MUSCLE: Strengthening Semi-Supervised Learning Via Concurrent Unsupervised Learning Using Mutual Information Maximization

Hanchen Xie[1], Mohamed E. Hussein[1,2], Aram Galstyan[1], Wael Abd-Almageed[1]
[1]USC Information Sciences Institute
[2]Alexandria University, Alexandria, Egypt
{hanchenx, mehussein, galstyan, wamageed}@isi.edu

Abstract

Deep neural networks are powerful, massively parameterized machine learning models that have been shown to perform well in supervised learning tasks. However, very large amounts of labeled data are usually needed to train deep neural networks. Several semi-supervised learning approaches have been proposed to train neural networks using smaller amounts of labeled data with a large amount of unlabeled data. The performance of these semi-supervised methods significantly degrades as the size of labeled data decreases. We introduce Mutual-information-based Unsupervised & Semi-supervised Concurrent LEarning (MUSCLE), a hybrid learning approach that uses mutual information to combine both unsupervised and semi-supervised learning. MUSCLE can be used as a stand-alone training scheme for neural networks, and can also be incorporated into other learning approaches. We show that the proposed hybrid model outperforms state of the art on several standard benchmarks, including CIFAR-10, CIFAR-100, and Mini-Imagenet. Furthermore, the performance gain consistently increases with the reduction in the amount of labeled data, as well as in the presence of bias. We also show that MUSCLE has the potential to boost the classification performance when used in the fine-tuning phase for a model pre-trained only on unlabeled data.

1. Introduction

Over the past decade, Deep Neural Networks (DNN) have been extensively employed and studied in various machine learning domains [10, 1]. DNNs have become the standard backbone for solving virtually all computer vision problems, such as image classification [26, 41, 19], object detection [35, 15, 37], image segmentation [48, 5, 6], and human motion prediction [45, 17]. However, due to their massive capacities, DNNs are infamous for requiring large amounts of labeled data.

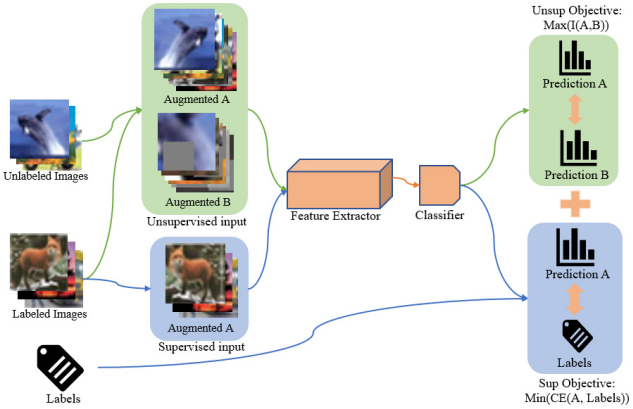


Figure 1: The Training Structure of MUSCLE

In the traditional supervised learning paradigm, large amounts of labeled data are essential for training well-performing models. To address this limitation, *few-shot adaptation* [14, 46, 42], has been studied. In this approach, using a handful of labeled data samples, a model that has been trained on a similar domain can be adapted to a new domain without compromising the performance on its original domain. While few-shot adaptation is an effective approach, the similarity between the original domain and the novel domains, and the generality of the source model – which requires a large amount of training data in the original domain – are crucial for its success. The commonly used evaluation protocols for few-shot adaptation use class-based splits of a single dataset (i.e. same domain) to create the original and novel domains [32, 46, 36, 3]. Semi-supervised learning (SSL) has been introduced [27, 44, 43] to leverage the massive amounts of available unlabeled data, instead of solely relying on labeled data.

Without loss of generality, SSL can generally be categorized into methods that use *consistency loss* [27, 44, 39, 31] and methods that use *pseudo labeling* [13, 29, 40, 11]. Although these two approaches are orthogonal, combining

them was shown to achieve better performance than each one of them individually [23, 43]. In both approaches, knowledge about the task is learned from labeled samples and transferred to unlabeled samples. In the case of pseudo labeling, unlabeled samples are explicitly labeled (using hard or soft labels), and assigned labels are used to provide supervisory signal in subsequent learning iterations. In the case of consistency loss, the label assignment or feature representation of unlabeled data is forced to be consistent across different models trained simultaneously or different variations of each sample. Despite their success, both approaches share two main weaknesses: (1) If the amount of labeled data significantly drops, i.e. knowledge about the task becomes too limited, they can either fall into degenerate solutions or fail to assign labels with high confidence to much of the unlabeled portion of the training data, and (2) Bias in the labeled portion can have a significantly negative impact on the models' performances. This is indeed a problem in all machine learning techniques. However, it can be immensely magnified when few labeled samples are available.

On the other hand, unsupervised learning (USL) techniques extract knowledge from the data without using any labels. Therefore, it is reasonable to assume that combining unsupervised and semi-supervised learning brings together the best of the two approaches. Some existing studies attempted at combining USL with Supervised Learning (SL). In such studies, USL can be used as a *pre-training* step, where the model trained via USL is either fine-tuned [21, 24] or frozen during the SL training [24]. Furthermore, in [12], which trains the network layer by layer, USL is used to train the layers and SL is used to learn the connection weights between layers. In these scenarios, the performance of the final SL model may only have a limited increment or even a drop compared to the same model trained directly on the labeled data. This is due to the possible contradiction between the USL's and the SL's training objectives, and hence, the lack of synergy between their two models during training.

