UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

> Reinventa el mundo <

Deep Learning

Week 10 : Mid-Term Review + Sparse Coding + Self-Supervised Learning

TRANSFORMATEC

Themes

50 % : GANs + VAEs

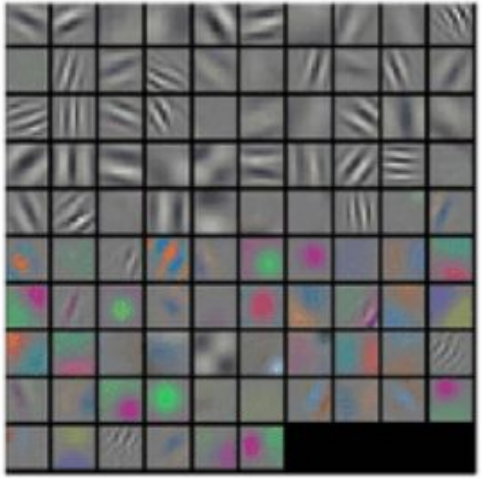
30 % : Fully Connected + Convolutional Neural Networks

20 % : NeuroAI + Sparse Coding + Self-Supervised Learning + Miscellaneous topics

Sample Problems (P1)

Fully Connected + CNNs

What determines the shape of the learned filter when training a neural network?

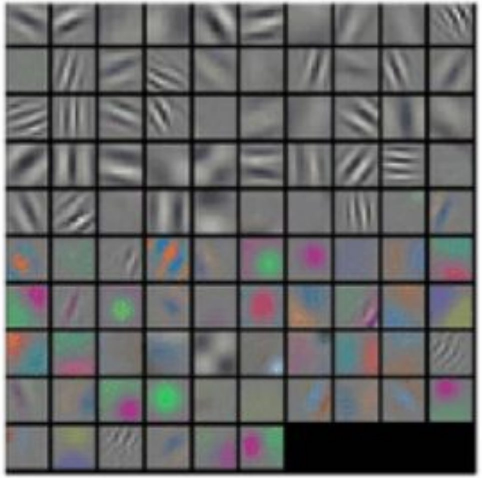


- ❏ Architecture
- ❏ Dataset at Training
- ❏ Dataset at Testing
- ❏ Regularization (eg. DropOut)
- ❏ Optimization Procedure (Learning Rule)
- ❏ Loss Function
- ❏ RAM available in CUDA Device
- ❏ Number of Epochs at Training

Sample Problems (P1)

Fully Connected + CNNs

What determines the shape of the learned filter when training a neural network?

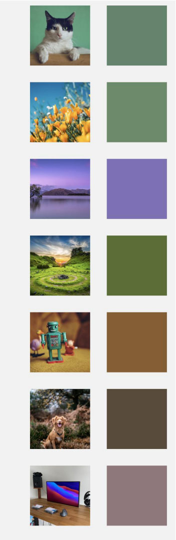


- ❑ Architecture
- ❑ Dataset at Training
- ❑ Dataset at Testing
- ❑ Regularization (eg. DropOut)
- ❑ Optimization Procedure (Learning Rule)
- ❑ Loss Function
- ❑ RAM available in CUDA Device
- ❑ Number of Epochs at Training

Sample Problems (P2)

Fully Connected + CNNs

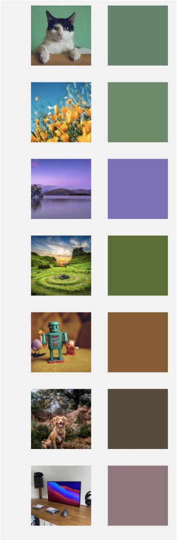
Write the Closed Form Expression function to compute the Average Color of an Image. What are the ideal weights of a Neural Network?

