Unsupervised Selective Labeling for More Effective Semi-Supervised Learning

Xudong Wang*[0000−0002−4973−780X], Long Lian*[0000−0001−6098−189X], and Stella X. Yu[0000−0002−3507−5761]

UC Berkeley / ICSI

A Appendix

A.1 Relationships Between Global Loss in USL-T and the K-Means Clustering Objective

Intuitively, the global loss in our proposed USL-T performs deep clustering. Furthermore, a connection can be observed between minimizing global loss and performing a generalized form of K-Means clustering, which reduces to K-Means clustering with an additional regularization term when $\tau = 0$ and the feature space is fixed.

Observation 1 *Assume that $\tau = 0$ and the feature space is fixed, minimizing L_{global} optimizes the objective of K-Means clustering with a regularization term on the inter-cluster distance that encourages additional diversity.*

Proof. Recall that we have one-hot assignment $y(x)$ and soft assignment $\hat{y}(x)$ defined as:

$$y_i(x) = \begin{cases} 1, & \text{if } i = \arg\min_{k \in \{1,...,C\}} s(f(x), c_k) \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

$$\hat{y}_i(x) = \frac{e^{s(f(x),c_i)}}{\sum_{j=1}^{C} e^{s(f(x),c_j)}} \tag{2}$$

where $c_i \in \mathbb{R}^d, k \in \{1, ..., C\}$ are learnable centroids with feature dimension d, $s(\cdot, \cdot)$ is a function that quantifies the similarity between two points in a feature space $\mathbb{R}^d$ and $f(x) \in \mathbb{R}^d$ is a function that maps an input x to a feature space, which is implemented by a CNN.

Then our global loss is defined as:

$$L_{\text{global}}(\mathcal{X}) = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} D_{\text{KL}}(y(x)||\hat{y}(x)) F(\hat{y}(x)) \tag{3}$$

* Equal contribution

When $\tau = 0$, the filtering function $F(\hat{y}(x)) = \mathbb{1}(\max(\hat{y}(x)) \geq \tau)$ has no effect. Then we can simplify our global loss and turn the loss into the following form:

$$L_{\text{global}}(\mathcal{X}) = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} D_{\text{KL}}(y(x)||\hat{y}(x)) \tag{4}$$

Since the feature space is fixed, i.e. does not change across the loss optimization, $f(x)$ remains constant, and the goal is to find the optimal centroids $\{c_i^*\}_{i=1}^C$ that minimizes the loss:

$$\{c_i^*\}_{i=1}^C = \arg\min_{\{c_i\}_{i=1}^C} L_{\text{global}}(\mathcal{X}) \tag{5}$$

We then get:

$$\begin{aligned}
\{c_i^*\}_{i=1}^C &= \arg\min_{\{c_i\}_{i=1}^C} L_{\text{global}}(\mathcal{X}) && (6)\\
&= \arg\min_{\{c_i\}_{i=1}^C} \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} D_{\text{KL}}(y(x)||\hat{y}(x)) && (7)\\
&= \arg\min_{\{c_i\}_{i=1}^C} \sum_{x \in \mathcal{X}} \sum_{i=1}^{C} -y(x) \log \frac{\hat{y}(x)}{y(x)} && (8)
\end{aligned}$$

Since $y(x)$ is a one-hot vector, we can simplify equation 8 further. Define $M(x) = \arg\min_k s(f(x), c_k)$ and $s(\cdot, \cdot) = -d(\cdot, \cdot)$ for some metric $d(\cdot, \cdot)$,

$$\begin{aligned}
\{c_i^*\}_{i=1}^C &= \arg\min_{\{c_i\}_{i=1}^C} \sum_{x \in \mathcal{X}} -\log \hat{y}(x)_{M(x)} && (9)\\
&= \arg\min_{\{c_i\}_{i=1}^C} \sum_{x \in \mathcal{X}} -\log \frac{e^{s(f(x), c_{M(x)})}}{\sum_{k=1}^C e^{s(f(x), c_k)}} && (10)\\
&= \arg\min_{\{c_i\}_{i=1}^C} \sum_{x \in \mathcal{X}} -\log e^{s(f(x), c_{M(x)})} + \log \sum_{k=1}^{C} e^{s(f(x), c_k)} && (11)\\
&= \arg\min_{\{c_i\}_{i=1}^C} \sum_{x \in \mathcal{X}} -\log e^{-d(f(x), c_{M(x)})} + \log \sum_{k=1}^{C} e^{-d(f(x), c_k)} && (12)\\
&= \arg\min_{\{c_i\}_{i=1}^C} \sum_{x \in \mathcal{X}} d(f(x), c_{M(x)}) + \log \sum_{k=1}^{C} e^{-d(f(x), c_k)} && (13)
\end{aligned}$$

If we let $d(\cdot, \cdot)$ be squared L2 distance, the expression can be decomposed into the sum of a square L2 distance with an regularization term:

$$\{c_i^*\}_{i=1}^C = \arg\min_{\{c_i\}_{i=1}^C} \ (\text{Main objective} + \text{Reg}) \tag{14}$$

where

$$\text{Main objective} = \sum_{x \in \mathcal{X}} ||x - c_{M(x)}||^2 \tag{15}$$

$$\text{Reg} = \log \sum_{k=1}^{C} e^{-d(f(x), c_k)} = \log \sum_{k=1}^{C} e^{-||f(x) - c_k||^2} \tag{16}$$

Minimizing the regularization term is equivalent to maximizing the sample's distance to all clusters $d(f(x), c_k)$, $\forall k \in \{1, ..., C\}$. This pushes apart different clusters and contributes to the diversity between clusters:

For $k \neq M(x)$, there is only force from the regularization term, which pushes apart a sample and other clusters that it does not belong to.

For $k = M(x)$, there are two forces: one from the main objective (equation 15) and one from the regularization term (equation 16). The regularization term pushes the sample away from its assigned cluster, i.e. the regularization term maximizes $d(x, c_k)$ also for $k = M(x)$, while the main objective minimizes $d(x, c_{M(x)})$.

