Linus Ericsson, Henry Gouk, Chen Change Loy, and Timothy M. Hospedales

Self-Supervised Representation Learning

Introduction, advances, and challenges

Self-supervised representation learning (SSRL) methods aim to provide powerful, deep feature learning without the requirement of large annotated data sets, thus alleviating the annotation bottleneck—one of the main barriers to the practical deployment of deep learning today. These techniques have advanced rapidly in recent years, with their efficacy approaching and sometimes surpassing fully supervised pretraining alternatives across a variety of data modalities, including image, video, sound, text, and graphs. This article introduces this vibrant area, including key concepts, the four main families of approaches and associated state-of-the-art techniques, and how self-supervised methods are applied to diverse modalities of data. We further discuss practical considerations including workflows, representation transferability, and computational cost. Finally, we survey major open challenges in the field, that provide fertile ground for future work.

Introduction

Deep neural networks (DNNs) now underpin state-of-the-art artificial intelligence (AI) systems for analysis of diverse data types [1], [2]. However, the conventional paradigm has been to train these systems using supervised learning, where performance has grown roughly logarithmically with annotated data set sizes [3]. The cost of such annotation has proven to be a scalability bottleneck for the continued advancement of state-of-the-art performance, and a more fundamental barrier for the deployment of DNNs in application areas where data and annotations are intrinsically rare, costly, dangerous, or time consuming to collect.

This situation has motivated a wave of research in SSRL [4], where freely available labels from carefully designed pretext tasks are used as supervision to discriminatively train deep representations. The resulting representations can then be reused for training a DNN to solve a downstream task of interest using comparatively little task-specific annotated data compared to conventional supervised learning.

Self-supervision refers to learning tasks that ask a DNN to predict one part of the input data—or a label programmatically derivable thereof—given another part of the input. This is in contrast to supervised learning, which asks the DNN to predict a manually provided target output, and generative modeling, which asks a DNN to estimate the density of the input data or learn a generator for input data. Self-supervised algorithms

Digital Object Identifier 10.1109/MSP.2021.3134634
Date of current version: 27 April 2022

differ primarily in their strategy for defining the derived labels to predict. This choice of pretext task determines the (in)variances of the resulting learned representation and thus how effective it is for different downstream tasks.

Self-supervised strategies have been leveraged successfully to improve sample efficiency of learning across a variety of modalities, from image [5]–[7], video [8], [9], speech [10], [11], text [12], [13], and graphs [14], [15]. Across these modalities, it can also be applied to boost diverse downstream tasks, including not only simple recognition but also detection and localization [16], dense prediction (signal transformation) [16], anomaly detection [17], and so on. Furthermore, some results suggest that self-supervised representation quality is also a logarithmic function of the amount of unlabeled pretraining data [16]. If this trend holds, then achievable performance may improve for "free" over time as improvements in data collection and computation power allow increasingly large pretraining sets to be used without the need for manually annotating new data.

There are various other strategies for improving the data efficiency of learning, such as transfer [18], [19], semisupervised [20], active, and metalearning. As discussed in this article, SSRL is an alternative to both conventional transfer and semisupervised learning pipelines; however, it can also be complementary to semisupervised and active learning.

©SHUTTERSTOCK.COM/LOCAL_DOCTOR

In this article, we focus on self-supervised algorithms and applications that address learning general-purpose features—or representations—that can be reused to improve learning in downstream tasks. We introduce SSRL and review its application and state of the art across several modalities (image, text, speech, graphs, and so on), with a specific focus on discriminative SSRL [we exclude generative models such as variational autoencoders (VAEs), generative adversarial networks (GANs), and flows, although they can also be used for representation learning]. Compared to existing surveys [4], we provide a broader introduction to the field; a wider coverage of different modalities rather than focusing on images; highlight more practical considerations such as representation transferability, computation cost, and deployment strategies; and provide a deeper discussion of open challenges.

Background

Problem definition

In this section, we introduce the necessary notation for defining the SSRL problem. We then contrast it to other common learning paradigms (see Figure 1).

Supervised learning requires a labeled data set for a target problem we wish to solve, $D_t = \{x_i^{(t)}, y_i^{(t)}\}_{i=1}^{N}$, from which we build a predictive model that makes estimates, $\hat{y} = f(x)$. In a deep learning context, the predictive model is usually composed of a representation extractor function, h_θ, and a classifier/regression function, g_ϕ, $f(x) = g_\phi(h_\theta(x)))$. We train this predictive model by minimizing a loss function $\mathcal{L}$, such as the negative log likelihood

$$\operatorname*{argmin}_{\theta,\phi} \sum_{(x_i^{(t)}, y_i^{(t)}) \in D_t} \mathcal{L}(g_\phi(h_\theta(x_i^{(t)})), y_i^{(t)}). \tag{1}$$

However, h_θ may have hundreds of millions of parameters, requiring millions of labeled data points in D_t to fit this correctly. These millions of annotated data points are not available in most applications, but many do have an essentially free supply of unlabeled data points; as an example, consider the wealth of raw audio signal data x versus the limited amount of transcribed speech data y in speech recognition.

Unsupervised learning techniques often learn from such unlabeled data by building generative models or density estimators. These range from classic shallow approaches, like Gaussian mixtures [21], to deep methods such as VAEs and GANs [18]. Other common unsupervised methods, such as autoencoders and clustering [18], learn compact latent representations. For example, autoencoders often optimize the following reconstruction objective:

$$\operatorname*{argmin}_{\theta,\phi} \sum_{x_i^{(t)} \in D_t} \mathcal{L}(g_\phi(h_\theta(x_i^{(t)})), x_i^{(t)}), \tag{2}$$

where $h_\theta(\cdot)$ extracts a compact feature from the input, and $g_\phi(\cdot)$ uses it to reconstruct the original input.

SSRL can be seen as a special case of unsupervised learning as both schemes learn without annotations, y. Although conventional unsupervised methods rely on reconstruction or density-estimation objectives, SSRL approaches rely on pretext tasks that exploit knowledge about the data modality used for training.

Although supervised learning methods tend to learn stronger features than unsupervised learning approaches, they require costly and time-consuming work from human annotators to generate the required labels. SSRL techniques aim for the best of both worlds: training a powerful feature extractor using discriminative learning without the need for manual annotation of training examples. Given an unlabeled source data set $D_s = \{x_i^{(s)}\}_{i=1}^{M}$, with $M \gg N$, self-supervised learning addresses how to make use of D_s and D_t together to learn the predictive model $f(x) = g_\phi(h_\theta(x))$.

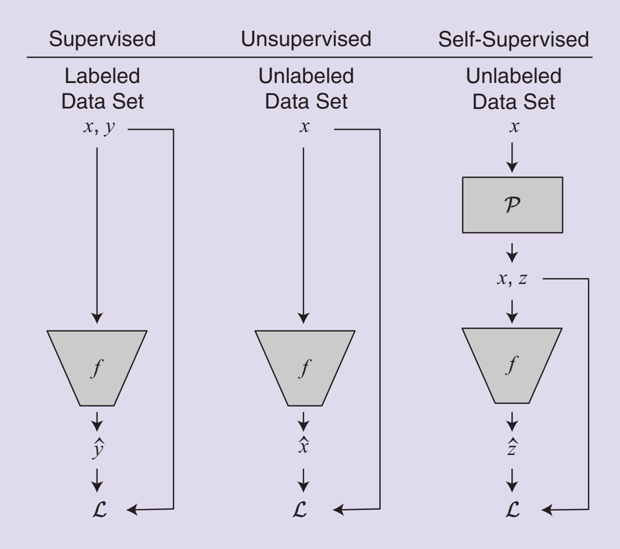

FIGURE 1. Contrasting supervised, unsupervised and self-supervised learning paradigms for training a model f using raw data x, labels y, and loss function $\mathcal{L}$. Self-supervision methods introduce pretext tasks $\mathcal{P}$ that generate pseudolabels z for discriminative training of f.

What defines a self-supervised method is its pretext task, consisting of a process, $\mathcal{P}$, to generate pseudolabels and an objective to guide learning. Given a raw data set like D_s, the pretext process programmatically generates pseudolabels z and possibly modified data points $\{x_i, z_i\}_{i=1}^{M} = \mathcal{P}(D_s)$. As an example, a portion of a speech signal x can be modified by masking out some part of the signal, and the pseudolabel z is defined as the masked-out portion of the input. An NN can then be trained on the objective of predicting the missing portion z, given the partially masked x.

Many self-supervision research activities address deriving pretext tasks $\mathcal{P}$, which enable learning general-purpose representations h_θ, which provide high performance and data-efficient learning of downstream tasks D_t. Different pretext tasks are discussed in detail in the "Pretext Tasks" section.

The workflow of self-supervision, also depicted in Figure 2, proceeds as follows:

1) Annotated data for the target task forms data set D_t, and available, unlabeled data forms the larger D_s.
2) The pretext task generates a new pseudolabeled data set, $\bar{D}_s = \{x_i, z_i\}_{i=1}^{M} = \mathcal{P}(D_s)$, as explained previously. (As process $\mathcal{P}$ often depends on sampling transformation or masking parameters, it is generally repeated at the start of each epoch of training.)
3) The pretext model, $k_\gamma(h_\theta(\cdot))$, is trained to optimize the self-supervised objective on $\bar{D}_s$:

$$\theta^* = \underset{\theta,\gamma}{\operatorname{argmin}} \sum_{(x_i,z_i)\in\mathcal{P}(\bar{D}_s)} \mathcal{L}(k_\gamma(h_\theta(x_i)), z_i). \tag{3}$$

Importantly, this provides a good estimate θ^* of the potentially hundreds of millions of parameters in h_θ, but without requiring label annotation. In many cases, input x_i is a single data point, and the pseudolabel z_i is a class label of scalar value. However, as discussed later in the "Pretext Tasks" section, in certain types of instance discrimination methods, the aforementioned input x_i can consist of multiple data

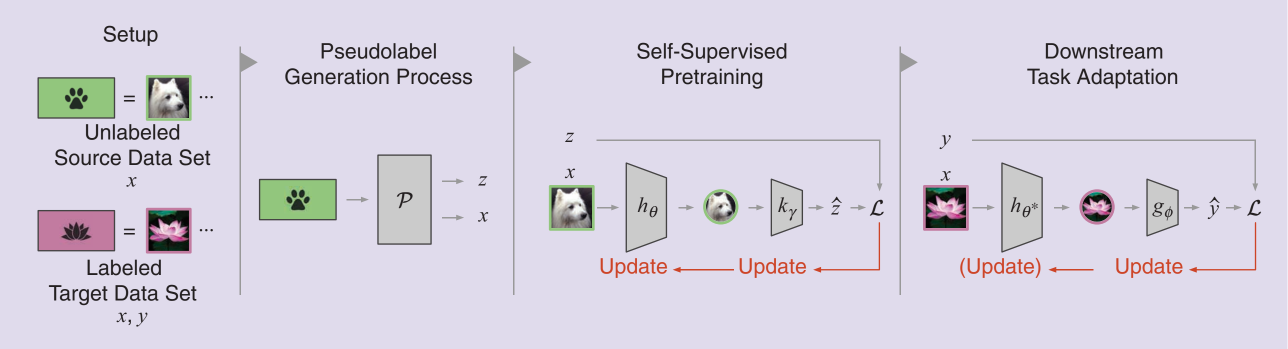

FIGURE 2. The self-supervised workflow starts with an unlabeled source data set and a labeled target data set. As defined by the pretext task, pseudolabels are programmatically generated from the unlabeled set. The resulting inputs, x and pseudolabels z, are used to pretrain the model $k_\gamma(h_\theta(\cdot))$—composed of feature extractor h_θ and output k_γ modules—to solve the pretext task. After pretraining is complete, the learned weights θ^* of the feature extractor h_{θ^*} are transferred and used together with a new output module g_ϕ to solve the downstream target task.

points, with pseudolabel z_i describing how the network should relate these data points. Similarly, in transformation prediction (TP), input x_i can consist of multiple shuffled chunks, while pseudolabel z_i relates the shuffled order to the original order.

4) Pretext output function k_γ is discarded, and representation function h_{θ^*} is transferred as a partial solution to solve the target problem of interest using model $g_\phi(h_{\theta^*}(\cdot))$. Crucially, when representation parameters θ^* are already well fitted from the self-supervision step in (3), only a minority of parameters may need to be learned or refined to solve the target problem, thus enabling it to be solved with a small, labeled target data set D_t. There are two common ways to solve the target problem using θ^*: fine-tuning and linear readout.

SSRL is an alternative to both conventional transfer and semisupervised learning pipelines; however, it can also be complementary to semisupervised and active learning.

The aforementioned presentation assumes that the target task is labeled and trained with supervised learning, as this is the most typical use case. However, unlabeled target tasks like clustering or retrieval can also obviously benefit from self-supervised pretraining if substituted into the previously mentioned step 4 [22].