In this paper, we show that concurrently using an USL objective along with a SSL model achieves better performance. We introduce Mutual-information-based Unsupervised & Semi-supervised Concurrent LEarning (MUSCLE). MUSCLE naturally involves an USL objective, which maximizes the mutual information between the predictions of variants of the same sample, from the very beginning of the training process. On one hand, when the amount of labeled data is limited, MUSCLE uses the USL objective to gain knowledge about the task. On the other hand, when there is ample labeled data, MUSCLE relies on its SSL objective, while the USL objective can work as a regularization term. MUSCLE can be used as a stand-alone SSL method, and can also be added to an existing SSL approach. We show that combining MUSCLE with three of the leading SSL approaches [43, 44, 23] consistently improves the performance on all evaluated benchmarks. Furthermore, the performance gain achieved by MUSCLE consistently increases as the amount of labeled data decreases. We also show that MUSCLE makes the SSL model less sensitive to data bias. Moreover, we show that MUSCLE can be useful in fine-tuning a model pre-trained only on unlabeled data. We provide a thorough discussion about the reasons for such combination to work, and ablation studies on different design parameters to better explain the inner working of the model.

2. Related Work

As mentioned in Section 1, there are two main approaches for semi-supervised learning based on either *consistency loss* or *pseudo labeling*.

Consistency Loss has been well studied and included in many SSL techniques [27, 44, 39, 31]. The basic form of the consistency loss can be expressed as:

$$L_{Consist} = \frac{1}{N}\sum_{i=1}^{N} l_c(f_\theta(x_i), f_{\theta'}(x'_i)) \qquad (1)$$

where f_θ is a classification function, x'_i is a variant of the input sample x_i, and l_c is a measure of divergence between $f_\theta(x_i)$ and $f_{\theta'}(x'_i)$, such as L_1 or L_2 distance [39, 44], Jensen-Shannon divergence [34], and KL-divergence. The source of the variation between x'_i and x_i can be data augmentation [43, 47], different network parameters [27, 44], or the randomness inside the network, e.g. dropout [20] or noise [31]. The basic idea of the consistency loss is that, in the absence of a ground truth label for an input sample x_i, the model ensures that variations of the same sample are consistently predicted. However, consistency loss must be accompanied with a supervised learning loss. Otherwise, we will end up with the trivial solution in which $f_\theta(x_i)$ and $f_{\theta'}(x'_i)$ take one value for all classes.

Pseudo Labeling, on the other hand, explicitly assigns labels to unlabeled data, such that the pseudo-labeled data can be used to train regular supervised learning methods, e.g. using cross entropy loss. In [4], for example, K nearest neighbors (K-NN) was used to assign labels to unlabeled samples based on their proximity to labeled samples. Then, SL, using cross entropy loss, was applied repeatedly to update the model and refine the labels until convergence. Such *hard labeling* approach provides a performance gain compared to using only supervised learning on the labeled data as it makes use of the unlabeled data. However, the gain can be limited due to the poor accuracy of the hard-assigned pseudo labels. *Soft labeling* [23] assigns confidence weights to the pseudo labels to reduce the negative impact of incorrect pseudo labels.

Completely unsupervised methods have also been suggested for classification tasks without using any labeled data [22, 24, 21, 12]. Dundar et al. [12] proposed using the k-means clustering algorithm for learning the layers and the connections between layers. Mutual information has also been used in [21, 24] for unsupervised classification.

3. Preliminaries

In this section, we explain the three leading SSL techniques, which are used as baseline for our proposed method. Beyond the basic idea of the consistency loss, the Π-Model and Temporal Ensembling [27] showed the effectiveness of updating the network parameters via Exponential Moving Average (EMA). The Mean-Teacher (MT) model [44] extended this idea by deploying two networks: the "student" and the "teacher" networks, both of which have the exact same architecture. In each training iteration, the gradient only back-propagates through the student network, and the parameters in the teacher network are updated by EMA, as shown in Equation 2

$$\theta'_t = (1-\mu) * \theta'_s + \mu * \theta_t \tag{2}$$

where θ'_t is the updated teacher parameters, θ_t is the previous teacher parameters, θ'_s is the updated student parameters, and μ is the EMA factor. The divergence between the outputs of the teacher and the student networks is minimized by minimizing the Mean Square Error (MSE) between the predictions of the two networks over training samples, as shown in Equation 3

$$L_{MT} = \frac{1}{N}\sum_{i}^{N} MSE(f_{teacher}(x_i), f_{student}(x_i)). \tag{3}$$

Label Propagation: Label propagation is a popular pseudo labeling technique, in which labels propagate from labeled samples to unlabeled samples in their proximity. In [11], label propagation was studied in the context of few shot learning. In [23, 18], label propagation was applied on SSL. Label propagation with diffusion [23] deploys a nearest-neighbor graph, which is represented using an affinity matrix. At each iteration, K unlabeled samples are selected based on their proximity to other samples and are assigned pseudo labels with confidence weights, in a process called *diffusion prediction*. We refer the reader to [23] for more details.