We can quantify the net effect for $k = M(x)$ scenario. The gradient of the regularization term w.r.t $d(x, c_{M(x)})$ is $-\hat{y}(x)_{M(x)}$, and the gradient from the main objective to $d(x, c_{M(x)})$ is always 1. As $\hat{y}(x)$ is a probability distribution, $0 \leq \hat{y}(x)_{M(x)} \leq 1$. Therefore, the net effect is still minimizing $d(x, c_{M(x)})$, i.e. attracting x to its cluster center $c_{M(x)}$ and $c_{M(x)}$ to x.

Therefore, equation 16 is a regularization term aiming for additional diversity.

Now we consider the objective without the regularization term, and we define the centroids without the regularization term $\{c'_i\}_{i=1}^{C}$ as:

$$\{c'_i\}_{i=1}^{C} = \arg\min_{\{c_i\}_{i=1}^{C}} \sum_{x \in \mathcal{X}} ||x - c_{M(x)}||^2 \tag{17}$$

Since there is no interdependence between c_i and c_j, where $i \neq j$, we can define

$$\mathcal{X}'_k = \{x \in \mathcal{X} \mid M(x) = k\} \tag{18}$$

and write equation 17 as

$$\{c'_i\}_{i=1}^{C} = \arg\min_{\{c_i\}_{i=1}^{C}} \sum_{k=1}^{C} h_k \tag{19}$$

$$h_k = \sum_{x \in \mathcal{X}'_k} ||x - c_k||^2 \tag{20}$$

Then the solution to equation 19 is to minimize the individual $h_k, \forall k \in \{1, 2, ..., C\}$, i.e. the sum of squared L2 distances between a cluster and the samples that belong to it.

Without loss of generalizability, we analyze c'_1,

$$\{c_i'\}_{i=1}^{C} = \arg\min_{c_1} h_1 \tag{21}$$

$$= \arg\min_{c_1} \sum_{x \in \mathcal{X}_1'} ||x - c_1||^2 \tag{22}$$

The gradient of the objective in equation 22 w.r.t c_1 is

$$\nabla_{c_1} h_1 = -2 \sum_{x \in \mathcal{X}_1'} x - c_1 \tag{23}$$

since the objective is convex, equation 23 indicates the unique minimum is reached when

$$\sum_{x \in \mathcal{X}_1'} x - c_1 = 0 \tag{24}$$

$$c_1 = \frac{1}{|\mathcal{X}_1'|} \sum_{x \in \mathcal{X}_1'} x \tag{25}$$

This means that c_i' is the mean of all sample vectors that belong to cluster i. This indicates that equation 17 is equivalent to the objective of K-Means clustering, which aims to minimize the square L2 distance between a group of samples that are assigned to a specific cluster and the mean of this group of samples.

Therefore, $L_{\text{global}}(\mathcal{X})$ has same objective with K-Means clustering with an extra regularization term on maximizing the inter-cluster sample distances for cluster diversity.

A.2 Non-optimality of Two Types of Collapses

Observation 2 *Neither one-cluster nor even-distribution collapse is optimal to our local constraint, i.e.* $P(z(x'), \bar{z}, t) \neq \hat{y}(x)$ *for either collapse.*

Proof. Let $z(x) \in \mathbb{R}^d$ be the logits of x and $\bar{z} \in \mathbb{R}^d$ be the moving average of the batch mean of $\sigma(z(x'))$, with $\sigma(\cdot)$ as the softmax function and μ as the momentum:

$$z_k(x) = s(f(x), c_k) = f(x)^\intercal c_k \tag{26}$$

$$\bar{z} \leftarrow \mu(\frac{1}{n} \sum_{i=1}^{n} \sigma(z(x_i'))) + (1-\mu)\bar{z} \quad \text{at each iteration} \tag{27}$$

Recall that we define our anti-collapsing function $P(z, \bar{z}, t)$ with two components, as:

$$\hat{P}(z, \bar{z}) = z - \alpha \log \bar{z} \tag{28}$$

$$[P'(\hat{z}, t)]_i = \frac{\exp(\hat{z}_i/t)}{\sum_j \exp(\hat{z}_j/t)} \tag{29}$$

$$P(z, \bar{z}, t) = P'(\hat{P}(z, \bar{z}), t) \tag{30}$$

$$\tag{31}$$

where α is the adjustment factor and t is the temperature.

Then the local loss is formulated as:

$$l_{\text{local}}(x_i, x_i') = D_{\text{KL}}(P(z(x_i'), \bar{z}, t) || \hat{y}(x_i)) \tag{32}$$

$$L_{\text{local}}(\{x_i\}_{i=1}^n) = \frac{1}{n} \sum_{i=1}^{n} l_{\text{local}}(x_i, x_i') \tag{33}$$

where x_i' is a randomly picked neighbor from the k nearest neighbors of x_i.

According to Jensen's inequality, KL divergence $D_{\text{KL}}(p||q)$ only achieves optimality, with gradient norm 0, when $p = q$. To prove a solution is not optimal for $l_{\text{local}}(x, x')$, we only need to prove $P(z(x'), \bar{z}, t) \neq \hat{y}(x)$.

For one-cluster collapse, where the neural network assigns all samples to the same cluster with high confidence, both $\hat{y}(x')$ and $\hat{y}(x)$ are very close to a one-hot distribution, with $\sigma(z(x_i')) \approx \frac{1}{n} \sum_{i=1}^{n} \sigma(z(x_i')), \forall i \in \{1, 2, ..., n\}$.

Assuming that we have already taken enough iterations for the moving average to catch up in this collapsing situation, that the difference between $\bar{z}$ and $\frac{1}{n} \sum_{i=1}^{n} \sigma(z_i)$ is negligible: $\bar{z} \approx \frac{1}{n} \sum_{i=1}^{n} \sigma(z(x_i'))$. We have:

$$\hat{P}(z, \bar{z}) = z - \alpha \log \bar{z} \tag{34}$$

$$\approx z - \alpha \log \sigma(z) \tag{35}$$

$$= c\mathbf{1}_d \tag{36}$$

where c is a constant and $\mathbf{1}_d \in \mathbb{R}^d$ is a vector of 1 when $\alpha = 1$. With $\alpha > 1$, $\hat{P}(z, \bar{z})$ has an even stronger adjustment effect that pushes the target distribution even farther than uniform distribution, which is found to be beneficial in our circumstances with an optimizer that is using momentum for faster convergence.