Linear readout

For linear readout, let (θ, γ) be the weights of the pretrained model, consisting of a feature extractor, h_θ, followed by a task-specific head, k_γ. The simplest way to reuse h_θ for a new task is to replace the head with a new one, g_ϕ, designed for the new task. This head is then trained with the feature extractor frozen. Given a target data set of N instances, $D_t = \{x_i^{(t)}, y_i^{(t)}\}_{i=1}^N$, the training objective is

$$\underset{\phi}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(g_\phi(h_\theta(x_i^{(t)})), y_i^{(t)}). \tag{4}$$

The head is often a simple linear function, leading to the term *linear readout*. This is often used in the very sparse data regime where the number of unique parameters to learn for the target task must be aggressively limited to avoid overfitting [15], [23], [24].

If enough downstream data are available, it may be better to fit a more complex nonlinear function on top of the features. This may consist of multiple linear layers interspersed with nonlinearities and potential task-specific modules. In academic literature, however, it is very common to fit only linear functions to simplify the comparison of methods.

Fine-tuning

Instead of just training a new head, we can retrain the entire network for the new task. We usually still need to replace the pretext head with one suited to the target task, but now we train both the feature extractor and head, as follows:

$$\underset{\theta,\phi}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(g_\phi(h_\theta(x_i^{(t)})), y_i^{(t)}). \tag{5}$$

Crucially, one must initialize θ with the values θ^* obtained during the self-supervised pretraining phase. Given that DNN optimization is usually nonconvex, and assuming a small learning rate, this results in the aforementioned optimization converging toward a local optimum on the target task objective that lies near the local optimum attained for the source task, thus providing knowledge transfer from the pretext source task.

Fine-tuning is often used in the moderately sparse data regime where there is enough target data to at least refine all the model parameters, or in regimes where the pretext task/data are not perfectly suited to the downstream task [5], [12], [25]–[27]. If the source and target domains are well aligned, it may not be necessary—or even beneficial—to fine-tune all the parameters. Often only the final few layers of a network need tuning to adapt to a new task. In other cases, it is enough to tune a specific type of layer, like batch normalization, to adapt to a slight change in domain.

In summary, SSRL uses unlabeled data to generate pseudolabels for learning a pretext task. The learned parameters then provide a basis for knowledge transfer to a target task of interest. After pretraining, the transfer can be completed by linear readout or fine-tuning of the labeled target data.

Canonical use cases

When should one consider using self-supervision? SSRL may have diverse benefits in terms of adversarial robustness [28], model calibration [29], and interpretability [29], which is reviewed further in the "Discussion" section. However, its main use case is to improve data efficiency in situations where there are limited labels for the downstream target task (e.g., semantic segmentation or object detection) and/or domain (e.g., medical or Earth-observation images) of interest. The following few typical problem templates explain how SSRL can be applied to different situations:

- If dense labels are available for the target task and the domain, then direct supervised learning may be the most effective approach, and SSRL may not be helpful.
- If the target domain of interest is very different from any available background data sets (e.g., radar versus ImageNet data in imagery), and annotation is expensive in the target domain, then collecting unlabeled target data for target-domain specific self-supervision, followed by sparse data fine-tuning may be effective. This setting can also be addressed by semi-supervised methods [20], which should then be evaluated as competitors against SSRL.
- If the target domain of interest is similar enough to large source data sets (e.g., everyday objects versus ImageNet), then one can leverage self-supervised pretraining on the source data set before directly transferring the representation to the target domain of interest. Note that here,

conventional supervised pretraining is a competitor that should be evaluated against SSRL. However, in many cases, state-of-the-art SSRL has the edge on supervised pretraining for such transfer settings [29].

Deployment considerations

In this section, we discuss common ways of using a pretrained encoder h_θ for a labeled target data set. Although there are often domain- or task-specific methods in the literature for how to best do this, we focus on some of the most widely adopted approaches.

The target input data are often assumed to lie in the same space as the source data so that the encoder can be used without modification; however, the label spaces will most likely differ. This means that the head of the pretrained network, k_γ, is not suited to solve the target task. The design of the new head, g_ϕ, depends mainly on the label space of the target task. For example, in object recognition, the output is likely a vector of class probabilities; for visual object detection, additional bounding-box locations must be predicted; and for dense predictions, a deconvolutional decoder may be introduced.

> **Self-supervised algorithm design requires and exploits human prior knowledge about structure in the data to help define meaningful pretext tasks.**

Layer choice

Given the model pretrained on the source task, determining which feature layer is best for extracting features to solve a downstream task is an active research question [16]. This problem concerns finding the correct layer to split encoder h_θ and source head k_γ. The optimal choice can differ from task to task and data set to data set, and can involve combining features from several layers, but a general rule is that earlier layers tend to encode simple patterns while later layers can combine these simpler patterns into more complex and abstract representations.

Fine-tuning versus a fixed extractor

An important design choice in deployment phase is whether to fix encoder h_θ and just train a new classifier module g_ϕ using the target data, or fine-tune the encoder while training the classifier. Many SSRL benchmarks use an experimental design relying on the linear classifier readout of a frozen encoder. This makes SSRL methods easier to compare due to there being fewer parameters to tune in linear readout.

There have been mixed results reported in the literature with regard to whether linear readout is sufficient or whether fine-tuning the entire encoder should improve performance [29], [30]. Which performs better may depend on the amount of available data (fine-tuning is more reliable with more data), the similarity between the source and target domain data, and how well suited the (in)variances of the SSRL pretext task used are to the requirements of the downstream task. The conditions with a larger domain/task discrepancy are likely to benefit from more fine-tuning. Of course, there are numerous ways to control the amount of fine-tuning allowed in terms of learning rate, and explicitly regularizing the fine-tuning step to prevent it from overfitting by limiting the deviation from the initial pretrained conditions [19].

Other considerations

A unique issue for SSRL is that it can be difficult to determine the ideal stopping condition for the pretext task as no simple validation signal can be used. There is not yet an efficient solution for this issue.

Multiple studies have observed that downstream performance after SSRL improves with the capacity of the network architecture used for pretraining [5], [16]. Although convenient for extracting more performance at the cost of computation and memory, it may create a bottleneck for deploying the resulting fat representation on an embedded or other memory-constrained downstream platform. To alleviate this issue, high-performance and high-parameter count SSRL features can be distilled into smaller networks while retaining their good performance [5], [31] (see the "Architecture Choice and Deployment Costs" section).

Pretext tasks

In the absence of human-annotated labels, self-supervision uses the intrinsic structure of the raw data and automated process $\mathcal{P}$ to synthesize a labeled source data set, $\bar{D}_s = \{x_i, y_i\} = \mathcal{P}(D_s)$. One can then make use of $\bar{D}_s$ as they would any other labeled data set when pretraining a model by applying a discriminative supervised learning algorithm. As the pseudolabels are created from some intrinsic structure in the data, a model learning to predict those labels must recognize and exploit this structure to solve the task successfully. Thus, self-supervised algorithm design requires and exploits human prior knowledge about structure in the data to help define meaningful pretext tasks. Furthermore, different pretext tasks will induce different (in) variance properties in the learned representations, so the choice of method can also be informed by which properties of the representation are required by the downstream task. In the following sections, we divide the various self-supervised pretexts in the literature into four broad families: masked prediction, TP, instance discrimination, and clustering (see Figure 3).

Masked prediction

This family of methods is characterized by training the model to fill in the missing data removed by $\mathcal{P}$. It relies on the assumption that context can be used to infer some types of missing information in the data if the domain is well modeled. Given a raw example, $x_i^{(s)}$, a subset of the elements is extracted to form pseudolabel z^i, and the remaining components that were not used to create the label are used as the new input example, x_i. The pseudolabel generation process therefore looks like $x_i, z_i = \mathcal{P}(x_i^{(s)})$ and is described in full in Algorithm 1.

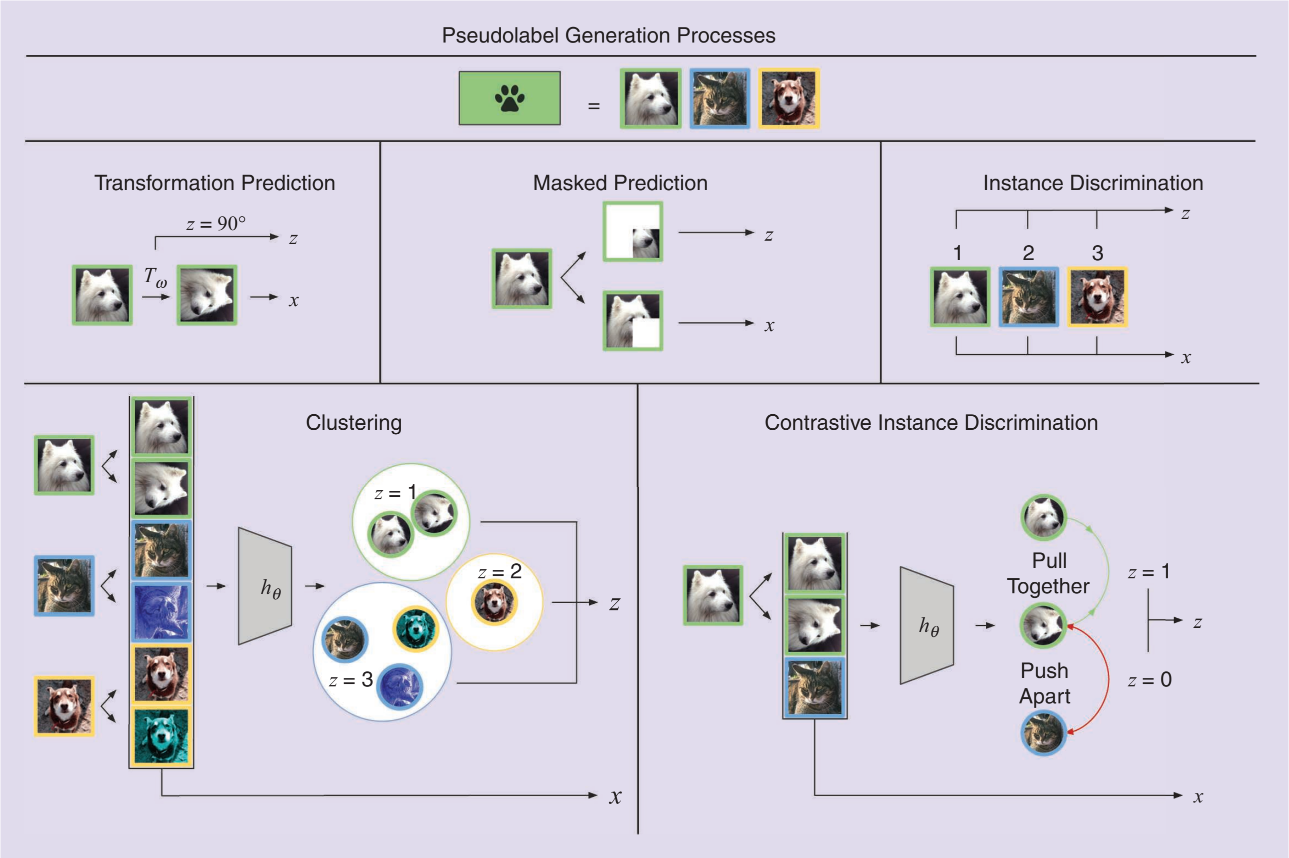

FIGURE 3. Illustrative examples of the way pseudolabels are generated in the four families of pretext tasks of our taxonomy: TP, masked prediction, instance discrimination, and clustering. An additional depiction is included of the popular version of instance discrimination using contrastive losses. The squares represent inputs x, while circles portray the feature vectors of those inputs, $h_\theta(x)$.

As an example of this on real data, a square region of an image can be masked out in the raw example. In this scenario, I is the set of indices inside the square-mask region, the pixels in the masked region will correspond to z_i, and the pixels outside the masked region will be x_i. Given x_i, z_i, the model can now be trained to minimize, e.g., the following reconstruction loss-like mean square error:

$$\theta^* = \operatorname*{argmin}_{\theta,\gamma} \frac{1}{|\mathcal{P}(D_s)|} \sum_{(x_i,z_i)\in\mathcal{P}(D_s)} (k_\gamma(h_\theta(x_i)) - z_i)^2. \qquad (6)$$

A major variant of masked prediction approaches are autoregressive methods, which treat x as a sequence, and the task is to autoregressively predict the $t+1$ element of the sequence, given the t elements seen thus far. By factorizing the joint distribution over x into a product of conditionals, these schemes can also be seen as unsupervised generative models.

Examples

Common masking methods involve hiding words in sentences for language modeling [12], [22], [32], hiding time slices in speech [10], hiding regions of images for inpainting [26], or hiding edges in graphs [27]. In a multimodal setting, it could correspond to, e.g., predicting the audio signal accompanying a video input, or vice versa.

Considerations

Defining an ideal masking strategy (how much, when, and where to mask; which context to provide in predicting the masked information) is important in making effective use of masked predictions. For example, masking too much of a speech signal will make it impossible to infer the missing words, while masking too little of it makes the task too easy to require a rich speech model to be learned.

