technologies

MDPI

Review

A Survey on Contrastive Self-Supervised Learning

Ashish Jaiswal *, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee and Fillia Makedon

Department of Computer Science and Engineering, The University of Texas at Arlington, Arlington, TX 76019, USA; ashwin.rameshbabu@mavs.uta.edu (A.R.B.); mohammad.zakizadehgharie@mavs.uta.edu (M.Z.Z.); debapriya.banerjee2@mavs.uta.edu (D.B.); makedon@uta.edu (F.M.)

* Correspondence: ashish.jaiswal@mavs.uta.edu

Abstract: Self-supervised learning has gained popularity because of its ability to avoid the cost of annotating large-scale datasets. It is capable of adopting self-defined pseudolabels as supervision and use the learned representations for several downstream tasks. Specifically, contrastive learning has recently become a dominant component in self-supervised learning for computer vision, natural language processing (NLP), and other domains. It aims at embedding augmented versions of the same sample close to each other while trying to push away embeddings from different samples. This paper provides an extensive review of self-supervised methods that follow the contrastive approach. The work explains commonly used pretext tasks in a contrastive learning setup, followed by different architectures that have been proposed so far. Next, we present a performance comparison of different methods for multiple downstream tasks such as image classification, object detection, and action recognition. Finally, we conclude with the limitations of the current methods and the need for further techniques and future directions to make meaningful progress.

Keywords: contrastive learning; self-supervised learning; discriminative learning; image/video classification; object detection; unsupervised learning; transfer learning

Citation: Jaiswal , A.; Ramesh Babu, A.; Zaki Zadeh, M.; Banerjee, D.; Makedon, F. A Survey on Contrastive Self-Supervised Learning. *Technologies* **2021**, *9*, 2. https://dx.doi.org/10.3390/technologies9010002

Received: 31 October 2020
Accepted: 23 December 2020
Published: 28 December 2020

1. Introduction

The advancements in deep learning have elevated the field to become a core component in most intelligent systems . The ability to learn rich patterns from the abundance of data available today has made the use of deep neural networks (DNNs) a compelling approach in the majority of computer vision (CV) tasks such as image classification, object detection, image segmentation, and activity recognition, as well as natural language processing (NLP) tasks such as sentence classification, language models, machine translation, etc. However, the supervised approach to learning features from labeled data has almost reached its saturation due to intense labor required in manually annotating millions of data samples. This is because most of the modern computer vision systems (that are supervised) try to learn some form of image representations by finding a pattern between the data points and their respective annotations in large datasets. Works such as GRAD-CAM [1] have proposed techniques that provide visual explanations for decisions made by a model to make them more transparent and explainable.

Traditional supervised learning approaches rely heavily on the amount of annotated training data available. Even though there is a plethora of data available, the lack of annotations has pushed researchers to find alternative approaches that can leverage them. This is where self-supervised methods play a vital role in fueling the progress of deep learning without the need for expensive annotations and learn feature representations where data provide supervision.

Supervised learning not only depends on expensive annotations, but also suffers from issues such as generalization error, spurious correlations, and adversarial attacks [2]. Recently, self-supervised learning methods have integrated both generative and contrastive approaches that have been able to utilize unlabeled data to learn the underlying representations. A popular approach has been to propose various pretext tasks that help in learning

features using pseudolabels. Tasks such as image-inpainting, colorizing grayscale images, jigsaw puzzles, super-resolution, video frame prediction, audio-visual correspondence, etc. have proven to be effective for learning good representations.

Generative models gained their popularity after the introduction of Generative Adversarial Networks (GANs) [3] in 2014. The work later became the foundation for many successful architectures such as CycleGAN [4], StyleGAN [5], PixelRNN [6], Text2Image [7], DiscoGAN [8], etc. These methods inspired more researchers to switch to training deep learning models with unlabeled data in an self-supervised setup. Despite their success, researchers started realizing some of the complications in GAN-based approaches. They are harder to train because of two main reasons: (a) non-convergence—the model parameters oscillate a lot and rarely converge, and (b) the discriminator gets too successful that the generator network fails to create real-like fakes due to which the learning cannot be continued. Furthermore, proper synchronization is required between the generator and the discriminator that prevents the discriminator converging and the generator diverging.

Unlike generative models, contrastive learning (CL) is a discriminative approach that aims at grouping similar samples closer and diverse samples far from each other as shown in Figure 1. To achieve this, a similarity metric is used to measure how close two embeddings are. Especially, for computer vision tasks, a contrastive loss is evaluated based on the feature representations of the images extracted from an encoder network. For instance, one sample from the training dataset is taken and a transformed version of the sample is retrieved by applying appropriate data augmentation techniques. During training, referring to Figure 2, the augmented version of the original sample is considered as a positive sample, and the rest of the samples in the batch/dataset (depends on the method being used) are considered negative samples. Next, the model is trained in a way that it learns to differentiate positive samples from the negative ones. The differentiation is achieved with the help of some pretext task (explained in Section 2). In doing so, the model learns quality representations of the samples and is used later for transferring knowledge to downstream tasks. This idea is advocated by an interesting experiment conducted by Epstein [9] in 2016, where he asked his students to draw a dollar bill with and without looking at the bill. The results from the experiment show that the brain does not require complete information of a visual piece to differentiate one object from the other. Instead, only a rough representation of an image is enough to do so.

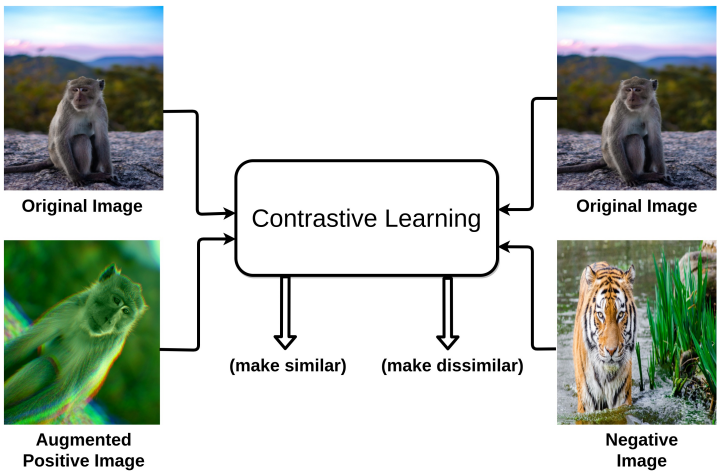


Figure 1. Basic intuition behind contrastive learning paradigm: push original and augmented images closer and push original and negative images away.

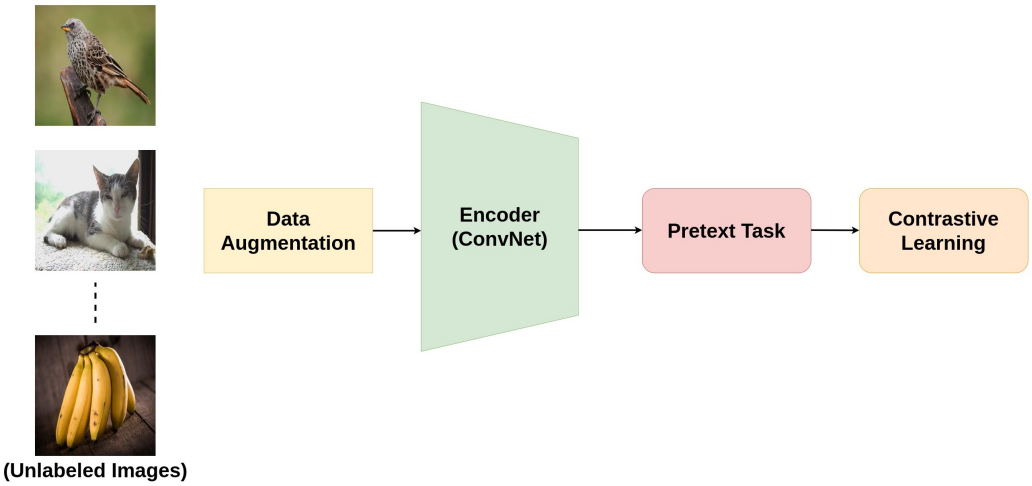


Figure 2. Contrastive learning pipeline for self-supervised training.

Most of the earlier works in this area combined some form of instance-level classification approach [10–12] with contrastive learning and were successful to some extent. However, recent methods such as SwAV [13], MoCo [14], and SimCLR [15] with modified approaches have produced results comparable to the state-of-the-art supervised method on ImageNet [16] dataset as shown in Figure 3. Similarly, PIRL [17], Selfie [18], and the work in [19] are some papers that reflect the effectiveness of the pretext tasks being used and how they boost the performance of their models.