Note that $\hat{y}(x)$ is a distribution close to one-hot, as above, we have:

$$P(z, \bar{z}, t) = P'(\hat{P}(z, \bar{z}), t) \tag{37}$$

$$\approx P'(c\mathbf{1}_d, t) \tag{38}$$

$$= \frac{1}{C}\mathbf{1}_d \tag{39}$$

$$\neq \hat{y}(x) \tag{40}$$

where C is the number of clusters.

Therefore, our loss will drive the distribution back to one that is less extreme, i.e. close to uniform distribution, and thus one-cluster collapse is not an optimum for $l_{\text{local}}(x)$.

Now we consider even distribution collapse, where samples are assigned to a distribution close to uniform distribution, with mean distribution of all samples being uniform, i.e. the mapping function $f(x)$ assigns similar logits and thus similar distributions to all clusters with small variations that are drawn from a distribution with zero-mean. Note that exact uniform distribution, where the variation exactly equals to 0, is hardly achieved in the optimization process and thus is not a concern for us.

Consider one such sample x with neighbors x'_i and predicted logits z from the batch. Here we also assume that the exponential moving average catches up, so that we have: $\bar{z} \approx \frac{1}{C}\mathbf{1}_d$. In this case, $\hat{P}(z, \bar{z})$:

$$\hat{P}(z, \bar{z}) = z - \alpha \log \bar{z} \tag{41}$$

$$\approx z - \alpha \log \frac{1}{C}\mathbf{1}_d \tag{42}$$

$$= z - \alpha \log \frac{1}{C} \tag{43}$$

Note that $P'(\hat{z}(x), t)$ is by design invariant to an additive constant on the $\hat{z}(x)$:

$$[P'(\hat{z}(x) + c, t)]_k = \frac{\exp((\hat{z}_k(x) + c)/t)}{\sum_j \exp((\hat{z}_j(x) + c)/t)} \tag{44}$$

$$= \frac{\exp(\hat{z}_k(x)/t)\exp(c/t)}{\sum_j \exp(\hat{z}_j(x)/t)\exp(c/t)} \tag{45}$$

$$= \frac{\exp(\hat{z}_k(x)/t)}{\sum_j \exp(\hat{z}_j(x)/t)} \tag{46}$$

$$= [P'(\hat{z}(x), t)]_k \tag{47}$$

Then we consider the net effect of $\hat{P}$ and P':

$$I(z(x'), \bar{z}, t) = P'(\hat{P}(z(x'), \bar{z}), t) \tag{48}$$

$$\approx P'(z(x') - \alpha \log \frac{1}{C}, t) \tag{49}$$

$$= P'(z(x'), t) \tag{50}$$

$$\neq \hat{y}(x) \tag{51}$$

The last step comes from the fact that $z(x')$ contains some variations and is not a uniform distribution. In this case, $P'(z(x'), t)$ will enlarge the dimension of z which has maximum value and make other dimension smaller in the output probability, forcing the softmax distribution to be spikier during training. Therefore, $I(z(x'), \bar{z}, t)$ will have a distribution that makes the variation more significant, driving the distribution out of mean cluster collapse.

A.3 Additional Experiment Results

Varying Budgets. Table 1 and 2 indicate the accuracy with different budget levels on SimCLRv2-CLD and FixMatch, respectively. For SimCLRv2-CLD, our method consistently outperforms not only random selection but also stratified selection for *all* the low-label settings. Our improvement is prominent especially when the number of selected samples is low. In 40 (250) labels case, we are able to achieve a 15.8% (2.7%) improvement. For FixMatch, we consistently outperform random baselines and even outperform stratified sampling, which makes use of ground truth labels of unlabeled data, in most of the settings.

Sampling Method	40 labels	100 labels	250 labels
Random	60.8	73.7	79.4
Stratified†	66.5	74.5	80.4
USL (Ours)	**76.6** ↑ **15.8**	**79.0** ↑ **5.3**	**82.1** ↑ **2.7**
USL-T (Ours)	**76.1** ↑ **15.3**	-	-

Table 1: CIFAR-10 experiments with transfer-learning based SSL method SimCLRv2-CLD [7, 25], with the mean of 5 different folds and 2 runs in each fold. †: Even though stratified selection uses more information and is not a fair comparison, we still outperform stratified selection.

	Accuracy (%)		
Sample Selection	40 labels	100 labels	250 labels
Random	82.9	88.7	93.3
Stratified†*	88.6	90.2	94.9
USL (Ours)	**90.4** ↑ **7.5**	**93.2** ↑ **4.5**	**94.0** ↑ **0.7**
USL-T (Ours)	**93.5** ↑ **10.6**	-	-

Table 2: CIFAR-10 experiments with FixMatch [23]. †: Not a fair comparison with us because it assumes balanced labeled data available and leaks information about ground truth labels. *: results from [23].

USL-T on ImageNet. We also provide experimental results of USL-T on ImageNet in Table 3. As for the hyperparams for USL-T, we use the same hyperparams as shown in the hyperparam table in the main text. For ImageNet, to create a fair comparison, USL-T model is initialized with weights of MoCov2.

ImageNet	SimCLRv2
Random	33.2
Stratified†	36.4
USL-MoCo (Ours)	39.8 ↑**6.6**
USL-CLIP (Ours)	40.4 ↑**7.2**
USL-T (Ours)	**41.3** ↑**8.1**

Table 3: Additional USL-T experiments with SimCLRv2 [7] on ImageNet. On ImageNet, USL-T also shows promising improvements, reaching a 6.6% improvement when compared to baseline. †: Although stratified selection utilizes ground truth, we still outperform it without using labeled information.

