Self-supervised Visual Feature Learning with Deep Neural Networks: A Survey

Longlong Jing, *Student Member, IEEE* and Yingli Tian*, *Fellow, IEEE*

Abstract—Large-scale labeled data are generally required to train deep neural networks in order to obtain better performance in visual feature learning from images or videos for computer vision applications. To avoid extensive cost of collecting and annotating large-scale datasets, as a subset of unsupervised learning methods, self-supervised learning methods are proposed to learn general image and video features from large-scale unlabeled data without using any human-annotated labels. This paper provides an extensive review of deep learning-based self-supervised general visual feature learning methods from images or videos. First, the motivation, general pipeline, and terminologies of this field are described. Then the common deep neural network architectures that used for self-supervised learning are summarized. Next, the schema and evaluation metrics of self-supervised learning methods are reviewed followed by the commonly used datasets for images, videos, audios, and 3D data, as well as the existing self-supervised visual feature learning methods. Finally, quantitative performance comparisons of the reviewed methods on benchmark datasets are summarized and discussed for both image and video feature learning. At last, this paper is concluded and lists a set of promising future directions for self-supervised visual feature learning.

Index Terms—Self-supervised Learning, Unsupervised Learning, Convolutional Neural Network, Transfer Learning, Deep Learning.

1 Introduction

1.1 Motivation

Due to the powerful ability to learn different levels of general visual features, deep neural networks have been used as the basic structure to many computer vision applications such as object detection [1], [2], [3], semantic segmentation [4], [5], [6], image captioning [7], etc. The models trained from large-scale image datasets like ImageNet are widely used as the pre-trained models and fine-tuned for other tasks for two main reasons: (1) the parameters learned from large-scale diverse datasets provide a good starting point, therefore, networks training on other tasks can converge faster [8], (2) the network trained on large-scale datasets already learned the hierarchy features which can help to reduce over-fitting problem during the training of other tasks, especially when datasets of other tasks are small or training labels are scarce.

The performance of deep convolutional neural networks (ConvNets) greatly depends on their capability and the amount of training data. Different kinds of network architectures were developed to increase the capacity of network models, and larger and larger datasets were collected these days. Various networks including AlexNet [9], VGG [10], GoogLeNet [11], ResNet [12], and DenseNet [13] and large scale datasets such as ImageNet [14], OpenImage [15] have been proposed to train very deep ConvNets. With the sophisticated architectures and large-scale datasets, the performance of ConvNets keeps breaking the state-of-the-arts for many computer vision tasks [1], [4], [7], [16], [17], [18].

However, collection and annotation of large-scale datasets are time-consuming and expensive. As one of the most widely used datasets for pre-training very deep 2D convolutional neural networks (2DConvNets), ImageNet [14] contains about 1.3 million labeled images covering $1,000$ classes while each image is labeled by human workers with one class label. Compared to image datasets, collection and annotation of video datasets are more expensive due to the temporal dimension. The Kinetics dataset [19], which is mainly used to train ConvNets for video human action recognition, consists of $500,000$ videos belonging to 600 categories and each video lasts around 10 seconds. It took many Amazon Turk workers a lot of time to collect and annotate a dataset at such a large scale.

To avoid time-consuming and expensive data annotations, many self-supervised methods were proposed to learn visual features from large-scale unlabeled images or videos without using any human annotations. A popular solution is to propose various pretext tasks for networks to solve, while the networks can be trained by learning objective functions of the pretext tasks and the features are learned through this process. Various pretext tasks have been proposed for self-supervised learning including colorizing grayscale images [20], image inpainting [21], playing image jigsaw puzzle [22], etc. The pretext tasks share two common properties: (1) visual features of images or videos need to be captured by ConvNets to solve the pretext tasks, (2) the supervisory signal is generated from the data itself (self-supervision) by leveraging its structure.

- *L. Jing is with the Department of Computer Science, The Graduate Center, The City University of New York, NY, 10016. E-mail: ljing@gradcenter.cuny.edu*
- *Y. Tian is with the Department of Electrical Engineering, The City College, and the Department of Computer Science, the Graduate Center, the City University of New York, NY, 10031. E-mail: ytian@ccny.cuny.edu*
  *Corresponding author

This material is based upon work supported by the National Science Foundation under award number IIS-1400802.

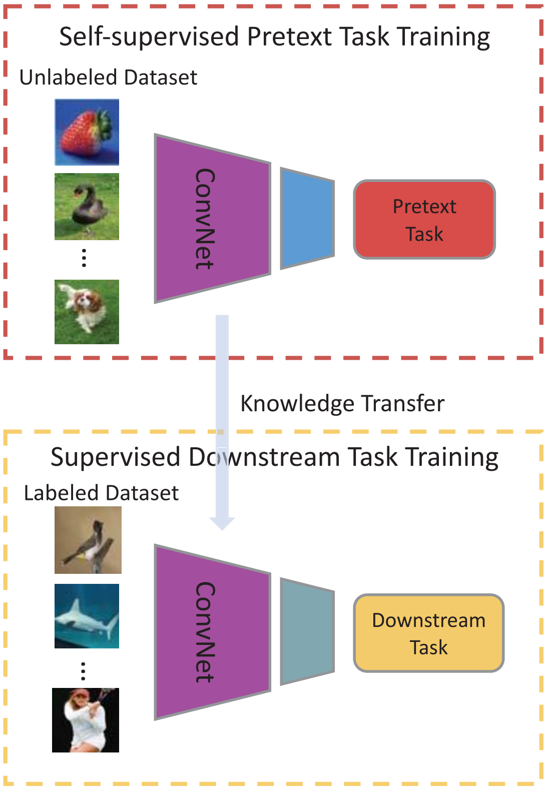

Fig. 1: The general pipeline of self-supervised learning. The visual feature is learned through the process of training ConvNets to solve a pre-defined pretext task. After self-supervised pretext task training finished, the learned parameters serve as a pre-trained model and are transferred to other downstream computer vision tasks by fine-tuning. The performance on these downstream tasks is used to evaluate the quality of the learned features. During the knowledge transfer for downstream tasks, the general features from only the first several layers are unusually transferred to downstream tasks.

The general pipeline of self-supervised learning is shown in Fig. 1. During the self-supervised training phase, a predefined pretext task is designed for ConvNets to solve, and the pseudo labels for the pretext task are automatically generated based on some attributes of data. Then the ConvNet is trained to learn object functions of the pretext task. When trained with pretext tasks, the shallower blocks of ConvNet focus on the low-level general features such as corners, edges, and textures, while the deeper blocks focus on the high-level task-specific features such as objects, scenes, and object parts [23]. Therefore, ConvNets trained with pretext tasks can learn kernels that to capture low-level features and high-level features that are helpful for other downstream tasks. After the self-supervised training finished, the learned visual features can be further transferred to downstream tasks (especially when only relatively small data available) as pre-trained models to improve performance and overcome over-fitting. Usually, visual features from only the first several layers are transferred during the supervised downstream task training phase.

1.2 Term Definition

To make this survey easy to read, we first define the terms used in the remaining sections.

- **Human-annotated label:** Human-annotated labels refer to labels of data that are manually annotated by human workers.
- **Pretext Task:** Pretext tasks are pre-designed tasks for networks to solve, and visual features are learned by learning objective functions of pretext tasks. The pretext tasks can be predictive tasks, generative tasks, contrasting tasks, or a combination of them. The supervision signal for pretext tasks is generated from the data itself based on its structure.
- **Pseudo label:** The labels used in pretext task is referred as Pseudo labels which are generated based on the structure of data for pretext tasks.
- **Downstream Task:** Downstream tasks are computer vision applications that can be used to evaluate the quality of features learned by self-supervised learning. These applications can greatly benefit from the pre-trained models when training data are scarce. In general, human-annotated labels are needed to solve the downstream tasks. However, in some applications, the downstream task can be the same as the pretext task without using any human-annotated labels.
- **Supervised Learning:** Supervised learning indicates learning methods using data with fine-grained human-annotated labels to train networks.
- **Semi-supervised Learning:** Semi-supervised learning refers to learning methods using a small amount of labeled data in conjunction with a large amount of unlabeled data.
- **Weakly-supervised Learning:** Weakly supervised learning refers to learning methods to learn with coarse-grained labels or inaccurate labels. The cost of obtaining weak supervision labels is generally much cheaper than fine-grained labels for supervised methods.
- **Unsupervised Learning:** Unsupervised learning refers to learning methods without using any human-annotated labels.
- **Self-supervised Learning:** Self-supervised learning is a subset of unsupervised learning methods. Self-supervised learning refers to learning methods in which ConvNets are explicitly trained with supervisory signals that are generated from the data itself (self-supervision) by leveraging its structure. This review only focuses on self-supervised learning methods for visual feature learning with ConvNets in which the features can be transferred to multiple different computer vision tasks.

The formulation of different learning schemes can be found in Appendix Section 1. Since no human annotations are needed to generate pseudo labels during self-supervised training, a main advantage of self-supervised learning methods is that they can be easily scaled to large-scale datasets with very low cost. Trained with these pseudo labels, self-supervised methods achieved promising results and the gap with supervised methods in performance on downstream tasks becomes smaller. This paper provides a comprehensive survey of deep ConvNets-based self-supervised visual feature learning methods. The key contributions of this paper are as follows:

- To the best of our knowledge, this is the first comprehensive survey about self-supervised visual feature learning with deep ConvNets which will be helpful for researchers in this field.
- An in-depth review of recently developed self-supervised learning methods and datasets.
- Quantitative performance analysis and comparison of the existing methods are provided.
- A set of possible future directions for self-supervised learning is pointed out.

2 Common Deep Network Architectures

No matter the categories of learning methods, they share similar network architectures. Various 2DConvNets have been designed for image feature learning including AlexNet [9], VGG [10], GoogLeNet [11], ResNet [12], and DenseNet [13], etc. To extract both spatial and temporal information from videos, several architectures have been designed for video feature learning including 2DConvNet-based methods [24], 3DConvNet-based methods [17], and LSTM-based methods [25]. The 2DConvNet-based methods apply 2DConvNet on every single frame and the image features of multiple frames are fused as video features. The 3DConvNet-based methods employ 3D convolution operation to simultaneously extract both spatial and temporal features from multiple frames. The LSTM-based methods employ LSTM to model long term dynamics within a video. The detailed description of these ConvNets can be found in Section 2 of the Appendix.

Deep ConvNets have demonstrated great potential in various computer vision tasks. And the visualization of the image and video features has shown that these networks truly learned meaningful features that required by the corresponding tasks [23], [26], [27], [28]. However, one common drawback is that these networks can be easily over-fit when training data are scarce since there are over millions of parameters in each network.

Take 3DResNet for an example, the performance of an 18-layer 3DResNet on UCF101 action recognition dataset [29] is 42% when trained from scratch. However, with a supervised pre-trained model on the large-scale Kinetics dataset ($500,000$ videos of 600 classes) with human-annotated class labels and then fine-tuned on UCF101 dataset, the performance can increase to 84%. Pre-trained models on large-scale datasets can speed up the training process and improve the performance on relatively small datasets. However, the cost of collecting and annotating large-scale datasets is very expensive and time-consuming.

In order to obtain pre-trained models from large-scale datasets without expensive human annotations, many self-supervised learning methods were proposed to learn visual features from pre-designed pretext tasks. The next section describes the general pipeline of the self-supervised feature learning.

3 Commonly used Pretext and Downstream Tasks

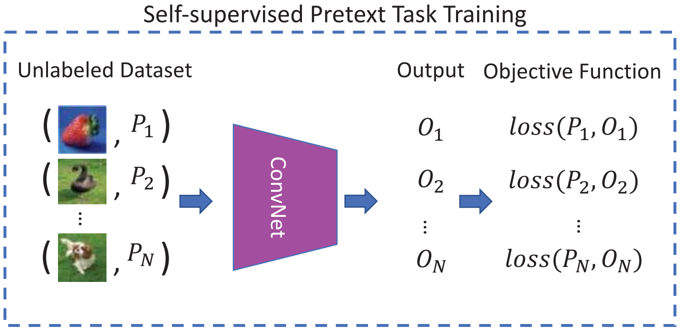

Fig. 2: Self-supervised visual feature learning schema. The ConvNet is trained by minimizing errors between pseudo labels P and predictions O of the ConvNet. Since the pseudo labels are generated based on the structure of the data, no human annotations are involved during the whole process.

Most existing self-supervised learning approaches follow the schema shown in Fig. 2. Generally, a pretext task is defined for ConvNets to solve and visual features can be learned through the process of accomplishing this pretext task. The pseudo labels P for pretext task can be automatically generated without human annotations. ConvNet is optimized by minimizing the error between the prediction of ConvNet O and the pseudo labels P. After the training on the pretext task is finished, ConvNet models that can capture visual features for images or videos are obtained.

3.1 Learning Visual Features from Pretext Tasks

To relieve the burden of large-scale dataset annotation, a pretext task is generally designed for networks to solve while pseudo labels for the pretext task are automatically generated based on data attributes. Many pretext tasks have been designed and applied for self-supervised learning such as foreground object segmentation [30], image inpainting [21], clustering [31], image colorization [32], temporal order verification [33], visual audio correspondence verification [34], and so on. Effective pretext tasks ensure that semantic features are learned through the process of accomplishing the pretext tasks.

Take image colorization as an example, image colorization is a task to colorize gray-scale images into colorful images. To generate realistic colorful images, networks are required to learn the structure and context information of images. In this pretext task, the data X is the gray-scale images which can be generated by performing a linear transformation in RGB images, while the pseudo label P is the RGB image itself. The training pair X_i and P_i can be generated in real time with negligible cost. Self-Supervised learning with other pretext tasks follows a similar pipeline.

3.2 Commonly Used Pretext Tasks

According to the data attributes used to design pretext tasks, we summarize the pretext tasks into four categories: generation-based, context-based, free semantic label-based,

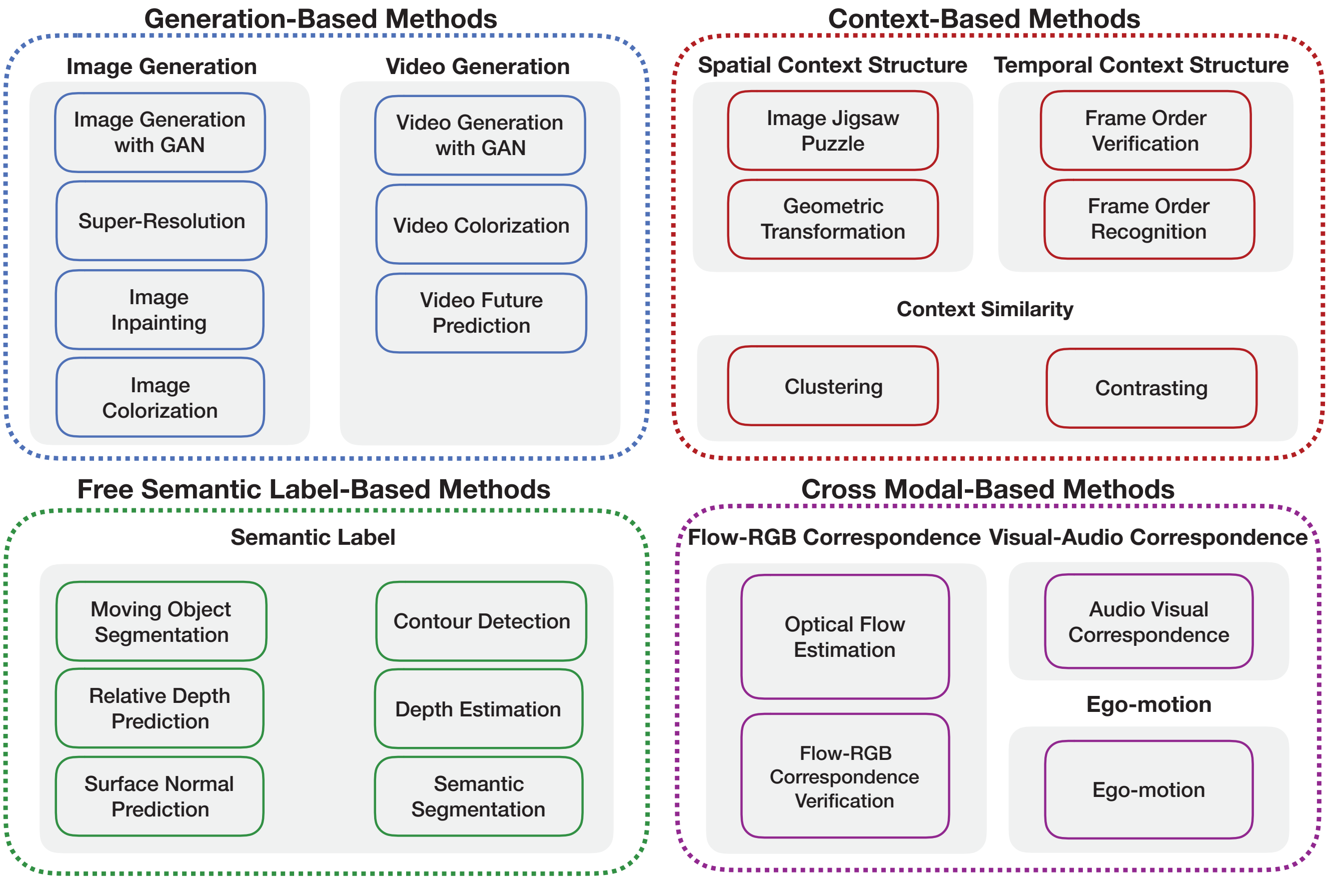

Fig. 3: Categories of pretext tasks for self-supervised visual feature learning: generation-based, context-based, free semantic label-based, and cross modal-based.

and cross modal-based. Most of the methods belong to one category. However, some methods may belong to more than one category.

Generation-based Methods: This type of methods learn visual features by solving pretext tasks that involve image or video generation.

- **Image Generation:** Visual features are learned through the process of image generation tasks. This type of methods includes image colorization [20], image super resolution [16], image inpainting [21], image generation with Generative Adversarial Networks (GANs) [35], [36].
- **Video Generation:** Visual features are learned through the process of video generation tasks. This type of methods includes video generation with GANs [37], [38] and video prediction [39].

Context-based pretext tasks: The design of context-based pretext tasks mainly employ the context features of images or videos such as context similarity, spatial structure, temporal structure, etc.

- **Context Similarity:** Pretext tasks are designed based on the context similarity between image patches. This type of methods includes image clustering-based methods [31], [40], and graph constraint-based methods [41].
- **Spatial Context Structure:** Pretext tasks are used to train ConvNets are based on the spatial relations among image patches. This type of methods includes image jigsaw puzzle [22], [42], [43], [44], context prediction [45], and geometric transformation recognition [46], [47], etc.
- **Temporal Context Structure:** The temporal order from videos is used as supervision signal. The ConvNet is trained to verify whether the input frame sequence in correct order [33], [48] or to recognize the order of the frame sequence [49], [50].