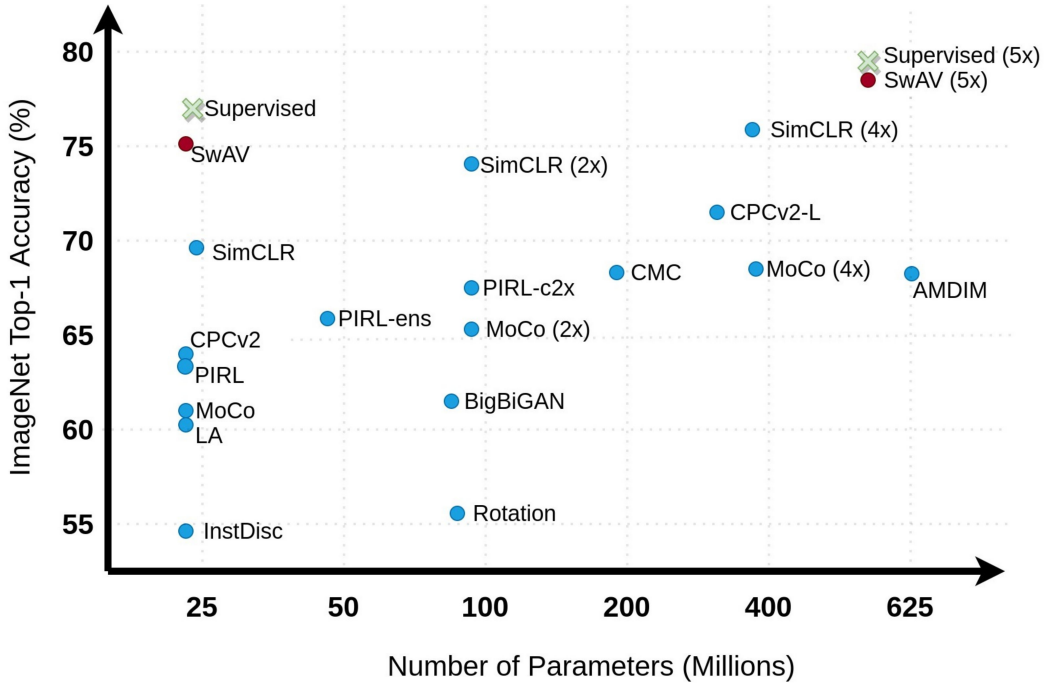


Figure 3. Top-1 classification accuracy of different contrastive learning methods against baseline supervised method on ImageNet.

2. Pretext Tasks

Pretext tasks are self-supervised tasks that act as an important strategy to learn representations of the data using pseudolabels. These pseudolabels are generated automatically based on the attributes found in the data. The learned model from the pretext task can be used for any downstream tasks such as classification, segmentation, detection, etc. in computer vision. Furthermore, pretext tasks can be designed for any kind of data such as image,

video, speech, signals, and so on. For a pretext task in contrastive learning, the original image acts as an anchor, its augmented(transformed) version acts as a positive sample, and the rest of the images in the batch or in the training data act as negative samples.

Most of the commonly used pretext tasks are divided into four main categories: color transformation, geometric transformation, context-based tasks, and cross-modal-based tasks. These pretext tasks have been used in various scenarios based on the problem intended to be solved.

2.1. Color Transformation

Color transformation involves basic adjustments of color levels in an image such as blurring, color distortions, converting to grayscale, etc. Figure 4 represents an example of color transformation applied on a sample image from the ImageNet dataset [15]. During this pretext task, the network learns to recognize similar images invariant to their colors.

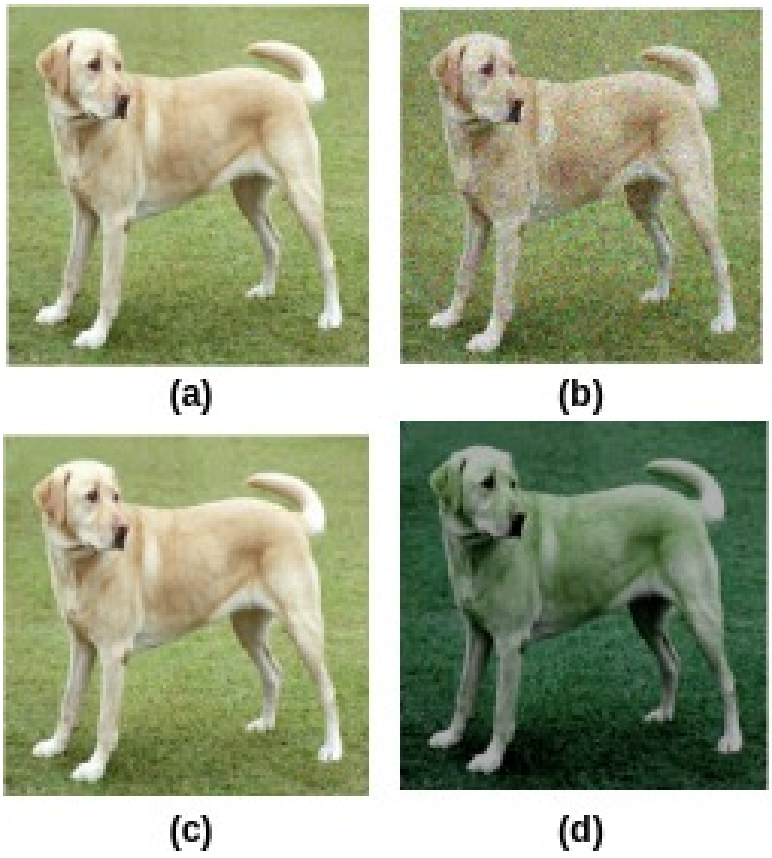


Figure 4. Color Transformation as pretext task [15]. (**a**) Original. (**b**) Gaussian noise. (**c**) Gaussian blur. (**d**) Color distortion (Jitter).

2.2. Geometric Transformation

A geometric transformation is a spatial transformation where the geometry of the image is modified without altering its actual pixel information. The transformations include scaling, random cropping, flipping (horizontally, vertically), etc. as represented in Figure 5, through which global-to-local view prediction is achieved. Here, the original image is considered as the global view and the transformed version is considered as the local view. Chen et al. [15] performed such transformations to learn features during pretext task.

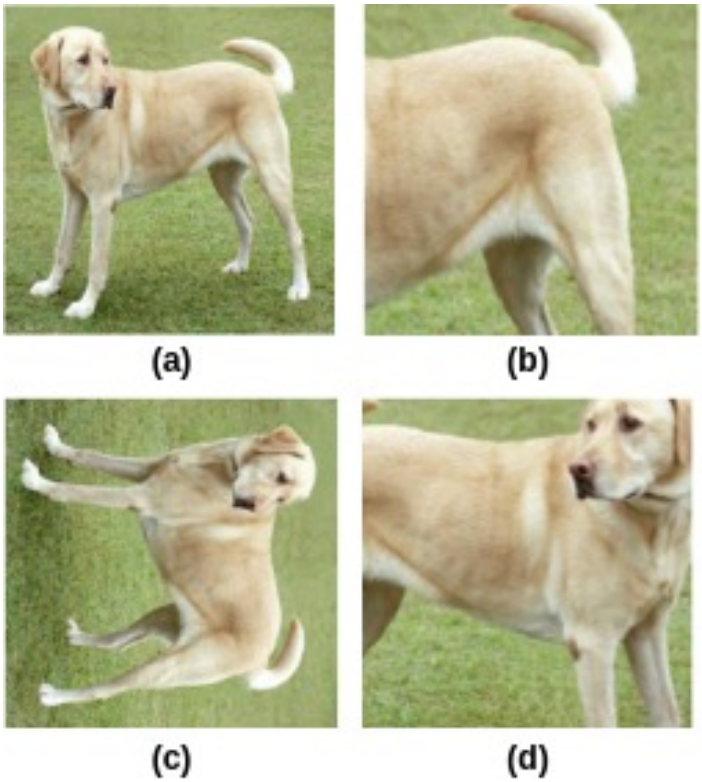


Figure 5. Geometric transformation as pretext task [15]. (**a**) Original. (**b**) Crop and resize. (**c**) Rotate (90°, 180°, 270°). (**d**) Crop, resize, and flip.

2.3. Context-Based

2.3.1. Jigsaw Puzzle

Traditionally, solving jigsaw puzzles has been a prominent task in learning features from an image in an unsupervised way. It involves identifying the correct position of the scrambled patches in an image by training an encoder (Figure 6). In terms of contrastive learning, the original image is the anchor, and an augmented image formed by scrambling the patches in the original image acts as a positive sample. The rest of the images in the dataset/batch are considered to be negative samples [17].

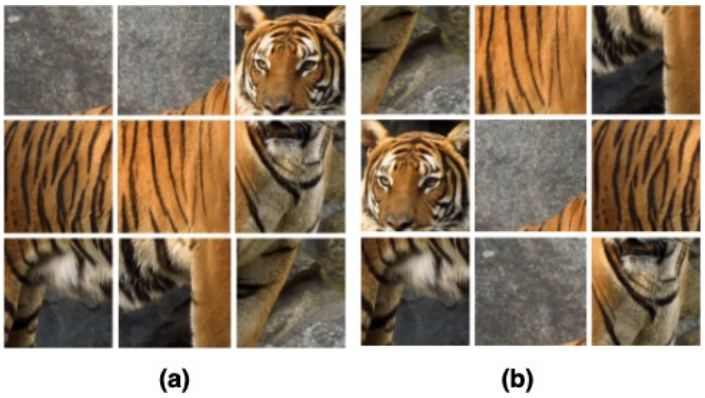


Figure 6. Solving jigsaw puzzle being used as a pretext task to learn representation. (**a**) Original Image; (**b**) reshuffled image. The original image is the anchor and the reshuffled image is the positive sample.

2.3.2. Frame Order Based

This approach applies to data that extends through time. An ideal application would be in the case of sensor data or a sequence of image frames (video). A video contains a

sequence of semantically related frames. This implies that frames that are nearby with respect to time are closely related and the ones that are far away are less likely to be related. Intuitively, the motive for using such an approach is solving a pretext task that allows the model to learn useful visual representations while trying to recover the temporal coherence of a video. Here, a video with shuffled order in the sequence of its image frames acts as a positive sample while all other videos in the batch/dataset would be negative samples.

Similarly, other possible approaches include randomly sampling two clips of the same length from a longer video or applying spatial augmentation for each video clip. The goal is to use a contrastive loss to train the model such that clips taken from the same video are arranged closer whereas clips from different videos are pushed away in the embedding space. In the work proposed by Qian et al. [20], the framework contrasts the similarity between two positive samples to those of negative samples. The positive pairs are two augmented clips from the same video. As a result, it separates all encoded videos into non-overlapping regions such that an augmentation used in the training perturbs an encoded video only within a small region in the representation space.