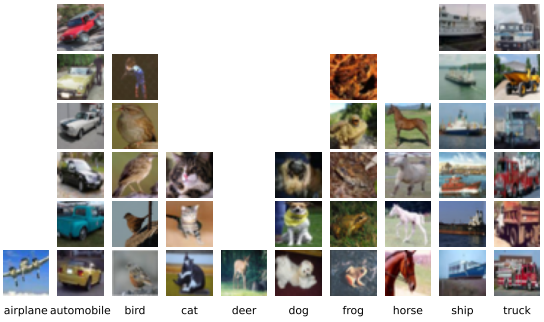
Selection Method	Accuracy
Random	77.17 ± 6.98
Stratified†	80.46 ± 7.88
USL (Ours)	**88.06** ± 1.41 ↑**10.89**

Table 4: USL shows remarkable generalizability across domains without *any* pretraining or fine-tuning on the target domain in BloodMNIST [28]. Annotated samples are chosen by a self-supervised CLD model trained on CIFAR-10 and *never exposed to medical images*. We adopt the same hyperparams as FixMatch on CIFAR-10, except that we train only for 64 epochs. Mean and standard deviation are taken over three runs. †: outperforming stratified with less information.

Cross-domain Generalizability on MedMNIST. We show USL's impressive generalizability in the main text through selective labeling with CLIP features in the ImageNet training set. Furthermore, to analyze whether USL's generalizability holds *across domains*, we use the exact same CLD model pretrained on CIFAR-10 to select samples in the BloodMNIST dataset of the MedMNISTv2 collection [28], which is a dataset in medical imaging domain. BloodMNIST contains about 18k blood cell images under microscope in 8 classes, which is drastically different from CIFAR-10's domain, but as shown in Table 4, our model with FixMatch performs 10.89% better than random sampling and 7.60% better than stratified sampling, further illustrating the possibility of a general sample selection model across image domains.

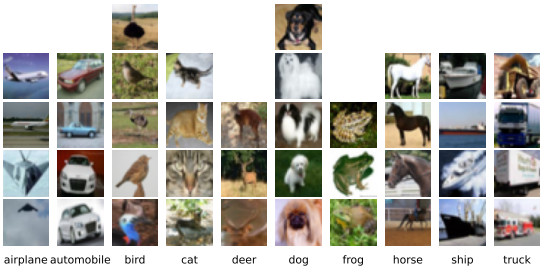
A.4 CIFAR-10 Visualizations on Selected Samples

We visualize the top-40 and least-40 of our USL and USL-T selected samples in CIFAR-10, as in Fig. 1. For clarity, we put images into buckets according to their labels. Samples from random selection are highly imbalanced in terms of semantic class distribution and coverage. Our top selected samples from USL and USL-T are representative and diverse. The representativeness could be seen from that the objects are almost always appear without any occlusion or any truncation. In contrast, the 40 samples that we are least likely to select are mainly outliers that could mislead the classifier.

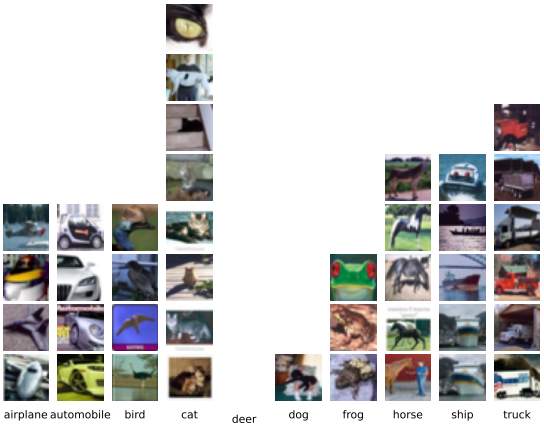


(a) Random Selection: 40 Samples

(b) Ours (USL): Top-40 Selection



(c) Ours (USL-T): Top-40 Selection



(d) Ours (USL): 40 Samples with Least Utility

Fig. 1: Visualizations of selected samples in CIFAR-10: Our selections are mostly balanced and representative. In contrast, random selection is very imbalanced and the samples that we are least likely to select are almost always outliers.

A.5 Pseudo-code for the Regularization Algorithm

We summarize the regularization algorithm in pseudo-code in Alg. 1. In Alg. 1, we first obtain $\hat{\mathbb{V}}^0$, the selection without regularization, and set the moving average regularizer $\hat{\text{Reg}}(V_i, 0)$ to 0 for every $V_i \in \mathbb{V}$; then in each iteration, we update $\hat{\text{Reg}}(V_i, t)$ with moving average from a closeness measurement to other previously selected samples, where t is the index of current iteration. We re-select samples according to regularized utility at the end of each iteration, with λ being a balancing factor. In the end, the selection from the last iteration is returned.

Algorithm 1 The iterative regularization algorithm

Require:
$\{U(V_i)|V_i \in \mathbb{V}\}$: The unregularized utility for each vertex V_i
λ: weight for applying regularization
m_{reg}: momentum in exponential moving average
l: the number of iterations
Procedure:
$\bar{\text{Reg}}(V_i, 0) \leftarrow 0, \forall V_i \in \mathbb{V}$
$\hat{\mathbb{V}}^0 \leftarrow$ samples with largest $U(V_i)$ in each cluster
for $t = 1$ **to** l **do**
  for all $V_i \in \mathbb{V}$ **do**
    $\text{Reg}(V_i, t) \leftarrow \sum_{\hat{V}_j^{t-1} \notin S_i} \frac{1}{\|V_i - \hat{V}_j^{t-1}\|^\alpha}$
    $\bar{\text{Reg}}(V_i, t) \leftarrow m_{\text{reg}} \cdot \bar{\text{Reg}}(V_i, t-1) + (1 - m_{\text{reg}}) \cdot \text{Reg}(V_i, t)$
    $U'(V_i, t) \leftarrow U(V_i) - \lambda \cdot \bar{\text{Reg}}(V_i, t)$
  end for
  $\hat{\mathbb{V}}^t \leftarrow$ samples with largest $U'(V_i, t)$ in each cluster
end for
return $\hat{\mathbb{V}}^l$

A.6 Using Euclidean Distance or Cosine Similarity?

Because the features of all instances are projected to a unit hypersphere with L2 normalization, theoretically, maximizing the cosine similarity between two nodes is equivalent to maximizing the inverse of Euclidean distance between two nodes:

$$\arg\max_{i,j}(\|f(x_i) - f(x_j)\|_2)^{-1} = \arg\max_{i,j}(2 - 2\cos(f(x_i), f(x_j)))^{-1} \tag{52}$$

$$= \arg\max_{i,j}(\cos(f(x_i), f(x_j))) \tag{53}$$

However, empirically, using maximizing the inverse of Euclidean distance $1/d(\cdot)$ as the objective function performs better than maximizing the cosine similarity $\cos(x)$. The reason is that, when two nodes are very close to each

others, $1/d(\cdot)$ is more sensitive to the change of its Euclidean distance, whereas $\cos(\cdot)$ tends to be saturated and insensitive to small changes. Therefore, the function $1/d(\cdot)$ has the desired property of non-saturating and can better focus on the distance difference with closest neighbors.

