SSL++: Improving Self-supervised Learning by Mitigating the Proxy Task-Specificity Problem

Song Chen, Jing-Hao Xue, Jianlong Chang, Jianzhong Zhang, Jufeng Yang, Qi Tian, *Fellow, IEEE*

***Abstract*—The success of deep convolutional networks (ConvNets) generally relies on a massive amount of well-labeled data, which is labor-intensive and time-consuming to collect and annotate in many scenarios. To eliminate such limitation, self-supervised learning (SSL) is recently proposed. Specifically, by solving a pre-designed proxy task, SSL is capable of capturing general-purpose features without requiring human supervision. Existing efforts focus obsessively on designing a particular proxy task but ignore the semanticity of samples that are advantageous to downstream tasks, resulting in the inherent limitation that the learned features are specific to the proxy task, namely the proxy task-specificity of features. In this work, to improve the generalizability of features learned by existing SSL methods, we present a novel self-supervised framework SSL++ to incorporate the proxy task-independent semanticity of samples into the representation learning process. Technically, SSL++ aims to leverage the complementarity, between the low-level generic features learned by a proxy task and the high-level semantic features newly learned by the generated semantic pseudo-labels, to mitigate the task-specificity and improve the generalizability of features. Extensive experiments show that SSL++ performs favorably against the state-of-the-art approaches on the established and latest SSL benchmarks. The code will be available to the public.**

***Index Terms*—Proxy task-specificity, self-supervised learning, representation learning, convolutional neural networks.**

I. Introduction

TODAY, because of smartphones, we are taking more photos than ever before. It is estimated that trillions of images were captured globally last year. However, current fully supervised learning paradigm is incapable of handling such a gigantic amount of unannotated images. To overcome the bottleneck of labor-intensive and time-consuming data annotation, a straightforward way is to employ the self-supervised learning (SSL) paradigm [1], [2]. Specifically, a model is trained to solve a well-designed proxy task, in which the supervision signal can be easily generated as free pseudo-labels, and the features that encode favorable auxiliary visual information can be learned [3], [4], essentially as an information encoding process. In this process, the pseudo-label underpins the success of feature learning and stimulates research on approaches to designing various proxy tasks for generating informative pseudo-labels.

S. Chen, J. Zhang, and J. Yang are with the College of Computer Science, Nankai University, Tianjin 300350, China (e-mail: songcheney@mail.nankai.edu.cn; zhangjz@nankai.edu.cn; yangjufeng@nankai.edu.cn).

J.-H. Xue is with the Department of Statistical Science, University College London, London WC1E 6BT, U.K. (e-mail: jinghao.xue@ucl.ac.uk).

J. Chang is with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: jianlong.chang@nlpr.ia.ac.cn).

Q. Tian is with the Huawei Cloud AI, Shenzhen, 518000, China (e-mail: tian.qi1@huawei.com).

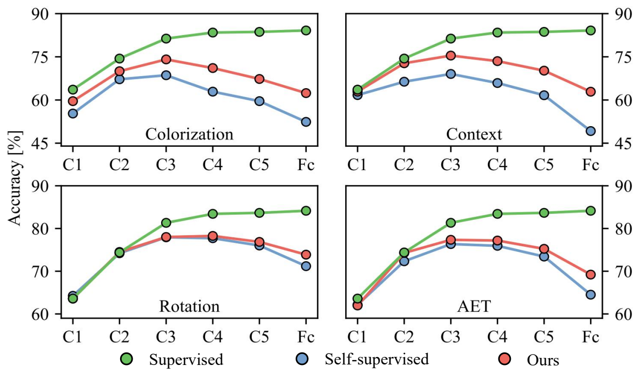

Fig. 1: **Results of compared methods on STL-10 [5].** The performance of *Supervised* witnesses a gradual increase while all *Self-supervised* methods (*Colorization* [6], *Context* [7], *Rotation* [8], and *AET* [9]) witness a gradual decrease after *C3*, indicating that the learned features with increasing specificity to the proxy task. The four panels show that SSL++ can mitigate the task-specificity of features, resulting in improving their performance largely, especially in the *C5* and *Fc* layers. Note that *Ck* denotes the *k*-th convolutional layer in the AlexNet [10] architecture.

Generally, based on how attributes are excavated from data, the pseudo-labeling schemes can be categorized into two groups: explicit methods [7], [11] and implicit methods [12], [13]. The notable examples of explicit methods are recoloring gray-scale images [6], [14], [15] and recovering the missing part of the data from the remaining part [7], [16]. However, these work mainly focuses on designing various specific proxy tasks to learn features, while seldom of them care about what characteristics are owned by the learned features and whether they are indeed generalizable to the downstream tasks that are usually different from the pre-designed proxy tasks. That is to say, over-focus on designing proxy tasks could induce inherent limitations in the generalizability of the learned features since these tasks are only proxy of downstream tasks in which the learned features will eventually be used [4], [8]. As such, the features learned by existing SSL methods can be specific to the proxy task, especially the features of the last few layers in the learned model. The results of our experiments (see details in Fig. 1), as well as a comprehensive review [4], demonstrate the adverse impact of task-specificity, *i.e.*, the quality of the learned features in SSL deteriorates towards the end of the network. Take the elegant work of [8] as an example, visual features can be learned by predicting image rotations. However, these features will be specific to those rotation variant samples and cannot possess discriminability for ones in favor of rotation invariance [17]. Moreover, not all examples in the real world are rotation determinable. For instance, ball types (*e.g. football* and *baseball*) are orientation agnostic, resulting in the ambiguity of what the pseudo-label of rotation is, while these ball types are semantically discriminative. Thus, utilizing the semantic pseudo-labels to learn task-independent semantic features is a feasible solution to mitigate the task-specificity in existing SSL methods. In this paper, the semanticity refers to the category attributes of samples, and we introduce IIC [18] (ConvNet-based clustering) to generate semantic pseudo-labels.

Typical of implicit approaches are clustering-based methods [19] that jointly optimize the cluster assignments and representation learning. Specifically, these methods alternate between generating pseudo-labels by using k-means on the extracted features to group samples and updating the parameters of ConvNet by predicting these labels. It means the cluster assignments and representation learning are coupled. For example, Caron *et al.* [20] iteratively use k-means to cluster deep features of samples to generate pseudo-labels for the supervisory signal. However, the success of these methods requires two conditions: 1) k-means suffices to cluster the samples; 2) cluster assignments and representation learning are complementary and supportive to each other. But in practice the two conditions may not be met, for two reasons: 1) k-means is inherently subject to some limitations, such as being dependent on centroid initialization and the predetermined value of k, and being prone to a local minimum, all of which could induce poor clustering results [21]; 2) if the extracted features are not discriminative and separably, the clustering results will be poor, and subsequently the updated ConvNet will extract features with worse discriminability. Such an iterative coupling strategy could induce degenerate solutions [18], [20], [22] that the cluster assignments are collapsed into a single entity, resulting in the generated pseudo-labels becoming noisy and unbalanced. Thus, a straightforward remedy is to leverage a more effective clustering algorithm and decouple it from the representation learning process.

Therefore, in this paper, we focus on tackling the two issues mentioned above in existing SSL work: 1) mitigating the proxy task-specificity of features by learning task-independent semantic features; 2) decomposing the coupling between cluster assignments and representation learning to avoid degenerate solutions. Towards this aim, we propose an effective decoupling framework SSL++ that first leverages IIC to perform clustering for generating semantic pseudo-labels, and then in the representation learning module, uses these labels to learn additional semantic features regardless of the proxy task to mitigate the limitations of task-specificity and degenerate solutions in existing methods (see Fig. 2). Specifically, in the clustering module, instead of iteratively using k-means, we utilize IIC (ConvNet-based clustering) and decouple it from the representation learning process to avoid degenerate solutions, and it also can generate less noisy and more balanced pseudo-labels when performing cluster assignments as semantic pseudo-labels [18]. For the representation learning module, to obtain more discriminative and generalizable features, we incorporate a classification branch into the SSL framework. The aim of adding such a classification branch is to learn proxy task-independent semantic features to complement the low-level information encoded by the SSL branch. By decoupling the two modules, the representation learning stage will be committed to learning more discriminative and generalizable features. Note that SSL++ can incorporate a variety of self-supervision tasks and promote their performance. For simplicity, we choose two representative tasks (*Colorization* [6] and *Context* [7]) and two state-of-the-art tasks (*Rotation* [8] and *AET* [9]) as illustration.

Our contributions can be summarized as follows:

- We propose a simple yet effective framework SSL++ that leverages proxy task-independent semanticity of samples to mitigate proxy task-specificity of features, which is in a position to consistently enhance the performance of proxy task-based SSL methods.
- We present a novel process decoupling clustering and self-supervision modules to avoid degenerate solutions [18], [20] in existing clustering-based SSL methods, which provides a successful attempt to improve the performance of SSL.
- We demonstrate the advantages of the above contributions on extensive experiments. On curated and uncurated benchmarks, and for classification, detection, and segmentation tasks as illustration, we show how to effectively leverage proxy task-independent semantic features and that this significantly improves the performance of existing typical and state-of-the-art SSL methods.

II. Related Work

A. Self-supervision via proxy tasks

Self-supervision via proxy tasks is the mainstream method in which ConvNets are trained with pseudo-labels that are automatically generated from the well-designed proxy tasks. In the past few years, a wide spectrum of proxy tasks have been proposed. Based on the attributes of generated pseudo-labels, proxy tasks can be roughly divided into five common categories: dense generation [6], [14], [15], spatial context [2], [23], [24], cross-modal methods [25]–[27], temporal structures of videos [28]–[30], and multitask-based models [31], [32]. Besides, the contrastive learning paradigm [33]–[35] can also be seen as a proxy task in which positive and negative pairs are constructed for comparison.

A notable example is the colorization [6], [15] that aims to reconstruct a coloring image given its gray-scale version as input. To avoid the mean effect in the regression paradigm of colorization [14], Zhang *et al.* [6] treat it as a classification problem by using K-nearest neighbors to quantize the color space. To correctly colorize each pixel, models need to recognize objects and group the pixel information of object parts together. However, there are two challenges in the colorization: 1) the number of the quantized values is a predetermined hyper-parameter, which is heavily dependent on the experimental configurations. For example, this number is set to 313 in [6] while 32 in [15]. 2) focusing on the low-level color information leads to large gap between the learned features and the downstream high-level tasks. Thus, its performance is unsatisfactory. Another example is the spatial context [7], [16], which is a task of recovering missing contents based on the surrounding. Committed to solving this task, networks need to learn the semantic structure of the surrounding. However, the spatial context methods have a drawback that the networks usually focus obsessively on exploiting incidental clues, such as chromatic aberration and edges between patches [36].