2.3.3. Future Prediction

One of the most common strategies for data that extends through time is to predict future or missing information. This is commonly used for sequential data such as sensory data, audio signals, videos, etc. The goal of a future prediction task is to predict high-level information of future time-step given a series of past ones. In [21,22], high-dimensional data are compressed into a compact lower-dimensional latent embedding space. Powerful autoregressive models are used to summarize the information in the latent space, and a context latent representation C_t is produced as represented in Figure 7. When predicting future information, the target (future) and context C_t are encoded into a compact distributed vector representation in a way that maximally preserves the mutual information of the original signals.

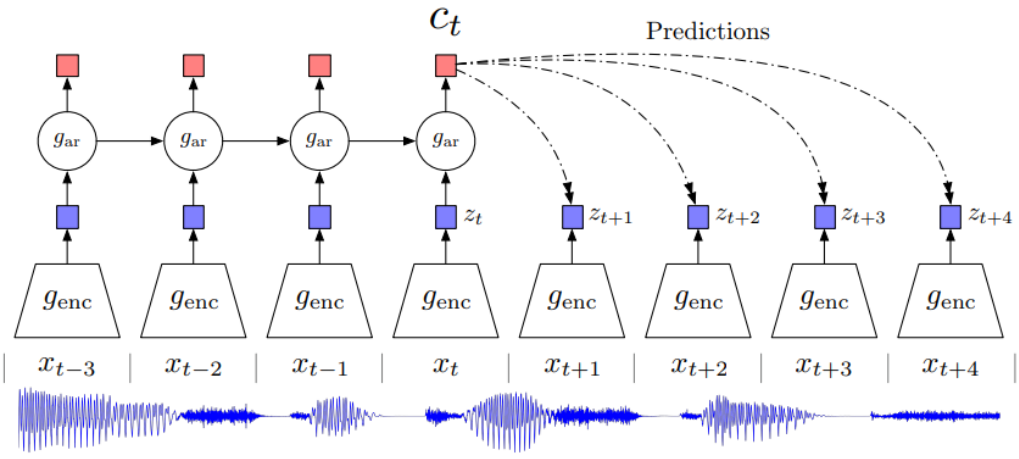


Figure 7. Contrastive Predictive Coding: Although the figure shows audio as input, similar setup can be used for videos, images, text etc. [21].

2.4. View Prediction (Cross-Modal-Based)

View prediction tasks are preferred for data that has multiple views of the same scene. Following this approach, in [23] the anchor and its positive images taken from simultaneous viewpoints are encouraged to be close in the embedding space while distant from negative images taken from a different time within the same sequence. The model learns by trying to simultaneously identify similar features between the frames from different angles and also trying to find the difference between frames that occur later in the sequence. Figure 8 represents their approach for view prediction. Similarly, recent work proposes an inter-intra

contrastive framework where inter-sampling is learned through multi-view of the same sample, and intra-sampling that learns the temporal relation is achieved through multiple approaches such as frame repetition and frame order shuffling that acts as the negative samples [24].

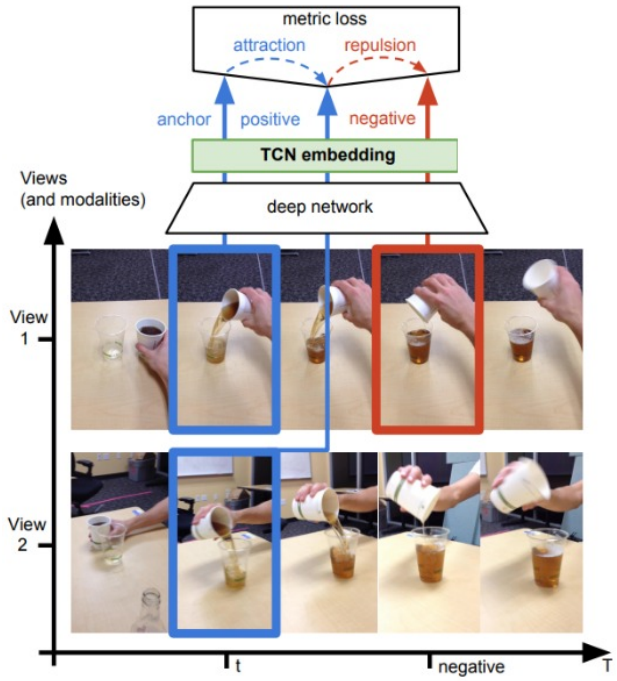


Figure 8. Learning representation from video frame sequence [23].

2.5. Identifying the Right Pre-Text Task

The choice of pretext task relies on the type of problem being solved. Although numerous methods have been proposed in contrastive learning, a separate track of research is still going on to identify the right pre-text task. Work has identified and proved that it is important to determine the right kind of pre-text task for a model to perform well with contrastive learning. The main aim of a pre-text task is to compel the model to be invariant to these transformations while remaining discriminative to other data points. However, the bias introduced through such augmentations could be a double-edged sword, as each augmentation encourages invariances to a transformation which can be beneficial in some cases and harmful in others. For instance, applying rotation may help with view-independent aerial image recognition but might significantly downgrade the performance while trying to solve downstream tasks such as detecting which way is up in a photograph for a display application [25]. Similarly, colorization-based pretext tasks might not work out in a fine-grain classification represented in Figure 9.

Figure 9. Most of the shapes of these two pairs of images are same. However, the low-level statistics are different (color and texture). Usage of right pre-text task here is necessary [26].

Similarly, the authors of [27] focus on the importance of using the right pretext task. The authors pointed out that in their scenario, except for rotation, other transformations such as scaling and changing aspect ratio may not be appropriate for the pretext task because they produce easily detectable visual artifacts. They also reveal that rotation does not work well when the image in a target dataset is constructed by color textures as in DTD dataset [28] as shown in Figure 10.

Figure 10. A sample from the DTD dataset [28]. An example of why rotation based pretext tasks may not work well.

2.6. Pre-Text Tasks in NLP

While self-supervised learning has been making significant progress in computer vision tasks for the past few years, it has been an active area of research in NLP for decades. Using a pretext task refers to generating labels such that supervised approaches can be applied to unsupervised problems to pretrain models. In NLP, text representations can be learned from large text corpora using any of the available pretext tasks that are discussed below.

2.6.1. Center and Neighbor Word Prediction

Back in 2013, Word2Vec [29] first introduced self-supervised methods to learn word representations in vector space. The continuous bag-of-words version of the model used "center word prediction" as the pretext task while the continuous skip-gram model implemented the "neighbor word prediction" task. In center word prediction, the input to the model is a sequence of words with a fixed window size and one word missing from the center of the sequence. The task of the model is to predict the missing word in the sequence. On the other hand, the input in skip-gram model is a single word where the model predicts its neighbor words. By performing these particular tasks, the model is able to learn word representations that can be further used to train models for downstream tasks.

2.6.2. Next and Neighbor Sentence Prediction

In "next sentence prediction", the model predicts whether two inputs sentences can be consecutive sentences or not. A positive sample in this case would be a sample that follows the original sentence while a negative sample is a sentence from a random document. BERT [30] used this method to drastically improve performance on downstream tasks that required an understanding of sentence relations such as question answering and language inference.

Similarly, given a sentence, a model has to predict its previous and the next sentence in "neighbor sentence prediction task". This approach was inherited by Skip-Thought Vectors [31] paper. It is similar to the skip-gram method but rather applied to sentences in place of words.

2.6.3. Autoregressive Language Modeling

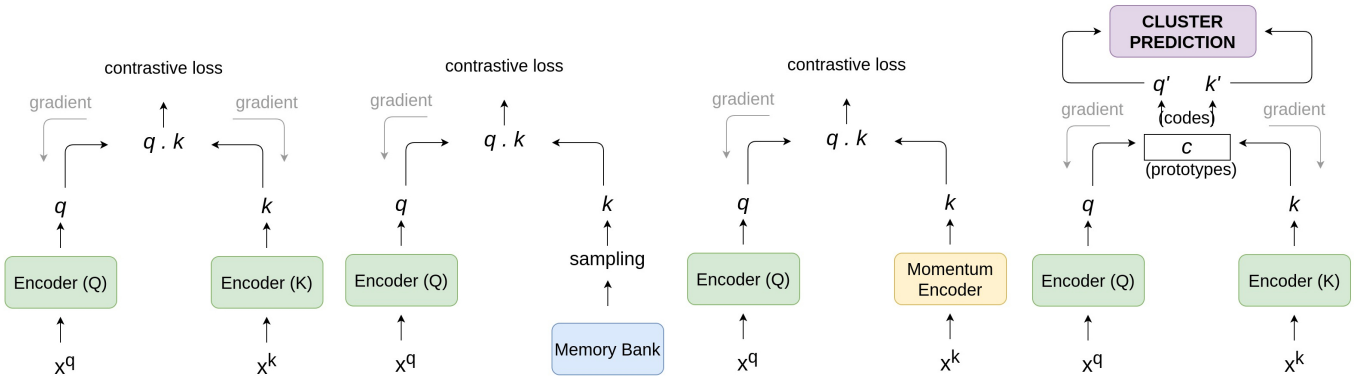
This task involves predicting the next word given previous words, or vice versa. A sequence of words from a text document is provided and the model tries to predict the next word that follows the sequence. This technique has been used by several n-gram models and neural networks such as GPT [32] and its recent versions.

2.6.4. Sentence Permutation

A recent paper known as BART [33] used a pretext task where a continuous span of text from the corpus is taken and broken into multiple sentences. The position of the sentences are randomly reshuffled and the task of the model is to predict the original order of the sentences.