A.7 General-domain Multi-modal Models: our method on CLIP features

Although our method works well in both small and large scale datasets, there are still two interesting aspects that we would like to explore. **1)** In our approach, self-supervised models need to be re-trained for each new dataset, which is time-consuming and could potentially delay the schedule for data annotation in real-world industry. **2)** Unsupervised models do not model semantic information explicitly, which may lead to confusion that could potentially be mitigated (e.g. datasets with varying intra-class variance will take regions of different sizes and may be treated differently in an unexpected way).

To address these issues, we put our focus on a large pretrained model that encodes semantic information. Fortunately, the availability of large-scale text-image pairs online makes it possible to train a large-scale model that encodes images in the general domain with semantic information. In this paper, we make use of publicly-available CLIP [21] models, a large-scale collection of models trained on Internet-crawled data with a wide general domain and use CLIP's image model as feature extractor.

Using models trained on multi-modal datasets resolves the above issues. Even though CLIP is never trained on our target dataset, nor does the categories in its training set match the dataset we are using, using it to select does not degrade our performance of sample selection and labeling pipeline. This indicates that the effectiveness of our label selection does not necessarily depend on whether the same pretrained model is used in the downstream task. In addition, we observe that such substitution even helps with a slightly larger annotation budget, demonstrating the effectiveness of making use of semantic information. Since we only perform inference on the CLIP model, the whole sample selection process could complete in *0.5 hours* on a commodity server using one GPU, indicating the possibility of our methods without delaying the schedule of human annotation or modifying the annotation pipeline and enables it to be used by industry on real-world dataset collection.

Note that although CLIP supports zero-shot inference by using text input (e.g. class names) to generate weights for its classifier, it is not always possible to define a class with names or even know all the classes beforehand. Since we only make use of the image part of the CLIP model, we do not make use of prior text information (e.g. class descriptions) that are sometimes available in the real world. We leave better integration of our methods and zero-shot multi-modal models to future work.

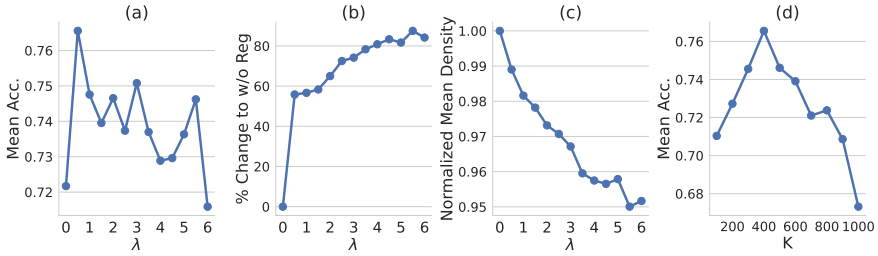


Fig. 2: Effect of different hyperparameters, λ (Fig. a,b,c) and k (Fig. d) on CIFAR-10 with SimCLRv2-CLD. λ balances representative and uniformity across the feature space. Larger λ indicates stronger regularization that pushes more selections to be different but potentially selects less individually representative samples, or vice versa. Larger k indicates that we are taking more neighbors into account when estimating the representativeness. Thanks to our stable formulation for density estimation, we found the optimal $k = 400$ on CIFAR-10 also work *consistently well* on CIFAR-100 and MedMNIST [28], indicating the hyperparameter's insensitivity to number of classes, number of images in each class, and image domains.

	Small-scale Dataset				Large-scale Dataset	
Hyperparam	CIFAR-10	CIFAR-100	MedMNIST	Hyperparam	ImageNet-100	ImageNet
k in kNN	400			k in kNN	20	
m_{reg}	0.9			Horizon	64	
α, λ	0.5, 0.5($\leq$ 100 samples) / 1.0, 1.0($>$ 100 samples)			α, λ	0.5, 1.5	
Iteration l	10					

Table 5: A list of hyperparams used in our USL experiments. The hyperparameters are slightly different for small-scale and large-scale datasets due to the introduction of regularization horizon in selective labeling in large-scale datasets. Following [23], we use different sets of hyperparameters for small-scale and large-scale datasets.

A.8 Hyperparameter Analysis

We focus on two hyperparameters in the analysis: λ, the weight for regularization, and k, the number of neighbors we use for kNN in Fig. 2. We use CIFAR-10 with SimCLRv2-CLD in a setting with a budget of 40 samples.

For hyperparam λ, we evaluated label selections with different λ values used in regularization. In the experiments, we select λ, ranging from 0 to 6 in a 0.5 increment, where 0 indicates no regularization and larger λ indicates a stronger regularization. We then evaluate the mean accuracy from 6 runs (using 2 runs per seed and 3 seeds per setting), the percent of samples that are different when compared to without regularization (i.e., $\lambda = 0$), and mean density normalized

w.r.t. without regularization. We observe that as λ gets larger, we select more different samples compared to without regularization, which indicates stronger adjustment. This comes with higher accuracy as we have more uniformity. As a trade-off, we could not sample from area which has as high density as before because selecting samples from that area leads to selections that are close to each other, leading to a high penalty. Here, uniformity and representativeness show a trade-off and the optimal choice is to balance each other at λ around 0.5. When λ is much greater than 0.5, outlier samples that are as far away as possible from other selections are chosen without considering whether the selected samples are representative, which leads to much lower accuracy.

For hyperparam k, we find that using a larger k contributes to a better representation estimation by considering more neighbors. Thanks to our formulation that considers not only the k^{th} sample for density estimation but the distance with all the k nearest neighbors, we found that our algorithm's choice for k is very generalizable: we found the optimal k for CIFAR-10 to be 400, and found that $k = 400$ also performs very well on CIFAR-100 and MedMNIST without *any* tuning, which indicates our hyperparam's insensitivity in the number of classes, number of samples per class, and the dataset domain. Similarly, for larger scale datasets with higher image resolution and lower sample noise, we find that simply set $k = 20$ leads to good performances on both 100 classes ImageNet and the full ImageNet with 1000 classes.