FixMatch: FixMatch [43] combines the consistency loss and pseudo labeling in one training strategy. As the training process progresses, the entropy of the prediction for unlabeled data decreases. Once the prediction probability of a given sample for a certain class exceeds a threshold τ, FixMatch uses that most probable class to pseudo-label the sample. Consistency loss is applied by minimizing the cross entropy between the assigned pseudo label and the prediction of a *hard-augmented* variant of the sample. The concept of *hard augmentation* is a critical component of FixMatch. Two different augmentation techniques are employed [7]: CTAugment [2], which learns the best augmentation from data, and RandAugment [8], in which the augmentation is randomly selected from a pool. FixMatch applies EMA to update the network parameters. Its initial learning rate is small compared to other state of the art methods [44, 23]. As a result, FixMatch requires 2^{20} training iterations to achieve its good performance, which is much larger than other methods.

4. Proposed Semi-supervised Learning Method

In SSL, the training dataset X is divided into two parts: X_l for the labeled data, where Y_l represents its labels, and X_u for the unlabeled data. The task is to learn features from $X = X_u \cup X_l$ leveraging Y_l. The key part of MUSCLE is involving USL from the very beginning so that we can extract meaningful features from X_u. In this section, we will first introduce the concept of MUSCLE, and then discuss its properties, functionality, and key aspects.

4.1. The Objective of MUSCLE

MUSCLE literally comes from the idea of training USL with semi-supervised or supervised Learning using Mutual Information (MI) [28] maximization, which has been proved to be useful in both *representation learning* and USL tasks [22, 21, 24]. Similar to [24], MUSCLE applies the Mutual Information Loss (MIL) to the network's likelihood prediction as shown in Equation 4

$$l_u = I(f_\theta(x_\alpha), f_\theta(x_\beta)) \tag{4}$$

where f_θ is the classification function, and x_α and x_β are the augmented data of x through transformation functions $A(x)$ and $B(x)$, respectively. The MI is calculated as shown in Equation 5 [28, 24]

$$I(z, z') = I(P) = \sum_{c=1}^{C}\sum_{c'=1}^{C} P_{cc'} \ln \frac{P_{cc'}}{P_c P_{c'}} \tag{5}$$

where

$$P = \frac{Q+Q^T}{2}, \;\; Q = \frac{1}{n}\sum_{i=1}^{N} f_\theta(x_i) \times f_\theta(x'_i)^T \tag{6}$$

where C is the number of classes, P is a $C \times C$ symmetric matrix, $P_{cc'}$ is the value at the c^{th} row and c'^{th} column of P, P_c and $P_{c'}$ are the summations over the c^{th} row and the c'^{th} column, respectively. The total loss function becomes

$$L_{MUSCLE} = l_s - \alpha l_u \tag{7}$$

where l_s can be any supervised loss from either real or pseudo labels, l_u is the MI between different outputs with

the same base sample x, and α is the factor of the MIL. From Equation 7, we can see that the loss is minimized when the MIL term αl_u is maximized. One of MUSCLE's advantages is that it could be combined with other existing SSL models or losses to mitigate their weakness instead of merely replacing them. For example, since pseudo labeling methods provide pseudo labels for supervised classification, they can be used with MUSCLE under the l_s loss term. Any consistency loss l_c can be also added to Equation 7 as:

$$L = L_{MUSCLE} + \beta l_c = l_s - \alpha l_u + \beta l_c \tag{8}$$

In this work, we use Label Propagation (LP) [23], the Mean Teacher model (MT) [44], and FixMatch [43] as base methods to highlight the advantage of combining other SSL techniques with MUSCLE.

4.2. Properties Of MUSCLE

Avoiding the Trivial Solutions: The reason that consistency loss does not have the capability of learning meaningful features from X_u without the prior knowledge generated by X_l and Y_l is that the network can simply output the same prediction for all classes of the input data, e.g. $[1, 0, \ldots, 0]$. In such a case, the consistency loss is zero but the solution is obviously meaningless. However, when maximizing the MI, this trivial solution is avoided. MI (Equation 5) can be expended to:

$$I(z, z') = H(z) - H(z|z') \tag{9}$$

where $H(z)$ is the entropy of z, or in other words, how much information z contains, and $H(z|z')$ is the conditional entropy of z given z'. Therefore, when maximizing the MI, the trivial solution is avoided because $H(z)$ is maximized when the average prediction for each class across the batch is the same. Thus, producing a fixed prediction for all samples should not maximize $H(z)$ with one exception: such fixed prediction has the same value for all classes, e.g. $[0.1, 0.1, \ldots, 0.1]$, which is avoided using $H(z|z')$. The necessary condition for minimizing $H(z|z')$ is when the samples' likelihood in z reach one-hot. Exceptions such as $[0.1, 0.1, \ldots, 0.1]$ increase $H(z|z')$ and should not exist in the optimal solution. Furthermore, the l_s term in the MUSCLE also acts as a stabilizer to MIL because l_s directly leads labeled data to meaningful predictions, and indirectly affects the unlabeled data since images within the same class are correlated with each other. The usage of maximizing the MI is also discussed in [24].