3. Architectures

Contrastive learning methods rely on the number of negative samples for generating good quality representations. Accessing negative samples can be seen as a dictionary-lookup task where the dictionary is sometimes the whole training set and the rest of the times some subset of the dataset. An interesting way to categorize these methods would be based on the technique used to collect negative samples against a positive data point during training. Based on the approach taken, we categorized the methods into four major architectures as shown in Figure 11. Each architecture is explained separately along with examples of successful methods that follow similar principles.



(**a**) End-to-End (**b**) Memory Bank (**c**) Momentum Encoder (**d**) Clustering

Figure 11. Different architecture pipelines for Contrastive Learning: (**a**) End-to-End training of two encoders where one generates representation for positive samples and the other for negative samples. (**b**) Using a memory bank to store and retrieve encodings of negative samples. (**c**) Using a momentum encoder which acts as a dynamic dictionary lookup for encodings of negative samples during training. (**d**) Implementing a clustering mechanism by using swapped prediction of the obtained representations from both the encoders using end-to-end architecture.

3.1. End-to-End Learning

End-to-end learning is a complex learning system that uses gradient-based learning and is designed in such a way that all modules are differentiable [34]. This architecture prefers large batch sizes to accumulate a greater number of negative samples. Except for the original image and its augmented version, the rest of the images in the batch are considered negative. The pipeline employs two encoders: a Query encoder (Q) and a Key encoder (K) as shown in Figure 11. The two encoders can be different and are updated end-to-end by backpropagation during training. The main idea behind training these

encoders separately is to generate distinct representations of the same sample. Using a contrastive loss, it converges to make positive samples closer and negative samples far from the original sample. Here, the query encoder Q is trained on the original samples and the key encoder K is trained on their augmented versions (positive samples) along with the negative samples in the batch. The features q and k generated from these encoders are used to calculate the similarity between the respective inputs using a similarity metric (discussed later in Section 5). Most of the time, the similarity metric used is cosine similarity, which is simply the inner product of two vectors normalized to have length 1 as defined in Equation (2).

Recently, a successful end-to-end model was proposed in SimCLR [15] where they used a batch size of 4096 for 100 epochs. It has been verified that end-to-end architectures are simple in complexity, but perform better with large batch sizes and a higher number of epochs as represented in Figure 12. Another popular work that follows end-to-end architecture was proposed by Oord et al. [21] where they learn feature representations of high-dimensional time series data by predicting the future in latent space by using powerful autoregressive models along with a contrastive loss. This approach makes the model tractable by using negative sampling. Moreover, several other work follow this approach [35–39].

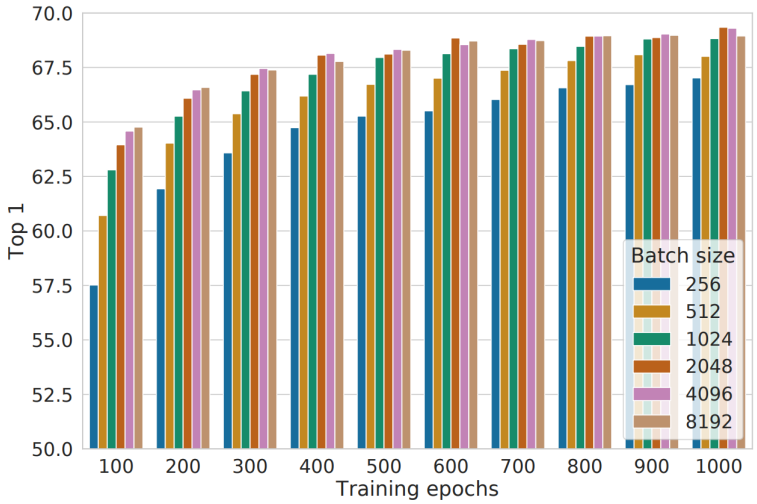


Figure 12. Linear evaluation models (ResNet-50) trained with different batch size and epochs. Each bar represents a single run from scratch [15].

The number of negative samples available in this approach is coupled with the batch size as it accumulates negative samples from the current batch. As the batch size is limited by the GPU memory size, the scalability factor with these methods remains an issue. Furthermore, for larger batch sizes, the methods suffer from a large mini-batch optimization problem and require effective optimization strategies as pointed out in [40].

3.2. Using a Memory Bank

With potential issues of having larger batch sizes that could inversely impact the optimization during training, a possible solution is to maintain a separate dictionary called Memory ank.

The aim of maintaining a memory bank is to accumulate a large number of feature representations of samples that are used as negative samples during training. For this purpose, a dictionary is created that stores and updates the embeddings of samples with the most recent ones at regular intervals. The memory bank (M) contains a feature representation m_I for each sample I in dataset D. The representation m_I is an exponential moving average of feature representations that were computed in prior epochs. It enables

replacing negative samples $m_{I'}$ by their memory bank representations without increasing the training batch size.

The representation of a sample in the memory bank gets updated when it is last seen, so the sampled keys are essentially about the encoders at multiple different steps all over the past epoch. PIRL [17] is one of the recent successful methods that learns good visual representations of images trained using a memory bank as shown in Figure 13. It requires the learner to construct representations of images that are covariant to any of the pretext tasks being used, though they focus mainly on the Jigsaw pretext task. Another popular work that uses a memory bank under contrastive setting was proposed by Wu et al. [12] where they implemented a nonparametric variant of softmax classifier that is more scalable for big data applications.

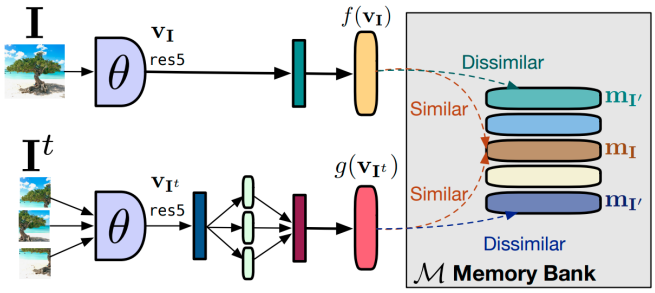


Figure 13. Usage of memory bank in PIRL: memory bank contains the moving average representations of all negative images to be used in contrastive learning [17].

However, maintaining a memory bank during training can be a complicated task. One of the potential drawbacks of this approach is that it can be computationally expensive to update the representations in the memory bank as the representations get outdated quickly in a few passes.

3.3. Using a Momentum Encoder for Contrastive Learning

To address the issues with a memory bank explained in the previous section, the memory bank gets replaced by a separate module called Momentum Encoder. The momentum encoder generates a dictionary as a queue of encoded keys with the current mini-batch enqueued and the oldest mini-batch dequeued. The dictionary keys are defined on-the-fly by a set of data samples in the batch during training. The momentum encoder shares the same parameters as the encoder Q as shown in Figure 11. It is not backpropagated after every pass, instead, it gets updated based on the parameters of the query encoder as represented by Equation (1) [14].

$$\theta_k \leftarrow m\theta_k + (1-m)\theta_q \tag{1}$$

In the equation, $m \in [0, 1)$ is the momentum coefficient. Only the parameters θ_q are updated by back-propagation. The momentum update makes θ_k evolve more smoothly than θ_q. As a result, though the keys in the queue are encoded by different encoders (in different mini-batches), the difference among these encoders can be made small.

The advantage of using this architecture over the first two is that it does not require training two separate models. Furthermore, there is no need to maintain a memory bank that is computationally and memory inefficient.

3.4. Clustering Feature Representations

All three architectures explained above focus on comparing samples using a similarity metric and try to keep similar items closer and dissimilar items far from each other allowing the model to learn better representations. On the contrary, this architecture follows an end-to-end approach with two encoders that share parameters, but instead of using

instance-based contrastive approach, they utilize a clustering algorithm to group similar features together.

One of the most recent works that employ clustering methods, SwAV [13], is represented in Figure 14. The diagram points out the differences between other instance-based contrastive learning architectures and the clustering-based methods. Here, the goal is not only to make a pair of samples close to each other but also, make sure that all other features that are similar to each other form clusters together. For example, in an embedded space of images, the features of cats should be closer to the features of dogs (as both are animals) but should be far from the features of houses (as both are distinct).

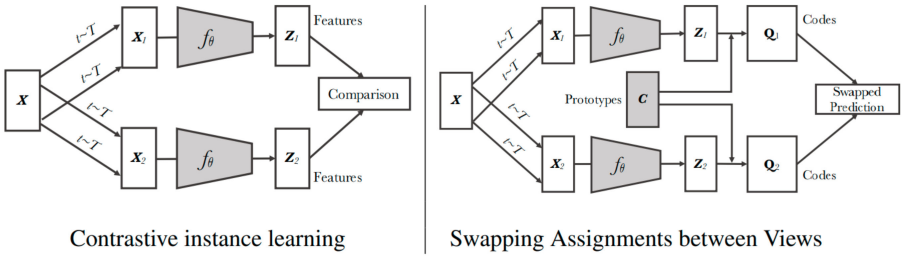


Figure 14. Conventional contrastive instance learning vs. contrastive clustering of feature representations in SwAV [13].

In instance-based learning, every sample is treated as a discrete class in the dataset. This makes it unreliable in conditions where it compares an input sample against other samples from the same class that the original sample belongs to. To explain it clearly, imagine we have an image of a cat in the training batch that is the current input to the model. During this pass, all other images in the batch are considered as negative. The issue arises when there are images of other cats in the negative samples. This condition forces the model to learn two images of cats as not similar during training despite both being from the same class. This problem is implicitly addressed by a clustering-based approach.

4. Encoders

Encoders play an integral role in any self-supervised learning pipeline as they are responsible for mapping the input samples to a latent space. Figure 15 reflects the role of an encoder in a self-supervised learning pipeline. Without effective feature representations, a classification model might have difficulty in learning to distinguish among different classes. Most of the works in contrastive learning utilize some variant of the ResNet [41] model. Among its variants, ResNet-50 has been the most widely used because of its balance between size and learning capability.