Free Semantic Label-based Methods: This type of pretext tasks train networks with automatically generated semantic labels. The labels are generated by traditional hard-code algorithms [51], [52] or by game engines [53]. As long as no human-annotations are involved through the design of hard-code algorithms, the detectors can be used to generate labels for self-supervised training. Strictly speaking, the methods based on data generated by game engines do not belong to the self-supervised learning methods since human intervention is needed during the data generation process. However, some recent work treat them as self-supervised learning methods [53], [54]. To make this survey paper extensive without missing important work, we also include this type of methods here such as moving object segmentation [30], [55], contour detection [53], [56], relative depth prediction [57], and etc.

Cross Modal-based Methods: This type of pretext tasks trains ConvNets to verify whether two different channels of input data are corresponding to each other such as Visual-Audio Correspondence Verification [34], [58], RGB-Flow Correspondence Verification [59], Contrasting [60], and egomotion [61], [62].

3.3 Commonly Used Downstream Tasks for Evaluation

To evaluate the quality of the learned image or video features by self-supervised methods, the learned parameters by self-supervised learning are employed as pre-trained models and then fine-tuned on downstream tasks such as image classification, semantic segmentation, object detection, and action recognition etc. The performance of the transfer learning on these high-level vision tasks demonstrates the generalizability of the learned features. If ConvNets of self-supervised learning can learn general features, then the pre-trained models can be used as a good starting point for other vision tasks that require capturing similar features from images or videos.

Image classification, semantic segmentation, and object detection usually are used as the tasks to evaluate the generalizability of the learned image features by self-supervised learning methods, while human action recognition in videos is used to evaluate the quality of video features obtained from self-supervised learning methods. Below are brief introductions of the commonly used high-level tasks for visual feature evaluation.

3.3.1 *Semantic Segmentation*

Semantic segmentation, the task of assigning semantic labels to each pixel in images, is of great importance in many applications such as autonomous driving, human-machine interaction, and robotics. The community has recently made promising progress and various networks have been proposed such as Fully Convolutional Network (FCN) [4], DeepLab [5], PSPNet [6] and datasets such as PASCAL VOC [63], CityScape [64], ADE20K [65]. Among all these methods, FCN [4] is a milestone work for semantic segmentation since it started the era of applying fully convolution network (FCN) to solve this task. 2DConvNet such as AlexNet, VGG, ResNet is used as the base network for feature extraction while the fully connected layer is replaced by transposed convolution layer to obtain the dense prediction. The network is trained end-to-end with pixel-wise annotations.

When using semantic segmentation as downstream task to evaluate the quality of image features learned by self-supervised learning methods, the FCN is initialized with the parameters trained with the pretext task and fine-tuned on the semantic segmentation dataset, then the performance on the semantic segmentation task is evaluated and compared with that of other self-supervised methods.

3.3.2 *Object Detection*

Object Detection, a task of localizing the position of objects in images and recognizing the category of the objects, is also very import for many computer vision applications such as autonomous driving, robotics, scene text detection and so on. Recently, many datasets such as MSCOCO [66] and OpenImage [15] have been proposed for object detection and many ConvNet-based models [1], [2], [3], [67], [68], [69], [70], [71] have been proposed and obtained great performance. Fast-RCNN [2] is a two-stage network for object detection. Object proposals are firstly generated based on feature maps produced by a convolution neural network, then these proposals are fed to several fully connected layers to generate the bounding box of objects and the categories of these objects.

When using object detection as downstream task to evaluate the quality of the self-supervised image features, networks that trained with the pretext task on unlabeled large data are served as the pre-trained model for the Fast-RCNN [2] and then fine-tuned on object detection datasets, then the performance on the object detection task is evaluated to demonstrate the generalizability of self-supervised learned features.

3.3.3 *Image Classification*

Image Classification is a task of recognizing the category of objects in each image. Many networks have been designed for this task such as AlexNet [9], VGG [10], ResNet [12], GoogLeNet [11], DenseNet [13], etc. Usually, only one class label is available for each image although the image may contains different classes of objects.

When choosing image classification as a downstream task to evaluate the quality of image features learned from self-supervised learning methods, the self-supervised learned model is applied on each image to extract features which then are used to train a classifier such as Support Vector Machine (SVM) [72]. The classification performance on testing data is compared with other self-supervised models to evaluate the quality of the learned features.

3.3.4 *Human Action Recognition*

Human action recognition is a task of identifying what people doing in videos for a list of pre-defined action classes. Generally, videos in human action recognition datasets contain only one action in each video [19], [29], [73]. Both the spatial and temporal features are needed to accomplish this task.

The action recognition task is often used to evaluate the quality of video features learned by self-supervised learning methods. The network is first trained on unlabeled video data with pretext tasks, then it is fine-tuned on action recognition datasets with human annotations to recognize the actions. The testing performance on action recognition task is compared with other self-supervised learning methods to evaluate the quality of the learned features.

3.3.5 *Qualitative Evaluation*

In addition to these quantitative evaluations of the learned features, there are also some qualitative visualization methods to evaluate the quality of self-supervised learning features. Three methods are often used for this purpose: kernel visualization, feature map visualization, and image retrieval visualization [31], [45], [46], [47].

Kernel Visualization: Qualitatively visualize the kernels of the first convolution layer learned with the pretext tasks and compare the kernels from supervised models. The

TABLE 1: Summary of commonly used datasets of images, videos, audios, and 3D object data. Note that video datasets can be used to learn both image and video features.

Dataset	Data Type	Size	Synthetic	# classes	Groundtruth Labeling
CIFAR10 [74]	Image	60,000 Images	No	10	Object category label
ImageNet [14]	Image	1.3 million images	No	1,000	Object category label
MNIST [75]	Image	70,000 images	No	10	Digit class label
PASCAL VOC [63]	Image	2,913 images	No	20	Category label, bounding box, segmentation mask
Places [76]	Image	2.5 million images	No	205	scene categories label
Places365 [77]	Image	10 million images	No	434	scene categories label
STL-10 [78]	Image	101,300 Images	No	10	Object category label
SUNCG [79]	Image	150,000 images	Yes	84	depth, volumetric data
SVHN [80]	Image	600,000 Images	No	10	Digit class label
AudioSet [81]	Video/Audio	2 million 10-second videos	No	632	Audio event class
HMDB51 [73]	Video	6,766 videos	No	51	Human action class
Kinetics [19]	Video	0.5 million 10-second videos	No	600	Human action class
KITTI [82]	Video	28 videos	No	—	Data captured by various sensors are available
Moment-in-Time [83]	Video	1 million 3-second videos	No	339	Video category class
SceneNet RGB-D [84]	Video	5 million images	Yes	13	Depth, Instance Segmentation, Optical Flow
UCF101 [29]	Video	10,031 videos	No	101	Human action class
YFCC100M [85]	Image/Video	100 million media data	No	—	Hashtags
ModelNet40 [86]	Mesh	12,311 mesh objects	Yes	40	Object category label
ShapeNet [87]	Mesh	57,447 mesh objects	Yes	55	Object category label
ShapeNet-PartSeg [88]	Point Cloud	12,137 point cloud objects	Yes	16	Object category label and point level part label
DCASE [89]	Audio	100 audio clips	No	10	Audio category label
ESC50 [90]	Audio	2,000 audio clips	No	50	Audio category label

similarity of the kernels learned by supervised and self-supervised models are compared to indicate the effectiveness of self-supervised methods [31], [47].

Feature Map Visualization: Feature maps are visualized to show the attention of networks. Larger activation represents the neural network pays more attention to the corresponding region in the image. Feature maps are usually qualitatively visualized and compared with that of supervised models [46], [47].

Nearest Neighbor Retrieval: In general, images with similar appearance usually are closer in the feature space. The nearest neighbor method is used to find the top K nearest neighbors from the feature space of the features learned by the self-supervised learned model [33], [41], [45], [50].

3.4 Beyond 2D Self-supervised Learning

Self-supervised learning methods have shown great potential in image and video domain, and some of these methods can be easily extended to other modalities of data like audio, natural language, and 3D vision data. As an important aspect of computer vision research, now more and more researchers attempt to apply self-supervised learning methods on 3D data including point cloud, mesh, and multi-view images [91]. Since some of the methods designed for images and videos can be directly applied to 3D self-supervised learning and there is a significant overlap between different mythologies, we only compare the performance of 3D self-supervised learning (see Section 7.3) to show the potential of self-supervised learning to different domains or modalities.

4 Datasets

This section summarizes the commonly used datasets for training and evaluating self-supervised visual feature learning methods. Datasets collected for supervised learning can be used for self-supervised training without using their human-annotated labels. Evaluations of the quality of learned features are usually conducted by fine-tuned on high-level vision tasks with relatively small datasets (normally with accurate labels) such as video action recognition, object detection, semantic segmentation, etc. It is worth noting that networks use these synthetic datasets for visual feature learning are included as self-supervised learning methods in this paper since labels of synthetic datasets are automatically generated by game engines and no human annotations are involved.

Commonly used image datasets include ImageNet [14], Places [76], Places365 [77], SUNCG [79], MNIST [75], SVHN [80], CIFAR10 [74], STL-10 [78], PASCAL VOC [63], commonly used video datasets include YFCC100M [85], SceneNet RGB-D [84], Moment-in-Time [83], Kinetics [19], AudioSet [81], KITTI [82], UCF101 [29], HMDB51 [73], commonly used audio datasets include AudioSet [81] ESC50 [90], DCASE [89], and commonly used 3D object datasets include ShapeNet [87], ModelNet40 [86], and ShapeNet-PartSeg [88]. Table 1 summarizes the commonly used image, video, audio, and 3D object datasets, and detailed descriptions of these datasets can be found in Section 3 of the Appendix.

5 Image Feature Learning

In this section, three groups of self-supervised image feature learning methods are reviewed including generation-based methods, context-based methods, and free semantic label-based methods. A list of the image feature self-supervised learning methods can be found in Table 2. Since the cross modal-based methods mainly learn features from videos and most methods of this type can be used for both image and video feature learning, so cross modal-based methods are reviewed in the video feature learning section.

5.1 Generation-based Image Feature Learning

Generation-based self-supervised methods for learning image features involve the process of generating images in-

TABLE 2: Summary of self-supervised image feature learning methods based on the category of pretext tasks. ”Multiple” means the method explicitly or implicitly uses multiple pretext tasks for image feature learning.

Method	Category	Code	Contribution
AutoColor [92]	Generation	Yes	Training ConvNet to predict per-pixel color histograms
BiGAN [93]	Generation	Yes	Bidirectional GAN to project data into latent space
ColorfulColorization [20]	Generation	Yes	Posing image colorization as a classification task
Colorization [32]	Generation	Yes	Using image colorization as the pretext task
CompletNet [94]	Generation	Yes	Employing two discriminators to guarantee local and global consistent
Context Encoder [21]	Generation	Yes	Employing ConvNet to solve image inpainting
DCGAN [95]	Generation	Yes	Deep convolutional GAN for image generation
GAN [35]	Generation	Yes	Forerunner of GAN
SelfGAN [96]	Multiple	No	Use rotation recognition and GAN for self-supervised learning
Split-Brain [97]	Generation	Yes	Using split-brain auto-encoder as the pretext task
SpotArtifacts [98]	Generation	Yes	Learning by recognizing synthetic artifacts in images
SRGAN [16]	Generation	Yes	Employing GAN for single image super-resolution
WGAN [99]	Generation	Yes	Proposed WGAN which makes the training of GAN more stable
Arbitrary Jigsaw [43]	Context	No	Learning with jigsaw puzzles with arbitrary grid size and dimension
Boosting [40]	Multiple	No	Using clustering to boost the self-supervised learning methods
ClusterEmbegging [100]	Context	Yes	Deep embedded clustering for self-supervised learning
CMC [101]	Context	Yes	Learning by contrasting multi-views of the data
Context Prediction [45]	Context	Yes	Learning by predicting the relative position of two patches from an image
CPC [102]	Context	Yes	Learning features by predicting the future in latent space with autoregressive models
Damaged Jigsaw [44]	Multiple	No	Learning by solving jigsaw puzzle, inpainting, and colorization together
DeepCluster [31]	Context	Yes	Using clustering as the pretext
DeepPermNet [103]	Context	Yes	A new method to solve image patch jigsaw puzzle
GraphConstraint [41]	Context	Yes	Learning with image pairs mined with Fisher Vector
ImproveContext [104]	Context	No	Techniques to improve context based self-supervised learning methods
Jigsaw [22]	Context	Yes	Image patch Jigsaw puzzle as the pretext task for self-supervised learning
JointCluster [105]	Context	Yes	Jointly learning of deep representations and image clusters
Learning2Count [106]	Context	Yes	Learning by counting visual primitive
MoCo [107]	Context	Yes	Contrastive learning of visual representations with a memory bank
MultiTask [108]	Multiple	Yes	Using multiple pretext tasks for self-supervised feature learning
PredictNoise [109]	Context	Yes	Learning by mapping images to a uniform distribution over a manifold
Ranking [110]	Context	Yes	Learning by ranking video frames with a triplet loss
RotNet [46]	Context	Yes	Learning by recognizing rotations of images
SimCLR [60]	Context	Yes	Contrastive learning of visual representations
CrossDomain [53]	Free Semantic Label	Yes	Utilizing synthetic data and its labels rendered by game engines
EdgeDetection [56]	Free Semantic Label	Yes	Learning by detecting edges
WatchingMove [30]	Free Semantic Label	Yes	Learning by grouping pixels of moving objects in videos

cluding image generation with GAN (to generate fake images), super-resolution (to generate high-resolution images), image inpainting (to predict missing image regions), and image colorization (to colorize gray-scale images into colorful images). For these tasks, pseudo training labels P usually are the images themselves and no human-annotated labels are needed during training, therefore, these methods belong to self-supervised learning methods.

The pioneer work about the image generation-based methods is the Autoencoder [111] which learns to compress an image into a low-dimension vector and then is uncompressed into the image that closes to the original image with a bunch of layers. With an auto-encoder, networks can reduce the dimension of an image into a lower-dimensional vector that contains the main information of the original image. Variational Autoencoder (VAE) is an improved version of Autoencoder which estimates the Probability Density Function (PDF) of the training data. The current image generation-based methods follow a similar idea but with different pipelines to learn visual features through the process of image generation.

5.1.1 Image Generation with GAN

Generative Adversarial Network (GAN) is a type of deep generative model that was proposed by Goodfellow *et al.* [35]. A GAN model generally consists of two kinds of networks: a generator which is to generate images from latent vectors and a discriminator which is to distinguish whether the input image is generated by the generator. By playing the two-player game, the discriminator forces the generator to generate realistic images, while the generator forces the discriminator to improve its differentiation ability. During the training, the two networks are competing against with each other and make each other stronger.

The common architecture for the image generation from a latent variable task is shown in Fig. 4. The generator is trained to map any latent vector sampled from latent space into an image, while the discriminator is forced to distinguish whether the image from the real data distribution or generated data distribution. Therefore, the discriminator is required to capture the semantic features from images to accomplish the task. The parameters of the discriminator can server as the pre-trained model for other computer vision tasks.

Most of the methods for image generation from random variables do not need any human-annotated labels. However, the main purpose of this type of task is to generate realistic images instead of obtaining better performance on downstream applications. Generally, the inception scores of the generated images are used to evaluate the quality of the generated images [112], [113]. And only a few methods evaluated the quality of the feature learned by the discriminator on the high-level tasks and compared with others [93], [95], [96].

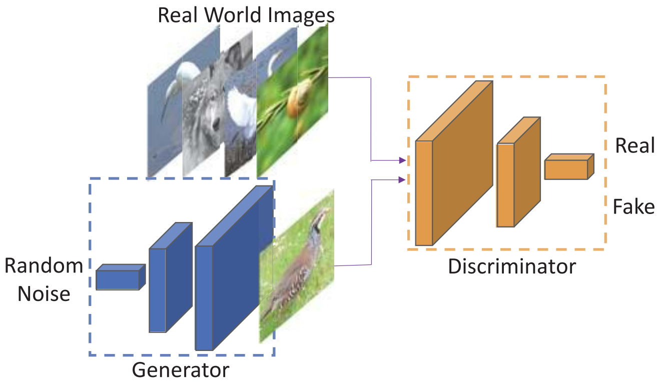

Fig. 4: The pipeline of Generative Adversarial Networks [35]. By playing the two-player game, the discriminator forces the generator to generate realistic images, while the generator forces the discriminator to improve its differentiation ability.

The adversarial training can help the network to capture the real distribution of the real data and generate realists data, and it has been widely used in computer vision tasks such as image generation [114], [115], video generation [37], [38], super-resolution [16], image translation [116], and image inpainting [21], [94]. When there is no human-annotated label involves, the method falls into the self-supervised learning.

5.1.2 Image Generation with Inpainting

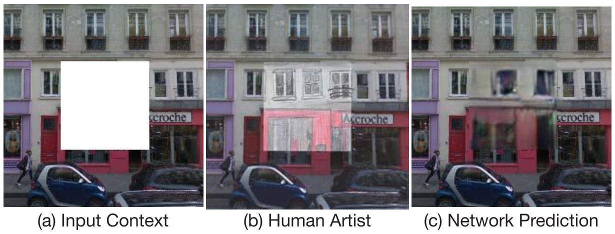
(a) Input Context
(b) Human Artist
(c) Network Prediction

Fig. 5: Qualitative illustration of image inpainting task. Given an image with a missing region (a), a human artist has no trouble inpainting it (b). Automatic inpainting using context encoder proposed in [21] trained with $L2$ reconstruction loss and adversarial loss is shown in (c). This figure is reproduced based on [21].

Image inpainting is a task of predicting arbitrary missing regions based on the rest of an image. A qualitative illustration of the image inpainting task is shown in Fig. 5. Fig. 5(a) is an image with a missing region, while Fig. 5(c) is the prediction of networks. To correctly predict missing regions, networks are required to learn the common knowledge including the color and structure of the common objects. Only by knowing this knowledge, networks are able to infer missing regions based on the rest part of the image.

By analogy with auto-encoders, Pathak *et al.* made the first step to train a ConvNet to generate the contents of an arbitrary image region based on the rest of the image [21]. Their contributions are in two folds: using a ConvNet to tackle image inpainting problem, and using the adversarial loss to help the network generate a realistic hypothesis. Most of the recent methods follow a similar pipeline [94]. Usually, there are two kinds of networks: a generator network is to generate the missing region with the pixel-wise reconstruction loss and a discriminator network is to distinguish whether the input image is real with an adversarial loss. With the adversarial loss, the network is able to generate sharper and realistic hypothesis for the missing image region. Both the two kinds of networks are able to learn the semantic features from images and can be transferred to other computer vision tasks. However, only Pathak *et al.* [21] studied the performance of transfer learning for the learned parameters of the generator from the image inpainting task.