Some methods explicitly exploit transformation equivariance and encourage the learned representations to be invariant to customized translations [37]. For example, Dosovitskiy *et al.* [37], [38] propose to apply various pre-defined transformations to individual images for generating a set of surrogate classes as supervision signals. However, they treat the visually similar images as different surrogate classes, which leads to the learned features could be over-discriminative. Besides, it adopts the parametric paradigm for classification, which leads to the training cost is expensive as every transformed example is treated as an individual surrogate class. Gidaris *et al.* [8] train a model to learn visual representations by predicting the rotation of an image with rotated transformation. However, the learned features cannot process discriminability for samples with rotation invariance, such as ball types. Thus, the work of [17] decouples representations by adding an instance discrimination task, which aims to learn features with rotation discriminative and rotation unrelated. Zhang *et al.* [9] propose a method in which images are converted by affine or projection transformations, then the transformation parameters are treated as pseudo-labels. Furthermore, Guo *et al.* [39], [40] explain transformation invariance from an information theory perspective and propose Autoencoding Variational Transformations (AVT), in which the mutual information between the transformations and representations is maximized for learning robust feature representation. Recently, PIRL [41] aims to learn invariant semantic information based on various transformations by the well-designed noise contrastive estimator (NCE). Specifically, PIRL uses jigsaw puzzles as its pretext task for achieving image transformation invariance. Following [42], it utilizes a memory bank to store the moving averaging of representations. Besides, there is also work using SSL paradigm to solve other problems in computer vision, such as tracking [43], face reconstruction [44], person re-identification [45], action recognition [46], and object detection [47]. To address the domain shift on both temporal dynamics and spatial representation in action recognition, the work of [46] utilizes spatio-temporal contrastive domain adaptation (STCDA) for establishing the cross-modal domain alignment.

In addition to designing various proxy tasks, the difficulty of proxy tasks is also an important factor [4]. A suitable proxy task should be not too difficult and not too easy. If a task is too difficult, the model may not converge, and if a task is too easy, the model can easily get trivial solutions. For example, in the Jigsaw puzzles task [24], [48], given 9 patches from an image, there are 9! (362,880) possible permutations. A network is unlikely to classify all the permutations due to the ambiguity of this task. To limit the difficulty of this task, the work of [2] utilizes hamming distance to choose a subset of permutations.

B. *Self-supervision based on clustering*

Clustering, grouping together similar samples, is one of the typical unsupervised algorithms and has proven promising for data mining. Recently, due to the unparalleled representation ability of ConvNet, various self-supervised methods utilizing clustering based on deep neural networks have been proposed [13], [49]–[52]. For example, Noroozi *et al.* [12] utilize k-means for performing knowledge distillation to relieve the restriction of architecture from SSL, resulting in improving the representation quality of the shallow model. Furthermore, Caron *et al.* [20] propose a framework that alternates between generating pseudo-labels by using k-means on the extracted features to group samples and updating the parameters of ConvNet by predicting these labels. Committed to bridging the performance gap between curated and uncurated data, Caron *et al.* [19] focus on combining the work of [8] with the typical clustering algorithm in [20]. Different from them coupling k-means to the representation learning process, which may lead to degenerate solutions [18], [20] so that the quality of learned representation would be degraded, we employ IIC (ConvNet-based clustering) and decouple it from the representation learning process. IIC is trained by maximizing the mutual information between paired images associated with geometric perturbations, such as cropping and rotation. The primary aim of IIC in [18] is cluster assignments, while ours is representation learning by utilizing it to generate less noisy and more balanced semantic pseudo-labels.

III. A Key Issue: Proxy Task-Specificity

In this section, we mainly describe the four self-supervised tasks that are used in this paper and elaborate on the key issue (*i.e.*, task-specificity) in existing SSL methods.

A. *Self-Supervised Learning Tasks*

Here we introduce two representative tasks and two newly proposed tasks in SSL.

Colorization [6]: The approach by Zhang *et al.* trains a model that takes lightness L as input and predicts the counterpart ab colors in the CIE Lab color space to learn visual features. It treats the colorization task as a classification problem rather than a regression problem to avoid graying and unsaturated results caused by the mean effect. Technically, it recasts this problem as a 313-way classification problem by leveraging K-nearest neighbors algorithm to quantize the ab space into 313 discrete categories. Naturally, the learned features in this task could be sensitive to color information.

Context-Encoder [7]: This task aims to recover the missing contents of the central region from the remaining part. Specifically, the encoder takes the incomplete image as input for producing latent visual features, and then the decoder takes them as input to recover the missing part. For producing more realistic image content, the adversarial loss [53] is combined with $L2$ reconstruction loss in their image generation experiment. However, the same as the experimental settings of representation learning in the original paper, we just use the reconstruction loss in our experiments. The idea is that the model needs to understand the image by reconstructing

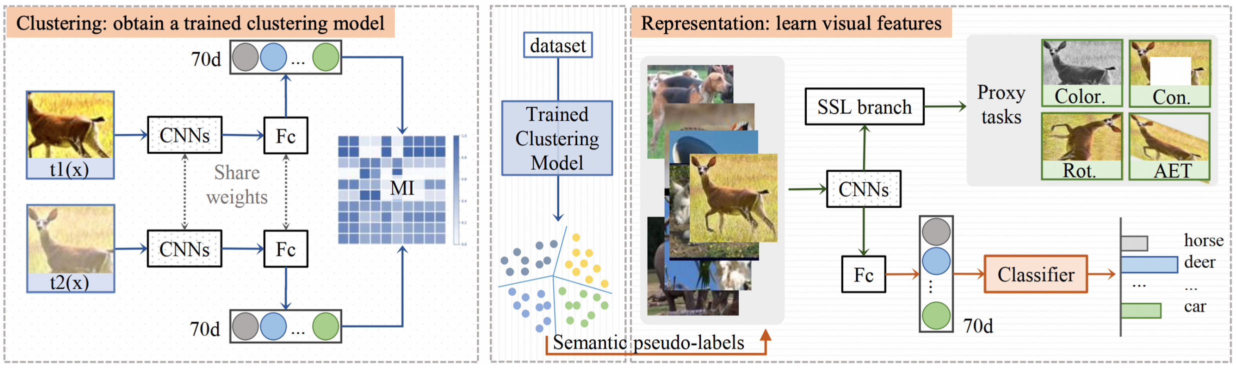

Fig. 2: **Illustration of the proposed framework**. SSL++ is decoupled into two modules for avoiding the degenerate solutions in existing methods [20]. In the clustering module, a clustering model is trained by maximizing the mutual information (MI) between paired transformed samples, as a result, we can obtain a well-trained clustering model and then semantic pseudo-labels can be generated from it. In the representation learning module, a model is trained to learn visual features by predicting the semantic and task-specific (*e.g.*, rotation categories [8]) pseudo-labels. The purpose of learning task-independent semantic features is to mitigate the task-specificity of features learned through proxy tasks.

the missing central part so that the learned features could be dependent on the central part.

Rotation [8]: Gidaris *et al*. propose an algorithm that first produces four different rotated copies of an image and then predicts what kind of rotation is applied in these copies. In their experiments, the rotation pseudo-label set is $\{0°, 90°, 180°, 270°\}$ and thus this task is converted to a 4-way classification problem. Intuitively, to predict the image rotation correctly, the model should focus on salient parts in an image that determine the rotations of the image, hence the learned features could be sensitive to image rotations.

AET [9]: Auto-Encoding Transformation (AET) aims to encode various transformations to learn visual features. Specifically, AET mainly focuses on the parameterized transformations, *i.e*., the affine and projective transformations. In detail, the two transformations can be denoted as a matrix $M(\theta) \in \mathbb{R}^{3\times3}$ that parametrically models the geometric perturbation between images. The AET model is trained to predict the parameters so that this task can be treated as a regression problem. The idea is that the AET model forces the representations to have transformation equivariance so that the learned features can encode the intrinsic information about their visual structures of original and transformed images. Intuitively, the learned features could be associated with the transformation parameters between samples.

B. Proxy Task-Specificity

Proxy task-specificity refers to an inherent limitation in SSL, *i.e*., the quality of features learned by a proxy task will deteriorate towards the end of the network. That is, as the depth of layers increases, the features of these layers will fall in the performance of downstream tasks. As multilayer feedforward networks are theoretically capable of approximating any function to any desired accuracy [54], Zhang *et al*. [55] demonstrate that ConvNets can easily fit completely random labels, thereby resulting in zero training error but not achieving generalizability. Hence, beyond question, ConvNets can fit any pseudo-labels easily. Therefore, if a model wants to learn generalizable features, it is crucial to get the pseudo-labels encoding meaningful visual information. Thus, a straightforward solution is to design various proxy tasks that exploit the attributes of data [2], [6], [23], [28]. Unfortunately, in being dedicated to solving these proxy tasks, the learned features will be increasingly specific to the proxy task with the depth of layers [8], [56]. Intuitively, features in shallow layers would encode general low-level information, such as edges and textures, while features in higher layers would encode task-specific high-level information [4] (*e.g*., rotation categories [8]). As such, the quality of the learned features in SSL deteriorates towards the end of the network.

Furthermore, a proxy task always has some task-specific data premises [17]. This limitation could induce performance degradation for downstream tasks that have the data violating the task-specific premises. Here, we briefly summarize the premises of the four approaches discussed in this paper: 1) *Colorization* [6]: an image ought to be chromatic or has sufficient color information; 2) *Context* [7]: the object in an image should be near the center of the image; 3) *Rotation* [8]: rotating an image should cause the orientation of objects in the image to change; and 4) *AET* [9]: the content of an image should be objects-dominated and has no extremely complicated or simple scenes that could lead to transformation agnostic. The prerequisites introduced in these approaches could be satisfied in most well-annotated datasets, such as ImageNet, in which the images generally are up-standing, colorful, objects-dominated, and center-located [57]. However, things go contrary to one's wishes, these premises may not be satisfied in real-world images, where the content is generally complicated, multifarious, and even monochrome. That is, the features learned by proxy tasks may not generalize well to downstream tasks that have ill-premised data. For example, the learned features in *Rotation* cannot generalize well to rotation agnostic tasks like the ball recognition challenge [58].