Incorporating MUSCLE into Other Approaches: By taking a closer look at the MI, we can see that maximizing the MI behaves similar to other SSL approaches with the ability of directly classifying unlabeled data X_u. It is easy to see that Maximizing the MI is an indirect method of doing pseudo-labeling since the prediction likelihood z will converge to one-hot for minimizing the second term in Equation 9, $H(z|z')$. According to Equation 1, consistency loss attempts to minimize the differences between two different predictions based on the same input data x_u without knowing its label y. For example, using the Euclidean Distance

$$d(z, z') = \sqrt{\sum (z_i - z'_i)^2} \ , \tag{10}$$

the distance between z and z' reaches its minimum value of zero if and only if $z = z'$. Using MI, both z and z' should converge to a one-hot vector. Furthermore, since z and z' are the predictions based on the same base sample x using similar or same network architectures and parameters, z and z' generally yield the same one-hot prediction due to the invariance behavior typical of DNNs. Therefore, in a sense maximizing the MI is equivalent to minimizing the consistency loss. The reason that MUSCLE can be combined with different SSL approaches is that they share common optimization goals. Thus, they can help each other for achieving those goals instead of competing with each other for different objectives.

4.3. Batch Composition

The batch for each training iteration can be expressed as: $[x_{u1}, \ldots, x_{uI}, x_{l1}, \ldots, x_{lJ}]$, where x_{ui} and x_{lj} indicate unlabeled and labeled data samples, respectively. Each batch contains I unlabeled data and J labeled data with the ratio of $r = \frac{I}{J}$. The ratio r is a critical parameter for MUSCLE, because MI attempts to predict each sample as one-hot while maintaining the predictions as a uniform distribution over the batch. Since we are randomly drawing data from the dataset, if the dataset itself is nearly balanced, then the selected data for each batch should also follow a uniform distribution over the classes. If we include J labeled data $[x_{li}, \ldots, x_{lJ}]$ in a training batch of size B, where the predictions $z_{lj} \forall j$ already converged to correct one-hot vectors based on the supervised learning term, we are revealing $\frac{J}{B}$ of correct answers to the MI term to learn the remaining samples in that batch. Therefore, r represents the balance of the batch's difficulty for MIL. A good r can prevent the batch from being overly "easy" or overly "hard". Section 5 includes an ablation study on the selection of r.

4.4. Data Augmentation

The effectiveness of the data augmentation in SSL has been well studied [31, 47, 43]. Often, only one augmentation function is used, which can be light augmentation [44, 23] or hard augmentation [24]. In [43], it was shown that using both light and hard augmentations into the consistency loss can have a much better result because it creates a larger divergence for the consistency loss to achieve better generalization. We also adopt the concept of two types of augmentation where the easy one is the classical augmentation used in [44] and the hard one is either the augmentation

Dataset	CIFAR10			
Num of Labeled Images	1000 (2%)	500 (1%)	250 (0.5%)†	100 (0.2%)†
Supervised Learning	59.97 ± 0.87	51.08 ± 1.02	45.74 ± 1.92	32.55 ± 2.02
Label Propagation [23]	77.98 ± 0.88	67.60 ± 1.80	59.55 ± 2.58	35.78 ± 3.98
Mean-Teacher [44]	80.90 ± 0.51	72.55 ± 2.64	61.26 ± 1.96	39.56 ± 1.73
LP+MT [23]	83.07 ± 0.70	75.98 ± 2.44	64.19 ± 1.95	41.62 ± 3.50
MUSCLE	85.46 ± 0.85	79.01 ± 0.99	70.37 ± 1.98	52.79 ± 4.81
MUSCLE+MT	86.42 ± 0.27	82.02 ± 0.22	75.86 ± 3.2	$\mathbf{59.57 \pm 4.53}$
MUSCLE+MT+LP	$\mathbf{86.71 \pm 0.36}$	$\mathbf{83.36 \pm 0.43}$	$\mathbf{76.46 \pm 3.05}$	59.03 ± 3.17

Table 1: Comparison with the SOTA methods on CIFAR-10 with 13-Layer CNN. Average accuracy and standard deviation are reported. The percentage of the labeled data, w.r.t the entire training dataset, is listed following the number of labels.
†Baseline results were generated by us.

Dataset	CIFAR-100				
Num of Labeled Images	10000 (20%)	4000 (8%)	2500 (5%)†	500 (1%)†	100 (0.2%)†
Supervised Learning	59.33 ± 0.49	44.57 ± 0.11	36.69 ± 0.70	16.02 ± 0.53	5.74 ± 0.64
Label Propagation [23]	61.57 ± 1.88	53.8 ± 0.76	48.84 ± 0.38	17.49 ± 1.01	4.94 ± 0.42
Mean-Teacher [44]	63.92 ± 0.51	54.64 ± 0.49	47.28 ± 0.82	20.45 ± 0.61	6.84 ± 0.89
LP+MT [23]	64.08 ± 0.47	56.27 ± 0.20	51.14 ± 0.44	21.40 ± 0.68	5.66 ± 0.84
MUSCLE+MT	$\mathbf{66.07 \pm 0.19}$	$\mathbf{59.31 \pm 0.45}$	$\mathbf{54.53 \pm 0.35}$	$\mathbf{29.86 \pm 0.85}$	$\mathbf{11.01 \pm 0.85}$
MUSCLE+MT+LP	64.79 ± 0.25	57.66 ± 0.62	52.48 ± 0.46	28.27 ± 0.73	10.51 ± 0.43

Table 2: Comparison with SOTA methods on CIFAR-100 with 13-Layer CNN. Average accuracy and standard deviation are reported. The percentage of the labeled data, w.r.t the entire training dataset, is listed following the number of labels.
†Baseline results were generated by us.

used in [24] without sobel processing or the RandAugment [8].