The generator network which is a fully convolutional network has two parts: encoder and decoder. The input of the encoder is the image that needs to be inpainted and the context encoder learns the semantic feature of the image. The context decoder is to predict the missing region based on this feature. The generator is required to understand the content of the image in order to generate a plausible hypothesis. The discriminator is trained to distinguish whether the input image is the output of the generator. To accomplish the image inpainting task, both networks are required to learn semantic features of images.

5.1.3 Image Generation with Super Resolution

Image super-resolution (SR) is a task of enhancing the resolution of images. With the help of fully convolutional networks, finer and realistic high-resolution images can be generated from low-resolution images. SRGAN is a generative adversarial network for single image super-resolution proposed by Ledig et al. [16]. The insight of this approach is to take advantage of the perceptual loss which consists of an adversarial loss and a content loss. With the perceptron loss, the SRGAN is able to recover photo-realistic textures from heavily downsampled images and show significant gains in perceptual quality.

There are two networks: one is generator which is to enhance the resolution of the input low-resolution image and the other is the discriminator which is to distinguish whether the input image is the output of the generator. The loss function for the generator is the pixel-wise $L2$ loss plus the content loss which is the similarity of the feature of the predicted high-resolution image and the high-resolution original image, while the loss for the discriminator is the binary classification loss. Compared to the network that only minimizing the Mean Squared Error (MSE) which generally leads to high peak signal-to-noise ratios but lacking high-frequency details, the SRGAN is able to recover the fine details of the high-resolution image since the adversarial loss pushes the output to the natural image manifold by the discriminator network.

The networks for image super-resolution task are able to learn the semantic features of images. Similar to other GANs, the parameters of the discriminator network can be transferred to other downstream tasks. However, no one tested the performance of the transferred learning on other tasks yet. The quality of the enhanced image is mainly compared to evaluate the performance of the network.

5.1.4 Image Generation with Colorization

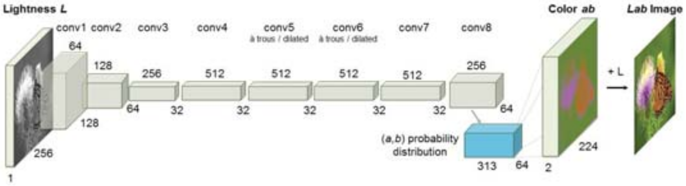

Fig. 6: The architecture of image colorization proposed in [20]. The figure is from [20] with author's permission.

Image colorization is a task of predicting a plausible color version of the photograph given a gray-scale photograph as input. A qualitative illustration of the image colorization task is shown in Fig. 6. To correctly colorize each pixel, networks need to recognize objects and to group pixels of the same part together. Therefore, visual features can be learned in the process of accomplishing this task.

Many deep learning-based colorization methods have been proposed in recent years [20], [117], [118]. A straightforward idea would be to employ a fully convolution neural network which consists of an encoder for feature extraction and a decoder for the color hallucination to colorization. The network can be optimized with $L2$ loss between the predicted color and its original color. Zhang *et al.* proposed to handle the uncertainty by posting the task as a classification task and used class-rebalancing to increase the diversity of predicted colors [20]. The framework for image colorization proposed by Zhang *et al.* is shown in Fig. 6. Trained in large-scale image collections, the method shows great results and fools human on 32% of the trials during the colorization test.

Some work specifically employs the image colorization task as the pretext for self-supervised image representation learning [20], [32], [92], [97]. After the image colorization training is finished, the features learned through the colorization process are specifically evaluated on other downstream high-level tasks with transfer learning.

5.2 Context-Based Image Feature Learning

The context-based pretext tasks mainly employ the context features of images including context similarity, spatial structure, and temporal structure as the supervision signal. Features are learned by ConvNet through the process of solving the pretext tasks designed based on attributes of the context of images.

5.2.1 Learning with Context Similarity

There are two ways of utilizing context similarity as supervision signals for self-supervised learning: formulating it as a predictive task or a contrastive task. For both methods, the data are firstly clustered into different groups under assumptions of the data from the same group have high context similarity, while data from different groups have low context similarity. The predictive tasks involve training networks to predict the group ID of the data and usually with cross entropy loss [31]. The contrastive tasks involve training networks to directly minimize feature distances from the same group and maximize feature distances from different groups with triplet loss or contrastive loss [60].

Clustering is a method of grouping sets of similar data in the same clusters [119]. In the self-supervised scenario, the clustering methods mainly employed as a tool to cluster image data. A naive method would be to cluster the image data based on the hand-designed feature such as HOG [120], SIFT [121], or Fisher Vector [122]. After the clustering, several clusters are obtained while the image within one cluster has a smaller distance in feature space and images from different clusters have a larger distance in feature space. The smaller the distance in feature space, the more similar the image in the appearance in the RGB space. Then a ConvNet can be trained to classify the data by using the cluster assignment as the pseudo class label. To accomplish this task, the ConvNet needs to learn the invariance within one class and the variance among different classes. Therefore, the ConvNet is able to learn semantic meaning of images.

The existing methods about using the clustering variants as the pretext task follow these principals [31], [40], [41], [100], [105]. Firstly, the image is clustered into different clusters which the images from the same cluster have smaller distance and images from different clusters have larger distance. Then a ConvNet is trained to recognize the cluster assignment [31], [40] or to recognize whether two imaged are from same cluster [41]. Since the clustering and self-supervised training are two separated steps, various clustering methods can be used to generate reliable clusters.

Another way of leveraging context similarly for self-supervised image feature learning is by contrasting [60], [101], [102], [107]. The general idea of the contrastive self-supervised learning is to train networks to maximum agreement of different views of same scene while minimizing agreement of views from different scenes. The recent state-of-the-art method is SimCLR proposed by Chen *et al.* which learns features by contrasting images after a composition of data augmentations [60]. The positive pairs are constructed by sampling two images after applying different augmentation techniques for the same image, while negative pairs include two different images. This method significantly outperforms other self-supervised learning methods on ImageNet dataset. It has been extended to other modalities of data [123], [124].

5.2.2 Learning with Spatial Context Structure

Images contain rich spatial context information such as the relative positions among different patches from an image which can be used to design the pretext task for self-supervised learning. The pretext task can be to predict the relative positions of two patches from same image [45], or to recognize the order of the shuffled a sequence of patches from same image [22], [43], [44]. The context of full images can also be used as a supervision signal to design pretext tasks such as to recognize the rotating angles of the whole images [46]. To accomplish these pretext tasks, ConvNets need to learn spatial context information such as the shape of the objects and the relative positions of different parts of an object.

The method proposed by Doersch *et al.* is one of the pioneer work of using spatial context cues for self-supervised visual feature learning [45]. Random pairs of image patches are extracted from each image, then a ConvNet is trained

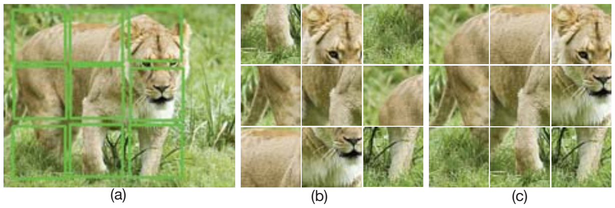

Fig. 7: The visualization of the Jigsaw Image Puzzle [22]. (a) is an image with 9 sampled image patches, (b) is an example of shuffled image patches, and (c) shows the correct order of the sampled 9 patches. Figure is reproduced based on [22].

to recognize the relative positions of the two image patches. To solve this puzzle, ConvNets need to recognize objects in images and learn the relationships among different parts of objects. To avoid the network learns trivial solutions such as simply using edges in patches to accomplish the task, heavy data augmentation is applied during the training phase.

Following this idea, more methods are proposed to learn image features by solving more difficult spatial puzzles [22], [42], [43], [44], [125]. As illustrated in Fig. 7, one typical work proposed by Noroozi *et al.* attempted to solve an image Jigsaw puzzle with ConvNet [22]. Fig. 7(a) is an image with 9 sampled image patches, Fig. 7(b) is an example of shuffled image patches, and Fig. 7(c) shows the correct order of the sampled 9 patches. The shuffled image patches are fed to the network which trained to recognize the correct spatial locations of the input patches by learning spatial context structures of images such as object color, structure, and high-level semantic information.

Given 9 image patches from an image, there are $362,880$ $(9!)$ possible permutations and a network is very unlikely to recognize all of them because of the ambiguity of the task. To limit the number of permutations, usually, hamming distance is employed to choose only a subset of permutations among all the permutations that with relative large hamming distance. Only the selected permutations are used to train ConvNet to recognize the permutation of shuffled image patches [22], [43], [44], [126], [127].

The main principle of designing puzzle tasks is to find a suitable task which is not too difficult and not too easy for a network to solve. If it is too difficult, the network may not converge due to the ambiguity of the task or can easily learn trivial solutions if it is too easy. Therefore, a reduction in the search space is usually employed to reduce the difficulty of the task.

5.3 Free Semantic Label-based Image Feature Learning

The free semantic label refers to labels with semantic meanings that obtained without involving any human annotations. Generally, the free semantic labels such as segmentation masks, depth images, optic flows, and surface normal images can be rendered by game engine or generated by hard-code methods. Since these semantic labels are automatically generated, the methods using the synthetic datasets or using them in conjunction with a large unlabeled image or video datasets are considered as self-supervised learning methods.

5.3.1 Learning with Labels Generated by Game Engines

Given models of various objects and layouts of environments, game engines are able to render realistic images and provide accurate pixel-level labels. Since game engines can generate large-scale datasets with negligible cost, various game engines such as Airsim [128] and Carla [129] have been used to generate large-scale synthetic datasets with high-level semantic labels including depth, contours, surface normal, segmentation mask, and optical flow for training deep networks. An example of an RGB image with its generated accurate labels is shown in Fig. 8.

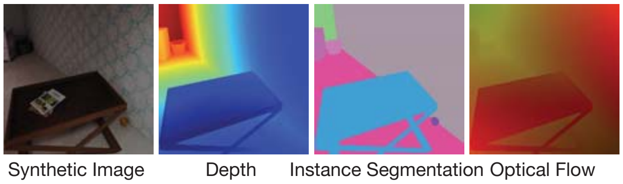

Fig. 8: An example of an indoor scene generated by a game engine [84]. For each synthetic image, the corresponding depth, instance segmentation, and optical flow can be automatically generated by the engine.

Game engines can generate realistic images with accurate pixel-level labels with very low cost. However, due to the domain gap between synthetic and real-world images, the ConvNet purely trained on synthetic images cannot be directly applied to real-world images. To utilize synthetic datasets for self-supervised feature learning, the domain gap needs to be explicitly bridged. In this way, the ConvNet trained with the semantic labels of the synthetic dataset can be effectively applied to real-world images.

To overcome the problem, Ren and Lee proposed an unsupervised feature space domain adaptation method based on adversarial learning [53]. As shown in Fig. 9, the network predicts surface normal, depth, and instance contour for the synthetic images and a discriminator network D is employed to minimize the difference of feature space domains between real-world and synthetic data. Helped with adversarial training and accurate semantic labels of synthetic images, the network is able to capture visual features for real-world images.

Compared to other pretext tasks in which the pretext tasks implicitly force ConvNets to learn semantic features, this type of methods are trained with accurate semantic labels which explicitly force ConvNets to learn features that highly related to the objects in images.

5.3.2 Learning with Labels Generated by Hard-code programs

Applying hard-code programs is another way to automatically generate semantic labels such as salience, foreground masks, contours, depth for images and videos. With these methods, very large-scale datasets with generated semantic labels can be used for self-supervised feature learning. This type of methods generally has two steps: (1) label generation by employing hard-code programs on images or videos to obtain labels, (2) train ConvNets with the generated labels.

Various hard-code programs have been applied to generate labels for self-supervised learning methods include

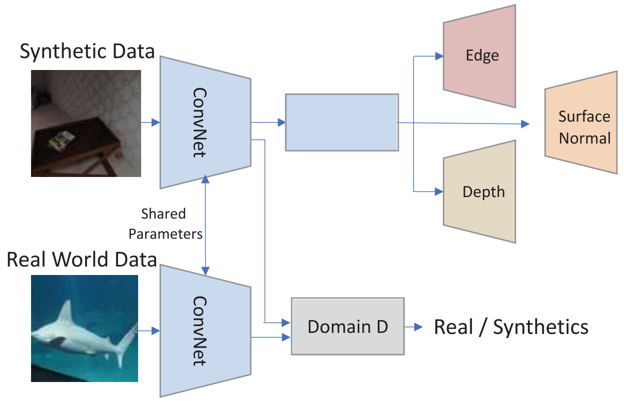

Fig. 9: The architecture for utilizing synthetic and real-world images for self-supervised feature learning [53]. Figure is reproduced based on [53].

methods for foreground object segmentation [30], edge detection [56], and relative depth prediction [57]. Pathak *et al.* proposed to learn features by training a ConvNet to segment foreground objects in each frame of a video while the label is the mask of moving objects in videos [30]. Li *et al.* proposed to learn features by training a ConvNet for edge prediction while labels are motion edges obtained from flow fields from videos [56]. Jing *et al.* proposed to learn features by training a ConvNet to predict relative scene depths while the labels are generated from optical flow [57].

No matter what kind of labels used to train ConvNets, the general idea of this type of methods is to distill knowledge from hard-code detector. The hard-code detector can be edge detector, salience detector, relative depth detector, etc.

Compared to other self-supervised learning methods, the supervision signal in these pretext tasks is semantic labels which can directly drive the ConvNet to learn semantic features. However, one drawback is that the semantic labels generated by hard-code detector usually are very noisy which need to specifically cope with.

6 Video Feature Learning

This section reviews the self-supervised methods for learning video features, as listed in Table 3, they can be categorized into four classes: generation-based methods, context-based methods, free semantic label-based methods, and cross modal-based methods. The networks employed for video feature learning include 2DConvNet, 3DConvNet, and LSTM combined with 2DConvNet or 3DConvNet.

6.1 Generation-based Video Feature Learning

Learning from video generation refers to the methods that visual features are learned through the process of video generation while without using any human-annotated labels. This type of methods includes video generation with GAN [37], video colorization [135] and video prediction [39]. For these pretext tasks, the pseudo training label P usually is the video itself and no human-annotated labels are needed during training, therefore, these methods belong to self-supervised learning.

6.1.1 Learning from Video Generation

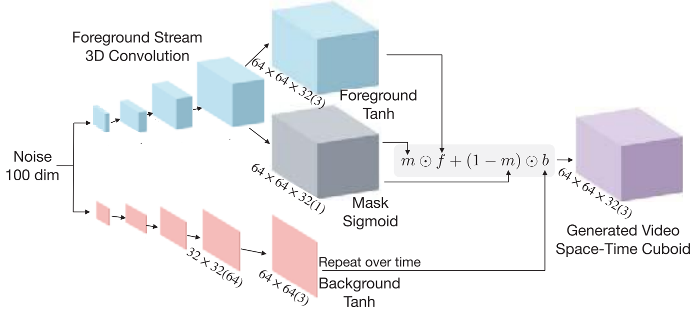

Fig. 10: The architecture of the generator in VideoGan for video generation with GAN proposed in [37]. The figure is from [37] with author's permission.

After GAN-based methods obtained breakthrough results in image generation, researchers employed GAN to generate videos [37], [38], [134]. One pioneer work of video generation with GAN is VideoGAN [37], and the architecture of the generator network is shown in Fig. 10. To model the motion of objects in videos, a two-stream network is proposed for video generation while one stream is to model the static regions in videos as background and another stream is to model moving object in videos as foreground [37]. Videos are generated by the combination of the foreground and background streams. The underline assumption is that each random variable in the latent space represents one video clip. This method is able to generate videos with dynamic contents. However, Tulyakov *et al.* argues that this assumption increases difficulties of the generation, instead, they proposed MocoGAN to use the combination of two subspace to represent a video by disentangling the context and motions in videos [38]. One space is context space which each variable from this space represents one identity, and another space is motion space while the trajectory in this space represents the motion of the identity. With the two sub-spaces, the network is able to generate videos with higher inception score.

The generator learns to map latent vectors from latent space into videos, while discriminator learns to distinguish the real world videos with generated videos. Therefore, the discriminator needs to capture the semantic features from videos to accomplish this task. When no human-annotated labels are used in these frameworks, they belong to the self-supervised learning methods. After the video generation training on large-scale unlabeled dataset finished, the parameters of discriminator can be transferred to other downstream tasks [37].

6.1.2 Learning from Video Colorization

Temporal coherence in videos refers to that consecutive frames within a short time have similar coherent appearance. The coherence of color can be used to design pretext tasks for self-supervised learning. One way to utilize color coherence is to use video colorization as a pretext task for self-supervised video feature learning.

Video colorization is a task to colorize gray-scale frames into colorful frames. Vondrick *et al.* proposed to constrain

TABLE 3: Summary of self-supervised video feature learning methods based on the category of pretext tasks.