To sum up, the task-specificity of features is triggered by the proxy task and the data premises required for this task. First, when a model is devoted to optimizing the objective of the proxy task, the corresponding features would be forced to encode high-level task-specific information (*e.g.*, rotation categories). Second, the features may suffer from performance degradation for the data violating the task-specific premises. Our approach, SSL++, adding proxy task-independent semantic features, as detailed in the following section, aims to mitigate the proxy task-specificity in SSL.

IV. Methodology

A. Overview

We propose a decoupled framework SSL++ that first generates semantic pseudo-labels, and then these pseudo-labels are used in the representation learning module to learn proxy task-independent features to mitigate the proxy task-specific features. Specifically, SSL++ consists of a clustering module and a representation learning module. In the clustering module, we utilize the mutual information (MI) between paired transformed samples to train a clustering model. Then, the well-trained clustering model is used to generate semantic pseudo-labels. In the representation learning module, a model is trained to learn visual features by predicting the generated semantic and task-specific pseudo-labels.

Compared with previous work, there are mainly two differences: 1) As opposed to some work [3], [56] focusing on dealing with the extrinsic factors (*e.g.*, architecture and data) of self-supervised methods, this paper aims to solve the intrinsic limitations that they often suffer from, namely the proxy task-specificity, to improve the performance of existing methods. 2) Clustering-based self-supervised methods [59], [60] always optimize the cluster assignments for generating semantic pseudo-labels and representation learning for learning visual features jointly. Such a coupling strategy could induce degenerate solutions [18], [20], [22], resulting in learning poor features. In our proposed SSL++, the clustering and representation learning modules are decoupled, thus, it can focus on learning good visual features without being disturbed by the instability of clustering assignments.

B. Pseudo-Labels Estimation

Here we first review the pseudo-labels estimation process in our SSL++ framework. Let $\mathcal{X}$ and $\mathcal{Y} = \{1, \ldots, C\}$ denote data space and class space, respectively. C means the number of classes, *i.e.*, the number of clusters. Let $\mathcal{T}$ denote a set that contains various geometric (such as cropping, rotating, flipping) and photometric (such as changing contrast and color saturation) transformations. We randomly sample two transformations (t_1 and t_2) from $\mathcal{T}$ and apply them to an sample $\mathbf{x}$ drawn from $\mathcal{X}$, resulting in two paired samples $\mathbf{x}_{t_1}$ and $\mathbf{x}_{t_2}$. We hope that our clustering model can learn a representation mapping function Φ: $\mathcal{X} \to \mathcal{Y}$, in which the commonality in paired samples is kept meanwhile the sample-specific details are thrown away. Both of them can be achieved by maximizing the following mutual information between the paired samples:

$$I\left(\Phi(\mathbf{x}_{t_1}), \Phi\left(\mathbf{x}_{t_2}\right)\right) \tag{1}$$

The neural network Φ ends with a softmax layer, thus it can be seen as a soft clustering paradigm. For a given sample $\mathbf{x}$, its representation is $\Phi(\mathbf{x}) \in [0,1]^C$, which can be interpreted as the probability of the cluster assignment random variables z over C classes, *i.e.*, $\Phi_c(\mathbf{x}) = P(z = c \mid \mathbf{x})$, the output probability that the cluster c is assigned to a sample $\mathbf{x}$. Consider a paired samples $\mathbf{x}_{t1}$ and $\mathbf{x}_{t2}$, z and z' denote a pair of clustering assignment variables for them, respectively. Now we define $P\left(z = c, z' = c' \mid \mathbf{x}_{t1}, \mathbf{x}_{t2}\right) = \Phi_c(\mathbf{x}_{t1}) \cdot \Phi_{c'}\left(\mathbf{x}_{t2}\right)$; after considering all the input batch data, $\mathbf{P} = \frac{1}{n}\sum_{i=1}^{n} \Phi\left(\mathbf{x}_{t1}^{i}\right) \cdot \Phi\left(\mathbf{x}_{t2}^{i}\right)^{\top}$. Note that $\mathbf{P}$ is a $C \times C$ matrix and each element at row c and column c' constitutes $\mathbf{P}_{cc'} = P\left(z = c, z' = c'\right)$. Summing along with the rows and columns of $\mathbf{P}$, we can get the marginals $\mathbf{P}_c = P(z = c)$ and $\mathbf{P}_{c'} = P\left(z' = c'\right)$. Now plugging $\mathbf{P}$ into Eq. (1), we get the final mutual information loss as follows:

$$I\left(z, z'\right) = I(\mathbf{P}) = \sum_{c=1}^{C}\sum_{c'=1}^{C} \mathbf{P}_{cc'} \cdot \ln \frac{\mathbf{P}_{cc'}}{\mathbf{P}_c \cdot \mathbf{P}_{c'}}. \tag{2}$$

By maximizing the mutual information between mapped features, Φ can encourage the same representations of paired samples. Compared with k-means, this algorithm employs mutual information to minimize the representation distance of samples, leading to generating less noisy and more balanced semantic pseudo-labels. In the early training stages, features of an image extracted by a training model are not always discriminative and separably. It leads to the clustering results based on these extracted features will be poor, that is, images of the same category may be divided into different clusters, or images of different categories may be included into the same cluster, both of them could lead to incorrect pseudo-labels. Then, learning these incorrect pseudo-labels, the updated ConvNet will extract features with worse discriminability. Such an iterative coupling strategy could induce degenerate solutions [18], [20], [22] that the cluster assignments are collapsed into a single entity, resulting in the generated pseudo-labels becoming noisy and unbalanced. While in SSL++, the cluster assignments process is decoupled from the representation learning process, so that it can focus on learning good visual features without being disturbed by the instability of clustering assignments. Thus, the degenerate solutions that are prone to occur in other clustering-based methods are avoided.

C. Self-Supervision Based on Proxy Tasks

In SSL, a proxy task is designed for generating pseudo-labels. Generally, the pseudo-labels can be categorized into two groups, namely the categorical labels and the continuous variables [19]. For example, the former predicts transformation or color attribute categories of an image, and the latter predicts transformation parameters or missing regions. In this work, we employ two classic SSL methods (*i.e.*, *Colorization* [6] and *Context* [7]) and two state-of-the-art methods (*i.e.*, *Rotation* [8] and *AET* [9]) to validate the effectiveness of the proposed SSL++. Note that *Colorization* and *Rotation* are classification problems while *Context* and *AET* are regression problems. We suppose that y_n denotes the pseudo-label of image x_n generated from a proxy task. The SSL branch (denoted as V_1) aims to predict the pseudo-label y_n upon the feature

TABLE I: **Experiment setups.** Experimental setups of the self-supervised training stage and the fine-tuning evaluation stage, including the used datasets, evaluation protocols, *etc*. The content in parentheses of the *data* columns denotes which part of the dataset is used, *Tr*, *Val*, *Te*, and *Un* represent the training set, validation set, test set, and unlabeled set, respectively.

Training Stage (Training without labels)			**Evaluation Stage (Fine-tune with labels)**			
Pre-training data	Category	Architecture	Re-training data	Test data	Task	Evaluation Protocol
CIFAR-10 (Tr)	Curated	NIN	CIFAR-10 (Tr)	CIFAR-10 (Te)	Classification	NIN Block/KNN
STL-10 (Tr+Un)	Unsupervised	NIN	STL-10 (Tr)	STL-10 (Te)	Classification	NIN Block/KNN
Tiny-YFCC100M	Uncurated	AlexNet	STL-10 (Tr)	STL-10 (Te)	Classification	Linear layer
			VOC 07 (Tr)	VOC 07 (Te)	Multi-label Class.	Fine-tune Fc6-8/All
			VOC 07 (Tr)	VOC 07 (Te)	Detection	Fine-tune All
			VOC 12 (Tr)	VOC 12 (Val)	Segmentation	Fine-tune All

TABLE II: **Implementation details**. Implementation details about the self-supervised training stage and the fine-tuning evaluation stage, including the used optimizer, batchsize, initial learning rate, *etc*.

Stage	**Method**	**Uniform setting**			**For CIFAR-10/STL-10**		**For Tiny-YFCC100M**	
		Optimizer	Batchsize	Initial Lr	Epoch	Lr schedule	Epoch	Lr schedule
SSL train.	Color. [6]	Adam	128	0.01	200	[100,150]	120	None
	Context [7]	SGD	512	0.01	200	[100,150]	200	[100,150]
	Rotation [8]	SGD	128	0.1	200	[60,120,160]	50	[15,35,45]
	AET [9]	SGD	512	0.1	1500	[240,480,640,1000]	200	[100,150]

Stage	**Pretrain data**	**Evaluation Data**	**Optimizer**	**Batchsize**	**Momentum**	**WD**	**Initial Lr**	**Epoch**	**Lr schedule**
Evaluation	CIFAR10	CIFAR10	SGD	128	0.9	0.0005	0.1	200	[100,150]
	STL10	STL10	SGD	64	0.9	0.0005	0.1	200	[100,150]
	Tiny-YFCC100M	STL10	SGD	64	0.9	0.0005	0.1	200	[100,150]
		VOC 07 Cls. (Fc6-8)	SGD	16	0.9	0.0005	0.003	200	None
		VOC 07 Cls. (All)	SGD	16	0.9	0.0005	0.001	200	None
		VOC 07 Det. (All)	SGD	2	0.9	0.0005	0.001	12	None
		VOC 12 Seg. (All)	SGD	2	0.9	0.0005	0.001	50	None

$f_\theta(x_n)$, where f_θ denotes a feature extraction model. The self-supervision loss function can be expressed as

$$\mathcal{L}_s = \frac{1}{N}\sum_{n=1}^{N} \ell\left(V_1\left(f_\theta\left(x_n\right)\right), y_n\right), \tag{3}$$

where ℓ denotes loss function. By solving proxy tasks, the features in shallow layers can capture sufficient general low-level visual information, such as edges and textures. Therefore, in this work, we take advantage of this benefit and treat them as a complement to our semantic features.

D. Learning Semantic Feature

Towards the goal of learning a proxy task-independent semantic feature, we append a classification branch to off-the-shelf SSL frameworks. Specifically, by exploiting the semanticity of samples, such a task-independent feature provides additional discrimination between images with similar task-specific features, thereby mitigating the side-effect of task-specificity. Formally, suppose that z_n denotes the pseudo-label of image x_n generated from cluster assignments, and $\mathcal{Z}$ denotes the set containing all possible z_n. A parameterized classifier (denoted as V_2) predicts the pseudo-label z_n on top of the feature $f_\theta(x_n)$. The classification loss function for learning the semantic feature can be expressed as

$$\mathcal{L}_c = \frac{1}{N}\sum_{n=1}^{N} \ell\left(V_2\left(f_\theta\left(x_n\right)\right), z_n\right). \tag{4}$$

Generally, pseudo-labels in $\mathcal{Z}$ are generated from the clustering module and still suffer from some noise. Thus, using this objective alone could lead to a model fitting these noisy labels, thereby resulting in performance degradation [55]. As with [19], we exploit the complementarity between task-specific features and semantic features to avoid this degradation problem.