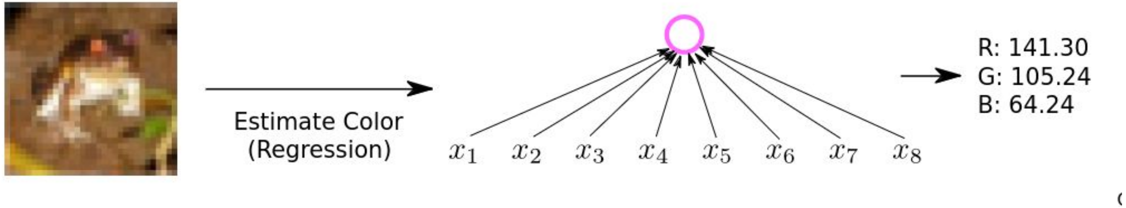


Sample Problems (P2)

Fully Connected + CNNs

Write the Closed Form Expression function to compute the Average Color of an Image. What are the ideal weights of a Neural Network?





Sample Problems (P3)

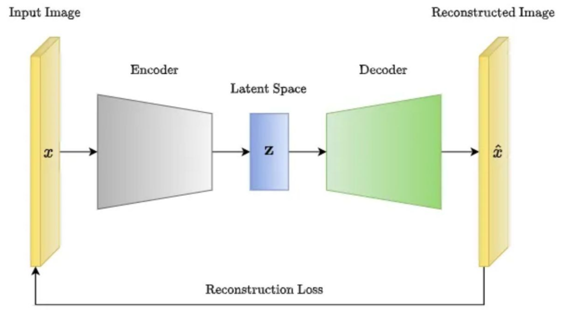
AutoEncoders + VAEs

Name 2 differences between the regular Auto-Encoder + Variational Auto-Encoder

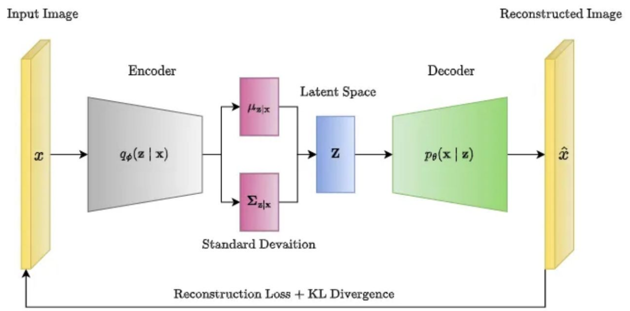
Sample Problems (P3)

AutoEncoders + VAEs

Name 2 differences between the regular Auto-Encoder + Variational Auto-Encoder



1. Deterministic vs Stochastic
2. Single-Term vs Dual-Term Loss



$$Loss_{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \hat{x}_i\right)^2$$

$$Loss_{BCE} = -\frac{1}{N}\sum_{n=1}^{N}\left[x_n \log \hat{x}_n + (1 - x_n)\log(1 - \hat{x}_n)\right]$$

$$\log p_\theta\left(x^{(i)}\right) \geq \mathcal{L}\left(x^{(i)}, \theta, \phi\right) = \underbrace{\mathbb{E}_z\left[\log p_\theta\left(x^{(i)} \mid z\right)\right]}_{\text{Reconstruct the Input Data}} - \underbrace{D_{KL}\left(q_\phi\left(z \mid x^{(i)}\right) \| p_\theta(z)\right)}_{\text{KL Divergence}}$$

$$\theta^*, \phi^* = \arg\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}\left(x^{(i)}, \theta, \phi\right)$$

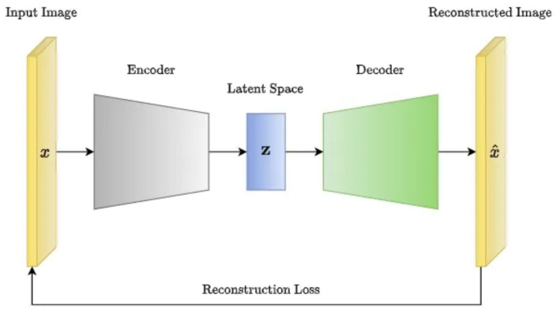
Sample Problems (P3)