Mehtod	SubCategory	Code	Contribution
ConvLSTM [130]	Generation	Yes	Employing Convolutional LSTM for video prediction
DPC [131]	Generation	Yes	Learning by recurrently predicting representations of future frames
LSTMDynamics [132]	Generation	No	Learning by predicting long-term temporal dynamic in videos
MCNet [133]	Generation	Yes	Disentangling motion and content for video future prediction
MocoGAN [38]	Generation	Yes	Decomposing motion and content for video generation with GAN
TemporalGAN [134]	Generation	Yes	Decomposing temporal and image generator for video generation
Un-LSTM [39]	Generation	Yes	Forerunner of video prediction with LSTM
Video Colorization [135]	Generation	Yes	Employing video colorization as the pretext task
VideoGAN [37]	Generation	Yes	Forerunner of video generation with GAN
AOT [48]	Context	Yes	Learning by recognizing the arrow of time in videos
CubicPuzzles [125]	Context	No	Learning by solving video cubic puzzles
LSTMPermute [136]	Context	Yes	Learning by temporal order verification with LSTM
OPN [49]	Context	Yes	Using frame sequence order recognition as the pretext task
O3N [137]	Context	No	Learning by identifying odd video sequences
ShuffleLearn [33]	Context	Yes	Employing temporal order verification as the pretext task
TemporalCoherence [138]	Context	No	Learning with the temporal coherence of features of frame sequence
Transitive [139]	Context	No	Learning inter and intra instance variations with a Triplet loss
VCOP [50]	Context	Yes	Learning by recognizing the order of video clips with 3DCNN
Video Jigsaw [42]	Context	No	Learning by jointly reasoning about spatial and temporal context
3DRotNet [47]	Context	No	Learning by recognizing rotations of video clips
ActivesStereoNet [140]	Cross Modal	Yes	End-to-end self-supervised learning of depth from active stereo systems
AmbientSound [141]	Cross Modal	No	Predicting a statistical summary of the sound from a video frame
AVTS [34]	Cross Modal	No	Visual and Audio correspondence verification as pretext task
AudioVisual [142]	Cross Modal	Yes	Jointly modeling visual and audio as fused multisensory representation
CrossLearn [59]	Cross Modal	No	Optical flow and RGB correspondence verification as pretext task
CrossPixel [143]	Cross Modal	No	Learning by predicting motion from a single image as the pretext task
DepthFlow [144]	Cross Modal	Yes	Depth and optical flow learning using cross-task consistency from videos
EgoMotion [145]	Cross Modal	Yes	Learning by predicting camera motion and the scene structure from videos
FlowNet [146]	Cross Modal	Yes	Forerunner of optical flow estimation with ConvNet
FlowNet2 [147]	Cross Modal	Yes	Better architecture and better performance on optical flow estimation
GDT [123]	Cross Modal	No	Learning video and audio features by contrasting across modalities and augmentations
GoNet [148]	Cross Modal	Yes	Jointly learning monocular depth, optical flow and ego-motion estimation from videos
L^3-Net [58]	Cross Modal	Yes	Forerunner of Audio-Visual Correspondence for self-supervised learning
LearnByMove [61]	Cross Modal	Yes	Learning by predicting the camera transformation from a pairs of images
MotionPred [149]	Cross Modal	Yes	Learning by predicting the appearance and motion statics of video clips
TiedEgoMotion [62]	Cross Modal	No	Learning from ego-motor signals and video sequence
UnFlow [150]	Cross Modal	Yes	An unsupervised loss for optical flow estimation
VisualOdometry [151]	Cross Modal	Yes	An unsupervised paradigm for deep visual odometry learning
XDC [152]	Cross Modal	No	Using clustering in one modality as a supervisory signal for the other modality

colorization models to solve video colorization by learning to copy colors from a reference frame [135]. Given the reference RGB frame and a gray-scale image, the network needs to learn the internal connection between the reference RGB frame and gray-scale image to colorize it.

Another perspective is to tackle video colorization by employing a fully convolution neural network. Tran *et al.* proposed an U-shape convolution neural network for video colorization [153]. The network is an encoder-decoder based 3DConvNet. The input of the network is a clip of grayscale video clip, while the output if a colorful video clip. The encoder is a bunch of 3D convolution layers to extract features while the decoder is a bunch of 3D deconvolution layers to generate colorful video clips from the extracted feature.

The color coherence in videos is a strong supervision signal. However, only a few work studied to employ it for self-supervised video feature learning [135]. More work can be done by studying using color coherence as a supervision signal for self-supervised video feature learning.

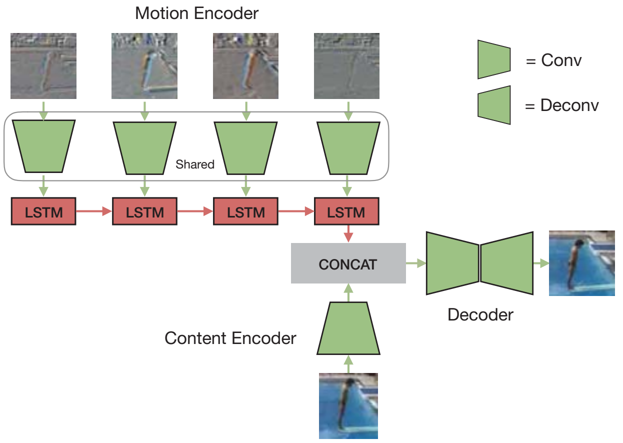

Fig. 11: The architecture for video prediction task proposed by [133]. Figure is reproduced based on [133].

6.1.3 Learning from Video Prediction

Video future prediction is a task of predicting future frame sequences [39] or features of future frame sequences [131] based on a limited number of frames of a video. To predict future frames, networks must learn dynamic changes in appearance within a given frame sequence and learn to infer the future.

The pioneer of applying deep learning for video prediction is Un-LSTM [39], and many methods have been proposed afterwards [39], [131], [133], [154], [155], [156], [157], [158]. Since its superior ability to model temporal dynamics, most of them use LSTM or LSTM variant to encode temporal dynamics in videos or to infer the future frames [39], [130], [133], [157], [158]. These methods can be employed for self-supervised feature learning without using human-annotations.

Most of the frameworks follow the encoder-decoder pipeline in which the encoder to model spatial and temporal features from the given video clips and the decoder to generate future frames or to generate features of the future frames based on feature extracted by encoder. Fig. 11 shows a pipeline of MCnet proposed by Villegas *et al.* in [133]. McNet is built on Encoder-Decoder Convolutional Neural Network and Convolutional LSTM for video prediction. It has two encoders, one is Content Encoder to capture the spatial layout of an image, and the other is Motion Encoder to model temporal dynamics within video clips. The spatial features and temporal features are concatenated to feed to the decoder to generate the next frame, and the network is optimized by content loss and adversarial loss. By separately modeling temporal and spatial features, this model can effectively generate future frames recursively.

Video future prediction is a self-supervised learning task and the learned features can be transferred to other tasks [39], [131]. Generally, for video future frame prediction tasks, Structural Similarity Index (SSIM) and Peak Signal to Noise Ratio (PSNR) are employed to evaluate the difference between the generated frame sequence and the ground truth frame sequence.

6.2 Temporal Context-based Learning

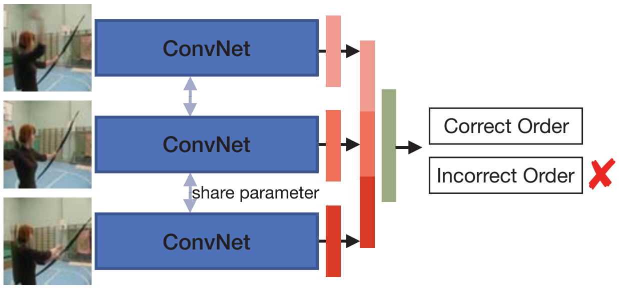

Fig. 12: The pipeline of Shuffle and Learn [33]. The network is trained to verify whether the input frames are in correct temporal order. Figure is reproduced based on [33].

Videos consist of various lengths of frames which have rich spatial and temporal information. The inherent temporal information within videos can be used as supervision signal for self-supervised feature learning. Various pretext tasks have been proposed by utilizing temporal context relations including temporal order verification [33], [48], [137] and temporal order recognition [49], [50], [125]. Temporal order verification is to verify whether a sequence of input frames is in correct temporal order, while temporal order recognition is to recognize the order of a sequence of input frames.

As shown in Fig. 12, Misra *et al.* proposed to use the temporal order verification as the pretext task to learn image features from videos with 2DConvNet [33] which has two main steps: (1) The frames with significant motions are sampled from videos according to the magnitude of optical flow, (2) The sampled frames are shuffled and fed to the network which is trained to verify whether the input data is in correct order. To successfully verify the order of the input frames, the network is required to capture the subtle difference between the frames such as the movement of the person. Therefore, semantic features can be learned through the process of accomplishing this task. The temporal order recognition tasks use networks of similar architecture.

However, the methods usually suffer from a massive dataset preparation step. The frame sequences that used to train the network are selected based on the magnitude of the optical flow, and the computation process of optical flow is expensive and slow. Therefore, more straightforward and time-efficiency methods are needed for self-supervised video feature learning.

6.3 Cross Modal-based Learning

Cross modal-based learning methods usually learn features from the correspondence of multiple data streams including RGB frame sequence, optical flow sequence, audio data, and camera pose.

In addition to rich temporal and spatial information in videos, optical flow sequence can be generated to specifically indicate the motion in videos, and the difference of frames can be computed with negligible time and space-time complexity to indicate the boundary of the moving objects. Similarly, audio data also provide a useful hint about the content of videos. Based on the type of data used, these methods fall into three groups: (1) methods that learn features by using the RGB and optical flow correspondence [59], [143], (2) methods that learn features by utilizing the video and audio correspondence [34], [58], (3) ego-motion that learn by utilizing the correspondence between egocentric video and ego-motor sensor signals [61], [62]. Usually, the network is trained to recognize if the two kinds of input data are corresponding to each other [34], [59], or is trained to learn the transformation between different modalities [61].

6.3.1 Learning from RGB-Flow Correspondence

Optical flow encodes object motions between adjacent frames, while RGB frames contain appearance information. The correspondence of the two types of data can be used to learn general features [59], [143], [146], [147]. This type of pretext tasks include optical flow estimation [146], [147] and RGB and optical flow correspondence verification [143].

Sayed *et al.* proposed to learn video features by verifying whether the input RGB frames and the optical flow corresponding to each other. Two networks are employed while one is for extracting features from RGB input and another is for extracting features from optical flow input [59]. To verify whether two input data correspond to each other, the network needs to capture mutual information between the two modalities. The mutual information across different modalities usually has higher semantic meaning compared to information which is modality specific. Through this pretext task, the mutual information that invariant to specific modality can be captured by ConvNet.

Optical flow estimation is another type of pretext tasks that can be used for self-supervised video feature learning. Fischer *et al.* proposed FlowNet which is an end-to-end convolution neural network for optical flow estimation from two consecutive frames [146], [147]. To correctly estimate optical flow from two frames, the ConvNet needs to capture appearance changes of two frames. Optical flow estimation can be used for self-supervised feature learning because it can be automatically generated by simulators such as game engines or by hard-code programs without human annotation.

6.3.2 Learning from Visual-Audio Correspondence

Recently, some researchers proposed to use the correspondence between visual and audio streams to design Visual-Audio Correspondence learning task [34], [58], [123], [141], [142]. Usually, this type of method jointly learns both video and audio features with heterogeneous networks.

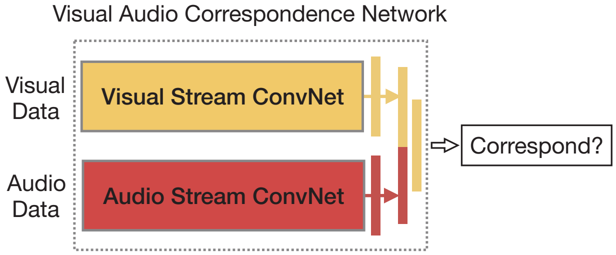

Fig. 13: The architecture of video and audio correspondence verification task [58].

The general framework of this type of pretext tasks is shown in Fig. 13. There are two subnetworks: the vision subnetwork and the audio subnetwork. The input of vision subnetwork is a single frame or a stack of image frames and the vision subnetwork learns to capture visual features of the input data. The audio network is a 2DConvNet and the input is the Fast Fourier Transform (FFT) of the audio from the video. Positive data are sampled by extracting video frames and audio from the same time of one video, while negative training data are generated by extracting video frames and audio from different videos or from different times of one video. Therefore, networks are trained to discover the correlation of video and audio data to accomplish the pretext task which could be verifying whether the input visual data and audio data are correspondents or not [34] with cross-entropy loss, or minimizing the distance between positive pairs while maximizing distances between negative pairs with contrastive or triplet loss [123]. Since the inputs of ConvNets are two kinds of data, networks are able to jointly learn both video and audio features by solving the pretext task.

Another way to utilize the correspondence of visual and audio streams is to use cross-modal deep clustering proposed by Alwassel *et al.* which leverages unsupervised clustering in one modality as a supervisory signal for other modalities [152]. With cross-modal supervision signal, models can utilize semantic correlations and differences among different modalities.

In many applications, multiple modalities of data are usually available such as video, audio, language, depth, point cloud, mesh, etc. Utilizing cross-modality correspondences as supervision signal for self-supervised learning can be easily extended to other modalities of data.

6.3.3 Ego-motion

With the self-driving car which usually equipped with various sensors, the large-scale egocentric video along with ego-motor signal can be easily collected with very low cost by driving the car in the street. Recently, some researchers proposed to use the correspondence between visual signal and motor signal for self-supervised feature learning [61], [62], [145].

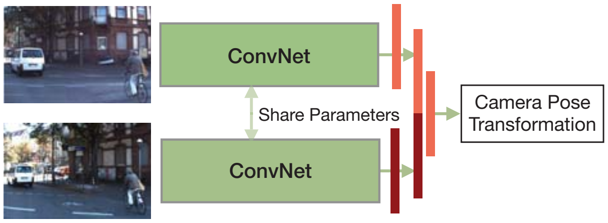

Fig. 14: The architecture of camera pose transformation estimation from egocentric videos [61].

The underline intuition of this type of methods is that a self-driving car can be treated as a camera moving in a scene and thus the egomotion of the visual data captured by the camera is as same as that of the car. Therefore, the correspondence between visual data and egomotion can be utilized for self-supervised feature learning. A typical network of using ego-motor signal is shown in Fig. 14 proposed by Agrawal *et al.* for self-supervised image feature learning [61]. The inputs to the network are two frames sampled from an egocentric video within a short time. The labels for the network indicate the rotation and translation relation between the two sampled images which can be derived from the odometry data of the dataset. With this task, the ConvNet is forced to identify visual elements that are present in both sampled images.

The ego-motor signal is a type of accurate supervision signal. In addition to directly applying it for self-supervised feature learning, it has also been used for unsupervised learning of depth and ego-motion [145]. All these networks can be used for self-supervised feature learning and transferred for downstream tasks.

7 Performance Comparison

This section compares the performance of image, video, audio, and 3D feature self-supervised learning methods on public datasets. For image feature self-supervised learning, the performance on downstream tasks including image classification, semantic segmentation, and object detection are compared. For video feature self-supervised learning, the performance on a downstream task which is human action recognition in videos is reported. Since some self-supervised video feature learning methods jointly learn video and audio features, we also include the performance comparison of

TABLE 4: Linear classification on ImageNet and Places datasets using activations from the convolutional layers of an AlexNet as features. "Convn" means the linear classifier is trained based on the n-th convolution layer of AlexNet. "Places Labels" and "ImageNet Labels" indicate using supervised model trained with human-annotated labels as the pre-trained model.

		ImageNet					Places				
Method	Pretext Tasks	conv1	conv2	conv3	conv4	conv5	conv1	conv2	conv3	conv4	conv5
Places labels [9]	—	—	—	—	—	—	22.1	35.1	40.2	43.3	44.6
ImageNet labels [9]	—	19.3	36.3	44.2	48.3	50.5	22.7	34.8	38.4	39.4	38.7
Random(Scratch) [9]	—	11.6	17.1	16.9	16.3	14.1	15.7	20.3	19.8	19.1	17.5
ColorfulColorization [20]	Generation	12.5	24.5	30.4	31.5	30.3	16.0	25.7	29.6	30.3	29.7
BiGAN [93]	Generation	17.7	24.5	31.0	29.9	28.0	21.4	26.2	27.1	26.1	24.0
SplitBrain [97]	Generation	17.7	29.3	35.4	35.2	32.8	21.3	30.7	34.0	34.1	32.5
ContextEncoder [21]	Context	14.1	20.7	21.0	19.8	15.5	18.2	23.2	23.4	21.9	18.4
ContextPrediction [45]	Context	16.2	23.3	30.2	31.7	29.6	19.7	26.7	31.9	32.7	30.9
Jigsaw [22]	Context	**18.2**	28.8	34.0	33.9	27.1	23.0	32.1	35.5	34.8	31.3
Learning2Count [106]	Context	18.0	30.6	34.3	32.5	25.7	**23.3**	**33.9**	36.3	34.7	29.6
DeepClustering [31]	**Context**	13.4	**32.3**	**41.0**	**39.6**	**38.2**	19.6	33.2	**39.2**	**39.8**	**34.7**

TABLE 5: Comparison of the self-supervised image feature learning methods on classification, detection, and segmentation on PASCAL VOC dataset. "ImageNet Labels" indicates using supervised model trained with human-annotated labels as the pre-trained model.

Method	Pretext Tasks	Classification (%)	Detection (%)	Segmentation (%)
ImageNet Labels [9]	—	79.9	56.8	48.0
Random(Scratch) [9]	—	57.0	44.5	30.1
ContextEncoder [21]	Generation	56.5	44.5	29.7
BiGAN [93]	Generation	60.1	46.9	35.2
ColorfulColorization [20]	Generation	65.9	46.9	35.6
SplitBrain [97]	Generation	67.1	46.7	36.0
RankVideo [110]	Context	63.1	47.2	35.4†
PredictNoise [109]	Context	65.3	49.4	37.1†
JigsawPuzzle [22]	Context	67.6	53.2	37.6
ContextPrediction [45]	Context	65.3	51.1	—
Learning2Count [106]	Context	67.7	51.4	36.6
DeepClustering [31]	**Context**	**73.7**	**55.4**	**45.1**
WatchingVideo [30]	Free Semantic Label	61.0	52.2	—
CrossDomain [53]	Free Semantic Label	68.0	52.6	—
AmbientSound [141]	Cross Modal	61.3	—	—
TiedToEgoMotion [62]	Cross Modal	—	41.7	—
EgoMotion [61]	Cross Modal	54.2	43.9	—

audio features on audio event classification benchmark [34], [123], [152]. For 3D feature learning, 3D object recognition with different modalities including point cloud, mesh, and multi-view images usually is used as downstream tasks to evaluate the quality of learned features.

7.1 Performance of Image Feature Learning

As described in Section 4.3, the quality of features learned by self-supervised learned models is evaluated by fine-tuning them on downstream tasks such as semantic segmentation, object detection, and image classification. This section summarizes the performance of the existing image feature self-supervised learning methods.

Table 4 lists the performance of image classification performance on ImageNet [14] and Places [76] datasets. During self-supervised pretext tasks training, most of the methods are trained on ImageNet dataset with AlexNet as based network without using the category labels. After self-supervised training finished, a linear classifier is trained on top of different frozen convolutional layers of the ConvNet on the training split of ImageNet and Places datasets. The classification performances on the two datasets are used to demonstrate the quality of the learned features.

As shown in Table 4, the overall performance of the self-supervised models is lower than that of models trained either with ImageNet labels or with Places labels. Three conclusions can be drawn based on the performance from the Table: (1) The features from different layers are always benefited from the self-supervised pretext task training. The performance of self-supervised learning methods is always better than the performance of the model trained from scratch. (2) All of the self-supervised methods perform well with the features from conv3 and conv4 layers while performing worse with the features from conv1, conv2, and conv5 layers. This is probably because shallow layers capture general low-level features, while deep layers capture pretext task-related features. (3) When there is a domain gap between dataset for pretext task training and the dataset of downstream task, the self-supervised learning method is able to reach comparable performance with the model trained with ImageNet labels.

In addition to image classification, object detection and semantic segmentation are also used as the downstream tasks to evaluate the quality of the features learned by self-supervised learning. Usually, ImageNet is used for self-supervised pretext task pre-training by discarding category

TABLE 6: Comparison of existing self-supervised methods for action recognition on UCF101 and HMDB51 datasets.