5. Experimental Evaluation

We first present the benchmark datasets used for the SSL evaluation, followed by the network structure and hyper-parameters settings. Then, we present a comparison between MUSCLE and the state-of-the-art methods. We also introduce a set of ablation studies on several key factors of MUSCLE. Finally, we include experiments to demonstrate an explanation for MUSCLE's main strengths.

5.1. Benchmarks Dataset

We conducted experiments on CIFAR-10 [25], CIFAR-100 [25], and Mini-Imagenet [46]. We put a special emphasis on the performance when the amount of labeled data is significantly reduced to showcase MUSCLE's clear advantage in data-starved scenarios. For example, for CIFAR-10, while experiments in [44] used a minimum of 1000 labeled data samples, and in [23] used a minimum of 500 labeled data samples, in our experiments we included evaluations on only 250 and 100 labeled data samples.

CIFAR-10 and CIFAR-100: Both CIFAR10 and CIFAR100 contain 60K of 32×32 RGB images, from 10 and 100 classes, respectively. In both datasets, all classes have the same number of samples, $\frac{1}{6}$ of which is dedicated for testing and the rest is for training. For CIFAR-10, we randomly selected 100, 50, 25, and 10 samples from each class to form the labeled dataset. For CIFAR-100, we randomly selected 100, 40, 25, 5, and 1 images from each class to form the labeled dataset. We use the rest of the training data as unlabeled samples.

Mini-Imagenet: Mini-Imagenet [46] is a subset of ImageNet [9] that contains 60K 84×84 3-channel images from 100 classes. However, different from normal classification datasets, it is split into 60-20-20 classes, where 60 classes are for training, 20 classes for validating, and 20 classes for testing. To evaluate SSL on this dataset, we followed the approach used in [23]. For each class, we randomly assigned 500 images to training and 100 images to testing. In total, we used 50K images for training and 10K images for testing. Then, we randomly selected 100, 40, and 25 images from each class to form the labeled samples, and use the rest of the training data as unlabeled samples.

5.2. Training

We implemented our method in PyTorch [33] and used the public implementations of LP [23] and the MT [44].

Dataset	Mini-ImageNet					
	Top 1 Accuracy			Top 5 Accuracy		
Num of Labeled Images	10000 (20%)	4000 (8%)	2500 (5%)†	10000 (20%)	4000 (8%)	2500 (5%)†
Supervised Learning	39.63 ± 0.53	25.62 ± 0.36	19.01 ± 0.71	61.39 ± 0.46	44.26 ± 0.45	35.48 ± 0.73
Label Propagation [23]	45.47 ± 0.47	29.71 ± 0.69	22.41 ± 0.62	69.46 ± 0.31	52.74 ± 0.49	43.13 ± 1.01
Mean-Teacher [44]	44.82 ± 0.51	28.23 ± 0.23	22.34 ± 0.52	69.38 ± 0.95	51.94 ± 0.42	44.61 ± 0.78
LP+MT [23]	45.92 ± 0.37	28.67 ± 0.41	23.38 ± 0.35	70.99 ± 0.56	52.26 ± 0.63	45.67 ± 0.86
MUSCLE	45.86 ± 0.06	34.90 ± 0.47	28.95 ± 0.45	71.26 ± 0.10	60.81 ± 0.21	54.45 ± 0.83
MUSCLE+MT	$\mathbf{49.47 \pm 0.30}$	$\mathbf{38.26 \pm 0.15}$	$\mathbf{32.56 \pm 0.12}$	$\mathbf{72.94 \pm 0.29}$	$\mathbf{63.35 \pm 0.21}$	$\mathbf{56.58 \pm 0.29}$
MUSCLE+MT+LP	47.30 ± 1.12	37.35 ± 0.25	30.86 ± 0.53	71.06 ± 0.64	$\mathbf{63.84 \pm 0.53}$	$\mathbf{56.79 \pm 0.48}$

Table 3: Comparison with the SOTA Methods on Mini-Imagenet with Resnet18. Average accuracy and standard deviation are reported. The percentage of the labeled data, w.r.t the entire training dataset, is listed following the number of labels.
†Baseline results were generated by us.

Dataset	CIFAR-10			CIFAR-100			
Num of Labeled Images	500 (1%)	250 (0.5%)	100 (0.2%)	10000 (20%)	4000 (8%)	2500 (5%)	500 (1%)
Hyper-Parameter Setting 1							
Fixmatch [43]	84.12 ± 0.55	81.96 ± 0.75	73.98 ± 1.46	53.35 ± 0.72	45.37 ± 0.56	41.02 ± 0.31	21.02 ± 0.84
MUSCLE+FixMatch	$\mathbf{84.59 \pm 0.61}$	$\mathbf{82.63 \pm 0.78}$	$\mathbf{76.06 \pm 1.81}$	$\mathbf{54.97 \pm 0.24}$	$\mathbf{47.71 \pm 0.71}$	$\mathbf{43.54 \pm 0.33}$	$\mathbf{23.73 \pm 1.67}$
Hyper-Parameter Setting 2							
Fixmatch [43]	90.53 ± 0.62	89.51 ± 0.71	81.62 ± 1.12	68.28 ± 0.19	62.28 ± 0.13	58.19 ± 0.31	32.52 ± 0.65
MUSCLE+FixMatch	$\mathbf{90.91 \pm 0.37}$	$\mathbf{90.25 \pm 0.39}$	$\mathbf{83.51 \pm 1.77}$	$\mathbf{68.51 \pm 0.23}$	$\mathbf{62.66 \pm 0.18}$	$\mathbf{58.53 \pm 0.49}$	$\mathbf{33.94 \pm 0.72}$

Table 4: Comparison with the FixMatch model on CIFAR10 and CIFAR100 datasets with 13-Layer CNN network. Two hyper-parameter settings (Section 5.2) are used. Average accuracy and standard deviation are reported. The percentage of the labeled data, w.r.t the entire training dataset, is listed following the number of labels. All results are generated by us.