A.9 Additional Discussions on Related Work

Related Work About Self-supervised Learning. Self-supervised Learning learns representations transferable to downstream tasks without annotations [13, 26]. *Contrastive learning* [6, 15, 25, 26] learns representations that map similar samples or different augmentations of the same instance close and dissimilar instances apart. *Similarity-based* methods [13] learn representations without negative pairs by predicting the embedding of a target network with an online network. *Feature learning with grouping* [3, 4, 25, 27, 29, 31] respects the natural grouping of data by exploiting clusters in the latent representation. We study unlabeled data in a unsupervisedly learned feature space, due to its high quality and low feature dimensions.

We make use of the high-quality representations and dimensionality-reduction property in self-supervised learning to facilitate sample selection.

Using the representation learned with unsupervised learning as the feature space of selecting labels has two main advantages: 1) Without leveraging any labeled data, self-supervised learning could generate high-quality representations for many downstream tasks. 2) It relieves us from dealing with high-dimensional feature, due to relatively low dimension of output feature.

Related Work About Our Deep Counterpart of *k*-Means Clustering in USL-T. In USL-T, we proposed a deep counterpart of k-Means clustering method that optimizes a unified global objective, which has an effect similar to

performing k-Means clustering but trains the feature space and cluster assignment jointly. We would like to offer a comparison to main related work of our proposed method that also involves k-Means clustering variants or deep clustering designs to jointly learn features and cluster assignments.

Deep k-Means [12] proposed a differentiable metric on auto-encoder features to perform clustering. However, [12] only scales to small datasets such as MNIST, while our formulation scales to datasets with around a million images. In addition, while [12] requires a reconstruction term in the loss function to support clustering throughout training, our clustering loss, i.e. global loss, requires only one term that matches the soft and hard distribution. Note that although we also employ a local loss to kick-start the training process due to our confidence-based filtering function, the local loss could be turned off early in the training process without negative impacts on the clustering quality.

DeepCluster [3] also jointly learns features and cluster assignments with k-Means clustering. However, our work and [3] have different contributions: while our work adapts k-Means clustering to a unified loss formulation, [3] simply uses the traditional k-Means as a part of their algorithm to provide supervision for feature learning. In other words, while we directly back-propagation from our adapted k-Means algorithm as a global loss term, [3] uses traditional k-Means that does not supply gradients and employs another branch for back-propagation and learning purpose. In addition, [3] applies k-Means on features of all data, which means all feature needs to be stored prior to clustering, whereas we apply our loss formulation on the current minibatch, which adheres to popular deep learning methods that do not require storing all features from the dataset. USL-T, with end-to-end backprop to jointly solve for cluster assignments and model optimization, is much more *scalable* and *easy to implement*.

Recent works [5,8] on implementing clustering in a deep-learning framework incorporate neural networks that output a categorical distribution through a softmax operator at the end of the network. In addition, DINO [5] also considers the potential collapses and proposes a carefully-designed loss function as mitigation. However, both methods mainly intend to learn a feature space/attention map used for downstream applications instead of acquiring a set of samples that are representative and diverse. Since the feature/attention maps are the goal of designing these methods, the ~60k clusters produced by DINO are extremely sparse and highly imbalanced. For ImageNet-1K, ~90% clusters from a fully-trained DINO model are *empty* (vs ~0 in USL-T). Therefore, the user has little control over the number of selections in DINO. Empirically, we observe that SSL models optimized on them perform much worse. Furthermore, in our unsupervised selective labeling setting, these methods require *full retraining* when the downstream budget changes. In contrast, USL-T, which leverages self-supervised pretraining, could complete a selection with new budget constraint with substantially less compute.

Also recently, SCAN/NNM/RUC [10,20,24] propose image clustering methods that intend to be evaluated with hungarian matching from image clusters to semantic classes. However, such methods are compared against semi-supervised

learning methods [24] instead of being proposed to be combined with semi-supervised learning methods. First of all, these methods make use of all labels on validation split to perform hungarian matching, which implicitly makes use of all the label information. In contrast, our USL/USL-T pipeline follows the standard assumption of semi-supervised learning that no labels, except the ones in the labeled dataset, are leveraged by the method to get the final classification. Furthermore, these methods generally do not generalize well to large datasets such as ImageNet [22], with [10,20] working on smaller datasets and [24] severely underperforms on ImageNet when a very limited amount (as low as 0.2%) of data labels are available.

A.10 Overview on Unsupervised Representation Learning

In self-supervised learning stage, we aim to learn a mapping function f such that in the $f(x)$ feature space, the positive instance x_i' is attracted to instance x_i, meanwhile, the negative instance x_j (with $j \neq i$) is repelled, and we model f by a convolutional neural network, mapping x onto a d-dimensional hypersphere with L^2 normalization. To make a fair comparison with previous arts [2], we use MoCo v2 [9] to learn representations on ImageNet with the instance-centric contrastive loss:

$$C\left(f_i, f_i^+, f_{\neq i}^-\right) = \\ -\log \frac{\exp(< f_i, f_i^+ > /T)}{\exp(< f_i, f_i^+ > /T) + \sum_{j \neq i} \exp(< f_i, f_j^- > /T)} \tag{54}$$

where T is a regulating temperature. Minimizing it can be viewed as maximizing the mutual information (MI) lower bound between the features of the same instance [14,19]. For experiments on ImageNet, the MoCo model pre-trained for 800 epochs is used for initializing the SSL model, as in [2].