AutoEncoders + VAEs

L2 - Loss

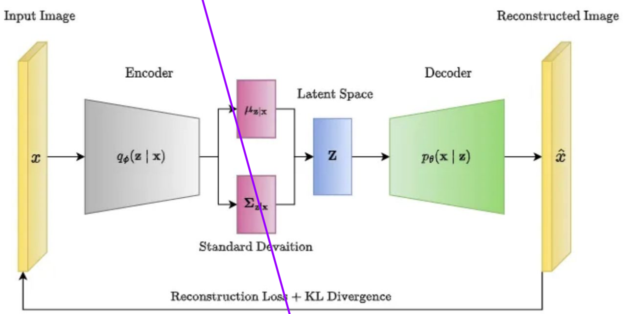
KL Divergence of Latent Code with Standarized Multi-Dimensional Normal

Name 2 differences between the regular Auto-Encoder + Variational Auto-Encoder



$$Loss_{MSE} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$$

$$Loss_{BCE} = -\frac{1}{N}\sum_{n=1}^{N}\left[x_n \log \hat{x}_n + (1 - x_n)\log(1 - \hat{x}_n)\right]$$



$$\log p_\theta\left(x^{(i)}\right) \geq \mathcal{L}\left(x^{(i)}, \theta, \phi\right) = \boxed{\underbrace{\mathbb{E}_z\left[\log p_\theta\left(x^{(i)} \mid z\right)\right]}_{\text{Reconstruct the Input Data}}} - \boxed{\underbrace{D_{KL}\left(q_\phi\left(z \mid x^{(i)}\right) \| p_\theta(z)\right)}_{\text{KL Divergence}}}$$

$$\theta^*, \phi^* = \arg\max_{\theta,\phi} \sum_{i=1}^{N} \mathcal{L}\left(x^{(i)}, \theta, \phi\right)$$

Recommended Reading : [Link here]

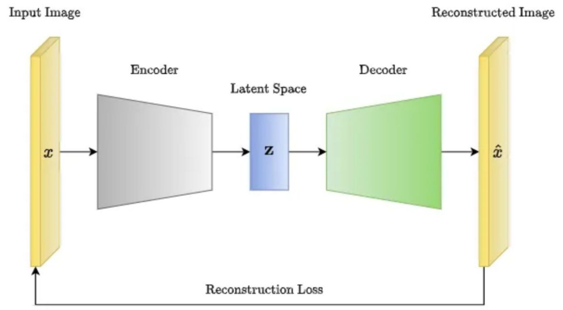
Sample Problems (P3)

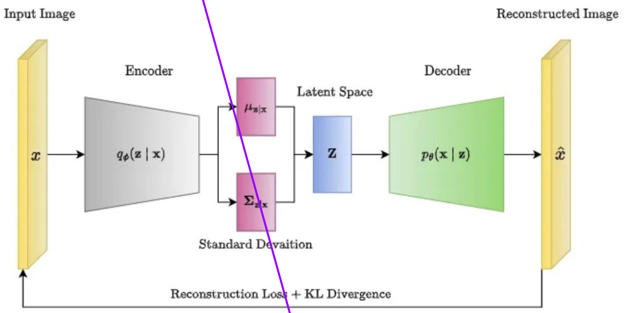
AutoEncoders + VAEs

L2 - Loss

KL Divergence of Latent Code with
Standarized Multi-Dimensional Normal

Name 2 differences between the regular Auto-Encoder + Variational Auto-Encoder





$$Loss_{MSE} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{x_i})^2$$

$$Loss_{BCE} = -\frac{1}{N} \sum_{n=1}^{N} \left[x_n \log \hat{x}_n + (1 - x_n) \log(1 - \hat{x}_n) \right]$$

$$\log p_\theta\left(x^{(i)}\right) \geq \mathcal{L}\left(x^{(i)}, \theta, \phi\right) = \boxed{\underbrace{\mathbb{E}_z\left[\log p_\theta\left(x^{(i)} \mid z\right)\right]}_{\text{Reconstruct the Input Data}}} - \boxed{\underbrace{D_{KL}\left(q_\phi\left(z \mid x^{(i)}\right) \| p_\theta(z)\right)}_{\text{KL Divergence}}}$$

$$\theta^*, \phi^* = \arg\max_{\theta, \phi} \sum_{i=1}^{N} \mathcal{L}\left(x^{(i)}, \theta, \phi\right)$$

Sample Problems (P4)

GANs

Write 3 reasons why optimizing GAN's are difficult?