SGD [38] was used to optimize all the models. We also implemented the loss function of FixMatch [43] for combining MUSCLE with FixMatch.

For CIFAR-10 and CIFAR-100, we used the 13-Layer CNN network that was used in [44, 23]. For Mini-Imagenet, we trained a Resnet18 network for the feature extractor. In a mini-batch, similar to [24], we performed hard augmentation on each image three times such that for each original image, a single weakly augmented version can be paired with three hardly augmented versions. This can increase the generality and improve the training stability.

When we compare MUSCLE with MT [44] and LP [23], we used the hyper-parameters in these methods. The network was trained over 180 epochs and the initial learning rate for MUSCLE was 0.05 for all datasets. A Cosine Learning Rate decay [30] was used to adjust the learning rate where the learning rate reaches 0 at the 210^{th} epoch. In each training batch, there are 128 images in total, including 64 labeled images. The ratio r (Section4.3) equals to 1. We followed the baselines' batch compositions and learning rate when MUSCLE is not involved.

Upon comparing MUSCLE with FixMatch [43], we noticed that FixMatch was trained on a TPU for 2^{20} iterations with a total batch size of 512 images, which is far beyond the computing resources available to us. For a fair comparison and demonstrating the potential that FixMatch can benefit from combining with MUSCLE, we evaluated FixMatch and MUSCLE with the same number of training iterations. We used two sets of training parameters: (1) the hyper-parameter settings in the MT and LP models, which were listed above. (2) the hyper-parameter settings in FixMatch with our batch composition and reduced number of iterations, such that the model is trained for 300 epochs with initial learning rate of 0.03. The learning rate is adjusted over a $\frac{7}{16}$ cycle of cosine learning rate decay. We separately listed the comparison with FixMatch to avoid confusion.

5.3. Comparison with the State of the Art

Tables 1, 2, 3, 4, and 5 compare the testing accuracy with supervised learning and the baseline methods [23, 44] with and without the use of MUSCLE. With MUSCLE, all baseline models consistently achieve better performance on all three datasets and all experimental setups. For comparing with FixMatch, the performance increases on both hyper-parameter settings for all datasets and all experimental setups. It is important to also note that the accu-

Dataset	Mini-ImageNet					
	Top 1 Accuracy			Top 5 Accuracy		
Num of Labeled Images	10000 (20%)	4000 (8%)	2500 (5%)	10000 (20%)	4000 (8%)	2500 (5%)
Hyper-Parameters Setting 1						
FixMatch [43]	26.71 ± 0.70	18.57 ± 0.32	14.12 ± 0.27	51.14 ± 1.80	39.38 ± 0.29	32.52 ± 1.29
MUSCLE+FixMatch	$\mathbf{27.33 \pm 0.19}$	$\mathbf{18.72 \pm 0.28}$	$\mathbf{14.58 \pm 0.29}$	$\mathbf{52.43 \pm 0.31}$	$\mathbf{39.83 \pm 0.65}$	$\mathbf{33.16 \pm 0.51}$
Hyper-Parameters Setting 2						
FixMatch [43]	36.05 ± 0.84	25.82 ± 0.99	19.11 ± 0.94	59.26 ± 1.03	48.61 ± 1.82	39.58 ± 1.47
MUSCLE+FixMatch	$\mathbf{37.71 \pm 0.48}$	$\mathbf{29.29 \pm 0.12}$	$\mathbf{24.73 \pm 0.68}$	$\mathbf{62.74 \pm 0.22}$	$\mathbf{53.60 \pm 0.33}$	$\mathbf{48.23 \pm 1.05}$

Table 5: Comparison with the FixMatch model on the Mini-ImangeNet dataset with Resnet18 using two hyper-parameter settings (Section 5.2). The top 1 and top 5 average accuracy and standard deviation are reported. The percentage of the labeled data w.r.t the entire training dataset is listed following the number of labels. All results are generated by us.