Transformation prediction

This family of procedures relies on the assumption that inputs have a canonical view, and that certain transformations can be applied to that view to change it. The canonical view can, for example, depend on the effects of gravity in vision (i.e., there is a correct notion of up and down in visual scenes) or

Algorithm 1. The pseudolabel generation process $\mathcal{P}$ for masked prediction.

```
Input: Unlabeled data set $D_s = \{x_i^{(s)}\}_{i=1}^{M}$.
  for $i$ from 1 to $M$ do
      Generate indices, $I$, of elements to remove from $x_i^{(s)}$
      $z_i \leftarrow \{x_{i,j}^{(s)} : j \in I\}$
      $x_i \leftarrow \{x_{i,j}^{(s)} : j \notin I\}$
  end for
Output: $\{x_i, z_i\}_{i=1}^{M}$.
```

temporal ordering in video, speech, or other time series. TP methods apply a transformation that maps from canonical to alternative views and train the model to predict which transformation has been applied. Given a raw input in its canonical view, $x_i^{(s)}$, a transformation, T_ω, is applied to produce $x_i = T_\omega(x_i^{(s)})$, which is fed into the model.

The parameters ω of this transform are used as the pseudolabel $z_i = \omega$ that the model is trained to predict. It is typical for these transformation parameters to be sampled from some distribution, Ω. The learning objective can be, e.g., a cross-entropy loss, in the case of categorical transformation parameters.

$$\theta^* = \operatorname*{argmin}_{\theta,\gamma} \sum_{(x_i,z_i)\in\mathcal{P}(D_s)} \mathcal{L}_{CE}(k_\gamma(h_\theta(x_i)), z_i). \qquad (7)$$

The full process, $\mathcal{P}(D_s)$, is described in Algorithm 2. Typically, one will generate several different views of each $x_i^{(s)}$, each with a different set of transformation parameters. To succeed, an SSRL method has to learn enough about the latent structure of the data to correctly predict the transformation while being invariant to intracategory variability.

Examples

In vision applications, one can apply rotations to the raw images and require the network to predict the angle of rotation [33]. In temporal data, such as videos and other time series, one can shuffle the temporal order of signal samples and force the network to predict the original order [8], [34].

Considerations

Whichever transformation is chosen, the model will learn to produce representations that are equivariant to that transformation. This is because the information regarding the transformation needs to be retained in the representation for the final layer to be able to correctly solve the pretext task. A second consideration is it that depends on data having a canonical view. If there is no canonical view with respect to the set of transformations, then the performance will be poor. For instance, satellite or drone Earth-observation image data may have no canonical view with respect to rotation, so training for rotation prediction on this data may be ineffective.

Instance discrimination

In this family of methods, each instance in the raw source data set D_s is treated as its own class, and the model is trained to discriminate between different instances. There are a few different variations on this framework, which we now describe.

Algorithm 2. The pseudolabel generation process $\mathcal{P}$ for TP.

```
Input: Unlabeled data set $D_s = \{x_i^{(s)}\}_{i=1}^M$.
  for $i$ from 1 to $M$ do
      Sample $\omega \sim \Omega$
      $x_i \leftarrow T_\omega(x_i^{(s)})$          ▷ Apply transformation to raw input
      $z_i \leftarrow \omega$
  end for
Output: $\{x_i, z_i\}_{i=1}^M$.
```

Cross entropy

The most straightforward way of tackling instance discrimination is to assign each instance in the data set a one-hot encoding of its class label; for example, instance number 126 in a data set of 100,000 images would be assigned a vector of length 100,000 with zeros everywhere except for a value of one at position 126. This enables training the network with a categorical cross-entropy loss to predict the correct instances. This was the approach taken by the early exemplar-convolutional NN (CNN) method [23]. However, as the size of the data set grows, the softmax operation used to compute class probabilities becomes prohibitively expensive. As such, it became difficult to scale this process to large modern data sets where the number of instances—and therefore classes—can be millions [35] or even billions [36]. This led to the development of the contrastive procedures discussed in the next sections.

Another problem within the instance discrimination framework is the lack of intraclass variability. As each instance in the data set is treated as its own class, we end up with only a single example of each class. In conventional supervised learning, there might be hundreds or thousands of examples within each class to aid the network with learning the inherent variation within in each class. This problem was tackled by exemplar-CNN via extensive data augmentation. Given a data point, we can apply many different transformations to obtain slightly different views of that same data point while preserving its core semantic information. For example, we can slightly change the color of an image of a car, and it will still be perceived as an image of a car. Figure 4 shows examples of common transformations across modalities. The use of data augmentation has become an important component for instance discrimination methods as we see in the more recent contrastive- and regularization-based techniques discussed next.

Contrastive

The issue with using a categorical cross-entropy loss to solve instance discrimination is that it becomes intractable for large data sets. Researchers have therefore looked for ways to approximate this loss in more efficient ways. The core idea leading to recent advances is inspired by metric learning as well as the work in [37] and [38]. The idea is to not predict the exact class of the input but to instead predict whether pairs of inputs belong to the same or different classes. This allows for the use a binary class label instead of massively high-dimensional class vectors. If a pair of inputs belongs to the same class, the label is one, and if it belongs to different classes, the label is zero. In this setting, however, the use of data augmentation becomes even more important as we need to introduce variation among inputs of the same class.

To formalize the contrastive-instance discrimination setup, multiple views of inputs are created via some process T (transformation or sensory based) and compared in representation

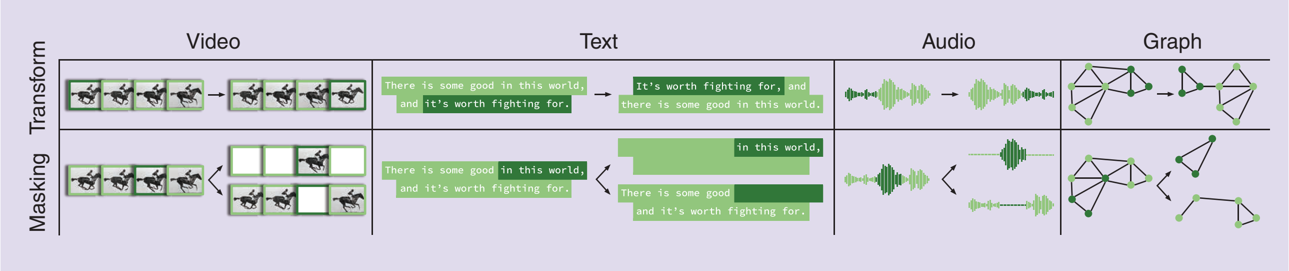
Video
Text
Audio
Graph
Transform
Masking
There is some good in this world,
and it's worth fighting for.
It's worth fighting for, and
there is some good in this world.
There is some good in this world,
and it's worth fighting for.
in this world,
There is some good
and it's worth fighting for.

FIGURE 4. The common transformation and masking methods for different modalities of data. The transformations can be applied to alter the order of sequential data, like the frames in a video, clauses in text, or chunks in audio waves. The graphs can be transformed by moving nodes or neighborhoods. Masking can be applied by hiding frames in videos, groups of words in text, chunks for audio data, or subgraphs in graphs. The darker-green area highlights the portion of the data point that is transformed or masked out. For examples of transforms and masking on image data, see Figure 3. (Source: Eadweard Muybridge, Human and Animal Locomotion, Plate 626, 1878–1887; Wikimedia Commons.)

space. One input, $x^a \sim T(x_i^{(s)})$, is chosen to be the anchor and is compared with a positive sample, $x^+ \sim T(x_i^{(s)})$, which is another view or transform of the same input. The anchor is also contrasted with a negative sample, which is a view of a different image, $x_j^- \sim T(x_j^{(s)})$. In the context of the general SSRL objective given in (3), this means that the pretext task generator $\mathcal{P}$ produces pretext inputs that each correspond to multiple pairs of raw input instances, with the associated pseudolabels indicating whether the pairs are matching or mismatching (see Algorithm 3 for a full description).

The samples are then encoded by the feature extractor to obtain their representations, $r^a = h_\theta(x^a), r^+ = h_\theta(x^+), r_j^- = h_\theta(x_j^-)$. A similarity function Φ is used to measure the similarity between positive (the anchor with a positive sample) and negative pairs (the anchor with a negative sample). The system is then trained to pull positive pairs closer and push negative pairs apart. A general formulation of the contrastive loss used in many works is

$$\mathcal{L}_{\text{con}} = -\mathbb{E}\left[\log \frac{\Phi(r^a, r^+)}{\Phi(r^a, r^+) + \sum_{j=1}^{k} \Phi(r^a, r_j^-)}\right], \tag{8}$$

where k different negative samples have been contrasted with the anchor. The model can then be updated by minimizing the contrastive loss

$$\theta^* = \underset{\theta,\gamma}{\operatorname{argmin}} \sum_{(x_i, z_i) \in \mathcal{P}(D_s)} \mathcal{L}_{\text{con}}(k_\gamma(h_\theta(x_i)), z_i). \tag{9}$$

Within this framework, methods differ in what similarity function they use, whether they use the same or different encoders for the anchor and other samples, which family of transformations T they use, and how they sample anchor, positive, and negative examples. Notable contrastive instance discrimination methods are SimCLR [5] and DGI [15].

Regularization based

Although the contrastive framework succeeds in scaling instance discrimination to large data sets, it still has some issues. To learn efficiently, a very large number of negative examples needs to be included in the loss. If we use too-few negative examples, the network will fail to learn the subtle differences between instances, but too many and training will be computationally expensive. If we were to remove negative examples altogether, the features of our network would all collapse to a single constant vector as there is no incentive to separate features.

Regularization-based approaches to instance discrimination avoid the use of negative examples altogether by regularization techniques that prevent feature collapse while keeping training efficient. There are many different schemes, like using asymmetrical encoding for the two inputs [6] or minimizing redundancy via cross correlation between features [39].

Examples

The established SSRL methods for computer vision, including MoCo [40] and SimCLR [5], fall into the family of regularization based. Other applications include speech [11] and multiview [41] and multimodal representation learning, including audiovisual [42] and visuolinguistic [43] data, where matching and mismatching views of the same instance are contrasted against each other.

Considerations

The representations learned here develop high sensitivity to instances while developing invariance to transformations or

Algorithm 3. The pseudolabel generation process $\mathcal{P}$ for con trastive-instance discrimination.

```
Input: Unlabeled data set $D_s = \{x_i^{(s)}\}_{i=1}^{M}$.
  for $i$ from 1 to $M$ do
      Sample $x^a \sim T(x_i^{(s)})$
      Sample $x^+ \sim T(x_i^{(s)})$
      for $k$ from 1 to $K$ do
          Sample $j \sim \mathcal{U}(1, M)$                ▷ Pick another raw input.
          Sample $x_k^- \sim T(x_j^{(s)})$                 ▷ Get a random transform
      end for
      $x_i \leftarrow \{(x^a, x^+), (x^a, x_1^-), ..., (x^a, x_K^-)\}$.
      $z_i \leftarrow \{1, 0, ..., 0\}$.
  end for
Output: $\{x_i, z_i\}_{i=1}^{M}$.
```

views. This means that the design of augmentation or view-selection function T is important due to its influence on the invariances learned. For example, aggressive color augmentation in T may lead to color-invariant representations [29], which could either be an issue or a benefit depending on the downstream task. When using different speakers as different views for audio data, the representations would become speaker invariant, which could be beneficial if the downstream task is speech recognition but an issue if it is speaker diarization.

Recent work has systematically demonstrated this intuition that the ideal transformations to use do indeed depend on the downstream task [44]. On one hand, this undermines the appealing and widely believed property of SSRL that a single pretrained model can be reused for diverse downstream tasks. On the other hand, it highlights a new route for research to further improve performance by customizing the transformation choice according to the downstream task requirements.

Instance discrimination methods implicitly assume that all instances in the raw data set represent unique semantic examples, which might not hold, e.g., if there are many images of the same object. When this assumption is violated, they suffer from false-positive pretext task labels [45]. Nevertheless, they are highly effective in practice despite this violated assumption.

A different issue that is not well understood in theory but crucial in practice is the sampling and batching strategy for anchor, positive, and negative instances for contrastive techniques. For example, how should negative samples be chosen (e.g., at random, via hard negative mining)? What proportion of positive and negative samples, and what batch size should be used [40]? These are all crucial design parameters that vary across the many approaches and significantly influence performance.

Clustering

This family of methods focuses on dividing the training data into a number of groups with high-intragroup and low-intergroup similarity. This relies on the assumption that there exists meaningful similarities by which the data can be grouped, which is likely the case, especially if the data are categorical in nature. There are multiple ways of determining cluster assignment, such as connectivity (hierarchical clustering), centroids fitting (e.g., k-means), likelihood maximization (e.g., Gaussian mixture modeling), and so on [21].

