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, pages 1735–1742, 2006. 2

[19] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9726–9735, 2020. 1, 2, 3, 4, 5, 7, 8

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5, 6

[21] Olivier J Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi, Carl Doersch, SM Eslami, and Aaron van den Oord. Data-efficient image recognition with contrastive predictive coding. In *ICML*, volume 119, pages 4182–4192, 2020. 1, 2, 6

[22] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015. 1, 2

[23] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alexander Shepard, Hartwig Adam, Pietro Perona, and Serge J. Belongie. The inaturalist species classification and detection dataset. In *CVPR*, pages 8769–8778, 2018. 7

[24] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E. Hopcroft, and Kilian Q. Weinberger. Snapshot ensembles: Train 1, get M for free. In *ICLR*, 2017. 1, 3

[25] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, volume 37, pages 448–456, 2015. 1, 2, 3

[26] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In Amir Globerson and Ricardo Silva, editors, *UAI*, pages 876–885. AUAI Press, 2018. 1, 3

[27] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020. 2

[28] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *ICLR*, 2017. 1, 2, 3

[29] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, 2013. 1, 2, 8

[30] Junnan Li, Pan Zhou, Caiming Xiong, Richard Socher, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*, 2020. 5, 6, 7

[31] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, volume 8693, pages 740–755, 2014. 1

[32] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *CVPR*, pages 6706–6716, 2020. 2, 6

[33] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(8):1979–1993, 2019. 2

[34] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, volume 9910, pages 69–84, 2016. 1, 2

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019. 5

[36] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018. 2

[37] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115(3):211–252, 2015. 1, 2, 5

[38] Zhiqiang Shen, Zhankui He, and Xiangyang Xue. MEAL: multi-model ensemble via adversarial learning. In *AAAI*, pages 4886–4893. AAAI Press, 2019. 2

[39] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *NeurIPS*, 2020. 1, 2, 4, 5, 8

[40] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014. 2

[41] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, pages 1195–1204, 2017. 1, 2, 3, 4, 8

[42] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *ECCV*, pages 776–794, 2020. 2

[43] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv:1607.08022*, 2016. 2, 6

[44] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, volume 11217, pages 3–19, 2018. 2, 6

[45] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, pages 3733–3742, 2018. 1, 2, 4, 5, 6, 7

[46] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. In *NeurIPS*, 2020. 8

[47] Qizhe Xie, Minh-Thang Luong, Eduard H. Hovy, and Quoc V. Le. Self-training with noisy student improves imagenet classification. In *CVPR*, pages 10684–10695. IEEE, 2020. 1

[48] Junjie Yan, Ruosi Wan, Xiangyu Zhang, Wei Zhang, Yichen Wei, and Jian Sun. Towards stabilizing batch statistics in backward propagation of batch normalization. In *ICLR*, 2020. 2

[49] Li Yuan, Tao Wang, Xiaopeng Zhang, Francis E. H. Tay, Zequn Jie, Wei Liu, and Jiashi Feng. Central similarity quantization for efficient image and video retrieval. In *CVPR*, pages 3080–3089, 2020. 7

[50] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. In *ECCV*, volume 9907, pages 649–666, 2016. 1, 2

[51] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005. 2

[52] Chengxu Zhuang, Alex Lin Zhai, and Daniel Yamins. Local aggregation for unsupervised learning of visual embeddings. In *ICCV*, pages 6001–6011. IEEE, 2019. 5, 7