E. Mitigating Task-Specificity in Self-Supervision

To combat the task-specificity of features, especially in higher layers, based on semantic pseudo-labels generated from IIC, we present a novel framework SSL++ to learn an additional task-independent feature with semantic pseudo-labels (see details in Fig. 2). By solving a well-designed proxy task, features in shallow layers encode general low-level information, such as edges and textures, while features in higher layers encode task-specific information [4]. Hence, we propose to use the semantic feature to complement the generic features in shallow layers and mitigate the task-specific features in higher layers.

With all components ready, the resulting model consists of two modules: proxy task module and semantic feature module, and the overall loss function is

$$\mathcal{L} = \mathcal{L}_s + \lambda\mathcal{L}_c, \tag{5}$$

in which $\mathcal{L}_s$ denotes the self-supervision loss and $\mathcal{L}_c$ denotes the classification loss. The parameter λ denotes the trade-off between the two losses. Intuitively, the aim of $\mathcal{L}_s$ is to make features encode general low-level information, and to alleviate fitting noisy labels from using $\mathcal{L}_c$ alone; meanwhile, the aim of $\mathcal{L}_c$ is to make features capture task-independent semanticity,

TABLE III: **Accuracy with NIN [61] block classifier on CIFAR-10 and STL-10.** We train a new NIN block classifier on various blocks of feature maps whose size is 192×8×8. The *Bk* denotes the *k*-th block in NIN architecture.

Method	**CIFAR-10**				**STL-10**			
	B2	B3	B4	Avg.	B2	B3	B4	Avg.
Supervised	92.2	92.6	91.7	92.2	80.2	80.3	79.2	79.9
Random	70.1	66.4	65.8	67.4	61.6	58.5	56.7	58.9
Color. [6]	78.4	71.2	66.5	72.0	78.0	72.9	67.9	72.9
Ours+Color.	**81.1**	**76.0**	**74.2**	**77.1**	**82.7**	**80.1**	**75.8**	**79.5**
Context [7]	78.8	70.7	63.9	71.1	62.6	59.4	52.6	58.2
Ours+Con.	**86.6**	**82.9**	**74.7**	**81.4**	**82.4**	**77.9**	**71.3**	**77.2**
Rotation [8]	**90.8**	86.4	60.1	79.5	**86.4**	82.7	58.7	75.1
Ours+Rot.	90.1	**86.5**	**75.9**	**84.2**	85.9	**83.6**	**74.2**	**81.2**
AET [9]	90.7	86.2	59.4	78.8	86.5	80.5	54.5	73.8
Ours+AET	**90.9**	**86.8**	**75.7**	**84.5**	**87.7**	**84.6**	**75.0**	**82.4**

TABLE IV: **Comparison with different classifiers on CIFAR-10.** Specifically, the n-FC denotes a classifier with n FC layers and *Conv* denotes a new NIN block as a convolutional classifier.

Method	**Classifiers**			
	1FC	2FC	3FC	Conv
Supervised	91.2	91.5	91.8	91.7
Random	57.3	65.0	65.6	65.8
Color. [6]	60.1	61.7	63.5	66.5
Ours+Color.	**72.5**	**75.1**	**75.4**	**74.2**
Context [7]	58.6	59.2	61.1	63.9
Ours+Con.	**72.8**	**75.8**	**75.6**	**74.7**
Rotation [8]	53.1	52.0	53.2	60.1
Ours+Rot.	**72.6**	**75.8**	**75.7**	**75.9**
AET [9]	59.8	61.6	62.6	59.4
Ours+AET	**74.1**	**76.1**	**75.9**	**75.7**

and to mitigate the task-specificity of features in higher layers from using $\mathcal{L}_s$ alone. With the joint optimization, both the task-independent generality and semanticity of features are captured. Hence the generalizability and discriminability of the features learned by SSL++ can be highly enhanced.

V. Experiments

Here we report on extensive experiments performed to evaluate the effectiveness of SSL++. Specifically, the evaluation process follows the protocols widely adopted by most existing self-supervised methods, through training a new classifier upon the learned representations for downstream tasks.

A. Experiment Setup

First, we summarize the used architectures, evaluation protocols, and implementation details for all experiments.

1) Architectures: For a fair comparison, we implement the two architectures (*i.e.*, Network-In-Network (NIN) [61] and AlexNet [10]) for all compared methods.

Network-In-Network (NIN). The NIN consists of four convolutional blocks, each of which contains three convolutional layers. In *AET* [9], the network has two shared-weights NIN branches, taking the original and transformed images as input, respectively. In other tasks, the network has one NIN branch that only takes the transformed images as input.

TABLE V: **Accuracy with KNN on CIFAR-10 and STL-10.** To avoid the model-distorted performance that re-training with labels could induce [20], we use KNN (set K = 10) to re-evaluate various blocks of feature maps.

Method	**CIFAR-10**				**STL-10**			
	B2	B3	B4	Avg.	B2	B3	B4	Avg.
Supervised	75.5	89.9	89.2	84.9	49.3	60.4	60.3	56.7
Random	37.7	38.0	36.9	37.5	25.2	25.4	25.7	25.4
Color. [6]	58.2	56.5	49.1	54.6	47.3	47.3	41.1	45.2
Ours+Color.	**60.9**	**61.4**	**61.1**	**61.1**	**50.8**	**53.5**	**55.5**	**53.4**
Context [7]	41.5	39.7	30.9	37.4	26.0	24.7	21.0	23.9
Ours+Con.	**62.8**	**67.3**	**64.0**	**64.7**	**47.6**	**51.9**	**49.1**	**49.5**
Rotation [8]	**69.9**	72.4	18.2	53.5	51.3	51.4	18.7	40.5
Ours+Rot.	69.3	**74.9**	**64.5**	**69.6**	**51.5**	**55.8**	**51.7**	**53.0**
AET [9]	70.4	68.3	25.1	54.6	50.5	48.1	18.1	38.9
Ours+AET	**72.1**	**75.9**	**66.4**	**71.5**	**53.5**	**58.3**	**53.5**	**55.1**

AlexNet. It contains eight layers, the first five are convolutional layers and the last three are fully connected layers. The setting of all methods is the same as mentioned above except that the architecture is changed from NIN to AlexNet.

2) Datasets: To adequately validate the effectiveness, we choose three types of datasets to evaluate the proposed SSL++, including the curated CIFAR-10 [62] and STL-10 [5], uncurated Tiny-YFCC100M [63], and PASCAL VOC [64].

CIFAR-10. The CIFAR-10 dataset consists of 60,000 32×32 images that represent 10 various semantic classes, each of which includes 6,000 images. There are 50,000 training images and 10,000 test images.

STL-10. Inspired by the CIFAR-10 dataset, STL-10 contains 113,000 images of 10 common classes. The difference is that STL-10 mainly focuses on evaluating unsupervised feature learning algorithms. In particular, each class has only 500 training images and 800 test images, while there are 100,000 unlabeled images for unsupervised learning. Note that all images are acquired from ImageNet.

Tiny-YFCC100M. YFCC100M [63] is a typical large-scale, diversified, and uncurated dataset, which contains roughly 99M images downloaded from the website. Different from curated datasets (*e.g.*, CIFAR10, ImageNet) that are balanced, with a well-behaved data distribution, YFCC100M is unbalanced, with a "long-tail" distribution and contains a variety of complex real-world scenes, ranging from massive street life-blogged photos to snapshots of everyday life, holidays and events [19]. Like Mini-ImageNet and Tiny-ImageNet sampled from ImageNet, we sample 300k images from YFCC100M (named Tiny-YFCC100M) for quick and effective validation of our algorithm. We will release it for public.

PASCAL VOC. This dataset is mainly used for evaluating the generalization of learned features on different downstream tasks. Specifically, it consists of three tasks, namely multi-label classification, object detection, and semantic segmentation.

3) Evaluation protocols: To evaluate the quality of features learned by self-supervised methods, we treat the learned model by SSL as a pre-trained model for downstream tasks in which a new classifier on top of frozen SSL features is trained with labels. Specifically, we follow the common evaluation protocols [8], [9] by building the following classifiers:

TABLE VI: **Comparison with varying numbers of K**. The comparison of the KNN accuracy with varying numbers K of nearest neighbors on CIFAR-10. All experiments are building a classifier on top of the fourth convolutional block.

Method	**Varying numbers of K**						
	1	3	5	10	15	20	Avg.
Supervised	88.2	88.8	89.0	89.2	89.3	89.3	89.0
Random	32.6	33.5	35.0	36.9	37.5	37.8	35.6
Color. [6]	42.3	44.3	46.7	49.1	50.1	50.7	47.2
Ours+Color.	**54.4**	**57.0**	**59.1**	**61.1**	**61.3**	**61.9**	**59.1**
Context [7]	27.5	27.0	29.4	30.9	31.5	32.2	29.8
Ours+Con.	**58.3**	**60.9**	**62.5**	**64.0**	**64.4**	**64.9**	**62.5**
Rotation [8]	15.6	15.6	16.9	18.2	18.8	19.0	17.4
Ours+Rot.	**57.1**	**60.3**	**62.4**	**64.5**	**65.1**	**65.3**	**62.5**
AET [9]	20.0	21.1	22.6	25.1	26.1	26.1	23.5
Ours+AET	**60.4**	**62.6**	**64.9**	**66.4**	**67.2**	**67.5**	**64.8**

TABLE VII: **Accuracy with linear classifier on STL-10.** We train a new linear classifier on various layers of feature maps that are spatially resized (with adaptive max pooling). *Ck* denotes the *k*-th convolutional layer in AlexNet architecture.