	Pretraining		Evaluation	
Method	Architecture	Dataset	UCF	HMDB
Fully supervised [159]	3DResNet18	Kinetics	84.4	56.4
Fully supervised [160]	R(2+1)D-18	Kinetics	93.1	63.6
ShuffleLearn [33]	CaffeNet	UCF/HMDB	50.2	18.1
GeometryGuide [161]	CaffeNet	UCF/HMDB	55.1	23.3
RL [126]	CaffeNet	UCF/HMDB	58.6	25.0
CMC [101]	CaffeNet	UCF/HMDB	59.1	26.7
CrossLearn [59]	CaffeNet	UCF/HMDB	58.7	27.2
OPN [49]	VGG	UCF/HMDB	59.8	23.8
L^3-Net [58]	VGG	AudioSet	72.3	40.2
MotionPred [149]	C3D	Kinetics	61.2	33.4
RotNet3D [47]	3D-ResNet18	Kinetics	62.9	33.7
DPC [131]	3D-ResNet18	Kinetics	68.2	34.5
ST-Puzzle [125]	3D-ResNet18	Kinetics	65.8	33.7
DPC [131]	3D-ResNet34	Kinetics	75.7	35.7
AVTS [34]	MC3-18	AudioSet	89.0	61.6
VCOP [50]	R(2+1)D-18	Kinetics	72.4	30.9
XDC [152]	R(2+1)D-18	Kinetics	84.2	47.1
GDT [123]	R(2+1)D-18	Kinetics	88.7	57.8
XDC [152]	R(2+1)D-18	IG65M	94.2	67.4

TABLE 7: Comparison of existing self-supervised methods for audio classification task on DCASE and ESC50 datasets.

Method	Pretraining		Evaluation	
	Dataset	#data	DCASE	ESC50
Random Forest [90]	ESC50	1.6K	-	44.3
Piczak ConvNet [162]	ESC50	1.6K	-	64.5
ConvRBM [163]	ESC50	1.6K	-	86.5
RNH [164]	DCASE	100	72	-
Ensemble [89]	DCASE	100	77	-
AVTS [34]	Kinetics-400	230K	91	76.7
XDC [152]	Kinetics-400	230K	-	78.0
GDT [123]	Kinetics-400	230K	94	78.6
Autoencoder [111]	SoundNet	2M+	-	39.9
SoundNet [165]	SoundNet	2M+	88	74.2
L^3-Net [58]	SoundNet	2M+	93	79.3
AVTS [34]	SoundNet	2M+	94	82.3
XDC [152]	AudioSet	1.8M	93	84.8

labels, while the AlexNet is used as the base network and fine-tuned on the three tasks. Table 5 lists the performance of image classification, object detection, and semantic segmentation tasks on the PASCAL VOC dataset. The performance of classification and detection is obtained by testing the model on the test split of PASCAL VOC 2007 dataset, while the performance of semantic segmentation is obtained by testing the model on the validation split of PASCAL VOC 2012 dataset.

As shown in Table 5, the performance of the self-supervised models on segmentation and detection dataset are very close to that of the supervised method which is trained with ImageNet labels during pre-training. Specifically, the margins of the performance differences on the object detection and semantic segmentation tasks are less than 3% which indicate that the learned features by self-supervised learning have a good generalizability. In general, context-based methods performance better than other types of methods.

7.2 Performance of Video Feature Learning

For self-supervised video feature learning methods, human action recognition task is used to evaluate the quality of learned features. Various video datasets have been used for self-supervised pre-training, and different network architectures have been used as the base network. Usually after the pretext task pre-training finished, networks are fine-tuned and tested on the commonly used UCF101 and HMDB51 datasets for human action recognition task. Table 6 compares the performance of existing self-supervised video feature learning methods on UCF101 and HMDB51 datasets, while Table 7 shows the performance of existing self-supervised learning methods on audio event classification on DCASE [89] and ESC50 [90] datasets.

As shown in Table 6, different backbone networks and datasets are used for self-supervised pre-training. In order to make fair comparisons, we group the methods based on the usage of datasets and backbone networks. Two conclusions can be drawn based on the performance from Table 6: (1) The performance on downstream tasks significantly depends on the capability of backbone networks. Stronger backbone networks usually achieve much better performance [131]. (2) The performance on downstream tasks can be significantly boosted up by adding the amount of data for self-supervised training. By pre-training on 65 million videos, the cross-modal based method XDC is able to outperform the supervised methods (pre-trained on Kinetics dataset with category labels) on both UCF101 and HMDB51 datasets with a large margin. Table 7 shows that the methods that jointly learning both video and audio features achieve relatively high accuracy on the audio classification task, and some models also benefit from the large-scale dataset for self-supervised pre-training [152]. This demonstrates a potential of scaling self-supervised learning methods to large-scale datasets.

7.3 Beyond 2D Self-supervised Learning

Table 3 lists the 3D self-supervised learning methods. Most of them learn point cloud features by various auto-encoder networks, and only a few learn features by designing novel pretext tasks [91], [124], [174]. For self-supervised 3D feature learning methods, 3D object recognition task is often employed as the downstream task to evaluate the quality of learned features. Usually, ShapeNet [87] or ModelNet40 [86] dataset is used to train self-supervised learning methods. After the self-supervised training finished, SVM classifier for 3D object classification is trained upon on the extracted features and the classification accuracy is reported to indicate the quality of learned features. Table 9 compares the performance of existing self-supervised 3D feature learning methods on ModelNet40 dataset.

As shown in Table 9, self-supervised learning methods with all modalities on ModelNet40 achieve very close performance as the supervised methods. These results demonstrate the potential of applying the self-supervised learning methods to other data modalities or other domains.

7.4 Summary

Based on the results, conclusions can be drawn about the performance and reproducibility of the self-supervised

TABLE 8: Summary of self-supervised 3D feature learning methods based on the category of pretext tasks.

Mehtod	SubCategory	Contribution
3D-GAN [166]	Generation	Learning features by generating 3D objects from a probabilistic space with 3D-GAN
FoldingNet [167]	Generation	Proposed an auto-encoder with a graph-based encoder and a novel folding-based decoder
Latent-GAN [168]	Generation	Learning point cloud features by a deep auto-encoder network
MRTNet [169]	Generation	Learning point cloud features by a multi-resolution tree-structured VAE network
PointCapsNet [170]	Generation	Learning point cloud features by using a 3D point-capsule auto-encoder network
T-L Network [171]	Generation	Learning with TL-embedding network to learn generative and predictive features
VConv-DAE [172]	Generation	Learning point cloud features with a convolutional volumetric auto-encoder
Contrast-Cluster [173]	Context	Learning point cloud features by part contrasting and clustering
3DMultiTask [174]	Multiple	Learning point cloud features with multiple pretext tasks
XMV [91]	Multiple	Jointly learning point cloud features and image features by exploring cross modalities and views
MVI [124]	Multiple	Jointly learning modality and view invariant features for 3D objects by contrasting

TABLE 9: The comparison with the state-of-the-art methods for 3D shape recognition on ModelNet40 dataset. * indicates the model is trained with human-annotated object category labels. 'Images' indicates the method is trained on multi-view images rendered from mesh objects.

Method	Pretext Task	Modality	Accuracy
PointNet* [18]	—	Point Cloud	89.2
DGCNN* [175]	—	Point Cloud	92.2
MeshNet* [176]	—	Mesh	91.9
MVCNN* [177]	—	Images	90.1
T-L Network [171]	Generation	Point Cloud	74.4
VConv-DAE [172]	Generation	Point Cloud	75.5
3D-GAN [166]	Generation	Point Cloud	83.3
Latent-GAN [168]	Generation	Point Cloud	85.7
MRTNet-VAE [169]	Generation	Point Cloud	86.4
Contrast-Cluster [173]	Context	Point Cloud	86.8
FoldingNet [167]	Generation	Point Cloud	88.4
PointCapsNet [170]	Generation	Point Cloud	88.9
3DMultiTask [174]	Multiple	Point Cloud	89.1
MVI [124]	Multiple	Point Cloud	89.3
XMV [91]	**Multiple**	**Point Cloud**	**89.8**
XMV [91]	Multiple	Images	87.3
MVI [124]	Multiple	Image	88.2
MVI [124]	Multiple	Mesh	87.7

learning methods.

Performance: For image feature self-supervised learning, due to the well-designed pretext tasks, the performance of self-supervised methods are comparable to the supervised methods on some downstream tasks, especially for the object detection and semantic segmentation tasks. The margins of the performance differences on the object detection and semantic segmentation tasks are less than 3% which indicate that the learned features by self-supervised learning have a good generalizability. However, the performance of self-supervised learning methods for videos greatly depends on the backbone networks and the amount of training data. And the video self-supervised learning methods usually are only evaluated on small datasets including UCF101 and HMDB51 for comparison. These two datasets are too small and may not fully reflect the generalizability and the full performance of self-supervised video feature learning methods.

Reproducibility: As we can observe, for the image feature self-supervised learning methods, most of the networks use AlexNet as a base network to pre-train on ImageNet dataset and then evaluate on same downstream tasks for quality evaluation. Also, the code of most methods are released which is a great help for reproducing results. However, for the video self-supervised learning, various datasets and networks have been used for self-supervised pre-training, therefore, it is unfair to directly compare different methods. Furthermore, some methods use UCF101 as self-supervised pre-training dataset which is a relatively small video dataset. With this size of the dataset, the power of a more powerful model such as 3DCovnNet may not be fully discovered and may suffer from server over-fitting. Therefore, larger datasets for video feature self-supervised pre-training should be used.

Evaluation Metrics: Another fact is that more evaluation metrics are needed to evaluate the quality of the learned features in different levels. The current solution is to use the performance on downstream tasks to indicate the quality of the features. However, this evaluation metric does not give insight what the network learned through the self-supervised pre-training. More evaluation metrics such as network dissection [23] should be employed to analysis the interpretability of the self-supervised learned features.

7.5 Discussion

The recent study analyzed the filters learned by various self-supervised methods and found that these self-supervised models learned kernels to detect objects, scenes, object parts, materials, textures, and colors [23]. This demonstrates that different pretext tasks lead to similar general and semantic features, and features learned by a well-designed pretext task [127] should be able to be effectively transferred to multiple downstream tasks. The self-supervised learning methods can be used to obtain pre-trained models in cases that only limited labeled data are available. Here are some guidelines to choose or design suitable pretext tasks for a given downstream task:

- The most challenging part of self-supervised learning is the design of an effective pretext task which can ensure a network to learn meaningful image/video features instead of trivial solutions. The design of the pretext task should balance the convergence and overfitting.
- In general, context-based self-supervised learning methods [31], [60], [125] have better performance than other methods for both image-based and video-based applications.
- Different pretext tasks provide different supervision signals which can help the network learn more representative features. Therefore, multiple pretext tasks should be used whenever possible.

- When the domain gap of the datasets between self-supervised pre-training and the downstream task is smaller, the performance on the downstream task is usually better.
- Recent work [127], [178] showed that increasing the capacity of ConvNet including the number of filters, the size of the representation, and the depth for self-supervised learning can significantly increase the quality of the learned visual representations. Therefore, ConvNet with more kernels and larger dimension of features should be employed.
- Larger dataset for self-supervised pre-training generally leads to better performance on downstream tasks [30], [47], [127], [152]. Therefore, larger datasets should be used whenever possible.

8 Future Directions

Self-supervised learning methods have been achieving great success and obtaining good performance that close to supervised models on some computer vision tasks. Here, some future directions of self-supervised learning are discussed.

Combining Self-supervised Learning Methods With Other Learning Methods: In real scenarios, the collection and annotation of data are generally expensive and time-consuming. Recently many semi-supervised learning and few-shot learning methods have been proposed to tackle the problem of only limited labeled data are available. The self-supervised learning methods can be incorporated into these methods to boost the performance in two ways: (1) the pre-trained models obtained by self-supervised pre-training can server as a start point for these methods, (2) the pretext task can be added as an auxiliary task for these methods to add regularization to networks. How to add self-supervised supervision to other tasks will be a good research direction.

Learning Features from Synthetic Data: A rising trend of self-supervised learning is to train networks with synthetic data which can be easily rendered by game engines with very limited human involvement [53], [54], [179], [180]. With the help of game engines, millions of synthetic images and videos with accuracy pixel-level annotations can be easily generated. With accurate and detailed annotations, various pretext tasks can be designed to learn features from synthetic data. One problem needed to solve is how to bridge the domain gap between synthetic data and real-world data. Only a few work explored self-supervised learning from synthetic data by using GAN to bridge the domain gap [53], [54]. With more available large-scale synthetic data, more self-supervised learning methods will be proposed.

Learning from Web Data: Another rising trend is to train networks with web collected data [181], [182], [183], [184], [185] based on their existing associated tags. With the search engine, millions of images and videos can be downloaded from websites like Flickr and YouTube with negligible cost. In addition to its raw data, the title, keywords, captions, and reviews can also be available as part of the data which can be used as extra information to train networks. With carefully curated queries, the web data retrieved by reliable search engines can be relatively clean. With large-scale web data and their associated metadata, the performance of self-supervised methods may be boosted up. One open problem about learning from web data is how to handle the noise in web data and their associated metadata.

Learning Features from Point Cloud Data: Image and video related computer vision applications greatly benefit from pre-trained models from large-scale classification datasets. Recently research studies on 3D draw more and more attention of the community, it would be a good research direction to obtain such a pre-trained model for point cloud data that can be used for tasks such as point cloud semantic segmentation and detection. Even though there are several self-supervised learning methods proposed to extract 3D features, most of them were trained and tested only on 3D CAD models and their generalizability to real-world point cloud datasets are not studied. It would be an interesting and practical direction to explore self-supervised learning methods on real-world 3D data.

Learning with Data from Different Modalities: Most existing self-supervised visual feature learning methods focused on learning features for only one modality. However, if other modalities of data from different sensors are available, the constraint between different modalities of data can be used as additional sources to train networks to learn features [91], [124], [145]. The self-driving cars usually are equipped with various sensors including RGB cameras, gray-scale cameras, 3D laser scanners, and high-precision GPS measurements and IMU accelerations. Very large-scale datasets with different modalities can be easily obtained through the driving, and the correspondence of data captured by different devices can be used as a supervision signal for self-supervised feature learning.

Learning with Multiple Pretext Tasks: Most existing self-supervised visual feature learning methods learn features by training ConvNet to solve one pretext tasks. Different pretext tasks provide different supervision signals which can help the network learn more representative features. Only a few work explored the multiple pretext tasks learning for self-supervised feature learning [53], [91], [108], [174], [186]. More work can be done by studying the multiple pretext task self-supervised feature learning.

9 Conclusion

Self-supervised image and video feature learning with deep convolutional neural network has obtained great success and the margin between the performance of self-supervised methods and that of supervised methods on some downstream tasks becomes very small. This paper has extensively reviewed recently deep convolution neural network-based methods for self-supervised image and video feature learning from all perspectives including common network architectures, pretext tasks, algorithms, datasets, performance comparison, discussions, and future directions etc. The comparative summary of the methods, datasets, and performance in tabular forms clearly demonstrate their properties which will benefit researchers in the computer vision community.

References

[1] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, pp. 580–587, 2014.

[2] R. Girshick, “Fast R-CNN,” in *ICCV*, 2015.

[3] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *NIPS*, pp. 91–99, 2015.

[4] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *CVPR*, pp. 3431–3440, 2015.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs,” *TPAMI*, 2018.

[6] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in *CVPR*, pp. 2881–2890, 2017.

[7] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural image caption generator,” in *CVPR*, pp. 3156–3164, 2015.

[8] K. He, R. Girshick, and P. Dollár, “Rethinking imagenet pre-training,” *ICCV*, 2018.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *NIPS*, pp. 1097–1105, 2012.

[10] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *ICLR*, pp. 1–14, 2015.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *CVPR*, pp. 1–9, 2015.

[12] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *CVPR*, pp. 770–778, 2016.

[13] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, “Densely connected convolutional networks,” in *CVPR*, p. 3, 2017.

[14] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *CVPR*, pp. 248–255, IEEE, 2009.

[15] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov, *et al.*, “The open images dataset v4,” *IJCV*, pp. 1–26, 2020.

[16] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, “Photo-realistic single image super-resolution using a generative adversarial network,” in *CVPR*, pp. 105–114, 2017.

[17] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, “Learning spatiotemporal features with 3D convolutional networks,” in *ICCV*, 2015.

[18] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” in *CVPR*, pp. 652–660, 2017.

[19] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, *et al.*, “The kinetics human action video dataset,” *arXiv preprint arXiv:1705.06950*, 2017.

[20] R. Zhang, P. Isola, and A. A. Efros, “Colorful image colorization,” in *ECCV*, pp. 649–666, Springer, 2016.

[21] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, “Context encoders: Feature learning by inpainting,” in *CVPR*, pp. 2536–2544, 2016.

[22] M. Noroozi and P. Favaro, “Unsupervised learning of visual represenions by solving jigsaw puzzles,” in *ECCV*, 2016.

[23] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, “Network dissection: Quantifying interpretability of deep visual representations,” in *CVPR*, 2017.

[24] K. Simonyan and A. Zisserman, “Two-stream convolutional networks for action recognition in videos,” in *NIPS*, pp. 568–576, 2014.

[25] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, “Long-term recurrent convolutional networks for visual recognition and description,” in *CVPR*, 2015.

[26] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, “Understanding deep learning requires rethinking generalization,” *ICLR*, 2017.

[27] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, “Gan dissection: Visualizing and understanding generative adversarial networks,” *ICLR*, 2019.

[28] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *ECCV*, pp. 818–833, Springer, 2014.

[29] K. Soomro, A. R. Zamir, and M. Shah, “UCF101: A dataset of 101 human actions classes from videos in the wild,” *CRCV-TR*, 2012.

[30] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan, “Learning features by watching objects move,” in *CVPR*, 2017.

[31] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, “Deep clustering for unsupervised learning of visual features,” in *ECCV*, 2018.

[32] G. Larsson, M. Maire, and G. Shakhnarovich, “Colorization as a proxy task for visual understanding,” in *CVPR*, 2017.

[33] I. Misra, C. L. Zitnick, and M. Hebert, “Shuffle and learn: unsupervised learning using temporal order verification,” in *ECCV*, pp. 527–544, Springer, 2016.

[34] B. Korbar, D. Tran, and L. Torresani, “Cooperative learning of audio and video models from self-supervised synchronization,” in *NIPS*, pp. 7773–7784, 2018.

[35] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *NIPS*, pp. 2672–2680, 2014.

[36] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *ICCV*, 2017.

[37] C. Vondrick, H. Pirsiavash, and A. Torralba, “Generating videos with scene dynamics,” in *NIPS*, pp. 613–621, 2016.

[38] S. Tulyakov, M.-Y. Liu, X. Yang, and J. Kautz, “Mocogan: Decomposing motion and content for video generation,” *CVPR*, 2018.