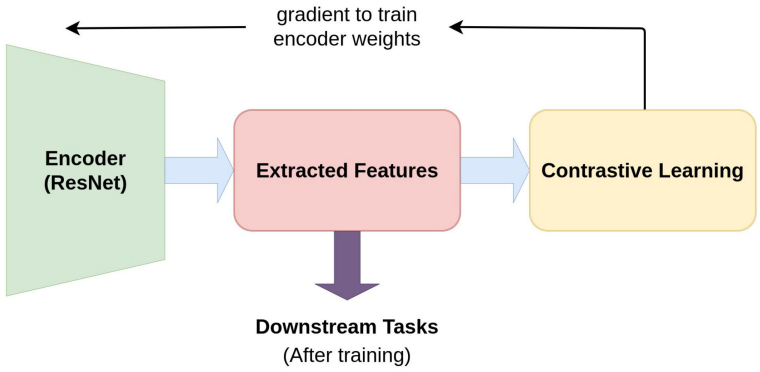


Figure 15. Training an encoder and transferring knowledge for downstream tasks.

In an encoder, the output from a specific layer is pooled to get a single-dimensional feature vector for every sample. Depending on the approach, they are either upsampled

or downsampled. For example, in the work proposed by Misra et al. [17], a ResNet-50 architecture is used where the output of the res5 (residual block) features are average-pooled to get a 2048-dimensional vector for the given sample (image in their case). They further apply a single linear projection to get a 128-dimensional feature vector. Furthermore, as part of their ablation test, they investigated features from various stages such as res2, res3, and res4 to evaluate the performance. As expected, features extracted from the later stages of the encoder proved to be a better representation of the input than the features extracted from the earlier stages.

Similarly, in the work proposed by Chen et al. [42], a traditional ResNet is used as an encoder where the features are extracted from the output of the average pooling layer. Further, a shallow MLP (1 hidden layer) maps representations to a latent space where a contrastive loss is applied. For training a model for action recognition, the most common approach to extract features from a sequence of image frames is to use a 3D-ResNet as encoder [22,24].

5. Training

To train an encoder, a pretext task is used that utilizes contrastive loss for backpropagation. The central idea in contrastive learning is to bring similar instances closer and push away dissimilar instances far from each other. One way to achieve this is to use a similarity metric that measures the closeness between the embeddings of two samples. In a contrastive setup, the most common similarity metric used is cosine similarity that acts as a basis for different contrastive loss functions. The cosine similarity of two variables (vectors) is the cosine of the angle between them and is defined as follows.

$$cos\_sim(A, B) = \frac{A.B}{\|A\|\|B\|} \tag{2}$$

Contrastive learning focuses on comparing the embeddings with a Noise Contrastive Estimation (NCE) [43] function that is defined as

$$L_{NCE} = -log\frac{exp(sim(q, k_+)/\tau)}{exp(sim(q, k_+)/\tau) + exp(sim(q, k\_)/\tau)} \tag{3}$$

where q is the original sample, k_+ represents a positive sample, and $k\_$ represents a negative sample. τ is a hyperparameter used in most of the recent methods and is called temperature coefficient. The sim() function can be any similarity function, but generally a cosine similarity as defined in Equation (2) is used. The initial idea behind NCE was to perform a nonlinear logistic regression that discriminates between observed data and some artificially generated noise.

If the number of negative samples is greater, a variant of NCE called InfoNCE is used as represented in Equation (4). The use of l2 normalization (i.e., cosine similarity) and the temperature coefficient effectively weighs different examples and can help the model learn from hard negatives.

$$L_{infoNCE} = -log\frac{exp(sim(q, k_+)/\tau)}{exp(sim(q, k_+)/\tau) + \sum_{i=0}^{K} exp(sim(q, k_i)/\tau)} \tag{4}$$

where k_i represents a negative sample.

Similar to other deep learning methods, contrastive learning employs a variety of optimization algorithms for training. The training process involves learning the parameters of encoder network by minimizing the loss function.

Stochastic Gradient Descent (SGD) has one of the most popular optimization algorithms used with contrastive learning methods [10,12,14,17]. It is an stochastic approximation of gradient descent optimization as it replaces the actual gradient (calculated from the entire data set) with an estimate calculated from a randomly selected subset of data. A crucial hyperparameter for the SGD algorithm is the learning rate, which in practice

should gradually be decreased over time. An improved version of SGD (with momentum) is used in most deep learning approaches.

Another popular optimization method known as adaptive learning rate optimization algorithm (Adam) [44] has been used in a few methods [21,45,46]. In Adam, momentum is incorporated directly as an estimate of the first-order moment. Furthermore, Adam includes bias corrections to the estimates of both the first-order moments and the second-order moments to account for their initialization at the origin.

As some of the end-to-end methods [13,15,47] use a very large batch size, training with standard SGD-based optimizers with a linear learning rate scaling becomes unstable. In order to stabilize the training, a Layer-wise Adaptive Rate Scaling (LARS) [48] optimizer along with cosine learning rate [49] was introduced. There are two main differences between LARS and other adaptive algorithms such as Adam: First, LARS uses a different learning rate for every layer, leading to better stability. Second, the magnitude of the update is based on the weight norm for better control of training speed. Furthermore, employing cosine learning rate involves periodically warm restarts of SGD, where in each restart, the learning rate is initialized to some value and is scheduled to decrease over time.

6. Downstream Tasks

Generally, computer vision pipelines that employ self-supervised learning involve performing two tasks: a pretext task and a downstream task. Downstream tasks are application-specific tasks that utilize the knowledge that was learned during the pretext task. Figure 16 represents the overview of how knowledge is transferred to a downstream task. The learned parameters serve as a pretrained model and are transferred to other downstream computer vision tasks by fine-tuning. The performance of transfer learning on these high-level vision tasks demonstrates the generalization ability of the learned features. Some of the common downstream tasks in computer vision are classification, detection, segmentation, future prediction, etc as represented in Figure 17. Using one or more of these tasks, the performance of models trained in an unsupervised or self-supervised way can be tested and evaluated.

To evaluate the effectiveness of the features learned with a self-supervised approach for downstream tasks, methods such as kernel visualization, feature map visualization, and nearest-neighbor-based approaches are commonly used. These methods also help in analyzing the contribution of a pretext task in efficiently training a model.

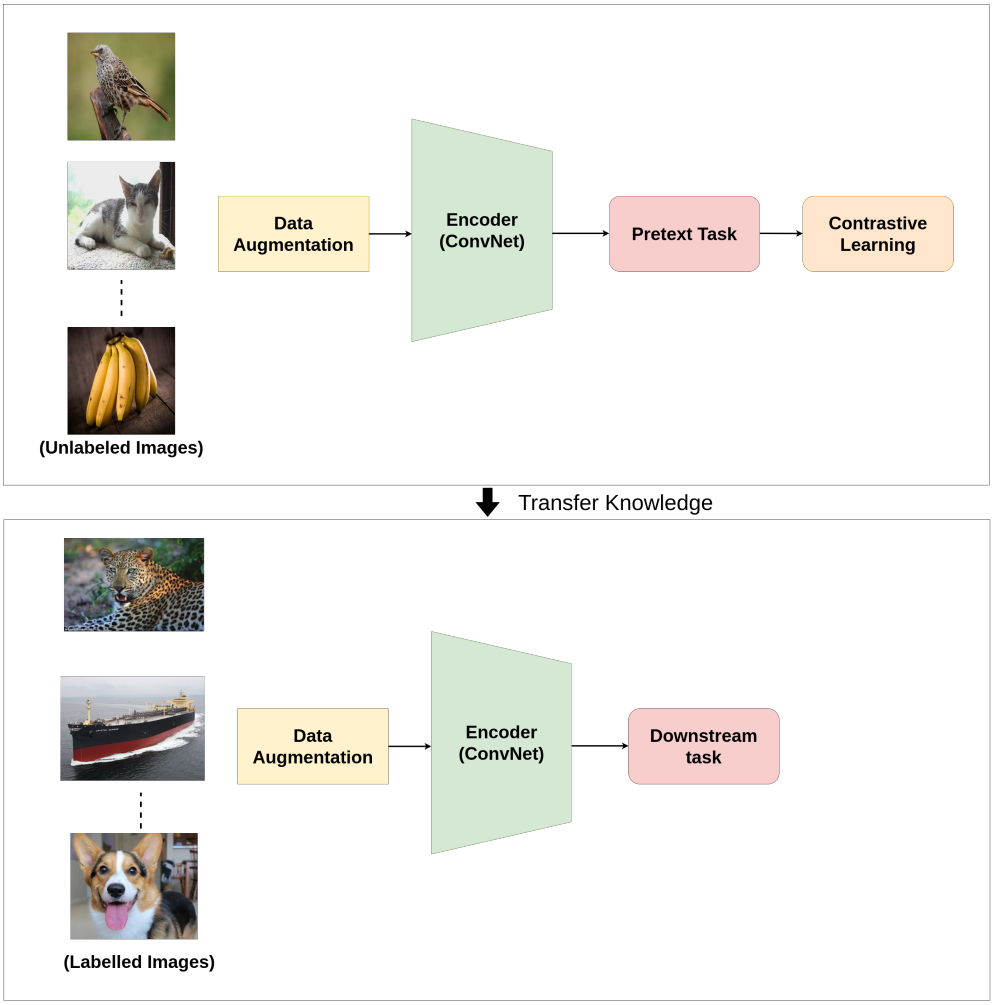


Figure 16. An overview of downstream task for images.