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Sample Problems (P4)

GANs

Write 3 reasons why optimizing GAN's are difficult?

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{z}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Min-Max Optimization Problem, where the Generator and the Discriminator are competing. Guarantees of convergence are tricky.

Highly susceptible to initialization of Generator and Discriminator

Playing with the Architecture (of G & D), Batch Size and other hyper-parameters are usually necessary to train GANs

Sample Problems (P5)

GANs + cGANs

In a Conditional GAN, the “Condition” is put in the :

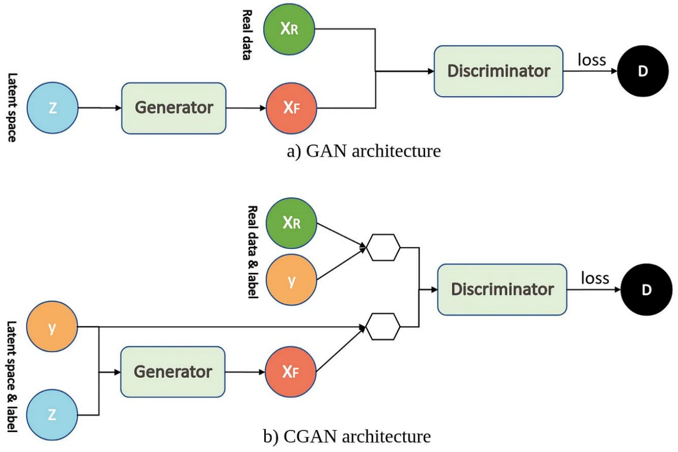
- ❏ Input of Discriminator
- ❏ Output of Discriminator
- ❏ Input of Generator
- ❏ Output of Generator

Sample Problems (P5)

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))]$$

In a Conditional GAN, the “Condition” is put in the :

- ❑ Input of Discriminator
- ❑ Output of Discriminator
- ❑ Input of Generator
- ❑ Output of Generator



Sample Problems (P6)

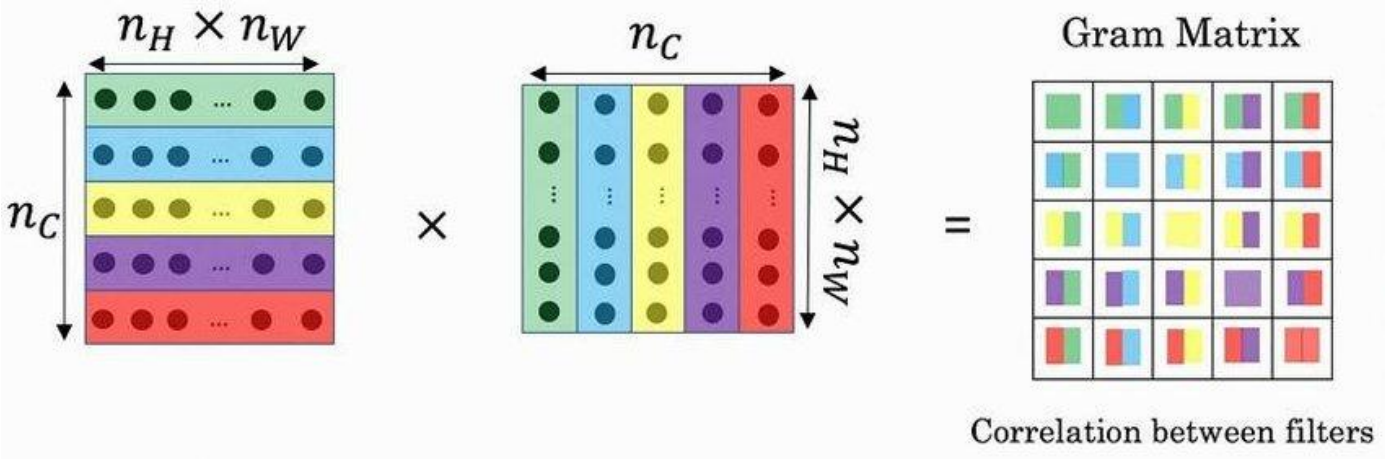
NeuroAI

How do you compute the Gramian Matrix in a Representational Similarity Analysis (RSA) ?