[39] N. Srivastava, E. Mansimov, and R. Salakhutdinov, “Unsupervised Learning of Video Representations using LSTMs,” in *ICML*, 2015.

[40] M. Noroozi, A. Vinjimoor, P. Favaro, and H. Pirsiavash, “Boosting self-supervised learning via knowledge transfer,” *CVPR*, 2018.

[41] D. Li, W.-C. Hung, J.-B. Huang, S. Wang, N. Ahuja, and M.-H. Yang, “Unsupervised visual representation learning by graph-based consistent constraints,” in *ECCV*, 2016.

[42] U. Ahsan, R. Madhok, and I. Essa, “Video jigsaw: Unsupervised learning of spatiotemporal context for video action recognition,” *WACV*, 2019.

[43] 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,” *CVPR*, 2019.

[44] D. Kim, D. Cho, D. Yoo, and I. S. Kweon, “Learning image representations by completing damaged jigsaw puzzles,” *WACV*, 2018.

[45] C. Doersch, A. Gupta, and A. A. Efros, “Unsupervised visual representation learning by context prediction,” in *ICCV*, pp. 1422–1430, 2015.

[46] S. Gidaris, P. Singh, and N. Komodakis, “Unsupervised representation learning by predicting image rotations,” in *ICLR*, 2018.

[47] L. Jing and Y. Tian, “Self-supervised spatiotemporal feature learning by video geometric transformations,” *arXiv preprint arXiv:1811.11387*, 2018.

[48] D. Wei, J. Lim, A. Zisserman, and W. T. Freeman, “Learning and using the arrow of time,” in *CVPR*, pp. 8052–8060, 2018.

[49] H.-Y. Lee, J.-B. Huang, M. Singh, and M.-H. Yang, “Unsupervised representation learning by sorting sequences,” in *ICCV*, pp. 667–676, IEEE, 2017.

[50] D. Xu, J. Xiao, Z. Zhao, J. Shao, D. Xie, and Y. Zhuang, “Self-supervised spatiotemporal learning via video clip order prediction,” in *CVPR*, pp. 10334–10343, 2019.

[51] A. Faktor and M. Irani, “Video segmentation by non-local consensus voting.,” in *BMVC*, 2014.

[52] O. Stretcu and M. Leordeanu, “Multiple frames matching for object discovery in video.,” in *BMVC*, 2015.

[53] Z. Ren and Y. J. Lee, “Cross-domain self-supervised multi-task feature learning using synthetic imagery,” in *CVPR*, 2018.

[54] P. Krhenbhl, “Free supervision from video games,” in *CVPR*, June 2018.

[55] I. Croitoru, S.-V. Bogolin, and M. Leordeanu, “Unsupervised learning from video to detect foreground objects in single images,” *ICCV*, 2017.

[56] Y. Li, M. Paluri, J. M. Rehg, and P. Dollár, “Unsupervised learning of edges,” *CVPR*, pp. 1619–1627, 2016.

[57] H. Jiang, G. Larsson, M. Maire Greg Shakhnarovich, and E. Learned-Miller, “Self-supervised relative depth learning for urban scene understanding,” in *ECCV*, pp. 19–35, 2018.

[58] R. Arandjelovic and A. Zisserman, “Look, listen and learn,” in *ICCV*, pp. 609–617, IEEE, 2017.

[59] N. Sayed, B. Brattoli, and B. Ommer, “Cross and learn: Cross-modal self-supervision,” *GCPR*, 2018.

[60] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," *arXiv preprint arXiv:2002.05709*, 2020.

[61] P. Agrawal, J. Carreira, and J. Malik, "Learning to see by moving," in *ICCV*, pp. 37–45, 2015.

[62] D. Jayaraman and K. Grauman, "Learning image representations tied to ego-motion," in *ICCV*, pp. 1413–1421, 2015.

[63] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *IJCV*, pp. 303–338, 2010.

[64] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *CVPR*, pp. 3213–3223, 2016.

[65] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Semantic understanding of scenes through the ade20k dataset," *IJCV*, pp. 302–321, 2019.

[66] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*, pp. 740–755, Springer, 2014.

[67] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, pp. 779–788, 2016.

[68] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *CVPR*, 2017.

[69] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *ECCV*, pp. 21–37, Springer, 2016.

[70] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection.," in *CVPR*, p. 4, 2017.

[71] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *TPAMI*, 2018.

[72] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, pp. 293–300, 1999.

[73] H. Kuehne, H. Jhuang, R. Stiefelhagen, and T. Serre, "Hmdb51: A large video database for human motion recognition," in *HPCSE*, pp. 571–582, Springer, 2013.

[74] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," tech. rep., Citeseer, 2009.

[75] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, pp. 2278–2324, 1998.

[76] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *NIPS*, pp. 487–495, 2014.

[77] B. Zhou, A. Khosla, A. Lapedriza, A. Torralba, and A. Oliva, "Places: An image database for deep scene understanding," *arXiv preprint arXiv:1610.02055*, 2016.

[78] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *AISTATS*, pp. 215–223, 2011.

[79] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," in *CVPR*, pp. 190–198, IEEE, 2017.

[80] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPSW*, p. 5, 2011.

[81] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *ICASSP*, pp. 776–780, IEEE, 2017.

[82] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *CVPR*, pp. 3354–3361, IEEE, 2012.

[83] M. Monfort, B. Zhou, S. A. Bargal, T. Yan, A. Andonian, K. Ramakrishnan, L. Brown, Q. Fan, D. Gutfruend, C. Vondrick, *et al.*, "Moments in time dataset: one million videos for event understanding," *TPAMI*, pp. 502–508, 2019.

[84] J. McCormac, A. Handa, S. Leutenegger, and A. J. Davison, "Scenenet rgb-d: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation," in *ICCV*, 2017.

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

[86] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *CVPR*, pp. 1912–1920, 2015.

[87] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[88] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas, "A scalable active framework for region annotation in 3d shape collections," *TOG*, pp. 1–12, 2016.

[89] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, "Detection and classification of acoustic scenes and events," *IEEE Transactions on Multimedia*, pp. 1733–1746, 2015.

[90] K. J. Piczak, "Esc: Dataset for environmental sound classification," in *Proceedings of the 23rd ACM international conference on Multimedia*, pp. 1015–1018, 2015.

[91] L. Jing, Y. Chen, L. Zhang, M. He, and Y. Tian, "Self-supervised feature learning by cross-modality and cross-view correspondences," *arXiv preprint arXiv:2004.05749*, 2020.

[92] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *ECCV*, pp. 577–593, Springer, 2016.

[93] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," *ICLR*, 2017.

[94] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and Locally Consistent Image Completion," *SIGGRAPH*, 2017.

[95] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *ICLR*, 2016.

[96] T. Chen, X. Zhai, M. Ritter, M. Lucic, and N. Houlsby, "Self-supervised gans via auxiliary rotation loss," in *CVPR*, pp. 12154–12163, 2019.

[97] R. Zhang, P. Isola, and A. A. Efros, "Split-brain autoencoders: Unsupervised learning by cross-channel prediction," in *CVPR*, 2017.

[98] S. Jenni and P. Favaro, "Self-supervised feature learning by learning to spot artifacts," *CVPR*, 2018.

[99] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *ICML*, 2017.

[100] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *ICML*, pp. 478–487, 2016.

[101] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," *ICLR*, 2020.

[102] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[103] R. Santa Cruz, B. Fernando, A. Cherian, and S. Gould, "Visual permutation learning," *TPAMI*, 2018.

[104] T. N. Mundhenk, D. Ho, and B. Y. Chen, "Improvements to context based self-supervised learning," in *CVPR*, 2018.

[105] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," in *CVPR*, pp. 5147–5156, 2016.

[106] M. Noroozi, H. Pirsiavash, and P. Favaro, "Representation learning by learning to count," in *ICCV*, 2017.

[107] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," *CVPR*, 2019.

[108] C. Doersch and A. Zisserman, "Multi-task self-supervised visual learning," in *ICCV*, 2017.

[109] P. Bojanowski and A. Joulin, "Unsupervised learning by predicting noise," *ICML*, 2017.

[110] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *ICCV*, 2015.

[111] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, pp. 504–507, 2006.

[112] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *NIPS*, pp. 2234–2242, 2016.

[113] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *NIPS*, pp. 6626–6637, 2017.

[114] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *ICLR*, 2019.

[115] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” *CVPR*, 2019.

[116] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” *CVPR*, 2017.

[117] R. Zhang, J.-Y. Zhu, P. Isola, X. Geng, A. S. Lin, T. Yu, and A. A. Efros, “Real-time user-guided image colorization with learned deep priors,” *TOG*, 2017.

[118] S. Iizuka, E. Simo-Serra, and H. Ishikawa, “Let there be color!: joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification,” *TOG*, p. 110, 2016.

[119] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data clustering: a review,” *CSUR*, pp. 264–323, 1999.

[120] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *CVPR*, pp. 886–893, IEEE, 2005.

[121] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *IJCV*, pp. 91–110, 2004.

[122] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, “Image classification with the fisher vector: Theory and practice,” *IJCV*, pp. 222–245, 2013.

[123] M. Patrick, Y. M. Asano, R. Fong, J. F. Henriques, G. Zweig, and A. Vedaldi, “Multi-modal self-supervision from generalized data transformations,” *arXiv preprint arXiv:2003.04298*, 2020.

[124] L. Jing, Y. Chen, L. Zhang, M. He, and Y. Tian, “Self-supervised modal and view invariant feature learning,” *arXiv preprint*, 2020.

[125] D. Kim, D. Cho, and I. S. Kweon, “Self-supervised video representation learning with space-time cubic puzzles,” *AAAI*, 2019.

[126] U. Büchler, B. Brattoli, and B. Ommer, “Improving spatiotemporal self-supervision by deep reinforcement learning,” in *ECCV*, pp. 770–786, 2018.

[127] P. Goyal, D. Mahajan, A. Gupta, and I. Misra, “Scaling and benchmarking self-supervised visual representation learning,” *ICCV*, 2019.

[128] S. Shah, D. Dey, C. Lovett, and A. Kapoor, “Airsim: High-fidelity visual and physical simulation for autonomous vehicles,” in *Field and service robotics*, pp. 621–635, Springer, 2018.

[129] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *CoRL*, pp. 1–16, 2017.

[130] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, “Convolutional lstm network: A machine learning approach for precipitation nowcasting,” in *NIPS*, pp. 802–810, 2015.

[131] T. Han, W. Xie, and A. Zisserman, “Video representation learning by dense predictive coding,” in *ICCVW*, 2019.

[132] Z. Luo, B. Peng, D.-A. Huang, A. Alahi, and L. Fei-Fei, “Unsupervised learning of long-term motion dynamics for videos,” in *CVPR*, 2017.

[133] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee, “Decomposing motion and content for natural video sequence prediction,” in *ICLR*, 2017.

[134] M. Saito, E. Matsumoto, and S. Saito, “Temporal generative adversarial nets with singular value clipping,” in *ICCV*, p. 5, 2017.

[135] C. Vondrick, A. Shrivastava, A. Fathi, S. Guadarrama, and K. Murphy, “Tracking emerges by colorizing videos,” in *ECCV*, 2018.

[136] B. Brattoli, U. Büchler, A.-S. Wahl, M. E. Schwab, and B. Ommer, “Lstm self-supervision for detailed behavior analysis,” in *CVPR*, pp. 3747–3756, IEEE, 2017.

[137] B. Fernando, H. Bilen, E. Gavves, and S. Gould, “Self-supervised video representation learning with odd-one-out networks,” in *CVPR*, 2017.

[138] D. Jayaraman and K. Grauman, “Slow and steady feature analysis: higher order temporal coherence in video,” in *CVPR*, pp. 3852–3861, 2016.

[139] X. Wang, K. He, and A. Gupta, “Transitive invariance for self-supervised visual representation learning,” in *ICCV*, 2017.

[140] Y. Zhang, S. Khamis, C. Rhemann, J. Valentin, A. Kowdle, V. Tankovich, M. Schoenberg, S. Izadi, T. Funkhouser, and S. Fanello, “Activestereonet: End-to-end self-supervised learning for active stereo systems,” in *ECCV*, pp. 784–801, 2018.

[141] A. Owens, J. Wu, J. H. McDermott, W. T. Freeman, and A. Torralba, “Ambient sound provides supervision for visual learning,” in *ECCV*, pp. 801–816, Springer, 2016.

[142] A. Owens and A. A. Efros, “Audio-visual scene analysis with self-supervised multisensory features,” *ECCV*, 2018.

[143] A. Mahendran, J. Thewlis, and A. Vedaldi, “Cross pixel optical flow similarity for self-supervised learning,” *ACCV*, 2018.

[144] Y. Zou, Z. Luo, and J.-B. Huang, “Df-net: Unsupervised joint learning of depth and flow using cross-task consistency,” in *ECCV*, pp. 38–55, Springer, 2018.

[145] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, “Unsupervised learning of depth and ego-motion from video,” in *CVPR*, p. 7, 2017.

[146] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, “Flownet: Learning optical flow with convolutional networks,” in *ICCV*, pp. 2758–2766, 2015.

[147] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, “Flownet 2.0: Evolution of optical flow estimation with deep networks,” in *CVPR*, pp. 1647–1655, IEEE, 2017.

[148] Z. Yin and J. Shi, “Geonet: Unsupervised learning of dense depth, optical flow and camera pose,” in *CVPR*, 2018.

[149] J. Wang, J. Jiao, L. Bao, S. He, Y. Liu, and W. Liu, “Self-supervised spatio-temporal representation learning for videos by predicting motion and appearance statistics,” in *CVPR*, pp. 4006–4015, 2019.

[150] S. Meister, J. Hur, and S. Roth, “Unflow: Unsupervised learning of optical flow with a bidirectional census loss,” in *AAAI*, pp. 7251–7259, 2018.

[151] G. Iyer, J. K. Murthy, G. Gupta, K. M. Krishna, and L. Paull, “Geometric consistency for self-supervised end-to-end visual odometry,” *CVPRW*, 2018.

[152] H. Alwassel, D. Mahajan, L. Torresani, B. Ghanem, and D. Tran, “Self-supervised learning by cross-modal audio-video clustering,” *arXiv preprint arXiv:1911.12667*, 2019.

[153] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, “Deep end2end voxel2voxel prediction,” in *CVPRW*, pp. 17–24, 2016.

[154] M. Mathieu, C. Couprie, and Y. LeCun, “Deep multi-scale video prediction beyond mean square error,” in *ICLR*, 2016.

[155] F. A. Reda, G. Liu, K. J. Shih, R. Kirby, J. Barker, D. Tarjan, A. Tao, and B. Catanzaro, “Sdc-net: Video prediction using spatially-displaced convolution,” in *ECCV*, pp. 718–733, 2018.

[156] M. Babaeizadeh, C. Finn, D. Erhan, R. H. Campbell, and S. Levine, “Stochastic variational video prediction,” *ICLR*, 2018.

[157] X. Liang, L. Lee, W. Dai, and E. P. Xing, “Dual motion gan for future-flow embedded video prediction,” in *ICCV*, 2017.

[158] C. Finn, I. Goodfellow, and S. Levine, “Unsupervised learning for physical interaction through video prediction,” in *NIPS*, pp. 64–72, 2016.

[159] K. Hara, H. Kataoka, and Y. Satoh, “Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet,” in *CVPR*, pp. 18–22, 2018.

[160] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, “A closer look at spatiotemporal convolutions for action recognition,” in *CVPR*, pp. 6450–6459, 2018.

[161] C. Gan, B. Gong, K. Liu, H. Su, and L. J. Guibas, “Geometry guided convolutional neural networks for self-supervised video representation learning,” in *CVPR*, pp. 5589–5597, 2018.

[162] K. J. Piczak, “Environmental sound classification with convolutional neural networks,” in *MLSP*, pp. 1–6, IEEE, 2015.

[163] H. B. Sailor, D. M. Agrawal, and H. A. Patil, “Unsupervised filterbank learning using convolutional restricted boltzmann machine for environmental sound classification.,” in *INTERSPEECH*, pp. 3107–3111, 2017.

[164] G. Roma, W. Nogueira, and P. Herrera, “Recurrence quantification analysis features for environmental sound recognition,” in *WASPAA*, pp. 1–4, IEEE, 2013.

[165] Y. Aytar, C. Vondrick, and A. Torralba, “Soundnet: Learning sound representations from unlabeled video,” in *NIPS*, pp. 892–900, 2016.

[166] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, “Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling,” in *NIPS*, pp. 82–90, 2016.

[167] Y. Yang, C. Feng, Y. Shen, and D. Tian, “Foldingnet: Point cloud auto-encoder via deep grid deformation,” in *CVPR*, pp. 206–215, 2018.

[168] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, “Learning representations and generative models for 3d point clouds,” *ICLRW*, 2018.

[169] M. Gadelha, R. Wang, and S. Maji, “Multiresolution tree networks for 3d point cloud processing,” in *ECCV*, pp. 103–118, 2018.

[170] Y. Zhao, T. Birdal, H. Deng, and F. Tombari, “3d point capsule networks,” in *CVPR*, pp. 1009–1018, 2019.

[171] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta, "Learning a predictable and generative vector representation for objects," in *ECCV*, pp. 484–499, Springer, 2016.

[172] A. Sharma, O. Grau, and M. Fritz, "Vconv-dae: Deep volumetric shape learning without object labels," in *ECCV*, pp. 236–250, Springer, 2016.

[173] L. Zhang and Z. Zhu, "Unsupervised feature learning for point cloud understanding by contrasting and clustering using graph convolutional neural networks," in *3DV*, pp. 395–404, IEEE, 2019.

[174] K. Hassani and M. Haley, "Unsupervised multi-task feature learning on point clouds," in *ICCV*, pp. 8160–8171, 2019.

[175] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *TOG*, pp. 1–12, 2019.

[176] Y. Feng, Y. Feng, H. You, X. Zhao, and Y. Gao, "Meshnet: mesh neural network for 3d shape representation," in *AAAI*, pp. 8279–8286, 2019.

[177] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," in *ICCV*, 2015.

[178] A. Kolesnikov, X. Zhai, and L. Beyer, "Revisiting self-supervised visual representation learning," in *CVPR*, June 2019.

[179] W. Hong, Z. Wang, M. Yang, and J. Yuan, "Conditional generative adversarial network for structured domain adaptation," in *CVPR*, pp. 1335–1344, 2018.

[180] Y. Cai, L. Ge, J. Cai, and J. Yuan, "Weakly-supervised 3d hand pose estimation from monocular rgb images," in *ECCV*, pp. 666–682, 2018.

[181] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "Youtube-8m: A large-scale video classification benchmark," *arXiv preprint arXiv:1609.08675*, 2016.

[182] W. Li, L. Wang, W. Li, E. Agustsson, and L. Van Gool, "Webvision database: Visual learning and understanding from web data," *arXiv preprint arXiv:1708.02862*, 2017.