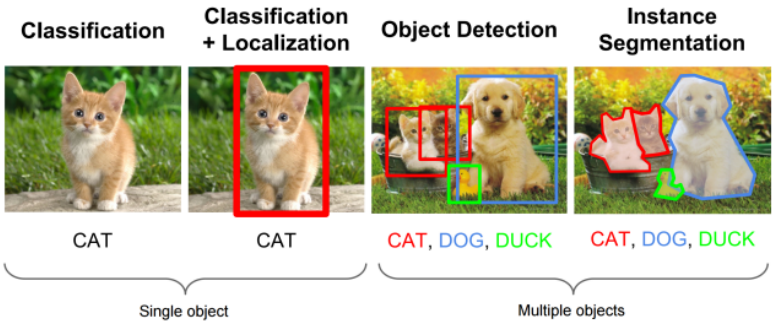


Figure 17. Image classification, localization, detection, and segmentation as downstream tasks in computer vision.

6.1. Visualizing the Kernels and Feature Maps

Here, the kernels of the first convolutional layer from encoders trained with both self-supervised (contrastive) and supervised approaches are compared. This helps to estimate the effectiveness of the self-supervised approach [50]. Similarly, attention maps generated from different layers of the encoders can be used to evaluate if an approach works or not. Gidaris et al. [51] assessed the effectiveness based on the activated regions observed in the input as shown in Figure 18.

Figure 18. Attention maps generated by a trained AlexNet. Left set of images represent a supervised approach and right set of images represent a self-supervised approach. The images represent the attention maps applied on features from different convolutional layers, (**a**) Conv1 27 × 27, (**b**) Conv3 13 × 13, and (**c**) Conv5 6 × 6.

6.2. Nearest-Neighbor Retrieval

In general, the samples that belong to the same class are expected to be closer to each other in the latent space. With the nearest-neighbor approach, for a given input sample, top-K retrieval of the samples from the dataset can be used to analyze whether a self-supervised approach performs as expected or not.

7. Benchmarks

Recently, several self-supervised learning methods for computer vision tasks have been proposed that challenge the existing state-of-the-art supervised models. In this section, we collect and compare the performances of these methods based on the downstream tasks they were evaluated on. For image classification, two popular datasets, ImageNet [16] and Places [52], have been used by most of the methods. Similarly, for object detection, the Pascal VOC dataset has often been referred to for evaluation where these methods have outperformed the best supervised models. For action recognition and video classification, datasets such as UCF-101 [53], HMDB-51 [54], and Kinetics [55] have been used.

Table 1 highlights the performance of several methods on ImageNet and reflects how these methods have evolved and performed better with time. At the moment, as seen in Figure 3, SwAV [13] produces comparable accuracy to the state-of-the-art supervised model in learning image representations from ImageNet. Similarly, for image classification task on Places [52] dataset, SwAV [13] and AMDIM [37] have outperformed the top supervised models with higher top-1 accuracies as shown in Table 2. The methods shown in the table were first pretrained on ImageNet and later inferred on Places dataset using a linear classifier. The results advocate that representations learned by contrastive learning methods performed better than the supervised approach when tested on a different dataset.

Table 1. Performance on ImageNet Dataset: Top-1 and Top-5 accuracies of different contrastive learning methods on ImageNet using self-supervised approach where models are used as frozen encoders for a linear classifier. The second half of the table (rightmost two columns) shows the performance (top-5 accuracy) of these methods after fine-tuning on 1% and 10% of labels from ImageNet.

Method	Architecture	ImageNet (Self-Supervised)		Semi-Supervised (Top-5)	
		Top-1	Top-5	1% Labels	10% Labels
Supervised	ResNet50	76.5	-	56.4	80.4
CPC [38]	ResNet v2 101	48.7	73.6	-	-
InstDisc [12]	ResNet50	56.5	-	39.2	77.4
LA [56]	ResNet50	60.2	-	-	-
MoCo [14]	ResNet50	60.6	-	-	-
BigBiGAN [57]	ResNet50 (4×)	61.3	81.9	55.2	78.8
PCL [58]	ResNet50	61.5	-	75.3	85.6
SeLa [59]	ResNet50	61.5	84.0	-	-
PIRL [17]	ResNet50	63.6	-	57.2	83.8
CPCv2 [38]	ResNet50	63.8	85.3	77.9	**91.2**
PCLv2 [58]	ResNet50	67.6	-	-	-
SimCLR [15]	ResNet50	69.3	89.0	75.5	87.8
MoCov2 [47]	ResNet50	71.1	-	-	-
InfoMin Aug [19]	ResNet50	73.0	91.1	-	-
SwAV [13]	ResNet50	**75.3**	-	**78.5**	89.9

Table 2. Image classification accuracy on Places dataset pretrained on ImageNet.

Method	Architecture	Parameters	Top-1 Accuracy
Supervised	ResNet50	25.6 M	53.2
BiGAN [60]	AlexNet	61 M	31.0
Context [61]	AlexNet	61 M	32.7
SplitBrain [62]	AlexNet	61 M	34.1
AET [63]	AlexNet	61 M	37.1
DeepCluster [50]	AlexNet	61 M	37.5
Color [64]	ResNet50	25.6 M	37.5
Jigsaw [64]	ResNet50	25.6 M	41.2
Rotation [51]	ResNet50	25.6 M	41.4
NPID [12]	ResNet50	25.6 M	45.5
PIRL [17]	ResNet50	25.6 M	49.8
LA [56]	ResNet50	25.6 M	50.1
AMDIM [37]	-	670 M	55.1
SwAV [13]	ResNet50	25.6 M	56.7

These methods have not only excelled in image classification, but have also performed well on other tasks like object detection and action recognition. As shown in Table 3, SwAV [13] outperforms the state-of-the-art supervised model in both linear classification and object detection in the Pascal VOC7 dataset. For linear classification, the models shown in the table were pretrained on VOC7 and features were taken for training a linear classification model. Similarly, for object detection, models were fine-tuned on VOC7+12 using Faster-RCNN. For video classification tasks, contrastive learning methods have shown promising results in datasets like UCF101, HMDB51, and Kinetics, as reflected in Table 4.

Table 3. (1) Linear classification top-1 accuracy on top of frozen features and (2) object detection with fine-tuned features on VOC7+12 using Faster-CNN.

Method	Architecture	Parameters	(1) Classification	(2) Detection
Supervised	AlexNet	61 M	79.9	56.8
Supervised	ResNet50	25.6 M	87.5	81.3
Inpaint [65]	AlexNet	61 M	56.5	44.5
Color [66]	AlexNet	61 M	65.6	46.9
BiGAN [60]	AlexNet	61 M	60.1	46.9
NAT [10]	AlexNet	61 M	65.3	49.4
Context [61]	AlexNet	61 M	65.3	51.1
DeepCluster [50]	AlexNet	61 M	72.0	55.4
Color [66]	ResNet50	25.6 M	55.6	–
Rotation [51]	ResNet50	25.6 M	63.9	72.5
Jigsaw [64]	ResNet50	25.6 M	64.5	75.1
LA [56]	ResNet50	25.6 M	69.1	–
NPID [12]	ResNet50	25.6 M	76.6	79.1
PIRL [17]	ResNet50	25.6 M	81.1	80.7
MoCo [14]	ResNet50	25.6 M	–	81.4
SwAV [13]	ResNet50	25.6 M	88.9	82.6

Table 4. Accuracy on Video Classification. All the proposed methods were pretrained with their proposed contrastive based approaches and a linear model was used for validation. R3D in model represents 3D-ResNet. † represents that the model has been trained on another dataset and further fine-tuned with the specific dataset. K represents Kinetics dataset.

Method	Model	UCF-101	HMDB-51	K (top 1)	K (top 5)
C3D (Supervised)	-	82.3 †	-	-	-
3DResNet-18 (Supervised)	R3D	84.4 †	56.4 †	-	-
P3D (Supervised)	-	84.4 †	-	-	-
ImageNet-inflated [67]	R3D	60.3	30.7	-	-
jigsaw [26]	-	51.5	22.5	-	-
OPN [68]	-	56.3	22.1	-	-
Cross Learn (with Optical Flow) [69]	-	58.7	27.2	-	-
O3N [70]	-	60.3	32.5	-	-
Shuffle and Learn [71]	-	50.2	18.1	-	-
IIC (Shuffle + res) [24]	R3D	74.4	38.3	-	-
inflated SIMCLR [20]	R3D-50	-	-	48.0	71.5
CVRL [20]	R3D-50	-	-	**64.1**	**85.8**
TCP [22]	R3D	77.9 (3 splits)	45.3	-	-
SeCo inter + intra + order [72]	R3D	88.26 †	55.5 †	61.91	-
DTG-Net [73]	R3D-18	**85.6**	**49.9**	-	-
CMC(3 views) [74]	R3D	59.1	26.7	-	-

8. Contrastive Learning in NLP

Contrastive learning was first introduced by Mikolov et al. [75] for natural language processing in 2013. The authors proposed a contrastive learning-based framework by using co-occurring words as semantically similar points and negative sampling [76] for learning word embeddings. The negative sampling algorithm differentiates a word from the noise distribution using logistic regression and helps to simplify the training method. This framework results in huge improvement in the quality of representations of learned words and phrases in a computationally efficient way. Arora et al. [77] proposed a theoretical framework for contrastive learning that learns useful feature representations from unlabeled data and introduced latent classes to formalize the notion of semantic similarity and performs well on classification tasks using the learned representations. Its performance is comparable to the state-of-the-art supervised approach on the Wiki-3029 dataset. Another recent model, CONtrastive Position and Ordering with Negatives Objective(CONPONO) [78], discourses coherence and encodes fine-grained sentence ordering in text and outperforms BERT-Large model despite having the same number of parameters as BERT-Base.