Algorithm 4. The pseudolabel generation process $\mathcal{P}$ for clustering.

```
Input: Unlabeled data set $D_s = \{x_i^{(s)}\}_{i=1}^{M}$.
Input: Representations $\{r_i\}_{i=1}^{M}$, where $r_i \leftarrow h_\theta(x_i^{(s)})$
Input: Cluster centers $\{c_j\}_{j=1}^{k}$, via clustering on $\{r_i\}_{i=1}^{M}$.
  for $i$ from 1 to $M$ do
      Sample $x_i \sim T(x_i^{(s)})$
      $z_i \leftarrow \mathrm{argmin}_{j \in [k]} \| c_j - r_i \|$
  end for
Output: $\{x_i, z_i\}_{i=1}^{M}$.
```

As opposed to traditional clustering, in SSRL, the aim of the algorithm is to obtain a good feature extractor, f_θ, instead of cluster assignments. Thus, one typically jointly performs feature extractor learning and clustering to pretrain the representation prior to downstream use. This is in contrast to classic clustering methods, which normally use a fixed set of features.

A common approach to self-supervised clustering is by alternating two steps: 1) optimizing the clustering objective by assigning data points into clusters based on their representations and (2) optimizing the model by using the cluster assignments as the pseudolabels in updates.

A unique feature of the clustering family is thus that pretext task $\mathcal{P}$ changes during the course of training. As the pseudolabels are created by clustering the current representations at each epoch, the labels are updated as the representations change. This means that the input to process $\mathcal{P}$ at each iteration is the representations and clusters in addition to the raw data. The full process $\mathcal{P}$ is described in Algorithm 4.

Given a cluster assignment where each input x_i has its cluster class assigned to z_i, we can optimize the model via a cross-entropy loss:

$$\theta^* = \underset{\theta,\gamma}{\mathrm{argmin}} \sum_{(x_i, z_i) \in \mathcal{P}(D_s)} \mathcal{L}_{CE}(k_\gamma(h_\theta(x_i)), z_i). \tag{10}$$

After this, we go back to the clustering step, now using the new representations of our updated model.

In the cluster assignment step, many works use k-means clustering [7], [46], where the number of clusters k is a hyperparameter set by evaluating on a validation set of a downstream task. A big problem is that there are degenerate solutions to this, such as assigning all instances to the same cluster [46]. To avoid this, methods often enforce that cluster assignments must be balanced [7]. Recent approaches such as ODC [47] aim to avoid the burden of alternating updates of the feature extractor and clusters by simultaneously updating both online.

Examples

The major examples include DeepCluster, ODC [46], [47], and SwAV [7] for vision, and XDC [9] for multimodal clustering, such as audio and video.

Considerations

Many clustering-based SSRL techniques [46] rely less heavily on data augmentation compared to contrastive methods [5]. This fact, as well as avoiding the need to sample triplets, have some benefit in terms of computation cost; however, the nonstationary nature of the clustering SSRL task (clusters coevolve with features) imposes additional cost compared to the other pretext tasks with stationary objectives. Compared to instance discrimination, TP, and masked prediction pretexts, it can be harder to analyze the kinds of (in)variances induced by clustering-based SSRL, making it harder to predict which downstream tasks they are suitable for without empirical evaluation.

Theoretical underpinning

The theoretical underpinnings of SSRL are lacking compared to standard supervised learning. When analyzing a conventional supervised method, the object of most interest is the expected performance of a model on unseen data. The model's performance is measured using a task-specific loss function. For example, consider the case of a binary classification problem where the model produces a real-valued score: the sign of this score indicates the predicted class, and the magnitude provides an indication of the confidence with which the model is making the prediction. One loss function that is commonly used for evaluation purposes is the following zero-one error:

$$\mathcal{L}_{0-1}(f, x, y) = \mathbb{I}(f(x)y > 0), \tag{11}$$

where $y \in \{-1, 1\}$ is the ground-truth label, f is the model, x is an input, and $\mathbb{I}(\cdot)$ is the indicator function. The expected performance of a model on unseen data are then denoted by

$$\mathbb{E}_{x,y}[\mathcal{L}_{0-1}(f, x, y)]. \tag{12}$$

The typical goal in statistical learning theory is to bound this quantity from (12) using the error measured on the training set and some measure of complexity of the class of models, $\mathcal{F}$, which the training algorithm is optimizing over. Such bounds are probabilistic due to the inherent randomness involved in sampling a training data set, and in some sense can be thought of as sophisticated confidence intervals. These bounds hold uniformly overall, $f \in \mathcal{F}$, and usually take the form

$$\mathbb{E}_{x,y}[\mathcal{L}_{0-1}(f, x, y)] \leq \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_{0-1}(f, x_i, y_i) + \mathcal{C}(\mathcal{F}, n, \delta), \tag{13}$$

where the inequality holds with a probability of at least $1 - \delta$, so δ is essentially defining the width of a confidence interval, as in classic statistical analysis. The complexity term $\mathcal{C}(\mathcal{F}, n, \delta)$ can be thought of as the upper bound of a confidence interval that takes into account multiple hypothesis testing, i.e., each $f \in \mathcal{F}$ can be thought of as a hypothesis. As more complex classes of models are considered, this term will grow larger. Crucially, these bounds assume no knowledge about the underlying data-generating distribution, and as such, they hold for all distributions.

One of the standard assumptions made in learning-theoretic analysis is that the elements in the training and test sets are sampled from the same distribution.

There are several roadblocks preventing the direct application of this framework to SSRL methods. The most fundamental issue is that the training loss used during self-supervised pretraining measures performance on a pretext task and is generally not the same loss function used for measuring the performance of the downstream task. As a consequence, the training loss cannot be interpreted as a biased estimate of the expected model performance, and analysis of the model's class complexity cannot be used to compensate for the bias in this estimate by widening the confidence interval. A further complication comes from the distribution shift. In many cases, one wishes to perform self-supervised pretraining on one data set (such as ImageNet) and then use the resulting features on another data set with a different marginal distribution. One of the standard assumptions made in learning-theoretic analysis is that the elements in the training and test sets are sampled from the same distribution.

Nevertheless, there is a small but growing body of literature devoted to the theoretical analysis of SSRL techniques. The key goal these papers share is relating a self-supervised training objective to a supervised one measured on a small set of labeled data by, e.g., showing that the SSRL loss can be interpreted as an upper bound to a supervised loss. Such analyses typically rely on making assumptions about the data-generating process that are hard to verify in practice. We briefly outline three recent approaches to connecting SSRL with conventional statistical learning theory: one approach that applies to only instance discrimination methods [48], another that primarily considers how SSRL learns useful representations for natural language tasks [49], and finally, a paper that makes use of conditional independence to further elucidate how masked prediction pretext tasks lead to useful representations.

The analysis of contrastive-instance discrimination methods for SSRL [48] is predicated on the assumption of a specific data-generating process. In particular, they assume that the data are generated by a mixture of distributions associated with latent classes. For example, there is a distribution over the pixels in an image associated with the concept "dog," and there is some prior probability that an image from a particular domain will contain a dog. They demonstrate that one can bound the supervised loss by

$$\mathbb{E}_{x,y}[\mathcal{L}_{0-1}(f, x, y)] \leq \mathbb{E}_{x}[\mathcal{L}^{-}_{ssrl}(f, x)] + s(\rho) + \mathcal{C}(\mathcal{F}, n, \delta), \tag{14}$$

where $\mathcal{L}^{-}_{ssrl}(\cdot,\cdot)$ is a modification to the contrastive loss that considers only negative pairs, and $s(\rho)$ is a function of the mixing coefficients, ρ, over the latent classes. This bound relies on f being a centroid classifier on top of the network trained with SSRL, and it is shown that this line of analysis is of limited use on more general families of models.

SSRL on text data is often formalized as a masked prediction problem where, given the first part of a sentence, the task is to predict the next word or remainder of the sentence. Recent work [49] has provided a concrete link between the performance on this pretext task and the performance one can expect to see on natural language classification problems. However, their analysis does require an assumption for how classification tasks can be reformulated to make them more comparable with the sentence-reconstruction pretext task. Their first contribution is to formalize this assumption as a falsifiable hypothesis and empirically verify that it holds, in practice. Their second main contribution investigates the transfer performance of ϵ-optimal language models, namely, those that achieve an expected cross-entropy loss within ϵ of the expected loss of the best possible model. They show that, conditioned on this empirically verified hypothesis being true, if one can find a model for next-word prediction with an ϵ-optimal cross-entropy loss, then the cross-entropy loss for a downstream classification task will be $O(\sqrt{\epsilon})$. This implies that developing models that are better at the next-word prediction pretext task will translate into better feature representations for natural language classifiers.

Lee et al. [50] conduct a more general analysis of masked prediction pretext tasks that is not restricted specifically to the NLP domain. Recall that masked prediction pretext tasks take each source instance $x_i^{(s)}$ and produce two new objects, x_i and z_i, which contain subsets of the elements in the original instance. It is shown in [50] that if there is conditional independence between x_i and z_i given the downstream label (and optionally some additional latent variables), then any model that successfully predicts z_i from x_s must be estimating the label (and optional latent variables). They further generalize their results to the case where one must only assume some notion of approximate conditional independence, which they quantify in terms of covariance matrix norms.

Table 1. The notable methods in each modality.

	Method	Pretext Task	Code/PT
Images	RotNet [33]	TP	Y/Y
	iGPT [52]	MP	Y/Y
	Colorization [53]	MP	Y/Y
	Inpainting [26]	MP	Y/Y
	MoCo [40]	ID	Y/Y
	SimCLR [5]	ID	Y/Y
	BYOL [6]	ID	Y/Y
	SwAV [7]	Cl	Y/Y
Video/MM	VCP [54]	MP	Y/N
	CLIP [43]	ID	Y/Y
	XDC [9]	Cl	N/Y
	ViLBERT [55]	MP + ID	Y/Y
Text	word2vec [22]	MP	Y/ Y
	ELMo [56]	MP	Y/ Y
	BERT [12]	MP	Y/Y
	GPT [13, 32]	MP	Y/Y N/N
S and TS	CPC [11]	MP	N/N
	wav2vec [10]	MP	Y/Y
	STRN [34]	TP	Y/Y
Graph	Node2Vec [14]	MP	Y/N
	GraphSAGE [57]	MP	Y/N
	DGI [15]	ID	Y/N
	GPT-GNN [27]	MP	Y/Y
	GraphTER [24]	TP	Y/Y

code/PT: indicates whether a code-base and pretrained models are available, respectively. MP: masked prediction; ID: instance discrimination; BYOL: bootstrap your own latent; Cl: clustering; S and TS: speech and time-series; Y/Y: yes/yes; Y/N: yes/no; N/Y: no/yes; N/N: no/no.

Although there have been some advances in understanding why contrastive and masked prediction schemes can lead to discriminative representations for downstream tasks, this work does rely on assumptions about the data (e.g., conditional independence) that have not been verified to occur in practice. Moreover, the empirical results associated with procedures from other parts of our taxonomy, such as TP and deep clustering, have still not been investigated. An example of how further work could address gaps in our current understanding is to extend theoretical frameworks analyzing (shallow) clustering methods [51] to the deep SSRL paradigm. Future work addressing these limitations would be useful for SSRL researchers and to the broader AI community that make use of pretrained features.

Methods and data sets

In this section, we review major techniques and considerations broken down by data modality. The summaries of major methods and data sets for image, video, text, time series, and graph modalities are provided in Tables 1 and 2, respectively.

Images

Computer vision tasks performed on still images vary broadly from recognition (whole image classification), detection (object localization within an image), and dense prediction

Table 2. The common source data sets used in each modality.

	Source	Size
Images	ImageNet [35]	1.3 million images
	YFCC100M [59]	100 million images
	iNaturalist [60]	2.7 million images
Video and MM	Kinetics [61]	650,000 videos
	YouTube-8M [62]	8 million videos
	HowTo100M [63]	136 million videos
Text	WikiText [64]	100 million tokens
	OpenWebText [65]	40 GB of text
	Common Crawl [66]	410 billion tokens
S and TS	Librispeech [67]	960 h of speech
	Libri-Light [68]	60,000 h of speech
	AudioSet [69]	580,000 h of audio
Graph	Open Academic Graph [70]	178 million nodes, 2 billion edges
	Amazon Review Recommendation [71]	113 million nodes
	PROTEINS [72]	1,100 graphs

(e.g., pixelwise segmentation). State-of-the-art performance on all of these tasks is achieved by supervised deep learning, and thus, SSRL aims to alleviate the annotation bottleneck in computer vision by providing self-supervised pretraining that can be combined with data-efficient fine-tuning.

Computer vision has long been dominated by the use of CNNs that use weight sharing to reduce the number of learnable parameters by exploiting the spatial properties of images. State-of-the-art architectures usually start with CNN representation encoding $h_\theta(\cdot)$, with residual networks (ResNet) [1] being widely used, before appending task-specific decoding heads g_ϕ. Many of the initially successful practices in SSRL used ResNet backbones [5], but a recent trend has brought transformer architectures into the vision domain [52]. One notable version is a vision transformer [58], which is increasingly being used by recent self-supervised methods on image data [43].

Methods

All types of pretext tasks (see the "Pretext Texts" section) have been widely applied in still imagery (see Table 1). The earliest example of a self-supervised system, given the modern interpretation of the phrase, is the work of [37]. This paper introduced two fundamental ideas still relevant to techniques being developed today: 1) metric learning with a contrastive loss, and a heuristic for generating training pairs that can be used to train an NN feature extractor; 2) side information, such as the relative position or viewing angle of training images, can be used to learn invariant or equivariant features. The subsequent methods that focused on SSRL for single images also pursued the goal of developing feature extractors that are invariant to different types of transformations through transformation augmentations [23].