[183] L. Gomez, Y. Patel, M. Rusiñol, D. Karatzas, and C. Jawahar, "Self-supervised learning of visual features through embedding images into text topic spaces," in *CVPR*, IEEE, 2017.

[184] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in *ICML*, 2018.

[185] D. Ghadiyaram, D. Tran, and D. Mahajan, "Large-scale weakly-supervised pre-training for video action recognition," in *CVPR*, pp. 12046–12055, 2019.

[186] A. Piergiovanni, A. Angelova, and M. S. Ryoo, "Evolving losses for unlabeled video representation learning," *CVPRW*, 2019.

Longlong Jing received the B.E. degree from Central South University and M.S. degrees from The Graduate Center of The City University of New York in 2015 and 2019, and is currently a Ph.D. candidate at The Graduate Center of The City University of New York. His current research focuses on computer vision and deep learning including human action recognition, self-supervised feature learning, weakly supervised semantic segmentation, and medical imaging analysis.

Yingli Tian (M'99–SM'01–F'18) received the B.S. and M.S. degrees from Tianjin University, China, in 1987 and 1990, and the Ph.D. degree from Chinese University of Hong Kong, Hong Kong, in 1996. After holding a faculty position at National Laboratory of Pattern Recognition, Chinese Academy of Sciences, Beijing, she joined Carnegie Mellon University in 1998, where she was a postdoctoral fellow at the Robotics Institute. She then worked as a research staff member in IBM T. J. Watson Research Center from 2001 to 2008. She is one of the inventors of the IBM Smart Surveillance Solutions. She is a CUNY Distinguished Professor in the Department of Electrical Engineering at the City College and the Department of Computer Science at the Graduate Center, the City University of New York since 2008. Her current research focuses on a wide range of computer vision problems from scene understanding, medical imaging analysis, human behavior analysis, to facial expression recognition and assistive technology. She is a fellow of IEEE.

Self-supervised Visual Feature Learning with Deep Neural Networks: A Survey Appendix

Longlong Jing, *Student Member, IEEE* and Yingli Tian*, *Fellow, IEEE*

Abstract—Large-scale labeled data are generally required to train deep neural networks in order to obtain better performance in visual feature learning from images or videos for computer vision applications. To avoid extensive cost of collecting and annotating large-scale datasets, as a subset of unsupervised learning methods, self-supervised learning methods are proposed to learn general image and video features from large-scale unlabeled data without using any human-annotated labels. This paper provides an extensive review of deep learning-based self-supervised general visual feature learning methods from images or videos. First, the motivation, general pipeline, and terminologies of this field are described. Then the common deep neural network architectures that used for self-supervised learning are summarized. Next, the schema and evaluation metrics of self-supervised learning methods are reviewed followed by the commonly used datasets for images, videos, audios, and 3D data, as well as the existing self-supervised visual feature learning methods. Finally, quantitative performance comparisons of the reviewed methods on benchmark datasets are summarized and discussed for both image and video feature learning. At last, this paper is concluded and lists a set of promising future directions for self-supervised visual feature learning.

Index Terms—Self-supervised Learning, Unsupervised Learning, Convolutional Neural Network, Transfer Learning, Deep Learning.

1 Formulation of Different Learning Schemas

Based on the training labels, visual feature learning methods can be grouped into the following four categories: supervised, semi-supervised, weakly supervised, and unsupervised. In this section, the four types of learning methods are compared and key terminologies are defined.

1.1 Supervised Learning Formulation

For supervised learning, given a dataset X, for each data X_i in X, there is a corresponding **human-annotated label** Y_i. For a set of N labeled training data $D = \{X_i\}_{i=0}^N$, the training loss function is defined as:

$$loss(D) = \min_{\theta} \frac{1}{N} \sum_{i=1}^{N} loss(X_i, Y_i). \quad (1)$$

Trained with accurate human-annotated labels, the supervised learning methods obtained break-through results on different computer vision applications [1], [2], [3], [4]. However, data collection and annotation usually are expensive and may require special skills. Therefore, semi-supervised, weakly supervised, and unsupervised learning methods were proposed to reduce the cost.

1.2 Semi-Supervised Learning Formulation

For semi-supervised visual feature learning, given a small labeled dataset X and a large unlabeled dataset Z, for each data X_i in X, there is a corresponding **human-annotated label** Y_i. For a set of N labeled training data $D_1 = \{X_i\}_{i=0}^N$ and M unlabeled training data $D_2 = \{Z_i\}_{i=0}^M$, the training loss function is defined as:

$$loss(D_1, D_2) = \min_{\theta} \frac{1}{N} \sum_{i=1}^{N} loss(X_i, Y_i) + \frac{1}{M} \sum_{i=1}^{M} loss(Z_i, R(Z_i, X)), \quad (2)$$

where the $R(Z_i, X)$ is a task-specific function to represent the relation between each unlabeled training data Z_i with the labeled dataset X.

1.3 Weakly Supervised Learning Formulation

For weakly supervised visual feature learning, given a dataset X, for each data X_i in X, there is a corresponding **coarse-grained label** C_i. For a set of N training data $D = \{X_i\}_{i=0}^N$, the training loss function is defined as:

$$loss(D) = \min_{\theta} \frac{1}{N} \sum_{i=1}^{N} loss(X_i, C_i). \quad (3)$$

Since the cost of weak supervision is much lower than the fine-grained label for supervised methods, large-scale datasets are relatively easier to obtain. Recently, many weakly supervised learning methods were proposed for

- L. Jing is with the Department of Computer Science, The Graduate Center, The City University of New York, NY, 10016. E-mail: ljing@gradcenter.cuny.edu
- Y. Tian is with the Department of Electrical Engineering, The City College, and the Department of Computer Science, the Graduate Center, the City University of New York, NY, 10031. E-mail: ytian@ccny.cuny.edu
  *Corresponding author

This material is based upon work supported by the National Science Foundation under award number IIS-1400802.

different tasks and achieved promising results [5], [6], [7], [8].

1.4 Unsupervised/Self-supervised Learning Formulation

Unsupervised learning refers to learning methods that do not need any human-annotated labels. This type of methods including fully unsupervised learning methods in which the methods do not need any labels at all, as well as self-supervised learning methods in which networks are explicitly trained with automatically generated pseudo labels without involving any human annotation.

Recently, many self-supervised learning methods for visual feature learning have been developed without using any human-annotated labels [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [19], [20], [21]. Some papers refer to this type of learning methods as unsupervised learning [**?**], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33]. Compared to supervised learning methods which require a data pair X_i and Y_i while Y_i is annotated by human labors, self-supervised learning also trained with data X_i along with its **pseudo label** P_i while P_i is automatically generated for a pre-defined pretext task without involving any human annotation. The pseudo label P_i can be generated by using attributes of images or videos such as the context of images [22], [34], [35], [36], or by traditional hand-designed methods [37], [38], [39].

Given a set of N training data $D = \{P_i\}_{i=0}^{N}$, the training loss function is defined as:

$$loss(D) = \min_{\theta} \frac{1}{N} \sum_{i=1}^{N} loss(X_i, P_i). \tag{4}$$

As long as the pseudo labels P are automatically generated without involving human annotations, then the methods belong to self-supervised learning. Recently, self-supervised learning methods have achieved great progress. This paper focuses on the self-supervised learning methods that mainly designed for visual feature learning, while the features have the ability to be transferred to multiple visual tasks and to perform new tasks by learning from limited labeled data. This paper summarizes these self-supervised feature learning methods from different perspectives including network architectures, commonly used pretext tasks, datasets, and applications, etc.

2 Common Deep Network Architectures

No matter the categories of learning methods, they share similar network architectures. This section reviews common architectures for learning both image and video features.

2.1 Architectures for Learning Image Features

Various 2DConvNets have been designed for image feature learning [1], [40], [41], [42], [43], [44], [45]. Here, five milestone architectures for image feature learning including AlexNet [1], VGG [40], GoogLeNet [41], ResNet [42], and DenseNet [43] are reviewed.

2.1.1 *AlexNet*

AlexNet obtained a big improvement in the performance of image classification on ImageNet dataset compared to the previous state-of-the-art methods [1]. With the support of powerful GPUs, AlexNet which has 62.4 million parameters were trained on ImageNet with 1.3 million images. As shown in Fig. 1, the architecture of AlexNet has 8 layers in which 5 are convolutional layers and 3 are fully connected layers. The ReLU is applied after each convolutional layer. 94% of the network parameters come from the fully connected layers. With this scale of parameters, the network can easily be over-fitting. Therefore, different kinds of techniques are applied to avoid over-fitting problem including data augmentation, dropout, and normalization.

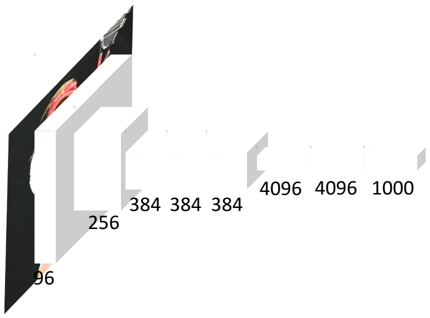

Fig. 1. The architecture of AlexNet [1]. The numbers indicate the number of channels of each feature map. Figure is reproduced based on AlexNet [1].

2.1.2 *VGG*

VGG is proposed by Simonyan and Zisserman and won the first place for ILSVRC 2013 competition [40]. Simonyan and Zisserman proposed various depths of networks, while the 16-layer VGG is the most widely used one due to its moderate model size and its superior performance. The architecture of VGG-16 is shown in Fig. 2. It has 16 convolutional layers belong to five convolution blocks. The main difference between VGG and AlexNet is that AlexNet has large convolution stride and large kernel size while all the convolution kernels in VGG have same small size (3×3) and small convolution stride (1×1). The large kernel size leads to too many parameters and large model size, while the large convolution stride may cause the network to miss some fine features in the lower layers. The smaller kernel size makes the training of very deep convolution neural network feasible while still reserving the fine-grained information in the network.

2.1.3 *ResNet*

VGG demonstrated that deeper networks are possible to obtain better performance. However, deeper networks are more difficult to train due to two problems: gradient vanishing and gradient explosion. ResNet is proposed by He *et al.* to use the skip connection in convolution blocks by sending the previous feature map to the next convolution block to overcome the gradient vanishing and gradient explosion [42]. The details of the skip connection are shown in Fig. 3. With the skip connection, training of very deep neural networks on GPUs becomes feasible.

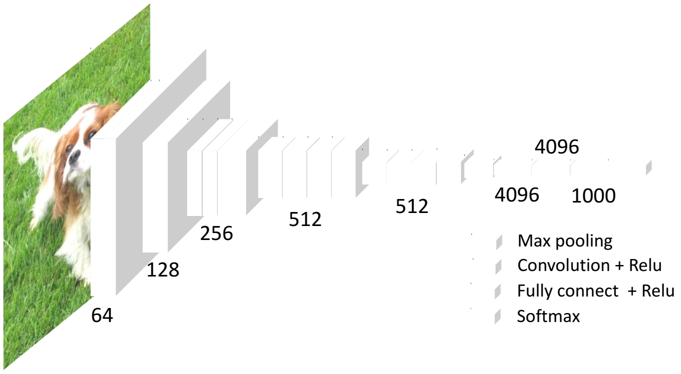

Fig. 2. The architecture of VGG [40]. Figure is reproduced based on VGG [40].

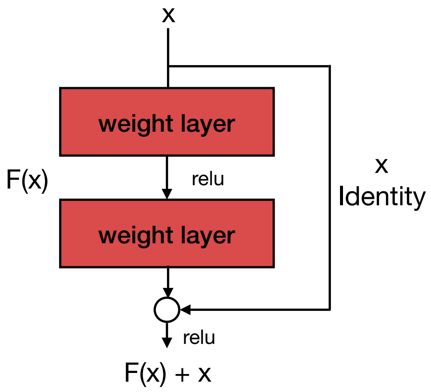

Fig. 3. The architecture of Residual block [42]. The identity mapping can effectively reduce gradient vanishing and explosion which make the training of very deep network feasible. Figure is reproduced based on ResNet [42].

In ResNet [42], He *et al.* also evaluated networks with different depths for image classification. Due to its smaller model size and superior performance, ResNet is often used as the base network for other computer vision tasks. The convolution blocks with skip connection also widely used as the basic building blocks.

2.1.4 GoogLeNet

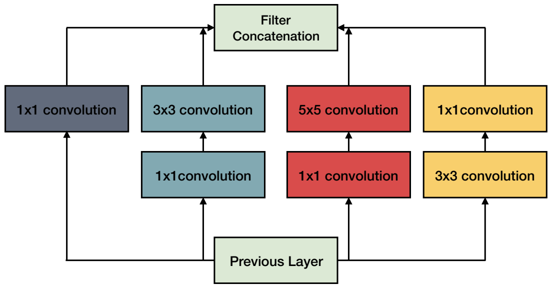

Fig. 4. The architecture of Inception block [41]. Figure is reproduced based on GoogLeNet [41].

GoogLeNet, a 22-layer deep network, is proposed by Szegedy *et al.* which won ILSVRC-2014 challenge with a top-5 test accuracy of 93.3% [41]. Compared to previous work that to build a deeper network, Szegedy *et al.* explored to build a wider network in which each layer has multiple parallel convolution layers. The basic block of GoogLeNet is inception block which consists of 4 parallel convolution layers with different kernel sizes and followed by 1×1 convolution for dimension reduction purpose. The architecture for the inception block of GoogLeNet is shown in Fig. 4. With a carefully crafted design, they increased the depth and width of the network while keeping the computational cost constant.

2.1.5 DenseNet

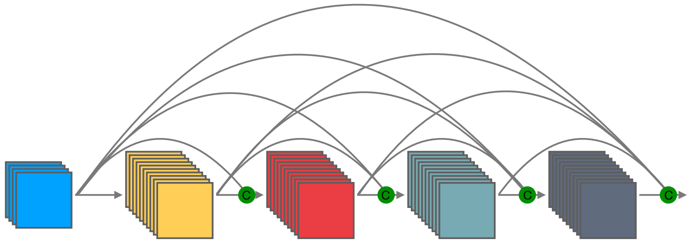

Fig. 5. The architecture of the Dense Block proposed in DenseNet [43]. Figure is reproduced based on [43].

Most of the networks including AlexNet, VGG, and ResNet follow a hierarchy architecture. The images are fed to the network and features are extracted by different layers. The shallow layers extract low-level general features, while the deep layers extract high-level task-specific features [46]. However, when a network goes deeper, the deeper layers may suffer from memorizing the low-level features needed by the network to accomplish the task.

To alleviate this problem, Huang *et al.* proposed the dense connection to send all the features before a convolution block as the input to the next convolution block in the neural network [43]. As shown in Fig. 5, the output features of all the previous convolution blocks serve as the input to the current block. In this way, the shallower blocks focus on the low-level general features while the deeper blocks can focus on the high-level task-specific features.

2.2 Architectures for Learning Video Features

To extract both spatial and temporal information from videos, several architectures have been designed for video feature learning including 2DConvNet-based methods [47], 3DConvNet-based methods [4], and LSTM-based methods [48]. The 2DConvNet-based methods apply 2DConvNet on every single frame and the image features of multiple frames are fused as video features. The 3DConvNet-based methods employ 3D convolution operation to simultaneously extract both spatial and temporal features from multiple frames. The LSTM-based methods employ LSTM to model long term dynamics within a video. This section briefly summarizes these three types of architectures of video feature learning.

2.2.1 Two-Stream Network

Videos generally are composed of various numbers of frames. To recognize actions in a video, networks are re-

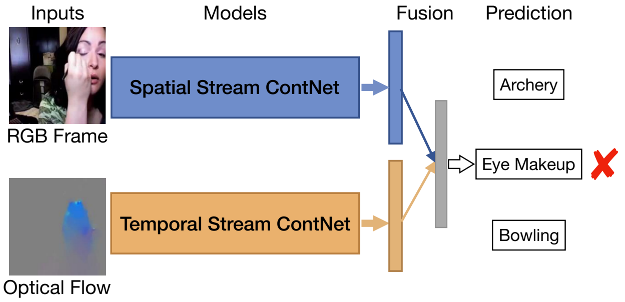

Fig. 6. The general architecture of the two-stream network which including one spatial stream and one temporal stream. Figure is reproduced based on [47].

quired to capture appearance features as well as temporal dynamics from frame sequences. As shown in Fig. 6, a two-stream 2DConvNet-based network is proposed by Simonyan and Zisserman for human action recognition, while using a 2DConvNet to capture spatial features from RGB stream and another 2DConvNet to capture temporal features from optical flow stream [47]. Optical flow encodes boundary of moving objects, therefore, the temporal stream ConvNet is relatively easier to capture the motion information within the frames.

Experiments showed that the fusion of the two streams can significantly improve action recognition accuracy. Later, this work has been extended to multi-stream network [49], [50], [51], [52], [53] to fuse features from different types of inputs such as dynamic images [54] and difference of frames [55].

2.2.2 Spatiotemporal Convolutional Neural Network

3D convolution operation was first proposed in 3DNet [56] for human action recognition. Compared to 2DConvNets which individually extract the spatial information of each frame and then fuse them together as video features, 3DConvNets are able to simultaneously extract both spatial and temporal features from multiple frames.

C3D [4] is a VGG-like 11-layer 3DConvNet designed for human action recognition. The network contains 8 convolutional layers, and 3 fully connected layers. All the kernels have the size of $3 \times 3 \times 3$, the convolution stride is fixed to 1 pixel. Due to its powerful ability of simultaneously extracting both spatial and temporal features from multiple frames, the network achieved state-of-the-art on several video analysis tasks including human action recognition [57], action similarity labeling [58], scene classification [59], and object recognition in videos [60].

The input of C3D is 16 consecutive RGB frames where the appearance and temporal cues from 16-frame clips are extracted. However, the paper of long-term temporal convolutions (LTC) [61] argues that, for the long-lasting actions, 16 frames are insufficient to represent whole actions which last longer. Therefore, larger numbers of frames are employed to train 3DConvNets and achieved better performance than C3D [61], [62].

With the success of applying 3D convolution on video analysis tasks, various 3DConvNet architectures have been proposed [63], [64], [65]. Hara *et al.* proposed 3DResNet by replacing all the 2D convolution layers in ResNet with 3D convolution layers and showed comparable performance with the state-of-the-art performance on action recognition task on several datasets [64].