Contrastive Learning has started gaining popularity on several NLP tasks in recent years. It has shown significant improvement on NLP downstream tasks such as cross-lingual pretraining [79], language understanding [80], and textual representations learning [81]. INFOXLM [79], a cross-lingual pretraining model, proposes a cross-lingual pretraining task based on maximizing the mutual information between two input sequences and learns to differentiate machine translation of input sequences using contrastive learning. Unlike TLM [82], this model aims to maximize mutual information between machine translation pairs in cross-lingual platform and improves the cross-lingual transferability in various downstream tasks, such as cross-lingual classification and question answering. Table 5 shows the recent contrastive learning methods on NLP downstream task.

Most of the popular language models, such as BERT [30] and GPT [32], approach pretraining on tokens and therefore may not capture sentence-level semantics. To address this issue, CERT [80] that pretrains models on the sentence level using contrastive learning was proposed. This model works in two steps: (1) creating augmentation of sentences using back-translation, and (2) predicting whether two augmented versions are from the same sentence or not by fine-tuning a pretrained language representation model (e.g., BERT and BART). CERT was also evaluated on 11 different natural language understanding tasks in the GLUE benchmark where it outperformed BERT on seven tasks. DeCLUTR [81] is self-supervised model for learning universal sentence embeddings. This model outperforms InferSent, a popular sentence encoding method. It has been evaluated based on the quality of sentence embedding on the SentEval benchmark.

Table 5. Recent contrastive learning methods in NLP along with the datasets they were evaluated on and the respective downstream tasks.

Model	Dataset	Application Areas
Distributed Representations [75]	Google internal	Training with Skip-gram model
Contrastive Unsupervised [77]	Wiki-3029	Unsupervised representation learning
CONPONO [78]	RTE, COPA, ReCoRD	Discourse fine-grained sentence ordering in text
INFOXLM [79]	XNLI and MLQA	Learning cross-lingual representations
CERT [80]	GLUE benchmark	Capturing sentence-level semantics
DeCLUTR [81]	OpenWebText	Learning universal sentence representations

9. Discussions and Future Directions

Although empirical results show that contrastive learning has decreased the gap in performance with supervised models, there is a need for more theoretical analysis to form a solid justification. For instance, a study by Purushwalkam et al. [83] reveals that approaches like PIRL [17] and MoCo [14] fail to capture viewpoint and category instance

invariance that are crucial components for object recognition. Some of these issues are further discussed below.

9.1. Lack of Theoretical Foundation

In an attempt to investigate the generalization ability of the contrastive objective function, the empirical results from Arora et al. [77] show that architecture design and sampling techniques also have a profound effect on the performance. Tsai et al. [84] provide an information-theoretical framework from a multi-view perspective to understand the properties that encourage successful self-supervised learning. They demonstrate that self-supervised learned representations can extract task-relevant information (with a potential loss) and discard task-irrelevant information (with a fixed gap). Ultimately, these findings propel such methods towards being highly dependent on the pretext task chosen during training. This affirms the need for more theoretical analysis on different modules in a contrastive pipeline.

9.2. Selection of Data Augmentation and Pretext Tasks

PIRL [17] emphasizes on methods that produce consistent results irrespective of the pretext task selected, but works like SimCLR [42] and MoCo-v2 [47], and Tian et al. [19] demonstrate that selecting robust pretext tasks along with suitable data augmentations can highly boost the quality of the representations. Recently, SwAV [13] beat other self-supervised methods by using multiple augmentations. It is difficult to directly compare these methods to choose specific tasks and transformations that can yield the best results on any dataset.

9.3. Proper Negative Sampling during Training

During training, an original (positive) sample is compared against its negative counterparts that contribute towards a contrastive loss to train the model. In cases of easy negatives (where the similarity between the original sample and a negative sample is very low), the contribution towards the contrastive loss is minimal. This limits the ability of the model to converge quickly. To get more meaningful negative samples, top self-supervised methods either increase the batch sizes [15] or maintain a very large memory bank [17]. Recently, Kalantidis et al. [85] proposed a few hard negative mixing strategies to facilitate faster and better learning. However, this introduces a large number of hyperparameters that are specific to the training set and are difficult to generalize for other datasets.

9.4. Dataset Biases

In any self-supervised learning task, the data provide supervision. In effect, the representations learned using self-supervised objectives are influenced by the underlying data. Such biases are difficult to minimize with the increasing size of the datasets.

10. Conclusions

This paper has extensively reviewed recent top-performing self-supervised methods that follow contrastive learning for both vision and NLP tasks. We clearly explain different modules in a contrastive learning pipeline; from choosing the right pretext task and selecting an architectural design, to using the learned parameters for a downstream task. The works based on contrastive learning have shown promising results on several downstream tasks such as image/video classification, object detection, and other NLP tasks. Finally, this work concludes by discussing some of the open problems of current approaches that are yet to be addressed. New techniques and paradigms are needed to tackle these issues.

Author Contributions: Conceptualization, A.J., A.R.B., and M.Z.Z.; introduction, A.J.; pretext tasks, A.R.B.; architectures, A.J.; encoder, A.J. and A.R.B.; training, M.Z.Z.; downstream tasks, A.J. and A.R.B.; benchmarks, A.J. and A.R.B., M.Z.Z. and D.B.; contrastive learning in nlp, D.B.; discussions, A.J. and M.Z.Z.; writing—original draft preparation, A.J., A.R.B., M.Z.Z., and D.B.;

writing—review and editing, A.J., A.R.B., M.Z.Z., and D.B.; visualization, A.J., A.R.B., and M.Z.Z.; supervision, F.M.; project administration, F.M.; funding acquisition, F.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by the National Science Foundation (NSF) under award numbers NSF-PFI 1719031 and NSF-CHS 1565328.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 618–626.
2. Liu, X.; Zhang, F.; Hou, Z.; Wang, Z.; Mian, L.; Zhang, J.; Tang, J. Self-supervised learning: Generative or contrastive. *arXiv* **2020**, arXiv:2006.08218.
3. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *arXiv* **2014**, arXiv:1406.2661.
4. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2223–2232.
5. Karras, T.; Laine, S.; Aila, T. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4401–4410.
6. Oord, A.V.d.; Kalchbrenner, N.; Kavukcuoglu, K. Pixel recurrent neural networks. *arXiv* **2016**, arXiv:1601.06759.
7. Reed, S.; Akata, Z.; Yan, X.; Logeswaran, L.; Schiele, B.; Lee, H. Generative adversarial text to image synthesis. *arXiv* **2016**, arXiv:1605.05396.
8. Kim, T.; Cha, M.; Kim, H.; Lee, J.K.; Kim, J. Learning to discover cross-domain relations with generative adversarial networks. *arXiv* **2017**, arXiv:1703.05192.
9. Epstein, R. The Empty Brain. 2016. Available online: https://aeon.co/essays/your-brain-does-not-process-information-and-it-is-not-a-computer (accessed on 1 November 2020).
10. Bojanowski, P.; Joulin, A. Unsupervised learning by predicting noise. *arXiv* **2017**, arXiv:1704.05310.
11. Dosovitskiy, A.; Fischer, P.; Springenberg, J.T.; Riedmiller, M.; Brox, T. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *38*, 1734–1747.
12. Wu, Z.; Xiong, Y.; Yu, S.X.; Lin, D. Unsupervised feature learning via non-parametric instance discrimination. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3733–3742.
13. Caron, M.; Misra, I.; Mairal, J.; Goyal, P.; Bojanowski, P.; Joulin, A. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. *arXiv* **2020**, arXiv:2006.09882.
14. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 9729–9738.
15. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. *arXiv* **2020**, arXiv:2002.05709.
16. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
17. Misra, I.; Maaten, L.V.D. Self-supervised learning of pretext-invariant representations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 6707–6717.
18. Trinh, T.H.; Luong, M.T.; Le, Q.V. Selfie: Self-supervised pretraining for image embedding. *arXiv* **2019**, arXiv:1906.02940.
19. Tian, Y.; Sun, C.; Poole, B.; Krishnan, D.; Schmid, C.; Isola, P. What makes for good views for contrastive learning. *arXiv* **2020**, arXiv:2005.10243.
20. Qian, R.; Meng, T.; Gong, B.; Yang, M.H.; Wang, H.; Belongie, S.; Cui, Y. Spatiotemporal Contrastive Video Representation Learning. *arXiv* **2020**, arXiv:2008.03800.
21. van den Oord, A.; Li, Y.; Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv* **2018**, arXiv:1807.03748.
22. Lorre, G.; Rabarisoa, J.; Orcesi, A.; Ainouz, S.; Canu, S. Temporal Contrastive Pretraining for Video Action Recognition. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision, Snowmass Village, CO, USA, 1–5 March 2020; pp. 662–670.
23. Sermanet, P.; Lynch, C.; Chebotar, Y.; Hsu, J.; Jang, E.; Schaal, S.; Levine, S.; Brain, G. Time-contrastive networks: Self-supervised learning from video. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 1134–1141.
24. Tao, L.; Wang, X.; Yamasaki, T. Self-supervised video representation learning using inter-intra contrastive framework. In Proceedings of the 28th ACM International Conference on Multimedia, Seattle, WA, USA, 12–16 October 2020; pp. 2193–2201.