Method	**Layers**						
	C1	C2	C3	C4	C5	Fc	Avg.
Supervised	63.6	74.4	81.3	83.4	83.7	84.2	78.4
Random	55.6	61.6	61.7	61.1	57.9	32.0	55.0
Color. [6]	55.3	67.2	68.6	62.9	59.6	52.4	61.0
Ours+Color.	**59.6**	**70.0**	**74.1**	**71.1**	**67.3**	**62.4**	**67.4**
Context [7]	61.7	66.4	69.1	65.9	61.7	49.2	62.3
Ours+Con.	**62.9**	**72.8**	**75.4**	**73.5**	**70.2**	**62.9**	**69.6**
Rotation [8]	**64.2**	74.2	77.9	77.7	76.0	71.2	73.5
Ours+Rot.	63.6	**74.5**	**78.0**	**78.3**	**76.9**	**73.9**	**74.2**
AET [9]	**62.1**	72.3	76.4	76.0	73.4	64.5	70.8
Ours+AET	62.0	**74.3**	**77.4**	**77.2**	**75.2**	**69.2**	**72.6**

NIN block: The NIN block is a convolutional classifier and contains three convolutional layers and a linear soft-max layer. The input size of the convolution part is 192×8×8, and its output feature map is spatially averaged pooled to 192 and then fed into the linear soft-max layer.

Linear layer: The linear classifier is designed for the AlexNet. It takes various layers of feature maps that are spatially resized to have about 9,000 elements as input. Following [17], the feature layers are *Conv1* to *Conv5* and *Fc7*.

KNN: Different from the above classifiers that need to be re-trained, the non-parametric KNN classifier can make a direct evaluation on the learned features instead of re-training with labels. Its input consists of the k closest training samples in the features, and an image is classified by a plurality voted of its neighbors in this space.

Note that the setups of all experiments are summarized in Table I, including the used datasets, architectures, and evaluation protocols, *etc*.

4) Implementation details: In Table II, we summarize the implementation details of all compared methods during the SSL training stage and the fine-tuning evaluation stage. To make a fair comparison, we adopt the Network-In-Network (NIN) and the AlexNet for all compared methods. We use the Pytorch [65] deep learning framework in all experiments. We adopt the NIN architecture for CIFAR-10 and STL-10 datasets and the AlexNet architecture for Tiny-YFCC100M and PASCAL VOC datasets. Except for *Colorization* trained by the Adam optimizer, the rest methods are trained by the SGD. The *Lr schedule* entry denotes the learning rate schedule of different methods. Take *AET* as an example, its learning rate is scheduled to drop by a factor of 5 after 240, 480, 640, 800 and 1000 epochs. In other methods, the factor is 10. Note that "None" means that there is no learning rate schedule, that is, using the initial learning rate fixedly. For all models, the momentum and weight decay (denoted as WD) are set to 0.9 and 5×10^{-4}, respectively. We observe the training log and find that the optimal hyper-parameter has relationship with the loss value of the proxy task module and semantic feature module. For simplicity, we set $\lambda = 1$ for *Colorization* and *Context* and set $\lambda = 0.05$ for *Rotation* and *AET*.

B. Classification on CIFAR-10 and STL-10

In this section, we conduct four types of experiments for validating the effectiveness of SSL++, including the improved performance of the corresponding self-supervised tasks, the model-free KNN classifier on various blocks of NIN architecture, the varying K of model-free KNN classifier, and the few-labeled data for fine-tuning evaluation stage.

Comparison with corresponding self-supervised methods. Following the evaluation protocols adopted by [8], [9], we evaluate all compared methods upon different blocks of NIN on the classification task of curated CIFAR-10 and STL-10 datasets. We observe from Table III that the learned features by the second block achieve the highest accuracy for all methods and those by the following blocks gradually degenerate. This phenomenon validates the issue that we stated in Section III-B, namely the task-specificity (*i.e.*, the learned features will be increasingly specific to the proxy task with the depth of layers). Importantly, in CIFAR-10 and STL-10, SSL++ can promote all blocks in all methods except for only a few ones with comparable results. SSL++ has different gains based on different self-supervised methods. Since the *Colorization* and *Context* generate pseudo-labels by utilizing low-level features of the image, *i.e.*, color information and image contents, their performance is unsatisfactory. Thus, our SSL++ achieves greater performance gains based on them. For example, on CIFAR-10, compared with *Colorization* and *Context*, SSL++ achieves 5.1% and 10.3% improvements on average, respectively. From the perspective of each block, SSL++ can improve all of them. The shallow blocks usually learn more general low-level features, thus the gains of SSL++ in shallow blocks are not as obvious as them in deep blocks, which suffer from more serious task-specificity. It means that our approach successfully mitigates the task-specificity. Take the *AET* [9] as an example, it achieves the good performance on CIFAR-10 and STL-10, while SSL++ can still improve its performance, *i.e.*, the average accuracy on CIFAR-10 increased from 78.8% to 84.5%, and on STL-10 increased from 73.8% to 82.4%. Note that the results of *Supervised* on STL-10 (Table III) are performed only with the 5,000 labeled samples, while all the other SSL methods (including ours) are trained on the whole STL-10 dataset, *i.e.*, 105,000 samples.

TABLE VIII: **Comparison with varying numbers of labeled samples.** 'Supervised' denotes that its model is trained with the labeled samples from scratch.

Method	20				100				400				1000			
	B2	B3	B4	Avg.	B2	B3	B4	Avg.	B2	B3	B4	Avg.	B2	B3	B4	Avg.
Supervised	37.3	37.6	37.5	37.5	58.9	59.1	59.2	59.1	78.7	78.5	78.9	78.7	86.0	85.7	86.1	85.9
Random	35.4	33.5	30.3	33.1	46.3	45.6	40.7	44.2	57.8	54.4	51.1	54.4	64.1	60.3	56.7	60.4
Color. [6]	50.9	52.4	49.5	50.9	61.1	59.3	55.0	58.5	68.3	62.9	59.2	62.6	72.2	66.4	63.1	67.2
Ours+Color.	**56.8**	**59.1**	**57.7**	**57.9**	**67.7**	**65.8**	**62.3**	**65.3**	**73.8**	**70.8**	**66.8**	**70.5**	**77.9**	**73.8**	**69.3**	**73.7**
Context [7]	42.9	37.0	30.7	36.9	55.8	48.0	40.7	48.2	65.4	58.0	51.1	58.2	71.6	64.4	56.9	64.3
Ours+Con.	**57.5**	**60.8**	**58.7**	**59.0**	**68.6**	**66.8**	**61.8**	**65.7**	**76.6**	**71.7**	**66.1**	**71.5**	**79.9**	**75.0**	**69.5**	**74.8**
Rotation [8]	**63.5**	67.6	38.8	56.6	**74.5**	**76.3**	45.4	65.4	**82.8**	**81.9**	52.2	72.3	**86.4**	**84.6**	56.7	75.9
Ours+Rot.	61.7	**68.0**	**60.4**	**63.4**	73.7	75.1	**64.0**	**70.9**	82.0	81.5	**68.9**	**77.5**	86.0	84.2	**70.5**	**80.2**
AET [9]	**68.1**	68.8	36.2	57.7	**78.3**	75.1	45.4	66.3	**83.8**	79.5	56.0	73.1	**87.4**	82.5	58.8	76.2
Ours+AET	67.6	**69.8**	**61.5**	**66.3**	77.6	**75.3**	**65.3**	**72.7**	83.6	**79.9**	**68.8**	**77.4**	86.7	**82.6**	**71.4**	**80.2**

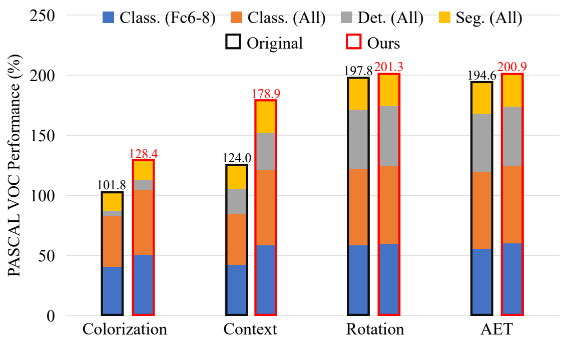

Fig. 3: **Task Generalization: results on PASCAL VOC classification, detection, and segmentation.** *Fc6-8* and *All* denote fine-tuning only the features after *Conv5* and the whole model, respectively.

Besides, following [9], [39], in Table IV we also conduct experiments to compare SSL++ with its baseline when trained with varying number of FC layers. As the number of FC layers increases, the performance of these self-supervised methods gradually increases. In our SSL++, the "2FC" and "3FC" have similar performance while both of them perform better than "1FC". It indicates that the learned features in SSL++ have touched the representation bottleneck. Compared with these basic self-supervised methods, our SSL++ can consistently make the highest performance no matter which classifiers are used.

Comparison with Model-free KNN Classifiers on various blocks. The evaluation protocol of re-training with labels could induce a model-distorted performance [20], *i.e.*, the performance gap between different feature extractors would be decreased. For example, a re-trained NIN classifier on top of the second block of a random NIN achieves 70.1% in accuracy on CIFAR-10 while the chance should be at 10%. The KNN is model-free classifier without training from labeled examples. Thus, the performance of the model-free KNN classifier can reveal the discriminative and separability of the learned feature representation. Following [8], [9], we utilize a model-free KNN classifier to make a direct evaluation on the quality of learned features (see Table V). In a word, SSL++ can improve the performance of most blocks in all methods with significant margins.

Comparison based on Model-free KNN Classifiers with varying K. Based on the averaged-pooled feature representations from the fourth convolutional block, in Table VI we test the performance of model-free KNN classifier and we can make the following three observations. First, as the K increases, the performance of all methods will get better. The larger K is equivalent to that the KNN classifier can see more global information in the features. Second, the recent state-of-the-art methods, *Rotation* and *AET*, on the the fourth convolutional block, perform worse compared with the methods of focusing on low-level information. We think that *Rotation* and *AET* suffer from more serious proxy-task specificity, thus both of them have a larger semantic gap between the downstream tasks. Third, compared with all the four self-supervised methods, our SSL++ aims to exploit the class-related semantic pseudo-labels, which makes the images of the same class together and the images of different classes distant in the feature space. Thus, when using the model-free KNN classifier, our SSL++ can achieve great improvements. In a word, the performance of SSL++ outperforms all the compared methods with varying K nearest neighbors.