Several approaches fall into the TP family, focusing on modifying unlabeled images using a known transformation, like rotation [33], and then training the network to predict the angle of that rotation. The others mask out information in the training images and require the network to reconstruct it, leading to pretext tasks such as colorization [53] and inpainting [26], where color channels and image patches, respectively, are removed. A state-of-the-art example in this category is iGPT [52], which exploits a self-attention architecture and masked prediction for representation learning.

The majority of recent schemes focus on the relationships between different images in the data set, using instance discrimination [5], [40] or clustering [7]; and heavy data augmentation has become a vital component required by all procedures to achieve high performance. Progress has accelerated rapidly in the last two years, with the latest methods now systematically outperforming supervised pretraining in diverse downstream tasks and data sets [29], as shown in Figure 5.

Data sets

As in much of computer vision, ImageNet [35] is the most typical source data set for self-supervised pretraining [5]–[7], [40], consisting of 1.28 million training images across 1,000 object categories, with the most commonly used resolution at 224 × 224. Many methods are increasingly using data sets much larger than ImageNet. For example, YFCC100M [59], with 100 million images from Flickr, used by Caron et al. [46], and by the authors in [36], with 3.5 billion images from

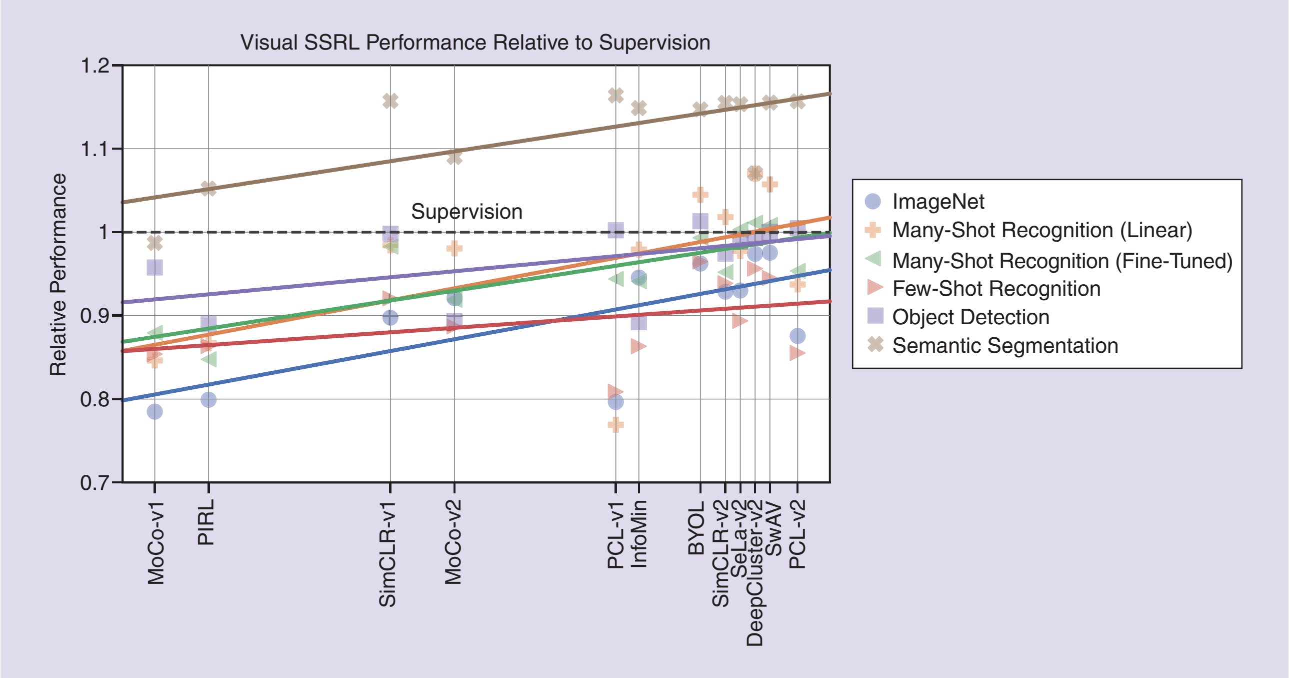

FIGURE 5. The relative performance of SSRL methods on visual tasks, compared to a supervised baseline. The figure was produced based on the results in [29].

Instagram. The subsets of the latter are used by the authors in [7] and [40].

The ImageNet benchmark is a highly curated data set, with certain biases that do not appear in natural images, such as centering of objects and clear isolation of object from background. iNaturalist [60] is a collection of wildlife data sets compiled by a citizen science project, where members upload their own photographs and others collectively annotate them. This forms a more natural data set that exhibits class imbalance and distractor objects, which often complicate real-world tasks. Although it has not yet served as the source data set for any new method, it has been used to benchmark the robustness of existing SSRL techniques to more uncurated data [73].

Applications

On established benchmarks, SSRL has had widespread and significant success in matching and surpassing supervised pretraining performance, especially for image recognition tasks, and in photo imagery of similar character to ImageNet (see Figure 5). Progress in transfer to more diverse downstream tasks such as detection and segmentation as well as downstream data sets, which are out of distribution with respect to pretraining data, has also been steady [29], if less rapid.

Beyond common benchmarks, SSRL has been successfully applied in application areas where labeled data are sparse, such as Earth-observation remote sensing [74]. In these cases, pretraining on the available, unlabeled target-domain data was beneficial to compensate for sparse annotations. A growing downstream consumer of SSRL is the medical imaging domain, where labeled data are often intrinsically sparse or too expensive to collect in bulk for end-to-end learning from scratch. For example, the authors in [75] used unlabeled brain scan images to perform image restoration (an inpainting-like task), improving upon random initialization for fine-tuning several downstream tasks. A somewhat unique feature of the medical imaging domain is the processing of 3D volumetric images, such as from magnetic resonance. This has also recently inspired various extensions of standard pretext tasks into 3D [76].

A final application where SSRL pretraining has been successfully applied is that of anomaly detection. SSRL-based approaches typically either train a feature to be used in conjunction with a classic generative anomaly detector, or more interestingly use the SSRL objective itself to produce an anomaly detection score. For example, current state-of-the-art anomaly detectors [77] rely on SSRL training of rotation prediction, with the rotation prediction accuracy providing the anomaly score.

Video and multimodal

In the domain of video and multimodality, diverse tasks are of interest, including video recognition, action/event detection (localization of an event within a longer video), tracking (localizing an object within frames and across time), and cross-modal retrieval (e.g., retrieving a video frame given associated subtitles). State-of-the-art architectures again dominate all of these tasks, given access to sufficient training data to train encoder and task-specific decoder components.

Common architectures $h_\theta(\cdot)$ for encoding videos include 3D CNNs or multistream encoders that process appearance and motion separately. In the case of multimodal processing of video and audio, or video and associated text, one requires a synchronized video CNN encoder as well as a text/audio encoder (e.g., a recurrent NN) to encode the multimodal streams. These data streams may then be fused into a single representation and decoded at each time step (e.g., for localization/detection), or first pooled over time (e.g., for video-level recognition).

Methods

TP and contrastive-instance discrimination methods are the most widely used for SSRL in video. There are a wide variety of TP pretexts in video. Rotation, and colorization discussed earlier, are also widely generalized to video data. Making more unique use of the temporal nature of video, one can, for example, predict the ordering of frames or clips [8], or the speeding up or slowing down of videos.

In terms of contrastive-instance discrimination methods, data augmentation has been the main mode of obtaining different views in still imagery. However, for videos, several approaches exploit multiple sensory views, like red, green, blue (RGB); optical flow; depth; and surface normals [41], [78], which provide different views for learning cross-view video clip matching.

A recent notable method in the instance discrimination family is CLIP [43], a visuolinguistic multimodal learning algorithm that has further advanced the state of the art in robust visual representation learning by crawling pairs of images and associated text from the Internet, and exploiting them for cross-view contrastive learning. Massive multimodal pretraining was shown to lead to excellent performance on diverse downstream tasks, including language-based image retrieval.

Clustering has been used in similar ways to match inputs from different modalities to the same clusters [9]. Finally, masked prediction has been applied through filling in masked-out clips [54].

Data sets

There are several data sets of videos used for pretraining in this modality. Kinetics [61] is a large, action recognition data set of human–object and human–human interactions, collected from YouTube videos. One version, Kinetics-400, contains approximately 300,000 videos. There are larger versions of the data set, with up to 700 classes and 650,000 videos. Recently, a group of very large-scale data sets have been constructed from publicly available videos on social platforms, like YouTube-8M [62] and HowTo100M [63], the latter containing 136 million YouTube instructional videos featuring narration with captions across 23,000 visual tasks.

For methods using multiple modalities, the visual and audio information often come from the large video data sets discussed previously [79]. An additional data set considered here is AudioSet [69], an audio event detection data set. For schemes using text information, this is often obtained from automated transcription using automatic speech recognition (ASR). Other data sets have textual information built in, such as subtitles or video descriptions.

Applications

As outlined in the previous section, the most common application and benchmark scenario for video SSRL is in video action recognition and detection in various guises. SSRL has made rapid progress in this area, and state-of-the-art methods trained on massive pretraining sources lead to significantly better performance than direct training on an array of standard benchmarks [9] but do not yet reliably surpass supervised pretraining on the same source data sets as in the case of still images earlier.

Similar to the still-image domain, SSRL has been successfully applied to video anomaly detection. For example, given a TP pretext task of arrow-of-time prediction (differentiating forward- versus reverse-frame sequences) among others, videos with a high probability of being reversed can be considered anomalous [17].

Video data are often multimodal, covering RGB-depth, video plus audio, or video plus text (e.g., from script- or text-to-speech) modalities. It is noteworthy that several studies [41], [79] have explored how SSRL on multimodal source data can be used to learn a stronger representation for single-modality downstream tasks, and ultimately outperform single-modality pretraining on diverse downstream tasks in unimodal video, still-image, or audio domains [79].

With regard to the video and text, several recent SSRL studies have learned joint multimodal representations. Notably, ViLBERT [55] exploited both BERT-like masked prediction and contrastive-instance discrimination to learn a multimodal representation, which then achieved state-of-the-art performance in downstream vision and language tasks such as caption-based retrieval, visual question answering (QA), and visual common sense reasoning.

Text and natural language

Natural language processing (NLP) methods aim to learn from raw input text and solve a wide variety of tasks, ranging from low level, such as word similarity, part of speech tagging and sentiment, to high-level tasks such as QA and language translation. The state-of-the-art approaches are often based on deep-sequence encoders such as long short-term memory (LSTM), and in recent years, self-attention-based approaches have been dominant [2]. With data annotation being a major bottleneck, NLP was the first discipline to make major and successful use of self-supervision [22].

Methods

SSRL has been a fundamental component in NLP for many years. Masked prediction methods have been particularly effective in this modality, with word embeddings becoming widely adopted as they succeed in producing representations that capture the semantic similarity of words as well as being able to deal with arbitrary vocabulary sizes. Word2vec [22] and related approaches work by either predicting a central word given its neighbors, called *continuous bag of words*, or predicting the neighbors given the central word, called *skip-gram*. Given such pretrained word embeddings, a target task is then solved by mapping input tokens to their vector embeddings and learning a model on top of them. As the embedding for a word is fixed after training, it cannot adapt to the context in which the word appears, causing a problem for words with many meanings.

As opposed to these noncontextual embedding methods, topical contextual approaches learn embeddings, which change depending on the surrounding words. The two most common approaches to this are next-word [13] and masked-word prediction [12], with the landmark BERT process combining the latter with next-sentence prediction [12]. For the encoder architecture, recurrent networks like LSTMs [18] have long been used to model the context while recent works have moved to transformer-based architectures with self-attention [2], which allow longer-range connections to be made across words in a sentence but require more data for training. A final trend is that new models are becoming bigger, counting ELMo [56] at 94 M, BERT [12] at 340 M, GPT-2 [13] at 1.5 billion and GPT-3 [32] at 175 billion parameters. The recent progress on this type of large-scale masked prediction has led to performance surpassing human baselines on language-understanding tasks. This can be seen in Figure 6, where we show the performance of selected top models from the leaderboard of the common SuperGLUE [80] benchmark.

All the techniques discussed previously belong to the masked prediction family of methods, and they have been the most successful and widely adopted. But there are examples of TP, such as recovering the order of permuted [81] or rotated [81] sentences. These have often been used as

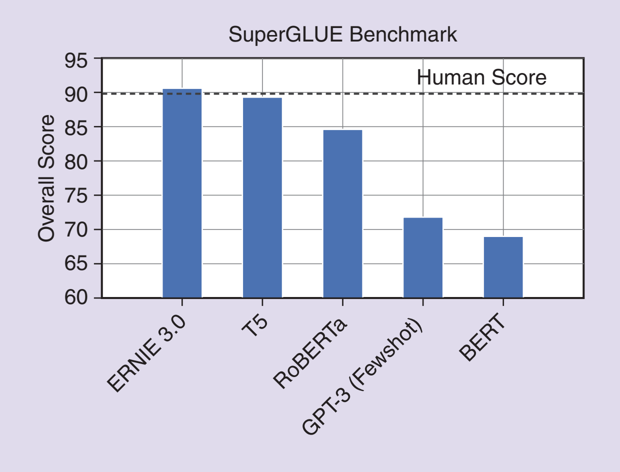

FIGURE 6. The performance of SSRL methods on the textual benchmark SuperGLUE, compared to a baseline of human performance. The selected techniques were taken from the official leaderboard at https://super.gluebenchmark.com/leaderboard.

complementary signals to improve downstream performance on a particular task.