The feature spaces of CIFAR-10 data we work on are extracted with CLD [25]. The instance-group contrastive loss is added in symmetrical terms over views x_i and x_i':

$$\begin{aligned} L(f; T_I, T_G, \lambda) = & \sum_i (C(f_I(x_i), v_i, v_{\neq i}; T_I) \\ & + C(f_I(x_i'), v_i, v_{\neq i}; T_I)) \\ & + \lambda \sum_i (C(f_G(x_i'), M_{\Gamma(i)}, M_{\neq \Gamma(i)}; T_G) \\ & + C(f_G(x_i), M'_{\Gamma'(i)}, M'_{\Gamma'(i)}; T_G)) \end{aligned} \tag{55}$$

Cross-level discrimination of Eqn. 55 (second term) can be understood as minimizing the cross entropy between hard clustering assignment based on $f_G(x_i)$ and soft assignment predicted from $f_G(x_i')$ in a different view, where f_G (f_I) is instance (group) branch, and $M_{\Gamma(i)}$ denotes the cluster centroid of instance x_i with

a cluster id $\Gamma(i)$ [25]. Empirically, we found that CLD has great feature quality on CIFAR-10 and better respects the underlying semantic structure of data. To be consistent with original FixMatch settings, our semi-supervised learner on CIFAR-10 is trained from scratch, without using pretrained weights.

A.11 Discussions About Run Time

CLD only takes about 4 hours to train on CIFAR-10 on a single GPU and sample selection with USL takes less than 10 minutes on CLD with one GPU. This takes significantly less GPU-time than FixMatch (120 GPU hours with 4 GPUs), which is, in turn, much less than the time for labelling the whole dataset of 50000 samples. On ImageNet, MoCo takes about 12 days with 8 GPUs to achieve 800 epochs [15], our algorithm takes about an hour on one GPU to select samples for both 1% and 0.2% labels, and in the end, FixMatch takes another 20 hours on 4 GPUs to train. Although it sounds like we are using a lot of compute time just to train a self-supervised learning model for selecting what samples to annotate, the fact is that FixMatch requires a self-supervised pretrained checkpoint to work well when the number of labeled samples is low, as shown in [2], even *without* our selection methods. The only compute overhead introduced is the sample selection process, which is *negligible* when compared to the other two stages. In addition, shown in our experiments, CLIP, as a model trained on a general and diverse image-text dataset, could also be used to select samples with comparable and sometimes even better samples to label. This indicates that the self-supervised training stage is not required in our method for sample selection when a model that sufficiently covers the current domain is available.

A.12 Experiment Setup and Implementation Details

CIFAR-10/100. For ***FixMatch*** experiments, to maintain consistency with the original FixMatch [23], we evaluate FixMatch trained on CIFAR-10 with 2^{20} steps in total. To illustrate the ability of our algorithm to select informative samples, we evaluate both approaches on an extremely-low setting from 40 samples to 250 samples in total (4 shots to 25 shots per class on average). Since the original FixMatch is evaluated with stratified sampling on CIFAR-10, we also retrain FixMatch with random sampling with the same number of samples in total as a fair comparison. Unless otherwise stated, we train FixMatch with a learning rate of 0.03, and weight decay 10^{-3} on 4 Nvidia RTX 2080 Ti GPUs with batch size 64 for labeled samples and with 2^{20} steps in total. All experiments are conducted with the same training and evaluation recipe for fair comparisons.

For ***SimCLRv2-CLD***, we also evaluate our algorithm on two-stage SSL method SimCLRv2-CLD based on transfer learning [7] by fine-tuning the linear layer of a ResNet-18 pretrained with self-supervised learning algorithm CLD [25]. Specifically, we fine-tune the linear layer on a ResNet-18 trained with CLD [25]. Since it is easy for the network to overfit the few-shot labeled samples, we freeze the backbone and fine-tune only the linear layer. We use SGD with learning rate

0.01, momentum 0.9, and weight decay 10^{-4} for 5 epochs because longer training time will lead to over-fitting.

For ***MixMatch***, we train for 1024 epochs with 1024 steps per epoch, following the original recipe. For each of labeled and unlabeled dataset, we use a batch size 64. We use a learning rate 0.002 with Adam optimizer. The results are evaluated with an weighted EMA module that has decay rate 0.999 and are averaged over 20 last epochs in the test set. For ***CoMatch***, we train for 512 epochs with official code and the default recipe.

ImageNet-100/1k. We evaluate our method on ImageNet [22] with approximately 1 million images and 1000 classes and ImageNet100 [24] with 100 classes from ImageNet.

We use different sets of hyperparameters in large-scale datasets, as described in Sec. A.8. For USL, we set a finite horizon in the large datasets to make evaluation feasible. Instead of using a momentum in regularization, we run one iteration without momentum for faster selection for both USL-MoCo and USL-CLIP. For USL-T, we freeze the backbone due to computational limitations in large-scale datasets. To maintain consistency with contrastive learning, we use L^2-normed linear layer as the last layer. We also initialize the last layer with features from random samples to greatly speed up convergence. As we find that providing only one label of the sample with top confidence in each cluster does not effectively convey the grouping information in low-shot SSL, we instead query the sample with top density in each cluster and annotate the 20 samples with max density using the label of the requested sample as the pseudo-label. We reduce the iterations in downstream for fair comparison. Similar to [3], we re-initialize centroids of tail or empty clusters to the perturbed centroid of the head cluster. Since this creates centroid competitions that reduces confidence value of the head cluster, we do not make use of confidence value and calculate global loss on all samples by default.

For ***SimCLRv2*** experiments, we fine-tune the released SimCLRv2 checkpoint on baseline selections and our selections. Due to differences in codebases, our reproduced accuracy differs from the one reported on SimCLRv2 paper pre-trained and fine-tuned on Cloud TPUs. Therefore, we report our reproduced baseline which is fine-tuned on stratified selection for fair comparison on the effectiveness on sample selection with our method with SimCLRv2. Similar to other ImageNet experiments, we use 1% and 0.2% labeled data. The labeled data selection is the same for SimCLRv2 as for our experiments in FixMatch. To keep the recipe as close to the original implementation as possible, we use ResNet-50 [16] with LARS [30] optimizer with learning rate 0.16 and use globally synced batch normalization [17]. While [7] employs a batch size of 1024, we found that under the same number of training epochs, setting batch size to 512 leads to better optimization outcomes on ImageNet-1k in our codebase. This is potentially due to more iterations with the same number of training epochs. Therefore, we set batch sizes to 512 on ImageNet-1k. In addition, to reduce the memory footprint, we use mixed precision training, which has no significant im-

pacts in training accuracy in our observation. We use 60 epochs for 1% task, following [7]. We use 240 epochs for 0.2% task without learning rate decay for all selection methods, since we find this gives better results.