Sample Problems (P6)

NeuroAI

How do you compute the Gramian Matrix in a Representational Similarity Analysis (RSA) ?

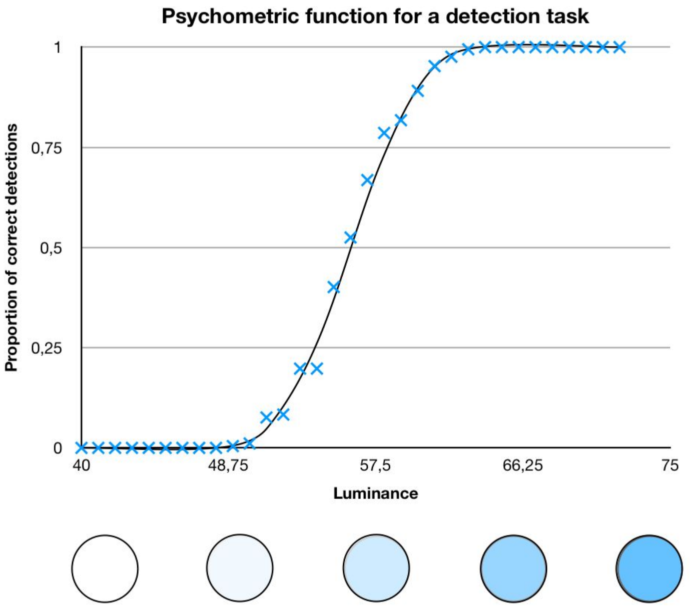


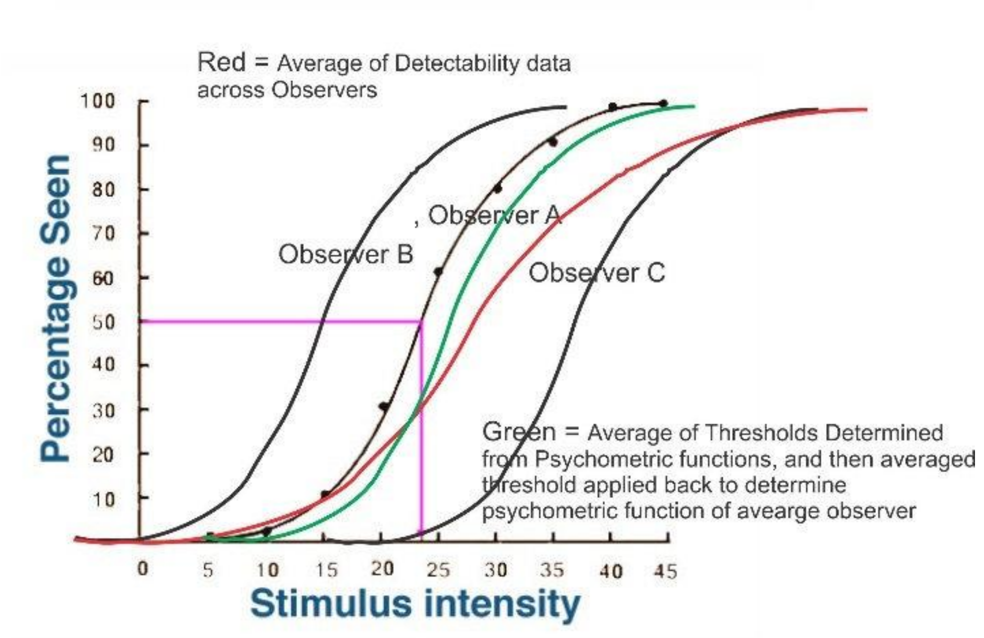
Sample Problems (P7)