Data sets

Self-supervision in language has been shown to benefit from ever-larger corpora of text. This has led to huge data sets being created, primarily by crawling the web for the data. The early word embeddings made heavy use of Wikipedia articles [64], or crawls of news sites and social media sites like Twitter. As models have become larger and require more text to train on, the organizations training these models have begun using private data sets, which are not publicly available [13], [32]. Attempts have been made at replicating the data used in such papers, for example the OpenWebText Corpus [65]. Another example is Common Crawl [66], a nonprofit project that makes data from billions of web pages freely accessible. Various data sets have been created from this data, and filtered versions of the entire corpus often form the bulk of training sets [32]. Using a combination of the aforementioned data sources, the total size of the training set used in state-of-the-art language modeling is now on the scale of 500 billion tokens [32].

A growing concern in language modeling is the extent to which biases implicit in the large training corpora for SSRL become baked into the resulting language models.

Applications

SSRL has made a major impact on a host of problems involving multiple languages, which introduces a new kind of source/target dichotomy besides the task- and domain-level dichotomies we have focused on thus far. In the simplest (within-language) scenario, SSRL can benefit all the standard language-understanding tasks (classification, QA, and so forth) for low-resource languages. One can pretrain SSRL models on large corpora of high-resource languages before fine-tuning them on smaller corpora of low-resource languages [25]. For cross-language tasks such as machine translation, one can pretrain SSRL models (e.g., for masked prediction) that are multilingual, in that they simultaneously encode/decode data from more than one language. These multilingual language models are then well primed for comparatively low-data fine-tuning for translation [25], or provide good representations to drive unsupervised [25] learning of translation models. This is valuable, as vanilla translation models are extremely expensive to supervise due to requiring a vast number of aligned (translated) sentence pairs across languages.

The conventional, task-specific fine-tuning, as outlined in the "Background" section, is the dominant paradigm for exploiting SSRL in language. However, a notable exception to this is in the recent GPT-3 [32] language model. A key observation in this work is that a sufficiently scaled-up 175 billion parameter generative language model can often perform few- or zero-shot learning of a new task in a purely feedforward manner (no backpropagation or fine-tuning) simply by prompting the model with a few training examples, the query, and allowing it to complete the answer.

Considerations

A growing concern in language modeling is the extent to which biases implicit in the large training corpora for SSRL become baked into the resulting language models, for example, sexist or racist stereotypes. Vast corpora must be used for SSRL, so training data cannot be manually filtered for appropriateness. A small but growing body of work aims to develop SSRL variants with reduced bias [82].

Audio and time series

The classic approaches to audio analysis tasks such as speech recognition compute mel-frequency cepstrum coefficients (MFCCs) from the raw audio data and then model the sequence via both Gaussian mixture and hidden Markov models. Meanwhile, contemporary NN approaches trained by supervised learning have dominated in settings where massive, annotated training data are available [83]. Against this backdrop, self-supervised methods have very recently made massive advances in alleviating this annotation bottleneck, enabling state-of-the-art audio analysis procedures to be trained with relatively sparse annotations.

Self-supervised methods in the audio analysis arena have exploited architectures h_θ spanning all the popular options for time-series data, including recurrent [84], convolutional [11], and self-attention [10] networks. These are usually applied directly to raw waveform data to build a representation without any preprocessing step, such as an MFCC.

Methods

In terms of self-supervision algorithms, numerous studies have successfully adapted the insights of self-attention-based language models [12] to audio data. As a pretext task, random segments of the input sequence are masked and predicted by a self-attention architecture. However, a key difference is that language models work with discrete token sequences, thus enabling the pretext to be formalized as a multiclass classification, while audio time series are naturally continuous. Thus, solutions to formalizing a masked prediction task for audio have either quantized the speech embedding for classification, including wav2vec-2.0 [10]; applied contrastive losses to differentiate the masked segment from alternative distractors, such as CPC [11]; or replaced classification-based predictions with regression layers to directly synthesize the masked frame, such as an APC [84]. Other approaches such as PASE [85] go beyond defining a single self-supervision pretext task to combine several losses, each predicting or classifying a different property of the input.

Data sets

Although it is not as strong as in the text modality, there is still a trend for newer models to train on ever-larger data sets. Small data sets historically used for model training are now reserved for downstream evaluation, with contemporary methods being pretrained on Librispeech [67], containing 960 h of speech from audiobook readings, and Libri-light [68], a much larger data set (60,000 h of similar audiobook recordings).

Applications

A notable success in speech was shown by wav2vec 2.0 [10], which used transformers plus masked prediction SSRL on 53,000 h of unlabeled data prior to fine-tuning a downstream speech recognition system. This was subsequently able to surpass prior state-of-the-art ASR performance with 10-fold less-supervised data than used before, and approach the state of the art with 100-fold less-supervised data than before. Albeit at the cost of 660-GPU days of SSRL compute, this is a dramatic improvement in data efficiency. In terms of representation learning on more general time-series data, masked prediction methods based on a transformer architecture have been shown to match supervised state of the art in a suite of benchmarks in diverse application areas [86].

A major application area for self-supervised time-series analysis is medical data, where annotations are hard to collect. There has been progress in applying SSRL to electroencephalogram (EEG) and electrocardiogram (ECG) data [34], [87]. For example, using TP SSRL prior to training ECG-based emotion recognition [34] and contrastive-instance discrimination SSRL prior to learning downstream, EEG-based, motor-movement classification and ECG-based anomaly detection [87]. In terms of time-series forecasting, transformers based on sequential masked prediction pretext have significantly outperformed conventional autoregressive models in predicting disease transmission [88].

Graphs

Graph-structured data are ubiquitous in the networked world and support a diverse array of tasks, including node, edge, and graph classification. These tasks should all be informed by both node/edge features where available, and graph connectivity. Graph NNs (GNNs) [89] have advanced all these tasks significantly, especially where massive-labeled data are available. Thus, a large body of work on self-supervised graph representation learning has emerged to facilitate downstream GNN-based tasks.

Graph-based SSRL can be somewhat unique in several aspects. Depending on whether the ultimate task of interest requires node- or graph-level predictions, methods may focus on learning node- [15], [27], [57] or graph-level [90] representations, or both [91]. Graph-based approaches also differ in whether they are oriented at training on a set of graphs [15], [90] (compare a set of images or audio clips in other modalities), or on a single large graph [14].

Methods

The early shallow methods for self-supervised graph representation learning used NLP-inspired masked prediction approaches to learn node embeddings, for example, based on random walks on the graph [14]. Much as shallow word embeddings have been eclipsed by deep language models in NLP, newer graph representation learning architectures that focus on graph convolutional networks or self-attention have driven progress in this modality.

In terms of self-supervision objectives, most of the work in this area falls into masked prediction and instance discrimination categories. Several recent techniques optimize mutual information-based, instance discrimination objectives, with DGI [15] and InfoGraph [90] performing contrastive-instance discrimination between pairs of nodes/patches and whole graphs. Masked prediction pretexts were used both by classic shallow approaches [14] as well as recent deep approaches such as GPT-GNN [27]. A minority of approaches have applied transformation prediction practices, such as GraphTER [24], where nodewise transformations are applied and predicted by a GNN.

An important dichotomy in graph-based representation learning is between transductive and inductive graph representation learning methods. The majority of schemes are transductive in that they learn embeddings specifically for nodes seen during, and so are primarily relevant in applications where the downstream task uses the same graph data as is used for pretraining. This is analogous to how the word2vec algorithm [22] in language learns embeddings for words in its training set, but cannot produce embeddings for unseen words. A minority of methods are inductive [27], [57] in that they learn embedding functions that do not depend on a specific choice of input graph, and thus can be transferred to new target nodes or graphs.

Data sets

As the graph-structured data occur so pervasively, they cover a wide range of data-type tasks. The major examples include social [57], citation [70], chemical [92], and biological networks [93]. Because there are many different kinds of graphs with different structures and sizes, there is no one-size-fits-all source data set, which consistently improves transfer, as in many of the previously discussed modalities. It instead depends on the tasks of interest.

For learning in the transductive setting, pretraining must necessarily be done on the same graph as the testing, thus limiting the task transfer, not domain-transfer. For the inductive setting, the source data can differ from the target, but in most evaluation cases, the test set consists of nodes that were hidden from the training graph [27], or unseen graphs from the same underlying data set [93]. Like in other modalities, we have seen increasingly large graphs being used for pretraining, like the Amazon Review Recommendation data [71] with 113 million nodes or the Open Academic Graph [70], which consists of more than 178 million nodes and 2 billion edges.

Applications

Self-supervised, graph-based representation learning is expected to benefit all graph-based prediction applications where data are limited. This is especially the case in computational chemistry and biology applications, where graphs and the associated annotations may correspond to molecules and corresponding molecular properties. In such applications, data are intrinsically hard to collect, but predicting graph properties can significantly impact tasks such as drug and material discovery [92], [93]. In computer vision, using lidar rather than RGB sensors leads to observations represented as point clouds or graphs, as opposed to conventional images. In this case, self-supervised graph representation learners such as GraphTER [24] have led to excellent performance in object segmentation (i.e., node classification) and classification (i.e., graph classification).

Discussion

Pretraining cost

The pretraining cost of different SSRL methods is not consistently documented, and hardware platform/GPU differences make them hard to compare quantitatively. Nevertheless, clearly, we can see that state-of-the-art techniques in computer vision, speech, and text (Tables 1 and 2) require massive resources on the order of 100 s of GPU days for training on ImageNet, Librispeech, and Wikipedia corpora, respectively. The general-purpose pretrained nature of these representations may amortize this cost somewhat by enabling many downstream problems to be solved with the same representation. This has largely been the case in the text modality where there has been strong success fine-tuning generic pretrained models to diverse tasks [12]. However, this may not be possible in other modalities such as graphs, which may require transductive training, or vision, where domain-specific pretraining may be necessary for data very different from ImageNet, such as hyperspectral imagery or volumetric magnetic resonance imaging. In this case, the pretraining cost poses an accessibility barrier to modestly resourced organizations, and an environmental issue [94] due to its energy requirement. Although there is also tremendous research activity in developing more efficient pretraining algorithms, the net cost of pretraining is trending upward due to the fact that bigger data sets and larger network architectures have systematically led to better performance.

Data requirement and curation

For text [32] and speech [10], the literature unambiguously shows that thus far, performance increases consistently with ever-larger data sets. In the case of text, this result further seems to be relatively insensitive to the degree of curation of the data.

For images, the majority of recent work still uses ImageNet, with its 1.28 million images as the source set [6], [7]. However, a number of studies have shown that using larger pretraining data sets [36], [59] benefits transfer performance [16], [36], with feature quality growing logarithmically with data volume [16]. For video pretraining, state-of-the-art models use the increasingly large YouTube-8M-2 [62] and HowTo100M [63] with combined video playtimes of 13 and 15 years, respectively.

The vision of SSRL is to enable representation learning on easily obtained, uncurated data. However, for benchmarking purposes (especially in vision and audio and graphs, but less so in text), methods are often actually trained on curated data while ignoring the labels. It is not clear how much existing algorithm design is overfitted to these curated data sets, and whether the relative performance of different approaches is maintained when real, uncurated data are used instead. For example, in computer vision, most of the pretraining is performed on ImageNet, which is large and diverse, yet uniformly focused on individual objects. If this was replaced with scene images of multiple cluttered objects, then typical instance discrimination tasks like mapping two different crops of one image to the same identity could create false-positive pretext label noise that maps different semantic objects to the same representation [95]. We are beginning to see new SSRL methods designed for data with different statistics, such as cluttered images [95].

Architecture choice and deployment costs

For both image [5], [16] and text [12] analysis, the trend has been that bigger architectures lead to better representation performance, especially when coupled with extremely large pretraining data sets, and challenging pretext tasks [16]. This is welcome from the perspective of near-"automatic" performance improvement as data sets and computation capabilities grow. However, it does pose a concern for deployment of the resulting models on resource-constrained or embedded devices with limited memory and/or computation capability, which may limit the benefit of this line of improvement for such applications.

The vision of SSRL is to enable representation learning on easily obtained, uncurated data.

A standard approach to alleviate this issue is to perform SSRL of large models as usual followed by using unlabeled data to perform posttraining distillation of the large, self-supervised model into a smaller, more compact but similarly performant student model. For example, in vision, this has been demonstrated to compress a ResNet-152 × 3 model to a ResNet-50 of similar performance [5]; in text, a 109-M parameter/22.5-gflop BERT model can be distilled to a 14.5-M/1.2-gflop BERT model with a similar performance [31].