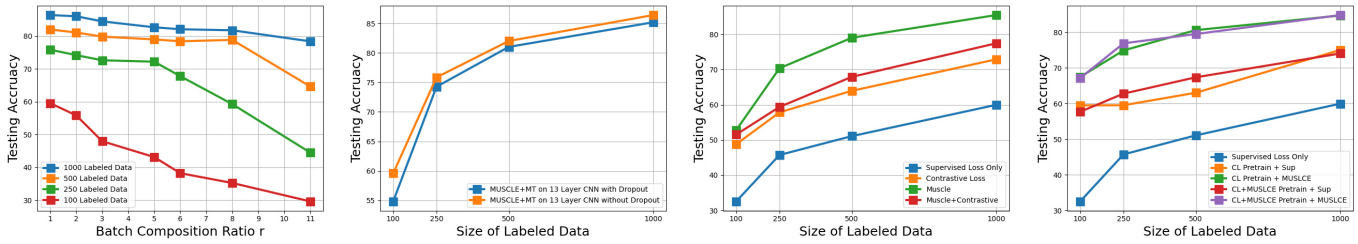
racy boost achieved with MUSCLE increases as the number of the labeled images decreases. This matches our expectations and show the advantage of MUSCLE in label-starved data scenarios. In Table 2 and Table 3, the performance of MT+MUSCLE is better than MUSCLE+MT+LP. The reason for this is that LP follows a two-stage training. The first stage trains the model without using LP for acquiring necessary preliminary knowledge about the task. Then, in the second stage, that knowledge is used to assign pseudo labels. Due to LP's properties, one hypothesis we have is that the second stage needs to start with either a highly accurate model, or a well-calibrated model [16]. However, in the case of a large number of classes (as in CIFAR-100 and Mini-ImageNet), the base model trained by MT+MUSCLE might not sufficiently satisfy either these requirements. Thus, adding another training stage with LP could be counter productive.

5.4. Ablation Study

Impact of r and Dropout Layer: We studied the impact of the ratio r, explained in Section 4.3. Figure 2 (a) shows the testing accuracy with different values of r on CIFAR-10. The performance is relatively flat for $1 < r < 2$. When r increases beyond 2, where unlabeled data amount is much larger than the labeled data amount, the accuracy decays significantly. For the Dropout Layer in the 13-layer CNN, MT [44] has provided a detailed ablation study for showing its importance to MT model. However, with MUSCLE included, as shown in Figure 2 (b), removing the Dropout layer can provide positive effect. Since most of the commonly used networks (e.g. ResNet or VGG) do not natively contain Dropout layers in the feature extractor, our method can be used with those networks without changing the architecture.

Comparison with Contrastive Loss: To show the benefit of the MIL in MUSCLE compared to simple Contrastive Loss (CL), which is commonly used in self-supervised learning, we provide two extra sets of ablation studies. First, we trained models from scratch on CIFAR10 by either replacing MIL with CL or combining MIL with CL. Second, we pre-trained models on CIFAR10 in an unsupervised learning manner by either only using CL or combining CL with MIL. Then, we fine-tuned them using either supervised learning or MUSCLE. Results are shown in Figure 2 (c) and (d), respectively. When training models from scratch, merely using CL performs better than the supervised learning baseline, but worse than either CL+MUSCLE or MUSCLE alone. This outcome is easy to understand as the CL considers each individual image as a standalone class. Even if two samples belong to the same class, the loss still attempts to push them away from each other. For the pre-training+fine-tuning experiment, the outcome shows that although adding MUSCLE to the pre-training stage hardly provides any benefit, adding MUSCLE to the fine-tuning stage clearly boosts the performance.

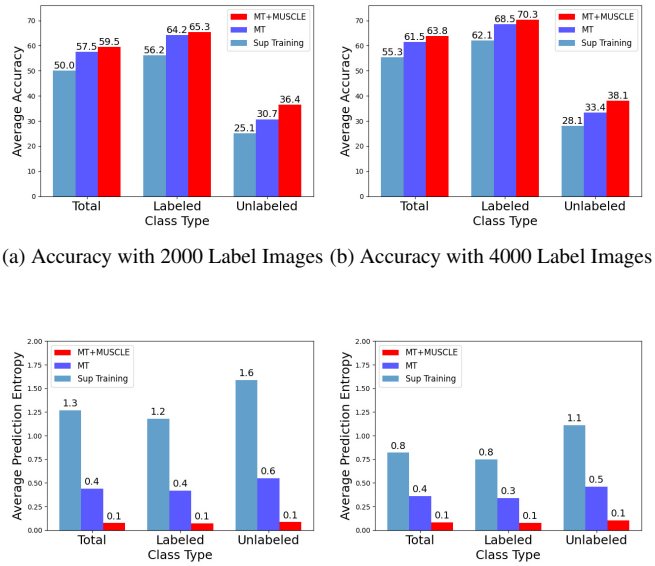
Sequestered Classes: The key point we claim for MUSCLE to work is that, compared with other SSL methods, MUSCLE can directly learn meaningful representations from unlabeled data due to involving USL early-on in the training process. Therefore, MUSCLE should have an advantage when the labeled data does not have enough samples for representing a specific class. For verifying this claim, we experimented on CIFAR-20, a hierarchical dataset based on the CIFAR-100. CIFAR-20 groups the 100 classes from CIFAR-100 into 20 super-classes, with each super-class having five sub-classes. In the same super-class, although the images from different sub-classes share some similarity, it is very hard to infer the super-class of a image based on another image from a different sub-class. For example, dolphins and otters both belong to aquatic mammals, but it is hard to classify a dolphin to aquatic mammal by only knowing otters are aquatic mammals. In this case, we introduce a new experimental setup. We randomly select a sub-class for each super-class and completely re-



(a) (b) (c) (d)

Figure 2: Ablation studies on CIFAR-10: (a) Batch composition ratio r (Section 4.3) vs. testing accuracy on MUSCLE+MT, (b) Impact of the dropout layer in the 13-layer CNN network, (c) Comparison on unsupervised learning losses when learning from scratch, (d) Comparison on unsupervised learning losses when performing pre-training followed by fine-tuning.