Comparison with Few Labeled Data. Finally, following [8], [66], we also evaluate the learned features on the few labeled data setting in Table VIII. More specifically, we first train the NIN model on various self-supervised tasks using the whole CIFAR-10 dataset, and then we only use a small number of labeled images to train the downstream classifiers on top of their feature maps. Through this part of the experiment, we can gain some insights into how the SSL could help to generate high-quality representation when only few labeled examples are available. Specifically, for each category we employ 20, 100, 400, and 1000 labeled images in the training process respectively, and three tendencies can be observed. First, for 1000 labeled images of each category, the supervised method achieves better performance than all self-supervised ones, while for 400 samples per class the performance gap between them is decreased. That is to say, within the massive labeled data, the supervised learning paradigm can show its strength. Second, for 20 or 100 labeled samples per class, the self-supervised methods turn the tables. It indicates that the supervised method does not make much sense with fewer labeled samples, while the self-supervised methods can make full use of unlabeled data to achieve better performance. Our model with *AET* greatly surpasses the supervised with only 20 labeled samples each class (the accuracy of ours is 66.3%

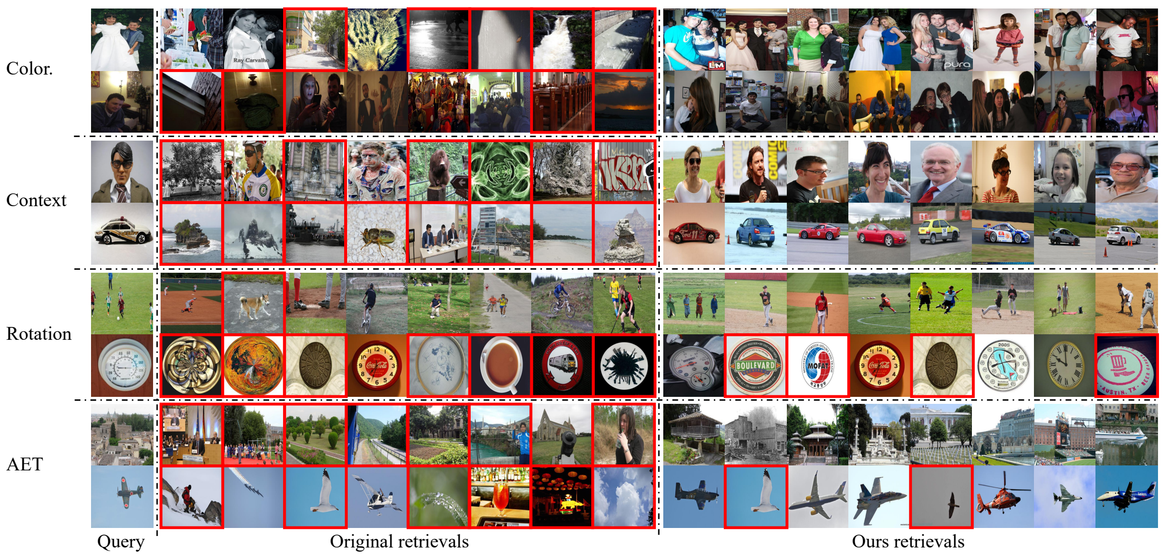

Fig. 4: **Visualization of nearest-neighbor retrieval results.** We show the eight nearest neighbors of all methods on additional 100,000 images acquired from YFCC100M [63]. Every two rows from top to bottom represent the results of Colorization [6], Context [7], Rotation [8], and AET [9], respectively. The red box marks the semantically incorrect retrieval results.

TABLE IX: **Ablations of different semantic pseudo-labeling schemes.** The *Labels* and *Random* entries denote the ground-truth labels and random pseudo-labels, respectively.

Scheme	**Rotation** [8]				**AET** [9]			
	B2	B3	B4	Avg.	B2	B3	B4	Avg.
Labels	90.3	88.1	85.0	87.8	92.6	91.9	91.4	92.0
Random	89.2	85.4	60.0	78.2	90.2	85.4	58.7	78.1
K-means	89.7	86.0	63.7	79.8	90.1	85.8	62.1	79.3
Ours	**90.1**	**86.5**	**75.9**	**84.2**	**90.9**	**86.8**	**75.7**	**84.5**

TABLE X: **Comparison to prior work**. *Decouple* is specially designed for the *Rotation* while other methods aim to promote various self-supervised tasks.

Base	Method	Layers						
		C1	C2	C3	C4	C5	Fc	Avg.
Rotation	Boosting [12]	63.3	73.6	77.2	76.3	74.3	72.0	72.8
	DeeperCluster [19]	55.3	70.5	75.1	72.9	68.6	59.2	66.9
	Decouple [17]	**64.2**	**74.9**	**78.2**	78.0	76.7	73.4	74.2
	Ours	63.6	74.5	78.0	**78.3**	**76.9**	**73.9**	**74.2**
AET	Boosting [12]	61.5	72.2	76.0	75.2	70.9	67.0	70.5
	DeeperCluster [19]	58.8	70.8	75.2	73.7	69.9	67.7	69.4
	Ours	**62.0**	**74.3**	**77.4**	**77.2**	**75.2**	**69.2**	**72.6**

on average while the supervised method is 37.5%). Third, SSL++ is in a position to improve the performance of all self-supervised models in most cases, which strongly demonstrates that SSL++ is a general framework for self-supervised models.

C. Pre-training on Tiny-YFCC100M

In curated datasets, most images are object-oriented, and the classes are well-balanced [57], so simply discarding their labels only removes part of the human supervision. In pursuit of the ultimate goal of SSL (*i.e.* no matter what kind of data is provided, SSL can still learn general visual features), we follow [19], [20] to conduct an experiment on the uncurated Tiny-YFCC100M dataset. We treat the learned weights as a pre-trained model and fine-tune it on various downstream tasks: classification, detection, and segmentation.

STL-10 classification. Following [8], [20], we report the classification accuracy of all methods in Table VII. As can be observed, compared with original methods, SSL++ can promote their performance in all layers except for *Conv1* with *Rotation* and *AET*. A possible reason for the exception is that the features in *Conv1* usually encode low-level information, while SSL++ pours more attention into high-level semantic information. Importantly, owing to the adverse impact of task-specificity, the performance of all the original methods gradually decreases from *Conv3*. However, SSL++ successfully mitigates the task-specificity and improve their performance, especially in higher layers. For example, the largest improvement (10%, 13.7%, 2.7%, and 4.7% in *Colorization*, *Context*, *Rotation*, and *AET*, respectively) is achieved on the *Fc* layer, which usually suffers from more serious task-specificity.

Classification, detection, and segmentation on PASCAL VOC. For validating generalizability, we investigate how SSL++ transfers to different visual tasks in Fig. 3. Following the conventions [20], we evaluate the learned features by fine-tuning them on the PASCAL VOC 2007 classification and detection tasks and the PASCAL VOC 2012 segmentation task. We use the publicly available testing protocol of Krähenbühl *et al.* [68] to perform multi-label classification, of Ren *et al.* [69] for detection, and of Long *et al.* [70] for segmentation. Note that we do not use the magic init in [68] in our experiments

Following [17], we fine-tune either the whole model or fix the features before *Conv5* for classification, and only fine-tune the whole model for detection and segmentation. As with the STL-10 classification task, SSL++ facilitates all original self-supervised methods in all tested tasks with a good margin. All those results suggest that, compared to the task-specific feature learned by only solving the proxy tasks, semantic features extracted by SSL++ facilitate better generalizability to various downstream visual tasks.

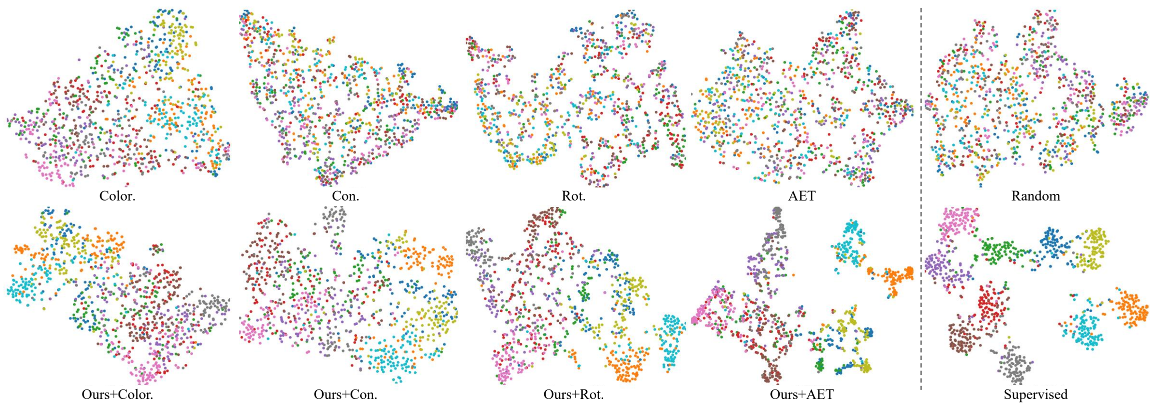

Fig. 5: **t-SNE visualization [67] of the learned feature space with the fourth blocks on randomly selected 1,000 CIFAR-10-test images.** The first four columns represent *Colorization*, *Context*, *Rotation*, and *AET*, respectively, the upper of the fifth column denotes *Random*, and the lower panel of the fifth column denotes *Supervised*. The visualization results show that SSL++ can preserve the semantics of self-supervised tasks and provide a more discriminative feature space.

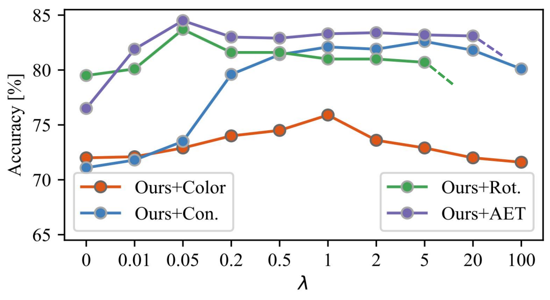

Fig. 6: **Performance with different λ values on CIFAR-10.** Following these results, we set λ= 1, 1, 0.05, and 0.05 for *Colorization*, *Context*, *Rotation*, and *AET*, respectively.

D. Qualitative Analysis

In this section we also analyze the representation quality by performing t-SNE visualizations and image retrieval.