Transferability

The vision of SSRL is to produce features that transfer to a wide range of downstream tasks. The extent to which this has been realized varies by discipline/modality. In vision this is on its way, with many studies evaluating transfer performance [16], [29], but no single benchmark has yet been widely agreed upon. Recognition has been the most common scene of transfer assessment, but recently, detection and dense prediction have also been embraced [6], [7], [29]. However, ImageNet Top-1 accuracy is still the main metric used in model comparisons. As reported by Ericsson et al. [29], this metric shows high correlation with downstream-recognition performance. Their results for detection and dense prediction, however, show markedly lower correlations, indicating that current SSRL methods are not optimized for such a broad transfer [29]. For practitioners with new data and tasks, this means that the best-performing SSRL model on ImageNet can be safely adapted to recognition tasks. However, if the task differs, then more models need to be considered. Additionally, if the images of the target domain are unstructured or exhibit different properties to ImageNet images, then further caution must be taken when choosing a pretrained model. This is further expanded on by Mac Aodha [73], who shows how SSRL models fail to compete with supervision on "in-the-wild" data sets containing plant and animal species, contrasting what has been found for curated data sets [29].

In video and multimodal settings, common transfer evaluation considers transfer from large source data sets, such as Kinetics [61], to standard target data sets such as UCF101. State-of-the-art procedures successfully leverage large source data sets and approaches but do not yet outperform supervised pretraining [78]. Nonetheless, there has been an uptake of SSRL methods in applications such as tracking [96] and detection.

In text, the field has matured more already. Here, several broad benchmarks, such as SuperGLUE [80], are regularly used to monitor progress. The main mode of transfer in NLP has long been to fit a linear model or fine-tune an SSRL model like BERT [12], and on many tasks on the above benchmarks, fine-tuned SSRL models achieve top results. The recent GPT-3 [32] has shown that huge SSRL models can achieve competitive performance via few-shot adaptation instead of full fine-tuning, especially on language modeling and QA. In summary, text models exhibit relatively high transferability, with SSRL pretraining dominating in a broad range of downstream tasks.

In speech and time series, the focus thus far has been narrow, with only a few tasks and data sets forming the evaluation landscape. These cover phoneme recognition, and occasionally, speaker identification or emotion classification. Most of the work focuses on English-language speech, both for pretraining and transfer. However, very rapid progress is currently being made in multilingual [97] speech models and cross-lingual transfer [98], so prospects for transferability seem promising.

The current state of the graph modality is that transferability is good to unseen nodes within the same graph and to unseen graphs within the same data set, e.g., protein-protein interactions [93]. However, there is little information to suggest transfer across graph types, like chemical-to-biological or citation-to-social, currently has any benefit.

Choosing the right pretext task

As we have seen, the four families of pretext tasks can be applied to all the different modalities. But because self-supervised pretexts rely on exploiting the structure of data, which in turn differs significantly across modalities, their efficacy can vary substantially across modalities. One such clear trend is that masked prediction is ubiquitous in the text modality [12], [22], [32], with other tasks being significantly less effective. And when other tasks are used, they are often complementary to a masked prediction loss [81]. In images, masked prediction and TP have been tried in various forms and drove initial progress, but the most recent advances in these modalities have been driven by instance discrimination [5], [78] and clustering [7], [9]. However, TP is still seeing success in videos, presumably because of the rich spatiotemporal information to be exploited. Finally, although there may be a dominant pretext strategy for a given modality, it is common that suitably designed combinations of pretexts applied in a multitask manner can improve performance compared to a single pretext [81].

Picking a pretext based on the bulk of successes for the modality of interest is a good start. However, to further inform choice, one can further consider the assumptions that underlie each family of methods. Masked prediction relies on context being enough to fill in the missing parts of a data point. TP relies on each data point possessing a canonical view. Instance discrimination relies on each data point representing a unique semantic example, distinguished from all other data points in the training set, which may not hold for cluttered images, as discussed previously. It is notable that clustering requires no strong assumptions other than the existence of meaningful similarities by which to group the data into a certain number of clusters. Therefore, if little is known of the structure of the data, then a method based on clustering may be a good start.

A final consideration when selecting a pretext task is, which properties do we want in our representations? If our data modality is images and we are interested in exploiting the orientation of objects in our data, do we want our representations to vary with orientation?; in which case, we might want to use a TP technique like that which is detailed in [33]. Or do we want all orientations of the input to produce the same output?; in which case, we might instead choose an instance discrimination method that uses rotation-based augmentation. This question of equivariance or invariance can greatly impact the downstream performance of certain tasks. For example, a visual object classification task might benefit from invariance to spatial translation, but a detection task would need this information to be preserved to correctly predict object locations.

If there are no specific downstream tasks in mind a priori and therefore no known required properties that must be learned, the ideal selection is not clear. In this case, we want to use the approach that best captures the core information in our data, which also has the best chance of being of use for later tasks. Finding such pretext tasks can be considered the main aim of the SSRL field of research.

Self-supervised versus semisupervised

In cases where the source and target data sets are the same or similar in content and label space, then both semisupervised and self-supervised approaches can potentially apply (see the "Background" section). As both families of methods are making rapid progress and there have been few direct comparisons, it is not yet clear if/when one family should be preferred. However, because SSRL deals with initialization and Secure

Sockets Layer (SSL) pertains to refinement, the two strategies can, in principle, both be applied to one learning problem. There has yet been very little investigation into the extent to which these strategies can be complementary and further boost performance when used together; a preliminary result in computer vision suggests that they cannot [99]. However, preliminary results in text [100] indicate that SSL and SSRL can be synergistic when used together.

Other benefits of SSRL

Although we have mainly focused on the benefits of SSRL with respect to accuracy in the low- and few-shot data regime, there are several other potential benefits: 1) The computational cost of fine-tuning a self-supervised model tends to be lower than training from scratch (although it is comparable to fine-tuning a supervised pretrained feature). 2) If the supervised target task suffers from label noise, training leads to a much worse performance compared to using clean labels. However, SSRL also increases resilience to such label noise [28], which often occurs in practice. 3) Given a trained system, SSRL can also improve the robustness of image recognition to adversarial attacks as well as common corruptions such as blur, noise, and compression artifacts [28]. 4) Furthermore, SSRL leads to better calibrated probabilities [28], [29], which can be used to drive abstention of automated predictions or out-of-distribution detection [28]. 5) Finally, in terms of model interpretability, feature extractors trained by self-supervision tend to lead to more reasonable and interpretable attention maps [29].

Recommendations for future work

The following few areas are recommended for future research:

- Develop wider benchmarks. Several of the modalities we look at have a few standard downstream tasks against which they are consistently evaluated. This creates a bias toward making new methods that optimize only for those particular tasks. Instead, we should create benchmark suites that study the performance of pretrained models across a wide range of tasks within a modality. This has been done successfully in NLP and has driven progress, making sure that it benefits many areas of the field [80], but such standardized benchmarks are lacking in the other modalities we have considered.
- Focus not only on tracking task performances in these benchmarks but also on other feature properties, like social biases, to obtain a broader understanding of how these models behave. Progress on reducing such biases can only really be done if we know about and can quantify them.
- Be wary of relying on only scale to improve performance. As we use ever-larger data sets to train these models, increasingly, we know less about the data themselves as there is very little human oversight in the data collection process. By developing methods that are more data efficient, i.e., don't need billions of instances to learn, we can create models that are easier to understand and control. Additionally, as we develop larger models, their carbon footprint grows significantly [94]. We must make sure that the efficiency of training these models is tracked in common benchmarks.
- Do not get stuck on training on only one specific source data set, as this will bias the type of methods that are created. As an example, the highly curated and single-centered object style of ImageNet has led to a particular style of data augmentation and instance discrimination. However, it has been shown that on less-curated, in-the-wild images, these procedures underperform. By continuously considering different types of source data sets, we get a better picture of when and where a method works.

Acknowledgments

This research was partially supported by Engineering and Physical Sciences Research Council (EPSRC) grant number EP/S000631/1 and the MOD University Defense Research Collaboration in Signal Processing; EPSRC Center for Doctoral Training in Data Science, funded by EPSRC grant EP/L016427/1 and the University of Edinburgh; and EPSRC grant EP/R026173/1. Linus Ericsson is the corresponding author.

Authors

Linus Ericsson (linus.ericsson@ed.ac.uk) received his MEng degree from Durham University and his MScR degree from the University of Edinburgh, Edinburgh, EH8 9AB, U.K., where he is currently pursuing his Ph.D. degree focusing on self-supervised representation learning. His recent work at the 2021 Conference on Computer Vision and Pattern Recognition examined the transfer performance of state-of-the-art self-supervised learners. His current research focuses on benchmarking and developing algorithms for self-supervised representation learning, with a particular focus on visual domains.

Henry Gouk (henry.gouk@ed.ac.uk) received his Ph.D. degree from the University of Waikato, New Zealand, where he worked on developing explicit inductive biases for representation learning. He is a postdoctoral research associate in the School of Informatics at the University of Edinburgh, Edinburgh, EH8 9AB, U.K. His current research interests focus on extending the scope of statistical learning theory beyond conventional supervised learning problems. In addition, he also works in the area of designing representation learning methods that can exploit relationships between different tasks and learn using imperfect data, such as a lack of labels.

Chen Change Loy (ccloy@ntu.edu.sg) received his Ph.D. degree in computer science from the Queen Mary University of London. He is an associate professor at Nanyang Technological University, 639798, Singapore, where he directs the MMLab@NTU. He is also an adjunct associate professor at the Chinese University of Hong Kong, Hong Kong, China. He is recognized as one of the 100 most influential scholars in computer vision by ArnetMiner and is particularly known for his major contributions to image superresolution, where his article in *IEEE Transactions on*

Pattern Analysis and Machine Intelligence on this topic remains one of the top-five most popular articles in the journal to date. He serves on the editorial board of *IEEE Transactions on Pattern Analysis and Machine Intelligence* and *International Journal of Computer Vision*.

Timothy M. Hospedales (t.hospedales@ed.ac.uk) received his Ph.D. degree in neuroinformatics from the University of Edinburgh. He is a full professor of artificial intelligence at the University of Edinburgh, Edinburgh, EH8 9AB, U.K., and a principal researcher at Samsung AI Centre Cambridge, Cambridge, CB1 2JH, U.K., where he directs the Machine Learning and Data Intelligence Programme. He has worked extensively on methods for learning with limited data, including unsupervised, self-supervised, weakly supervised, and meta learning, with applications in vision, language, reinforcement learning, and beyond. He has published numerous papers on these topics at major venues, including the Conference on Computer Vision and Pattern Recognition, International Conference on Computer Vision, European Conference on Computer Vision, International Conference on Machine Learning, Conference on Neural Information Processing Systems, International Conference on Learning Representations, Association for the Advancement of Artificial Intelligence, International Joint Conference on Artificial Intelligence, and Empirical Methods in Natural Language Processing.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. NIPS'17*, 2017, pp. 6000–6010.

[3] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *Proc. IEEE Int. Conf. Comput. Vision*, 2017, pp. 843–852, doi: 10.1109/ICCV.2017.97.

[4] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 11, pp. 4037–4058, 2020, doi: 10.1109/TPAMI.2020.2992393.

[5] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, "Big Self-Supervised models are strong semi-supervised learners," 2020, arXiv:2006.10029.

[6] J.-B. Grill *et al.*, "Bootstrap your own latent: A new approach to self-supervised learning," 2020, arXiv:2006.07733.

[7] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," 2020, arXiv:2006.09882.

[8] D. Xu, J. Xiao, Z. Zhao, J. Shao, D. Xie, and Y. Zhuang, "Self-supervised spatiotemporal learning via video clip order prediction," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 10,326–10,335, doi: 10.1109/CVPR.2019.01058.

[9] H. Alwassel, D. Mahajan, B. Korbar, L. Torresani, B. Ghanem, and D. Tran, "Self-supervised learning by cross-modal audio-video clustering," 2020, arXiv:1911.12667.

[10] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," 2020, arXiv:2006.11477.

[11] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, arXiv:2006.11477.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding,"2019, arXiv:1810.04805v2.

[13] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," San Francisco, CA, USA, Tech. Rep., 2019. [Online]. Available: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

[14] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proc. KDD*, 2016, pp. 855–864, doi: 10.1145/2939672.2939754.

[15] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengioy, and R. D. Hjelm, "Deep graph infomax," 2019, arXiv:1809.10341.

[16] P. Goyal, D. Mahajan, A. Gupta, and I. Misra, "Scaling and benchmarking self-supervised visual representation learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vision*, 2019, pp. 6390–6399, doi: 10.1109/ICCV.2019.00649.

[17] M.-I. Georgescu, A. Barbalau, R. T. Ionescu, F. S. Khan, M. Popescu, and M. Shah, "Anomaly detection in video via self-supervised and multi-task learning," 2021, arXiv:2011.07491.