25. Xiao, T.; Wang, X.; Efros, A.A.; Darrell, T. What Should Not Be Contrastive in Contrastive Learning. *arXiv* **2020**, arXiv:2008.05659.
26. Noroozi, M.; Favaro, P. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Computer Vision—ECCV 2016, Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016*; Springer:Berlin/Heidelberg, Germany, 2016; pp. 69–84.
27. Yamaguchi, S.; Kanai, S.; Shioda, T.; Takeda, S. Multiple Pretext-Task for Self-Supervised Learning via Mixing Multiple Image Transformations. *arXiv* **2019**, arXiv:1912.11603.
28. Cimpoi, M.; Maji, S.; Kokkinos, I.; Mohamed, S.; Vedaldi, A. Describing textures in the wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 3606–3613.
29. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. *arXiv* **2013**, arXiv:1301.3781.
30. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
31. Kiros, R.; Zhu, Y.; Salakhutdinov, R.R.; Zemel, R.; Urtasun, R.; Torralba, A.; Fidler, S. Skip-thought vectors. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 3294–3302.
32. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving language understanding by generative pre-training. **2018**, in progress.
33. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. *arXiv* **2019**, arXiv:1910.13461.
34. Glasmachers, T. Limits of end-to-end learning. *arXiv* **2017**, arXiv:1704.08305.
35. Hjelm, R.D.; Fedorov, A.; Lavoie-Marchildon, S.; Grewal, K.; Bachman, P.; Trischler, A.; Bengio, Y. Learning deep representations by mutual information estimation and maximization. *arXiv* **2018**, arXiv:1808.06670.
36. Ye, M.; Zhang, X.; Yuen, P.C.; Chang, S.F. Unsupervised Embedding Learning via Invariant and Spreading Instance Feature. *arXiv* **2019**, arXiv:1904.03436.
37. Bachman, P.; Hjelm, R.D.; Buchwalter, W. Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*; Vancouver, Canada, 8–14 December 2019; pp. 15535–15545.
38. Henaff, O. Data-efficient image recognition with contrastive predictive coding. In Proceedings of the International Conference on Machine Learning, Vienna, Austria, 12–18 July 2020; pp. 4182–4192.
39. Khosla, P.; Teterwak, P.; Wang, C.; Sarna, A.; Tian, Y.; Isola, P.; Maschinot, A.; Liu, C.; Krishnan, D. Supervised Contrastive Learning. *arXiv* **2020**, arXiv:2004.11362.
40. Goyal, P.; Dollár, P.; Girshick, R.; Noordhuis, P.; Wesolowski, L.; Kyrola, A.; Tulloch, A.; Jia, Y.; He, K. Accurate, large minibatch sgd: Training imagenet in 1 h. *arXiv* **2017**, arXiv:1706.02677.
41. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
42. Chen, T.; Zhai, X.; Ritter, M.; Lucic, M.; Houlsby, N. Self-supervised gans via auxiliary rotation loss. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 12154–12163.
43. Gutmann, M.; Hyvärinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Sardinia, Italy, 13–15 May 2010.
44. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
45. Srinivas, A.; Laskin, M.; Abbeel, P. CURL: Contrastive Unsupervised Representations for Reinforcement Learning. *arXiv* **2020**, arXiv:2004.04136.
46. Hafidi, H.; Ghogho, M.; Ciblat, P.; Swami, A. GraphCL: Contrastive Self-Supervised Learning of Graph Representations. *arXiv* **2020**, arXiv:2007.08025.
47. Chen, X.; Fan, H.; Girshick, R.; He, K. Improved Baselines with Momentum Contrastive Learning. *arXiv* **2020**, arXiv:2003.04297.
48. You, Y.; Gitman, I.; Ginsburg, B. Large Batch Training of Convolutional Networks. *arXiv* **2017**, arXiv:1708.03888.
49. Loshchilov, I.; Hutter, F. SGDR: Stochastic Gradient Descent with Warm Restarts. *arXiv* **2016**, arXiv:1608.03983.
50. Caron, M.; Bojanowski, P.; Joulin, A.; Douze, M. Deep Clustering for Unsupervised Learning of Visual Features. *arXiv* **2019**, arXiv:1807.05520.
51. Gidaris, S.; Singh, P.; Komodakis, N. Unsupervised Representation Learning by Predicting Image Rotations. *arXiv* **2018**, arXiv:1803.07728.
52. Zhou, B.; Lapedriza, A.; Khosla, A.; Oliva, A.; Torralba, A. Places: A 10 million image database for scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 1452–1464.
53. Soomro, K.; Zamir, A.R.; Shah, M. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv* **2012**, arXiv:1212.0402.
54. Kuehne, H.; Jhuang, H.; Garrote, E.; Poggio, T.; Serre, T. HMDB: A large video database for human motion recognition. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2556–2563.
55. Carreira, J.; Zisserman, A. Quo vadis, action recognition? A new model and the kinetics dataset. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6299–6308.
56. Zhuang, C.; Zhai, A.L.; Yamins, D. Local aggregation for unsupervised learning of visual embeddings. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 6002–6012.

57. Donahue, J.; Simonyan, K. Large scale adversarial representation learning. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 10542–10552.
58. Li, J.; Zhou, P.; Xiong, C.; Socher, R.; Hoi, S.C.H. Prototypical Contrastive Learning of Unsupervised Representations. *arXiv* **2020**, arXiv:2005.04966.
59. Asano, Y.M.; Rupprecht, C.; Vedaldi, A. Self-labelling via simultaneous clustering and representation learning. *arXiv* **2019**, arXiv:1911.05371.
60. Donahue, J.; Krähenbühl, P.; Darrell, T. Adversarial Feature Learning. *arXiv* **2017**, arXiv:1605.09782.
61. Doersch, C.; Gupta, A.; Efros, A.A. Unsupervised visual representation learning by context prediction. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1422–1430.
62. Zhang, R.; Isola, P.; Efros, A.A. Split-Brain Autoencoders: Unsupervised Learning by Cross-Channel Prediction. *arXiv* **2017**, arXiv:1611.09842.
63. Zhang, L.; Qi, G.J.; Wang, L.; Luo, J. AET vs. AED: Unsupervised Representation Learning by Auto-Encoding Transformations rather than Data. *arXiv* **2019**, arXiv:1901.04596.
64. Goyal, P.; Mahajan, D.; Gupta, A.; Misra, I. Scaling and Benchmarking Self-Supervised Visual Representation Learning. *arXiv* **2019**, arXiv:1905.01235.
65. Pathak, D.; Krahenbuhl, P.; Donahue, J.; Darrell, T.; Efros, A.A. Context encoders: Feature learning by inpainting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2536–2544.
66. Zhang, R.; Isola, P.; Efros, A.A. Colorful Image Colorization. *arXiv* **2016**, arXiv:1603.08511.
67. Kim, D.; Cho, D.; Kweon, I.S. Self-supervised video representation learning with space-time cubic puzzles. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 8545–8552.
68. Lee, H.Y.; Huang, J.B.; Singh, M.; Yang, M.H. Unsupervised representation learning by sorting sequences. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 667–676.
69. Sayed, N.; Brattoli, B.; Ommer, B. Cross and learn: Cross-modal self-supervision. In *GCPR 2018: Pattern Recognition, Proceedings of the German Conference on Pattern Recognition, Stuttgart, Germany, 9–12 October 2018*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 228–243.
70. Fernando, B.; Bilen, H.; Gavves, E.; Gould, S. Self-supervised video representation learning with odd-one-out networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3636–3645.
71. Misra, I.; Zitnick, C.L.; Hebert, M. Shuffle and learn: Unsupervised learning using temporal order verification. In *Computer Vision—ECCV 2016, Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 527–544.
72. Yao, T.; Zhang, Y.; Qiu, Z.; Pan, Y.; Mei, T. SeCo: Exploring Sequence Supervision for Unsupervised Representation Learning. *arXiv* **2020**, arXiv:2008.00975.
73. Liu, Z.; Gao, G.; Qin, A.; Li, J. DTG-Net: Differentiated Teachers Guided Self-Supervised Video Action Recognition. *arXiv* **2020**, arXiv:2006.07609.
74. Tian, Y.; Krishnan, D.; Isola, P. Contrastive Multiview Coding. *arXiv* **2019**, arXiv:1906.05849.
75. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, 3111–3119.
76. Gutmann, M.U.; Hyvärinen, A. Noise-Contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics. *J. Mach. Learn. Res.* **2012**, *13*, 307–361.
77. Arora, S.; Khandeparkar, H.; Khodak, M.; Plevrakis, O.; Saunshi, N. A Theoretical Analysis of Contrastive Unsupervised Representation Learning. *arXiv* **2019**, arXiv:1902.09229.
78. Iter, D.; Guu, K.; Lansing, L.; Jurafsky, D. Pretraining with Contrastive Sentence Objectives Improves Discourse Performance of Language Models. *arXiv* **2020**, arXiv:2005.10389.
79. Chi, Z.; Dong, L.; Wei, F.; Yang, N.; Singhal, S.; Wang, W.; Song, X.; Mao, X.L.; Huang, H.; Zhou, M. InfoXLM: An Information-Theoretic Framework for Cross-Lingual Language Model Pre-Training. *arXiv* **2020**, arXiv:2007.07834.
80. Fang, H.; Wang, S.; Zhou, M.; Ding, J.; Xie, P. CERT: Contrastive Self-supervised Learning for Language Understanding. *arXiv* **2020**, arXiv:2005.12766.
81. Giorgi, J.M.; Nitski, O.; Bader, G.D.; Wang, B. DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations. *arXiv* **2020**, arXiv:2006.03659.
82. Lample, G.; Conneau, A. Cross-lingual Language Model Pretraining. *arXiv* **2019**, arXiv:1901.07291.
83. Purushwalkam, S.; Gupta, A. Demystifying Contrastive Self-Supervised Learning: Invariances, Augmentations and Dataset Biases. *arXiv* **2020**, arXiv:2007.13916.
84. Tsai, Y.H.H.; Wu, Y.; Salakhutdinov, R.; Morency, L.P. Self-supervised Learning from a Multi-view Perspective. *arXiv* **2020**, arXiv:2006.05576.
85. Kalantidis, Y.; Sariyildiz, M.B.; Pion, N.; Weinzaepfel, P.; Larlus, D. Hard Negative Mixing for Contrastive Learning. *arXiv* **2020**, arXiv:2010.01028.