move all label information for that sub-class and call them *unlabeled class*. Then, we randomly select k images from the rest of the classes to form the labeled dataset and call them *labeled classes*. In other words, an *unlabeled class* will not contribute to the labeled data but they still contribute to the unlabeled data. By following this setup, we believe that the selected labeled images cannot fully represent their super-classes. In Figure 3, we can see that, compared with supervised learning and MT, MUSCLE delivers a performance boost in all three types of classes, but the majority of the performance improvement is in the unlabeled class. Furthermore, the entropy of the predictions on testing data shows that for both supervised learning method and MT model, the predictions can be affected by the class type and the amount of labeled data, whereas MUSCLE has a very constant prediction entropy across all class types and label amounts.



(a) Accuracy with 2000 Label Images (b) Accuracy with 4000 Label Images

(c) Entropy with 2000 Label Images (d) Entropy with 4000 Label Images

Figure 3: Average accuracy and prediction entropy by different class types on CIFAR-20: (a)-(b) testing accuracy, (c)-(d) average prediction entropy.

6. Conclusion

We presented Mutual-information-based Unsupervised and Supervised Concurrent LEarning (MUSCLE), which is a powerful framework for semi-supervised learning that combines the merits of leading SSL and USL techniques. In contrast to prior attempts, MUSCLE involves USL in the training process from the first iteration. MUSCLE achieved consistent improvement over the state of the art over three standard datasets, across all experimental setups. The performance boost gained by MUSCLE is maximum when the amount of training data is lowest, e.g. one sample per class for CIFAR-100. MUSCLE's power is further underscored by its extra robustness in the situation when the labeled data is biased. Finally, MUSCLE exhibited significant potential in fine-tuning pre-trained models.

7. Acknowledgements

This material is based on research sponsored by Air Force Research Laboratory (AFRL) under agreement number FA8750-19-1-1000. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation therein. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Laboratory, DARPA or the U.S. Government.

References

[1] Ossama Abdel-Hamid, Abdel-Rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. Convolutional neural networks for speech recognition. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 22(10):1533–1545, Oct. 2014.

[2] David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *International Conference on Learning Representations*, 2020.

[3] Luca Bertinetto, Joao F. Henriques, Philip Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations*, 2019.

[4] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision*, 2018.

[5] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation, 2017.

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 833–851, Cham, 2018. Springer International Publishing.

[7] E. D. Cubuk, B. Zoph, D. Mané, V. Vasudevan, and Q. V. Le. Autoaugment: Learning augmentation strategies from data. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 113–123, 2019.

[8] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Jun 2020.

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[11] M. Douze, A. Szlam, B. Hariharan, and H. Jégou. Low-shot learning with large-scale diffusion. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3349–3358, 2018.

[12] Aysegul Dundar, Jonghoon Jin, and Eugenio Culurciello. Convolutional clustering for unsupervised learning, 2015.

[13] Ismail Elezi, Alessandro Torcinovich, Sebastiano Vascon, and Marcello Pelillo. Transductive label augmentation for improved deep network learning. pages 1432–1437, 08 2018.

[14] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *ICML'17*, pages 1126–1135. JMLR.org, 2017.

[15] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[16] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1321–1330. JMLR.org, 2017.

[17] Xiao Guo and Jongmoo Choi. Human motion prediction via learning local structure representations and temporal dependencies. *AAAI*, abs/1902.07367, 2019.

[18] P. Haeusser, A. Mordvintsev, and D. Cremers. Learning by association - a versatile semi-supervised training method for neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[20] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors, 2012.

[21] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2019.

[22] Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1558–1567. JMLR.org, 2017.

[23] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[24] Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9865–9874, 2019.

[25] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[27] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *ICLR*, 2017.

[28] Erik G Learned-Miller. Entropy and mutual information.

[29] Dong-Hyun Lee. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*, 07 2013.

[30] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *international conference on learning representations*, 2017.

[31] T. Miyato, S. Maeda, M. Koyama, and S. Ishii. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1979–1993, 2019.

[32] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 721–731. Curran Associates, Inc., 2018.

[33] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[34] Siyuan Qiao, Wei Shen, Zhishuai Zhang, Bo Wang, and Alan Yuille. Deep co-training for semi-supervised image recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[35] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[36] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. Meta-learning for semi-supervised few-shot classification. In *Proceedings of 6th International Conference on Learning Representations ICLR*, 2018.

[37] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017.

[38] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[39] M. Sajjadi, M. Javanmardi, and T. Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *NIPS*, 2016.

[40] Weiwei Shi, Yihong Gong, Chris Ding, Zhiheng MaXiaoyu Tao, and Nanning Zheng. Transductive semi-supervised deep learning using min-max features. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[42] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, 2017.

[43] 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. *arXiv preprint arXiv:2001.07685*, 2020.

[44] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NIPS*, 2017.

[45] G. W. Taylor, L. Sigal, D. J. Fleet, and G. E. Hinton. Dynamical binary latent variable models for 3d human pose tracking. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 631–638, 2010.

[46] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3630–3638. Curran Associates, Inc., 2016.

[47] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019.

[48] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal. Context encoding for semantic segmentation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7151–7160, 2018.