[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[19] H. Gouk, T. M. Hospedales, and M. Pontil, "Distance-based regularisation of deep networks for fine-tuning," 2021, arXiv:2002.08253.

[20] J. E. van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Mach. Learn.*, vol. 109, no. 2, pp. 373–440, 2020, doi: 10.1007/s10994-019-05855-6.

[21] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Berlin, Heidelberg, Germany: Springer Science & Business Media, 2009.

[22] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst. NeurIPS*, 2013, pp. 3111–3119.

[23] A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with exemplar convolutional neural networks," 2014, arXiv:1406.6909.

[24] X. Gao, W. Hu, and G.-J. Qi, "GraphTER: Unsupervised learning of graph transformation equivariant representations via auto-encoding node-wise transformations," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2020, pp. 7161–7170, doi: 10.1109/CVPR42600.2020.00719.

[25] A. Conneau and G. Lample, "Cross-lingual language model pretraining," 2019, arXiv:2109.11129.

[26] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," 2016, arXiv:1604.07379v2.

[27] Z. Hu, Y. Dong, K. Wang, K.-W. Chang, and Y. Sun, "GPT-GNN: Generative pre-training of graph neural networks," 2020, arXiv:2006.15437v1.

[28] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, "Using self-supervised learning can improve model robustness and uncertainty," 2019, arXiv:1906.12340.

[29] L. Ericsson, H. Gouk, and T. M. Hospedales, "How well do self-supervised models transfer?" 2021, arXiv:2011.13377.

[30] A. Kolesnikov, X. Zhai, and L. Beyer, "Revisiting self-supervised visual representation learning," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 1920–1929, doi: 10.1109/CVPR.2019.00202.

[31] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "TinyBERT: Distilling BERT for natural language understanding," 2020, arXiv:1909.10351v5.

[32] T. B. Brown *et al.*, "Language models are few-shot learners," 2020, arXiv:2005.14165.

[33] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," 2018, arXiv:1803.07728.

[34] P. Sarkar and A. Etemad, "Self-supervised learning for ECG-based emotion recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, 2020, pp. 3217–3221, doi: 10.1109/ICASSP40776.2020.9053985.

[35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2009, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.

[36] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten, "Exploring the limits of weakly supervised pretraining," 2018, arXiv:1805.00932.

[37] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, 2006, pp. 1735–1742, doi: 10.1109/CVPR.2006.100.

[38] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proc. Mach. Learn. Res.*, 2010, pp. 297–304.

[39] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, "Barlow twins: Self-supervised learning via redundancy reduction," 2021, arXiv:2103.03230.

[40] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2020, pp. 9726–9735, doi: 10.1109/CVPR42600.2020.00975.

[41] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *Proc. Eur. Conf. Comput. Vision*, 2020, pp. 776–794.

[42] A. Owens and A. A. Efros, "Audio-visual scene analysis with self-supervised multisensory features," 2018, arXiv:1804.03641.

[43] A. Radford *et al.*, "Learning transferable visual models from natural language supervision," 2021, arXiv:2103.00020.

[44] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola, “What makes for good views for contrastive learning?” 2020, arXiv:2005.10243.

[45] C.-Y. Chuang, J. Robinson, Y.-C. Lin, A. Torralba, and S. Jegelka, “Debiased contrastive learning,” 2020, arXiv:2007.00224.

[46] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, “Deep clustering for unsupervised learning of visual features,” in *Proc. Eur. Conf. Comput. Vision,* 2018, pp. 132–149.

[47] X. Zhan, J. Xie, Z. Liu, Y. S. Ong, and C. C. Loy, “Online deep clustering for unsupervised representation learning,” in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2020, pp. 6687–6696, doi: 10.1109/CVPR42600.2020.00672.

[48] N. Saunshi, O. Plevrakis, S. Arora, M. Khodak, and H. Khandeparkar, “A theoretical analysis of contrastive unsupervised representation learning,” in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 5628–5637.

[49] N. Saunshi, S. Malladi, and S. Arora, “A mathematical exploration of why language models help solve downstream tasks,” 2021, arXiv:2010.03648.

[50] J. D. Lee, Q. Lei, N. Saunshi, and J. Zhuo, “Predicting what you already know helps: Provable self-supervised learning,” 2020, arXiv:2008.01064.

[51] U. von Luxburg, “Clustering stability: An overview,” *Found. Trends Mach. Learn.*, vol. 2, no. 3, pp. 235–274, 2010, doi: 10.1561/2200000008.

[52] M. Chen, A. Radford, J. Wu, H. Jun, P. Dhariwal, D. Luan, and I. Sutskever, “Generative pretraining from pixels,” in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 1691–1703.

[53] G. Larsson, M. Maire, and G. Shakhnarovich, “Learning representations for automatic colorization,” in *Proc. Eur. Conf. Comput. Vis.,* 2016, pp. 577–593, doi: 10.1007/978-3-319-46493-0_35.

[54] D. Luo, C. Liu, Y. Zhou, D. Yang, C. Ma, Q. Ye, and W. Wang, “Video cloze procedure for self-supervised spatio-temporal learning,” in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 7, pp. 11,701–11,708, doi: 10.1609/aaai.v34i07.6840.

[55] J. Lu, D. Batra, D. Parikh, and S. Lee, “Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks,” in *Proc. Neural Inf. Process. Syst.*, 2019, pp. 13–23.

[56] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2018, pp. 2227–2237.

[57] W. L. Hamilton, R. Ying, and J. Leskovec, “Inductive representation learning on large graphs,” in *Proc. Neural Inf. Process. Syst.*, 2017, pp. 1025–1035.

[58] A. Dosovitskiy *et al*., “An image is worth 16x16 words: Transformers for image recognition at scale,” in *Proc. Int. Conf. Learn. Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy

[59] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li, “YFCC100M: The new data in multimedia research,” *Commun. ACM*, vol. 59, no. 2, pp. 64–73, 2015, doi: 10.1145/2812802.

[60] G. Van Horn *et al*., “The iNaturalist species classification and detection dataset,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8769–8778.

[61] W. Kay *et al*., “The kinetics human action video dataset,” 2017, arXiv:1705.06950.

[62] “*YouTube-8M: A Large and Diverse Labeled Video Dataset for Video Understanding Research*,” Accessed: Jan., 2021. [Online Video]. Available: http://research.google.com/youtube8m/

[63] A. Miech, D. Zhukov, J.-B. Alayrac, M. Tapaswi, I. Laptev, and J. Sivic, “HowTo100M: Learning a text-video embedding by watching hundred million narrated video clips,” in *Proc. Int. Conf. Comput. Vision,* 2019, pp. 2630–2640.

[64] S. Merity, C. Xiong, J. Bradbury, and R. Socher, “Pointer sentinel mixture models,” 2017, arXiv:1609.07843.

[65] A. Gokaslan and V. Cohen, “OpenWebText corpus,” 2019. [Online]. Available: https://skylion007.github.io/OpenWebTextCorpus/

[66] C. Buck, K. Heafield, and B. Van Ooyen, “N-gram counts and language models from the common crawl,” in *Proc. Int. Conf. Lang. Resources Eval.*, 2014, p. 4.

[67] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, “Librispeech: An ASR corpus based on public domain audio books,” in *Proc. Int. Conf. Acoust., Speech Signal Process.*, 2015, pp. 5206–5210.

[68] J. Kahn *et al*., “Libri-light: A benchmark for ASR with limited or no supervision,” in *Proc. Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 7669–7673.

[69] J. F. Gemmeke, D. P. W. 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 *Proc. Int. Conf. Acoust., Speech Signal Process.*, 2017, pp. 776–780.

[70] F. Zhang *et al*., “OAG: Toward linking large-scale heterogeneous entity graphs,” in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2019, pp. 2585–2595.

[71] J. Ni, J. Li, and J. McAuley, “Justifying recommendations using distantly-labeled reviews and fine-grained aspects,” in *Proc. 2019 Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process.*, pp. 188–197.

[72] K. M. Borgwardt, C. S. Ong, S. Schonauer, S. V. N. Vishwanathan, A. J. Smola, and H.-P. Kriegel, “Protein function prediction via graph kernels,” *Bioinformatics*, vol. 21, no. Suppl 1, pp. i47–i56, 2005, doi: 10.1093/bioinformatics/bti1007.

[73] O. Mac Aodha, “Benchmarking representation learning on natural world image collections,” in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit.*, 2021, pp. 12,884–12,893.

[74] C. Tao, J. Qi, W. Lu, H. Wang, and H. Li, “Remote sensing image scene classification with self-supervised paradi under limited labeled samples,” *IEEE Geosci. Remote Sens. Lett.*, early access*,* 2020, doi: 10.1109/LGRS.2020.3038420.

[75] L. Chen, P. Bentley, K. Mori, K. Misawa, M. Fujiwara, and D. Rueckert, “Self-supervised learning for medical image analysis using image context restoration,” *Med. Image Anal.*, vol. 58, p. 101,539, Dec. 2019. doi: 10.1016/j.media.2019.101539.

[76] A. Taleb, W. Loetzsch, N. Danz, J. Severin, T. Gaertner, B. Bergner, and C. Lippert, “3d self-supervised methods for medical imaging,” 2020, arXiv:2006.03829.

[77] I. Golan and R. El-Yaniv, “Deep anomaly detection using geometric transformations,” 2018, arXiv:1805.10917.

[78] T. Han, W. Xie, and A. Zisserman, “Self-supervised co-training for video representation learning,” 2020, arXiv:2010.09709.

[79] J.-B. Alayrac *et al*., “Self-supervised multimodal versatile networks,” in *Proc. Conf. Neural Inf. Process. Syst.*, 2020, p. 7.

[80] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, “SuperGLUE: A stickier benchmark for general-purpose language understanding systems,” 2019, arXiv:1905.00537.

[81] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, “BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension,” 2020, arXiv:1910.13461.

[82] P.-S. Huang *et al*., “Reducing sentiment bias in language models via counterfactual evaluation,” 2019, arXiv:1911.03064.

[83] D. Amodei *et al*., “Deep speech 2: End-to-end speech recognition in English and Mandarin,” in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 173–182.

[84] Y.-A. Chung, W.-N. Hsu, H. Tang, and J. Glass, “An unsupervised autoregressive model for speech representation learning,” 2019, arXiv:1904.03240.

[85] S. Pascual, M. Ravanelli, J. Serrà, A. Bonafonte, and Y. Bengio, “Learning problem-agnostic speech representations from multiple self-supervised tasks,” 2019, arXiv:1904.03416.

[86] J.-Y. Franceschi, A. Dieuleveut, and M. Jaggi, “Unsupervised scalable representation learning for multivariate time series,” 2019, arXiv:1901.10738.

[87] J. Y. Cheng, H. Goh, K. Dogrusoz, O. Tuzel, and E. A. Apple, “Subject-aware contrastive learning for biosignals,” 2020, arXiv:2007.04871.

[88] N. Wu, B. Green, X. Ben, and S. O’Banion, “Deep transformer models for time series forecasting: The influenza prevalence case,” 2020, arXiv:2001.08317.

[89] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” 2017, arXiv:1609.02907.

[90] F.-Y. Sun, J. Hoffmann, V. Verma, and J. Tang, “InfoGraph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization,” 2020, arXiv:1908.01000.

[91] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. Pande, and J. Leskovec, “Strategies for pre-training graph neural networks,” 2020, arXiv:1905.12265.

[92] Z. Wu, B. Ramsundar, E. N. Feinberg, J. Gomes, C. Geniesse, A. S. Pappu, K. Leswing, and V. Pande, “MoleculeNet: A benchmark for molecular machine learning,” *Chem. Sci.*, vol. 9, no. 2, pp. 513–530, 2018, doi: 10.1039/C7SC02664A.

[93] M. Zitnik and J. Leskovec, “Predicting multicellular function through multi-layer tissue networks,” *Bioinformatics*, vol. 33, no. 14, pp. i190–i198, 2017, doi: 10.1093/bioinformatics/btx252.

[94] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, “Green AI,” *Commun. ACM*, vol. 63, no. 12, pp. 54–63, 2020, doi: 10.1145/3381831.

[95] S. Purushwalkam and A. Gupta, “Demystifying contrastive self-supervised learning: invariances, augmentations and dataset biases,” 2020, arXiv:2007.13916.

[96] X. Wang, A. Jabri, and A. A. Efros, “Learning correspondence from the cycle-consistency of time,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2566–2576.

[97] A. Kannan *et al*., “Large-scale multilingual speech recognition with a streaming end-to-end model,” 2019, arXiv:1909.05330.

[98] M. Rivière, A. Joulin, P.-E. Mazaré, and E. Dupoux, “Unsupervised pretraining transfers well across languages,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 7414–7418.

[99] B. Zoph, G. Ghiasi, T.-Y. Lin, Y. Cui, H. Liu, E. D. Cubuk, and Q. Le, “Rethinking pre-training and self-training,” 2020, arXiv:2006.06882.

[100] J. Du, E. Grave, B. Gunel, V. Chaudhary, O. Celebi, M. Auli, V. Stoyanov, and A. Conneau, “Self-training improves pre-training for natural language understanding,” 2021, arXiv:2010.02194.