Nearest-neighbor retrieval. As with [17], we perform image retrieval on the *Fc7* layer for all methods by calculating the cosine-similarity between features. Fig. 4 shows some retrieval examples (top-8) arranged from left to right in order of decreasing similarity. We can see that SSL++ can capture more semantic features, which are proxy task-independent, resulting in mitigating the side-effect of task-specificity. For example, on the sixth row, for retrieving clocks with rotation agnostic, SSL++ finds more clocks rather than just the semantic-unrelated circular objects in the *Rotation* method. Additionally, for some extremely complicated or simple scenes with projection ambiguity, *AET* fails to extract semantic information of objects. Take an airplane image with extremely simple background as an example, SSL++ can still excavate the high-level semanticity of *airplane* so that most of the retrieval results are *airplane*, while the original *AET* finds many irrelevant images (*e.g.*, skiing, plant, and sky).

t-SNE visualizations. We utilize the t-SNE algorithm [67] to visualize the feature space on the fourth blocks of NIN architecture in Fig. 5. For clear contrast, we present not only the feature space visualization results of the four original self-supervised tasks and our corresponding results, but also the results of fully-supervised and random model. It can be seen that the semantic relation of samples in our feature space is preserved to a good extent. Take the visualization result of *AET* as an example, it has similar chaotic feature visualizations to random models. Similar to the visualization result of supervised model, a visual sample in our feature space, especially the ‘Ours+AET’, will be close to its class-level semantic cluster while far away from clusters of other classes, resulting in enhanced generalizability of features for downstream tasks.

E. Discussion

In this section, we discuss the important aspects of SSL++, including the impact of different semantic pseudo-labeling acquiring schemes, the trade-off between two losses, *etc*.

Hyper-parameter analysis. The objective function in (5) of SSL++ contains two terms, *i.e*., self-supervision loss and classification loss. In detail, we explore different values of parameter λ in (5) varying in the set $\{0, 0.01, 0.05, 0.2, 0.5, 1, 2, 5, 20, 100\}$ to make a trade-off between the two losses. As shown in Fig. 6, the performance of the four methods will change with λ, and $\lambda = 0$ denotes using the task-specific loss alone, which suffers task-specificity, resulting in relatively poor performance. Empirically, the extreme imbalance between the two losses could affect the convergence of the model (*e.g*., *Rotation* with $\lambda > 5$ and *AET* with $\lambda > 20$). From the results, the optimal λ of *Colorization*, *Context*, *Rotation*, and *AET* are 1, 5, 0.05, 0.05, respectively. For simplicity in all our experiments, we set $\lambda = 1$ for *Colorization* and *Context* and set $\lambda = 0.05$ for *Rotation* and *AET*.

Ablation studies. To see the impact of different semantic pseudo-labeling schemes on SSL++, we conduct ablation studies on CIFAR-10 (see Table IX). We consider four pseudo-labeling schemes for the two state-of-the-art SSL tasks, *Rotation* and *AET*. In detail, we compare the ground-truth labels (*Labels*), the random pseudo-labels (*Random*), the k-means (*K-means*), and ours (using IIC for generating pseudo-labels). Note that CIFAR-10 is a 10-way classification dataset, Rotation is a 4-way classification task, and we use 70 clusters

TABLE XI: Performance of using the self-supervised proxy task pseudo-labels alone, of using the semantic pseudo-labels alone (denoted as *SPL*), and of our SSL++. *T-Y* denotes Tiny-YFCC100M dataset.

Dataset	Evaluation	Method	B2	B3	B4	Avg.
CIFAR-10	Re-training	AET	90.7	86.2	59.4	78.8
		SPL	84.1	78.7	71.1	78.0
		Ours	**90.9**	**86.8**	**75.7**	**84.5**
	KNN	AET	70.4	68.3	25.1	54.6
		SPL	61.4	64.5	61.3	62.4
		Ours	**72.1**	**75.9**	**66.4**	**71.5**
STL-10	Re-training	AET	86.5	80.5	54.5	73.8
		SPL	79.3	73.3	66.2	72.9
		Ours	**87.7**	**84.6**	**75.0**	**82.4**
	KNN	AET	50.5	48.1	18.1	38.9
		SPL	45.7	44.3	45.4	45.1
		Ours	**53.5**	**58.3**	**53.5**	**55.1**

Dataset	Method	Layers						
		C1	C2	C3	C4	C5	Fc	Avg.
T-Y	AET	62.1	72.3	76.4	76.0	73.4	64.5	70.8
	SPL	**62.4**	72.2	75.8	74.5	70.2	67.5	70.4
	Ours	62.0	**74.3**	**77.4**	**77.2**	**75.2**	**69.2**	**72.6**

with IIC (*i.e.*, a 70-way classification task). We analyze the experiments from two aspects: 1) Ours vs. Ground-Truth; 2) Ours vs. K-means. First, for fair comparison, we conduct experiments with 10 clusters based on *Rotation*. Its accuracy of *B2*, *B3*, and *B4* are 88.4%, 83.3%, and 73.5%, respectively. As expected, when the semantic pseudo-label classification branch is 10-way, the accuracy of using ground-truth labels is better than ours. Besides, using more fined clusters (#70), *i.e.*, over-clustering, the pseudo-labels are more refined. Similarly, the performance of using ground-truth labels is better. Second, we utilize PCA to make dimension reduction of original high-dimensional image data to perform stable k-means clustering. Benefiting from using deep clustering strategy, our method outperforms the k-means algorithm.

Comparison to prior work. In this experiment, we compare the prior work that aims to promote various self-supervised tasks. Specifically, we take the state-of-the-art *Rotation* and *AET* as basal tasks for all compared methods. Note that *Decouple* is specially designed for *Rotation*, and the others can employ various self-supervised methods as basal tasks. As shown in Table X, SSL++ significantly outperforms *Boosting* and *DeeperCluster* algorithms. Compared with *Decouple*, SSL++ achieves a better performance in higher layers, which suggests that high-level features extracted by our model have better generalizability. Furthermore, SSL++ is all-purpose and can promote various self-supervised tasks while *Decouple* is customized for *Rotation*.

Using the semantic pseudo-labels alone. In Section IV-D, we claim that using the semantic pseudo-labels alone can cause the model to fit noisy labels, resulting in poor performance. Here we take the state-of-the-art *AET* task as an illustration (see Table XI). Although *AET* has achieved high performance in various downstream tasks, it suffers from task specificity, *i.e.*, the quality of learned features will deteriorate towards the end of the network. Therefore, we leverage the task-independent semantic features to mitigate the task-specificity.

TABLE XII: **Performance of coupled model on CIFAR-10.** Based on AET, we train a coupled model that jointly optimizes the cluster assignments and representation learning process.

Method/Blocks	B2	B3	B4	Avg.
Coupled	85.2	81.5	67.7	78.1
Ours	**90.9**	**86.8**	**75.7**	**84.5**

TABLE XIII: Hyper-parameter about the number of clusters.

#Cluster	10	30	50	70	100
Cluster Acc.	42.1	46.1	52.3	**54.4**	50.4
(B2) Final performance	88.4	89.0	89.2	**90.1**	89.7
(B4) Final performance	73.5	74.2	75.2	**75.9**	75.1

Yet, using the semantic pseudo-labels alone can cause poor performance due to fitting noisy labels. Take the re-training evaluation protocol in CIFAR-10 as an example, *AET* is obviously affected by task-specificity (the performance gap between *B2* and *B4* is 37.9%). *SPL* returns a better performance in *B4* while achieves poor performance in *B2* and *B3*. SSL++ leverages the complementarity, between the low-level generic features learned by an SSL proxy task and the high-level semantic features newly learned by the semantic pseudo-labels, resulting in the state-of-the-art performance in all blocks.

The performance of coupled solution. In Table XII, we conduct experiments of coupled solution that jointly optimize the cluster assignments and representation learning. We maintain two branches in this coupled model, *i.e.*, the classification branch and proxy-task branch. Specifically, in the classification branch, following [20] we iteratively use k-means to cluster deep features of samples to generate pseudo-labels for the supervisory signal. The proxy-task branch aims to predict the proxy-specific pseudo-labels. From the results in Table XII, our decoupled SSL++ can achieve better performance than the coupled one, which verifies the presence of degenerate solutions [20], [22].

Hyper-parameter about the number of clusters. The number of clusters decides all the possible pseudo-labels, thus, we also conduct experiments to exploit its influence on the clustering accuracy and final classification performance. Note that the calculation of clustering accuracy refers from [18], *i.e.*, finding the best one-to-one permutation mapping between the learned clusters and the ground-truth using linear assignment [71]. Specifically, there are two situations corresponding to it: under-clustering and over-clustering. The former has a small number of clusters, so the same cluster could contain many samples of different categories. The latter has a large number of clusters, so samples of the same class could be divided into different clusters. Both of these situations will cause noise in the pseudo-label. Therefore, the number of clusters is also an important hyper-parameter. From the results in Table XIII, we fix the number of clusters to 70 in all experiments.

Ablation studies of transformations. Table XIV shows the performance of the clustering model and the fine-tuning performance of using a new *NIN block* classifier on the second or fourth block of the pre-trained model. We observe that different transformation combinations have a significant impact on the accuracy of the clustering model. For ex-

TABLE XIV: Ablation studies of transformation in the clustering module. Cr, F, Co, G, and S denote the transformation of cropping, flipping, color jitter, grey, and sobel filtering, respectively. We use the *Rotation* as a pretext task.

Weak	Cr			Cr+G	Cr+G+S
Strong	Cr	Cr+F	Cr+F+Co	Cr+F+Co+G	Cr+F+Co+G+S
B2	89.0	89.1	89.5	89.6	**90.1**
B4	60.1	61.0	66.4	74.4	**75.9**
Cluster Acc.	22.3	23.1	30.3	50.1	**54.4**

ample, when only *Crop* is used for weakly- and strongly-augmentation, the clustering accuracy is only 22.3%. However, when using a combination of the five transformations, the clustering accuracy can reach 54.4%. This is why multiple data augmentation strategies are used in [3] and [20], both of them focusing on the clustering performance. For our SSL++, the final classification performance on top of the frozen features of the second block changes slightly for different transformation combinations. We think this might be because low-level information is often learned in the first few layers of the network, such as edges and textures. These features are more versatile, thus, when evaluated with these features, the final classification performance is often not sensitive. The fourth block usually suffers from more serious task-specificity, in which case the final classification accuracy will be more affected by the clustering accuracy. The best result is 75.9% when using all the transformations.

VI. Conclusion

To combat the inherent limitation (*i.e.*, task-specificity) in SSL, we present a decoupled framework SSL++ that first leverages IIC for generating semantic pseudo-labels, and then embeds these labels into off-the-shelf self-supervision tasks. The resulting decoupled SSL++ naturally avoids the degenerate solutions in existing clustering-based SSL methods. Extensive experiments show that SSL++ can learn discriminative and generalizable features, thereby resulting in improved performance for existing methods with a good margin. We hope that this idea, utilizing semantic features to mitigate proxy task-specificity, can open a new perspective for closing the gap between SSL and the fully supervised counterparts.