For ***FixMatch*** experiments, we use either a MoCo-pretrained model with Exponential Moving Average Normalization (EMAN) [2] or a CLIP ViT/16 model [11] to select samples to annotate. For ImageNet 1%, we run K-Means clustering with 12900 clusters, which is slightly more than 12820 samples we are selecting, because we observe that there will sometimes be empty clusters. To maintain consistency with prior works, we use the same setting as in [2] besides the selection of input labeled data, unless otherwise stated. Specifically we use a learning rate of 0.03 with weight decay 10^{-4} and train a ResNet-50 for 50 epochs with a MoCo [15] model as pretrained model. We perform learning rate warmup for 5 epochs and decay the learning rate by 0.1 at 30 and 40 epochs. Note that we load MoCo model as the pretrained model for FixMatch for fair comparison so that the only difference between MoCo and CLIP setting is the sample selection.

A.13 Details About the Toolbox

Currently, different SSL/AL/SSAL implementations use different formats to represent what samples to label, making selective labeling methods hard to benchmark. Therefore, to standardize the benchmark, we intend to release a toolbox that includes implementations of following methods:

- Our selective labeling methods: USL and USL-T
- SSL methods that we experimented on, including SimCLRv2 [7], SimCLRv2-CLD [7, 25], FixMatch [23], CoMatch [18], MixMatch [1], that are adapted with the unified dataset representation as illustrated below
- Several AL/SSAL methods that we use as baselines

USL, USL-T, SSL methods, and the AL/SSAL baselines in the toolbox are implemented with *unified* data loaders that comes with *standard* and simple file formats to indicate what samples are requested to be labeled and what samples are unlabeled. We provide out-of-the-box data loaders that use this unified file representation for datasets used in our experiments. In addition, the training recipe will be provided for the methods mentioned above to facilitate future research and fair comparisons.

References

1. Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., Raffel, C.: Mixmatch: A holistic approach to semi-supervised learning. arXiv preprint arXiv:1905.02249 (2019) 19
2. Cai, Z., Ravichandran, A., Maji, S., Fowlkes, C., Tu, Z., Soatto, S.: Exponential moving average normalization for self-supervised and semi-supervised learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 194–203 (2021) 16, 17, 19
3. Caron, M., Bojanowski, P., Joulin, A., Douze, M.: Deep clustering for unsupervised learning of visual features. In: ECCV (2018) 14, 15, 18
4. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. Advances in Neural Information Processing Systems **33** (2020) 14
5. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9650–9660 (2021) 15
6. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning. pp. 1597–1607. PMLR (2020) 14
7. Chen, T., Kornblith, S., Swersky, K., Norouzi, M., Hinton, G.: Big self-supervised models are strong semi-supervised learners. arXiv preprint arXiv:2006.10029 (2020) 7, 8, 17, 18, 19
8. Chen, W., Pu, S., Xie, D., Yang, S., Guo, Y., Lin, L.: Unsupervised image classification for deep representation learning. In: European Conference on Computer Vision. pp. 430–446. Springer (2020) 15
9. Chen, X., Fan, H., Girshick, R., He, K.: Improved baselines with momentum contrastive learning. arXiv preprint arXiv:2003.04297 (2020) 16
10. Dang, Z., Deng, C., Yang, X., Wei, K., Huang, H.: Nearest neighbor matching for deep clustering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13693–13702 (2021) 15, 16
11. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020) 19
12. Fard, M.M., Thonet, T., Gaussier, E.: Deep k-means: Jointly clustering with k-means and learning representations. Pattern Recognition Letters **138**, 185–192 (2020) 15
13. Grill, J.B., Strub, F., Altché, F., Tallec, C., Richemond, P.H., Buchatskaya, E., Doersch, C., Pires, B.A., Guo, Z.D., Azar, M.G., et al.: Bootstrap your own latent: A new approach to self-supervised learning. arXiv preprint arXiv:2006.07733 (2020) 14
14. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: CVPR (2006) 16
15. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9729–9738 (2020) 14, 17, 19
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 18

17. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pp. 448–456. PMLR (2015) 18
18. Li, J., Xiong, C., Hoi, S.C.: Comatch: Semi-supervised learning with contrastive graph regularization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9475–9484 (2021) 19
19. Oord, A.v.d., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018) 16
20. Park, S., Han, S., Kim, S., Kim, D., Park, S., Hong, S., Cha, M.: Improving unsupervised image clustering with robust learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12278–12287 (2021) 15, 16
21. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. arXiv preprint arXiv:2103.00020 (2021) 12
22. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y 16, 18
23. Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Raffel, C.A., Cubuk, E.D., Kurakin, A., Li, C.L.: Fixmatch: Simplifying semi-supervised learning with consistency and confidence. Advances in Neural Information Processing Systems **33** (2020) 7, 13, 17, 19
24. Van Gansbeke, W., Vandenhende, S., Georgoulis, S., Proesmans, M., Van Gool, L.: Scan: Learning to classify images without labels. In: European Conference on Computer Vision. pp. 268–285. Springer (2020) 15, 16, 18
25. Wang, X., Liu, Z., Yu, S.X.: Unsupervised feature learning by cross-level instance-group discrimination. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12586–12595 (2021) 7, 14, 16, 17, 19
26. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3733–3742 (2018) 14
27. Xie, J., Girshick, R., Farhadi, A.: Unsupervised deep embedding for clustering analysis. In: ICML (2016) 14
28. Yang, J., Shi, R., Wei, D., Liu, Z., Zhao, L., Ke, B., Pfister, H., Ni, B.: Medmnist v2: A large-scale lightweight benchmark for 2d and 3d biomedical image classification. arXiv preprint arXiv:2110.14795 (2021) 8, 13
29. Yang, Y., Xu, D., Nie, F., Yan, S., Zhuang, Y.: Image clustering using local discriminant models and global integration. TIP (2010) 14
30. You, Y., Gitman, I., Ginsburg, B.: Large batch training of convolutional networks. arXiv preprint arXiv:1708.03888 (2017) 18
31. Zhuang, C., Zhai, A.L., Yamins, D., , et al.: Local aggregation for unsupervised learning of visual embeddings. In: ICCV (2019) 14