2.2.3 Recurrent Neural Network

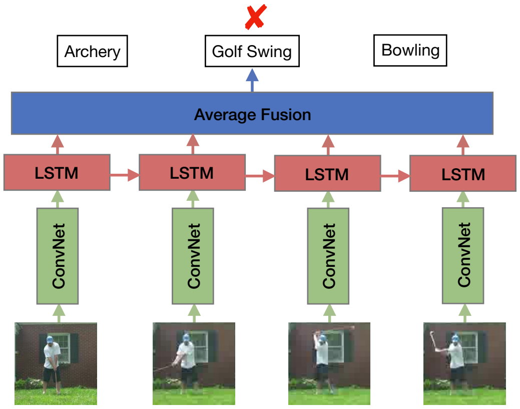

Fig. 7. The architecture of long-term recurrent convolutional networks (LRCN) [48]. LSTM is employed to model the long term temporal information within a frame sequence. Figure is reproduced based on [48].

Due to the ability to model the temporal dynamics within a sequence, recurrent neural networks (RNN) are often applied to videos as ordered frame sequences. Compared to standard RNN [66], long short term memory (LSTM) uses memory cells to store, modify, and access internal states, to better model the long-term temporal relationships within video frames [67].

Based on the advantage of the LSTM, Donahue *et al.* proposed long-term recurrent convolutional networks (LRCN) for human action recognition [48]. The framework of the LRCN is shown in Fig. 7. The LSTM is sequentially applied to the features extracted by ConvNets to model the temporal dynamics in the frame sequence. With the LSTM to model a video as frame sequences, this model is able to explicitly model the long-term temporal dynamics within a video. Later on, this model is extended to a deeper LSTM for action recognition [68], [69], video captioning [70], and gesture recognition tasks [71].

3 Datasets

This section summarizes the commonly used image, video, audio, and 3D object datasets for training and evaluating of self-supervised visual feature learning methods.

3.1 Image Datasets

- **CIFAR10:** The CIFAR10 dataset is a collection of tiny images for image classification task [72]. It consists of $60,000$ images of size 32×32 that covers 10 different classes. The 10 classes include airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The dataset is balanced and there are $6,000$ images of each class.
- **ImageNet:** The ImageNet dataset [73] contains 1.3 million images uniformly distributed into $1,000$

classes and is organized according to the WordNet hierarchy. Each image is assigned with only one class label. ImageNet is the most widely used dataset for self-supervised image feature learning.

- **MNIST:** The MNIST is a dataset of handwritten digits consisting of $70,000$ images while $60,000$ images belong to training set and the rest $10,000$ images are for testing [74]. All digits have been size-normalized and centered in fixed-size images.
- **PASCAL Visual Object Classes (VOC):** The VOC 2,012 dataset [75] contains 20 object categories including vehicles, household, animals, and other: aeroplane, bicycle, boat, bus, car, motorbike, train, bottle, chair, dining table, potted plant, sofa, TV/monitor, bird, cat, cow, dog, horse, sheep, and person. Each image in this dataset has pixel-level segmentation annotations, bounding box annotations, and object class annotations. This dataset has been widely used as a benchmark for object detection, semantic segmentation, and classification tasks. The PASCAL VOC dataset is split into three subsets: $1,464$ images for training, $1,449$ images for validation and a private testing [75]. All the self-supervised image representation learning methods are evaluated on this dataset with the three tasks.
- **Places:** The Places dataset [76] is proposed for scene recognition and contains more than 2.5 million images covering more than 205 scene categories with more than $5,000$ images per category.
- **Places365:** The Places365 is the 2nd generation of the Places database which is built for high-level visual understanding tasks, such as scene context, object recognition, action and event prediction, and theory-of-mind inference [77]. There are more than 10 million images covering more than 400 classes and $5,000$ to $30,000$ training images per class.
- **STL-10:** The STL-10 dataset is specifically designed for developing unsupervised feature learning [78]. It consists of 500 labeled training images, 800 testing images, and $100,000$ unlabeled images covering 10 classes which include airplane, bird, car, cat, deer, dog, horse, monkey, ship, and truck.
- **SUNCG:** The SUNCG dataset is a large synthetic 3D scene repository for indoor scenes which consists of over $45,000$ different scenes with manually created realistic room and furniture layouts [79]. The synthetic depth, object level semantic labels, and volumetric ground truth are available.
- **SVHN:** SVHN is a dataset for recognizing digits and numbers in natural scene images which obtained from house numbers from Google Street View images [80]. The dataset consists of over $600,000$ images and all digits have been resized to a fixed resolution of 32×32 pixels.

3.2 Video Datasets

- **AudioSet:** The AudioSet consists of $2,084,320$ human-labeled 10-second sound clips drawn from YouTube videos covers ontology of 632 audio event classes [81]. The event classes cover a wide range of human and animal sounds, musical instruments and genres, and common everyday environmental sounds. This dataset is mainly used for the self-supervised learning from video and audio consistence [12].
- **HMDB51:** Compared to other datasets, the HMDB51 dataset is a smaller video dataset for human action recognition. There are around $7,000$ video clips in this dataset belong to 51 human action categories [82]. The videos in HMDB51 dataset have 320×240 pixels spatial resolution and 30 FPS frame rate. In the self-supervised sensorial, the self-supervised models are fine-tuned on the dataset to evaluate the quality of the learned video features.
- **Kinetics:** The Kinetics dataset is a large-scale, high-quality dataset for human action recognition in videos [83]. The dataset consists of around $500,000$ video clips covering 600 human action classes with at least 600 video clips for each action class. Each video clip lasts around 10 seconds and is labeled with a single action class.
- **KITTI:** The KITTI dataset is collected from driving a car around a city which equipped with various sensors including high-resolution RGB camera, gray-scale stereo camera, a 3D laser scanner, and high-precision GPS measurements and IMU accelerations from a combined GPS/IMU system [84]. Videos with various modalities captured by these sensors are available in this dataset.
- **Moment in Time:** The Moment-in-Time dataset is a large balanced and diverse dataset for video understanding [85]. The dataset consists of 1 million video clips that cover 339 classes, and each video lasts around 3 seconds. The average number of video clips for each class is $1,757$ with a median of $2,775$. The video in this dataset contains videos that capturing visual and/or audible actions, produced by humans, animals, objects or nature [85].
- **SceneNet RGB-D:** The SceneNet RGB-D dataset is a large indoor synthetic video dataset which consists of 5 million rendered RGB-D images from over 15K trajectories in synthetic layouts with random but physically simulated object poses [86]. It provides pixel-level annotations for scene understanding problems such as semantic segmentation, instance segmentation, and object detection, and also for geometric computer vision problems such as optical flow, depth estimation, camera pose estimation, and 3D reconstruction [86].
- **UCF101:** The UCF101 is a widely used video dataset for human action recognition [57]. The dataset consists of $13,370$ video clips with more than 27 hours belonging to 101 categories in this dataset. The videos in this dataset have a spatial resolution of 320×240 pixels and 25 FPS frame rate. This dataset has been widely used for evaluating the performance of human action recognition. In the self-supervised sensorial, the self-supervised models are fine-tuned on the dataset and the accuracy of the action recognition are reported to evaluate the quality of the features.

- **YFCC100M:** The Yahoo Flickr Creative Commons 100 Million Dataset (YFCC100M) is a large public multimedia collection from Flickr, consisting of 100 million media data, of which around 99.2 million are images and 0.8 million are videos [87]. The statistics on hashtags used in the YFCC100M dataset show that the data distribution is severely unbalanced [88].

3.3 Audio Datasets

- **ESC50:** The ESC50 (Environmental Sound Classification) dataset consists of $2,000$ recordings. Each recording is about 5 seconds duration [89] and belongs to one of the five groups: animal sounds, natural soundscapes and water sounds, human (non-speech) sounds, interior/domestic sounds, and exterior/urban noises. This dataset is mainly used to evaluate the learned audio features.
- **DCASE:** The DCASE (Detection and Classification of Acoustic Scenes and Events) dataset consists of 100 training clips of 10 different audio classes (i.e. bus, busy street, office, open-air market, park, quiet street, restaurant, supermarket, tube, and tube station) [90]. This dataset is mainly used to evaluate the learned audio features.

3.4 3D Object Datasets

- **ModelNet40:** The ModelNet40 dataset consists of $12,311$ 3D CAD models belonging to 40 object classes [91]. The training split has $9,843$ objects and testing splits has $2,468$ objects. Different modalities like point cloud and multi-view images can be generated for each object, and all of these modalities can be used for training. This dataset is mainly used as an evaluation benchmark for 3D self-supervised learning methods.
- **ShapeNet:** The ShapeNet dataset consists of $57,445$ 3D CAD models belonging to 55 object classes [92]. Different modalities like point cloud and multi-view images can be generated for each object. This dataset is mainly used to pre-trained 3D self-supervised learning methods.
- **ShapeNet-PartSeg:** The ShapeNet-PartSeg contains $12,137$ 3D point cloud models belonging to 16 categories with part semantic segmentation labels [93]. In the self-supervised learning, this dataset usually is used to evaluate the generalizability of point cloud networks on part-segmentation task.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, pp. 1097–1105, 2012.

[2] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, pp. 3431–3440, 2015.

[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, pp. 580–587, 2014.

[4] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *ICCV*, 2015.

[5] D. Mahajan, R. B. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten, "Exploring the limits of weakly supervised pretraining," in *ECCV*, pp. 185–201, 2018.

[6] W. Li, L. Wang, W. Li, E. Agustsson, and L. Van Gool, "Webvision database: Visual learning and understanding from web data," *arXiv preprint arXiv:1708.02862*, 2017.

[7] D. Zhang, J. Han, L. Zhao, and D. Meng, "Leveraging prior-knowledge for weakly supervised object detection under a collaborative self-paced curriculum learning framework," *IJCV*, pp. 363–380, 2019.

[8] J. Han, Y. Yang, D. Zhang, D. Huang, D. Xu, and F. De La Torre, "Weakly-supervised learning of category-specific 3d object shapes," *TPAMI*, 2019.

[9] A. Mahendran, J. Thewlis, and A. Vedaldi, "Cross pixel optical flow similarity for self-supervised learning," *ACCV*, 2018.

[10] N. Sayed, B. Brattoli, and B. Ommer, "Cross and learn: Cross-modal self-supervision," *GCPR*, 2018.

[11] B. Korbar, D. Tran, and L. Torresani, "Cooperative learning of audio and video models from self-supervised synchronization," in *NIPS*, pp. 7773–7784, 2018.

[12] A. Owens and A. A. Efros, "Audio-visual scene analysis with self-supervised multisensory features," *ECCV*, 2018.

[13] D. Kim, D. Cho, and I. S. Kweon, "Self-supervised video representation learning with space-time cubic puzzles," *AAAI*, 2019.

[14] L. Jing and Y. Tian, "Self-supervised spatiotemporal feature learning by video geometric transformations," *arXiv preprint arXiv:1811.11387*, 2018.

[15] B. Fernando, H. Bilen, E. Gavves, and S. Gould, "Self-supervised video representation learning with odd-one-out networks," in *CVPR*, 2017.

[16] Z. Ren and Y. J. Lee, "Cross-domain self-supervised multi-task feature learning using synthetic imagery," in *CVPR*, 2018.

[17] X. Wang, K. He, and A. Gupta, "Transitive invariance for self-supervised visual representation learning," in *ICCV*, 2017.

[18] C. Doersch and A. Zisserman, "Multi-task self-supervised visual learning," in *ICCV*, 2017.

[19] T. N. Mundhenk, D. Ho, and B. Y. Chen, "Improvements to context based self-supervised learning," in *CVPR*, 2018.

[20] M. Noroozi, A. Vinjimoor, P. Favaro, and H. Pirsiavash, "Boosting self-supervised learning via knowledge transfer," *CVPR*, 2018.

[21] U. Büchler, B. Brattoli, and B. Ommer, "Improving spatiotemporal self-supervision by deep reinforcement learning," in *ECCV*, pp. 770–786, 2018.

[22] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *ICLR*, 2018.

[23] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised Learning of Video Representations using LSTMs," in *ICML*, 2015.

[24] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *ICCV*, 2015.

[25] H.-Y. Lee, J.-B. Huang, M. Singh, and M.-H. Yang, "Unsupervised representation learning by sorting sequences," in *ICCV*, pp. 667–676, IEEE, 2017.

[26] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and learn: unsupervised learning using temporal order verification," in *ECCV*, pp. 527–544, Springer, 2016.

[27] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *ICCV*, pp. 1422–1430, 2015.

[28] R. Zhang, P. Isola, and A. A. Efros, "Split-brain autoencoders: Unsupervised learning by cross-channel prediction," in *CVPR*, 2017.

[29] D. Li, W.-C. Hung, J.-B. Huang, S. Wang, N. Ahuja, and M.-H. Yang, "Unsupervised visual representation learning by graph-based consistent constraints," in *ECCV*, 2016.

[30] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *ECCV*, 2018.

[31] E. Hoffer, I. Hubara, and N. Ailon, "Deep unsupervised learning through spatial contrasting," *arXiv preprint arXiv:1610.00243*, 2016.

[32] P. Bojanowski and A. Joulin, "Unsupervised learning by predicting noise," *ICML*, 2017.

[33] Y. Li, M. Paluri, J. M. Rehg, and P. Dollár, "Unsupervised learning of edges," *CVPR*, pp. 1619–1627, 2016.

[34] M. Noroozi and P. Favaro, "Unsupervised learning of visual representions by solving jigsaw puzzles," in *ECCV*, 2016.

[35] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *ECCV*, pp. 649–666, Springer, 2016.

[36] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *CVPR*, pp. 2536–2544, 2016.

[37] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: Theory and practice," *IJCV*, pp. 222–245, 2013.

[38] A. Faktor and M. Irani, "Video segmentation by non-local consensus voting.," in *BMVC*, 2014.

[39] O. Stretcu and M. Leordeanu, "Multiple frames matching for object discovery in video.," in *BMVC*, 2015.

[40] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *ICLR*, pp. 1–14, 2015.

[41] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, pp. 1–9, 2015.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, pp. 770–778, 2016.

[43] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," in *CVPR*, p. 3, 2017.

[44] G. Cheng, J. Han, P. Zhou, and D. Xu, "Learning rotation-invariant and fisher discriminative convolutional neural networks for object detection," *TIP*, pp. 265–278, 2018.

[45] M. Jaderberg, K. Simonyan, A. Zisserman, *et al.*, "Spatial transformer networks," in *NIPS*, pp. 2017–2025, 2015.

[46] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *ICLR*, 2017.

[47] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *NIPS*, pp. 568–576, 2014.

[48] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *CVPR*, 2015.

[49] C. Feichtenhofer, A. Pinz, and R. Wildes, "Spatiotemporal residual networks for video action recognition," in *NIPS*, pp. 3468–3476, 2016.

[50] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Spatiotemporal multiplier networks for video action recognition," in *CVPR*, pp. 7445–7454, 2017.

[51] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *CVPR*, pp. 1933–1941, 2016.

[52] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *CVPR*, pp. 4305–4314, 2015.

[53] L. Wang, Y. Xiong, Z. Wang, and Y. Qiao, "Towards good practices for very deep two-stream convnets," *arXiv preprint arXiv:1507.02159*, 2015.

[54] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, and S. Gould, "Dynamic image networks for action recognition," in *CVPR*, 2016.

[55] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: towards good practices for deep action recognition," in *ECCV*, 2016.

[56] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *TPAMI*, pp. 221–231, 2013.

[57] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," *CRCV-TR*, 2012.

[58] X. Peng, L. Wang, X. Wang, and Y. Qiao, "Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice," *CVIU*, pp. 109–125, 2016.

[59] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Bags of spacetime energies for dynamic scene recognition," in *CVPR*, pp. 2681–2688, 2014.

[60] X. Ren and M. Philipose, "Egocentric recognition of handled objects: Benchmark and analysis," in *CVPRW*, pp. 1–8, IEEE, 2009.

[61] G. Varol, I. Laptev, and C. Schmid, "Long-term Temporal Convolutions for Action Recognition," *TPAMI*, 2017.

[62] L. Jing, X. Yang, and Y. Tian, "Video you only look once: Overall temporal convolutions for action recognition," *JVCIR*, pp. 58–65, 2018.

[63] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *CVPR*, pp. 4724–4733, IEEE, 2017.

[64] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet," in *CVPR*, pp. 18–22, 2018.

[65] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3d residual networks," in *ICCV*, 2017.

[66] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, pp. 157–166, 1994.

[67] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, pp. 1735–1780, 1997.

[68] J. Y. Ng, M. J. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *CVPR*, 2015.

[69] Z. Li, K. Gavrilyuk, E. Gavves, M. Jain, and C. G. Snoek, "Videolstm convolves, attends and flows for action recognition," *CVIU*, pp. 41–50, 2018.

[70] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, "Sequence to sequence – video to text," in *ICCV*, 2015.

[71] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, and J. Kautz, "Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network," in *CVPR*, pp. 4207–4215, 2016.

[72] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," tech. rep., Citeseer, 2009.

[73] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, pp. 248–255, IEEE, 2009.

[74] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, pp. 2278–2324, 1998.

[75] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *IJCV*, pp. 303–338, 2010.

[76] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *NIPS*, pp. 487–495, 2014.

[77] B. Zhou, A. Khosla, A. Lapedriza, A. Torralba, and A. Oliva, "Places: An image database for deep scene understanding," *arXiv preprint arXiv:1610.02055*, 2016.

[78] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *AISTATS*, pp. 215–223, 2011.

[79] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," in *CVPR*, pp. 190–198, IEEE, 2017.

[80] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPSW*, p. 5, 2011.

[81] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *ICASSP*, pp. 776–780, IEEE, 2017.

[82] H. Kuehne, H. Jhuang, R. Stiefelhagen, and T. Serre, "Hmdb51: A large video database for human motion recognition," in *HPCSE*, pp. 571–582, Springer, 2013.

[83] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, *et al.*, "The kinetics human action video dataset," *arXiv preprint arXiv:1705.06950*, 2017.

[84] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *CVPR*, pp. 3354–3361, IEEE, 2012.

[85] M. Monfort, B. Zhou, S. A. Bargal, T. Yan, A. Andonian, K. Ramakrishnan, L. Brown, Q. Fan, D. Gutfruend, C. Vondrick, *et al.*, "Moments in time dataset: one million videos for event understanding," *TPAMI*, pp. 502–508, 2019.

[86] J. McCormac, A. Handa, S. Leutenegger, and A. J. Davison, "Scenenet rgb-d: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation," in *ICCV*, 2017.

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

[88] A. Joulin, L. van der Maaten, A. Jabri, and N. Vasilache, "Learning visual features from large weakly supervised data," in *ECCV*, pp. 67–84, 2016.

[89] K. J. Piczak, "Esc: Dataset for environmental sound classification," in *Proceedings of the 23rd ACM international conference on Multimedia*, pp. 1015–1018, 2015.

[90] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, "Detection and classification of acoustic scenes and events," *IEEE Transactions on Multimedia*, pp. 1733–1746, 2015.

[91] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *CVPR*, pp. 1912–1920, 2015.

[92] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[93] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas, "A scalable active framework for region annotation in 3d shape collections," *TOG*, pp. 1–12, 2016.