Our proposed SSL++ consists of a clustering module and a representation learning module, which achieves performance gains based on existing self-supervised methods. However, due to the unsupervised way to exploit the pattern of high-dimensional images, the clustering algorithm usually converges slowly and requires a lot of training time. Thus, in the future, efficient clustering algorithms are crucial so that it can be extended to large-scale datasets. Furthermore, different proxy tasks provide different supervision signals. Therefore, combining multiple proxy tasks in the representation learning module can help the network learn more discriminative visual features.

Acknowledgement: This work was supported by the National Key Research and Development Program of China Grant (NO. 2018AAA0100403), NSFC (NO.61876094, U1933114), Natural Science Foundation of Tianjin, China (NO.20JCJQJC00020, 18JCYBJC15400, 18ZXZNGX00110).

References

[1] J. Donahue and K. Simonyan, “Large scale adversarial representation learning,” *Advances in neural information processing systems (NIPS)*, pp. 10 542–10 552, 2019.

[2] M. Noroozi and P. Favaro, “Unsupervised learning of visual representations by solving jigsaw puzzles,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 69–84.

[3] P. Goyal, D. Mahajan, A. Gupta, and I. Misra, “Scaling and benchmarking self-supervised visual representation learning,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 6391–6400.

[4] L. Jing and Y. Tian, “Self-supervised visual feature learning with deep neural networks: A survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[5] A. Coates, A. Ng, and H. Lee, “An analysis of single-layer networks in unsupervised feature learning,” in *Proceedings of the fourteenth international conference on artificial intelligence and statistics (AISTATS)*, 2011, pp. 215–223.

[6] R. Zhang, P. Isola, and A. A. Efros, “Colorful image colorization,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 649–666.

[7] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, “Context encoders: Feature learning by inpainting,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2536–2544.

[8] S. Gidaris, P. Singh, and N. Komodakis, “Unsupervised representation learning by predicting image rotations,” in *International Conference on Learning Representations (ICLR)*, 2018.

[9] L. Zhang, G.-J. Qi, L. Wang, and J. Luo, “AET vs. AED: Unsupervised representation learning by auto-encoding transformations rather than data,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2547–2555.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Advances in neural information processing systems (NIPS)*, 2012, pp. 1097–1105.

[11] G. Larsson, M. Maire, and G. Shakhnarovich, “Learning representations for automatic colorization,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 577–593.

[12] M. Noroozi, A. Vinjimoor, P. Favaro, and H. Pirsiavash, “Boosting self-supervised learning via knowledge transfer,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 9359–9367.

[13] J. Yang, D. Parikh, and D. Batra, “Joint unsupervised learning of deep representations and image clusters,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5147–5156.

[14] A. Deshpande, J. Rock, and D. Forsyth, “Learning large-scale automatic image colorization,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 567–575.

[15] G. Larsson, M. Maire, and G. Shakhnarovich, “Colorization as a proxy task for visual understanding,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6874–6883.

[16] R. Zhang, P. Isola, and A. A. Efros, “Split-brain autoencoders: Unsupervised learning by cross-channel prediction,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1058–1067.

[17] Z. Feng, C. Xu, and D. Tao, “Self-supervised representation learning by rotation feature decoupling,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 10 364–10 374.

[18] X. Ji, J. F. Henriques, and A. Vedaldi, “Invariant information clustering for unsupervised image classification and segmentation,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 9865–9874.

[19] M. Caron, P. Bojanowski, J. Mairal, and A. Joulin, “Unsupervised pre-training of image features on non-curated data,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 2959–2968.

[20] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, “Deep clustering for unsupervised learning of visual features,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 132–149.

[21] M. E. Celebi, H. A. Kingravi, and P. A. Vela, “A comparative study of efficient initialization methods for the k-means clustering algorithm,” *Expert Systems with Applications*, vol. 40, no. 1, pp. 200–210, 2013.

[22] W. Van Gansbeke, S. Vandenhende, S. Georgoulis, M. Proesmans, and L. Van Gool, “Scan: Learning to classify images without labels,” in

Proceedings of the European Conference on Computer Vision (ECCV), 2020, pp. 268–285.

[23] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *The IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1422–1430.

[24] C. Wei, L. Xie, X. Ren, Y. Xia, C. Su, J. Liu, Q. Tian, and A. L. Yuille, "Iterative reorganization with weak spatial constraints: Solving arbitrary jigsaw puzzles for unsupervised representation learning," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1910–1919.

[25] D. Jayaraman and K. Grauman, "Learning image representations tied to ego-motion," in *The IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1413–1421.

[26] R. Arandjelovic and A. Zisserman, "Look, listen and learn," in *The IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 609–617.

[27] B. Korbar, D. Tran, and L. Torresani, "Cooperative learning of audio and video models from self-supervised synchronization," in *Advances in neural information processing systems (NIPS)*, 2018, pp. 7763–7774.

[28] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan, "Learning features by watching objects move," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2701–2710.

[29] P. Agrawal, J. Carreira, and J. Malik, "Learning to see by moving," in *The IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 37–45.

[30] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *The IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2794–2802.

[31] C. Doersch and A. Zisserman, "Multi-task self-supervised visual learning," in *The IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2051–2060.

[32] X. Wang, K. He, and A. Gupta, "Transitive invariance for self-supervised visual representation learning," in *The IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 1329–1338.

[33] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning (ICML)*, 2020, pp. 1597–1607.

[34] O. Henaff, "Data-efficient image recognition with contrastive predictive coding," in *International Conference on Machine Learning (ICML)*, 2020, pp. 4182–4192.

[35] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization," in *International Conference on Learning Representations (ICLR)*, 2018.

[36] T. N. Mundhenk, D. Ho, and B. Y. Chen, "Improvements to context based self-supervised learning," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 9339–9348.

[37] A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with exemplar convolutional neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 9, pp. 1734–1747, 2015.

[38] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with convolutional neural networks," in *Advances in neural information processing systems (NIPS)*, 2014, pp. 766–774.

[39] G.-J. Qi, L. Zhang, C. W. Chen, and Q. Tian, "AVT: Unsupervised learning of transformation equivariant representations by autoencoding variational transformations," in *The IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 8130–8139.

[40] X. Wang, D. Kihara, J. Luo, and G.-J. Qi, "Enaet: A self-trained framework for semi-supervised and supervised learning with ensemble transformations," *IEEE Transactions on Image Processing*, 2020.

[41] I. Misra and L. v. d. Maaten, "Self-supervised learning of pretext-invariant representations," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 6707–6717.

[42] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 3733–3742.

[43] D. Yuan, X. Chang, P.-Y. Huang, Q. Liu, and Z. He, "Self-supervised deep correlation tracking," *IEEE Transactions on Image Processing*, vol. 30, pp. 976–985, 2020.

[44] Y. Chen, F. Wu, Z. Wang, Y. Song, Y. Ling, and L. Bao, "Self-supervised learning of detailed 3d face reconstruction," *IEEE Transactions on Image Processing*, vol. 29, pp. 8696–8705, 2020.

[45] K. Jiang, T. Zhang, Y. Zhang, F. Wu, and Y. Rui, "Self-supervised agent learning for unsupervised cross-domain person re-identification," *IEEE Transactions on Image Processing*, vol. 29, pp. 8549–8560, 2020.

[46] X. Song, S. Zhao, J. Yang, H. Yue, P. Xu, R. Hu, and H. Chai, "Spatio-temporal contrastive domain adaptation for action recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 9787–9795.

[47] X. Pan, F. Tang, W. Dong, Y. Gu, Z. Song, Y. Meng, P. Xu, O. Deussen, and C. Xu, "Self-supervised feature augmentation for large image object detection," *IEEE Transactions on Image Processing*, vol. 29, pp. 6745–6758, 2020.

[48] D. Kim, D. Cho, D. Yoo, and I. S. Kweon, "Learning image representations by completing damaged jigsaw puzzles," in *Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 793–802.

[49] P. Bojanowski and A. Joulin, "Unsupervised learning by predicting noise," in *International Conference on Machine Learning (ICML)*, 2017, pp. 517–526.

[50] R. Liao, A. Schwing, R. Zemel, and R. Urtasun, "Learning deep parsimonious representations," in *Advances in neural information processing systems (NIPS)*, 2016, pp. 5076–5084.

[51] S.-A. Rebuffi, S. Ehrhardt, K. Han, A. Vedaldi, and A. Zisserman, "Lsd-c: Linearly separable deep clusters," in *The IEEE International Conference on Computer Vision (ICCV)*, 2021, pp. 1038–1046.

[52] J. Zhang, C.-G. Li, C. You, X. Qi, H. Zhang, J. Guo, and Z. Lin, "Self-supervised convolutional subspace clustering network," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5473–5482.

[53] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems (NIPS)*, 2014, pp. 2672–2680.

[54] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[55] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv preprint arXiv:1611.03530*, 2016.

[56] A. Kolesnikov, X. Zhai, and L. Beyer, "Revisiting self-supervised visual representation learning," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1920–1929.

[57] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[58] K. Dixit and A. Balakrishnan, "Deep learning using CNNs for ball-by-ball outcome classification in sports," *Stanford University: Stanford, CA, USA*, 2016.

[59] Y. Asano, C. Rupprecht, and A. Vedaldi, "Self-labelling via simultaneous clustering and representation learning," in *International Conference on Learning Representations (ICLR)*, 2019.

[60] K. Han, S. Rebuffi, S. Ehrhardt, A. Vedaldi, and A. Zisserman, "Automatically discovering and learning new visual categories with ranking statistics," in *Intennational Conference on Learning Representations (ICLR)*, 2020.

[61] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[62] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," Tech. Rep., 2009.

[63] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li, "YFCC100M: The new data in multimedia research," *arXiv preprint arXiv:1503.01817*, 2015.

[64] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[65] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *Advances in neural information processing systems (NIPS)*, 2017.

[66] G. Qi, L. Zhang, F. Lin, and X. Wang, "Learning generalized transformation equivariant representations via autoencoding transformations." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[67] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[68] P. Krähenbühl, C. Doersch, J. Donahue, and T. Darrell, "Data-dependent initializations of convolutional neural networks," *arXiv preprint arXiv:1511.06856*, 2015.

[69] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Advances in neural information processing systems (NIPS)*, 2015, pp. 91–99.

[70] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.

[71] H. W. Kuhn, “The hungarian method for the assignment problem,” *Naval Research Logistics (NRL)*, vol. 52, no. 1, pp. 7–21, 2005.