Draw an Example of a Psychometric Function

Sample Problems (P7)

Draw an Example of a Psychometric Function





Sample Problems (P8)

Explain the Pipeline of the YOLO Image Classification Model

Sample Problems (P8)

Explain the Pipeline of the YOLO Image Classification Model

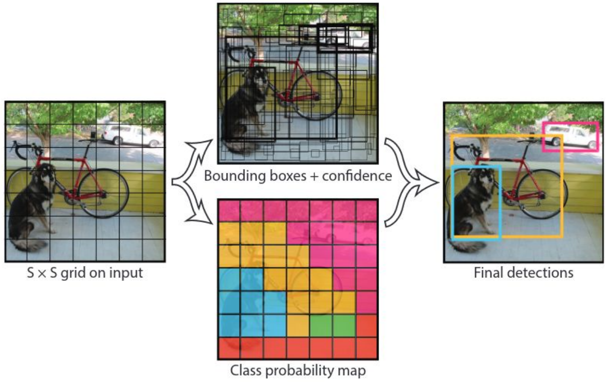


Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

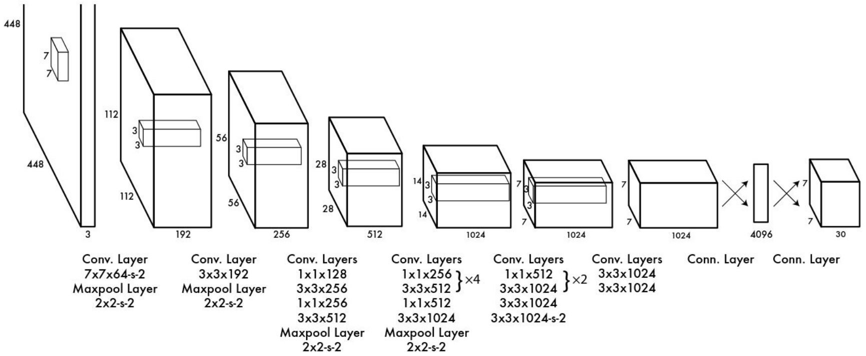


Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

Sparse Coding Fundamentals

$$x = Weights * factors + bias + noise$$

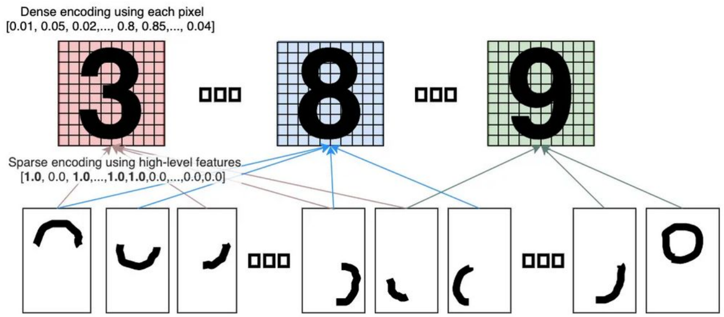
`x` — vector of size `u`

`Weights` — matrix of size (`k` , *u), estimated model parameter*

`factors` — vector of size k, random variables

`bias` — vector of size u, estimated model parameter, usually standard Gaussian

`noise` — vector of size u, random variables, zero-centered



Sparse Coding Fundamentals

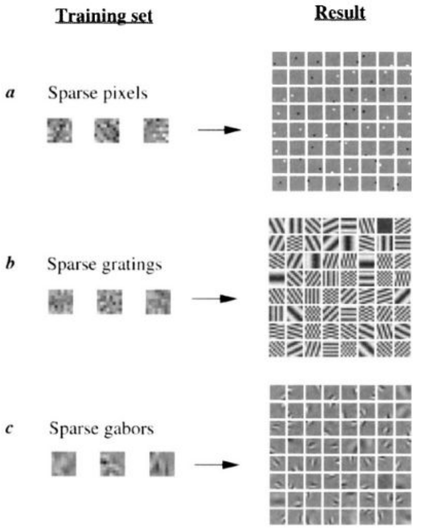
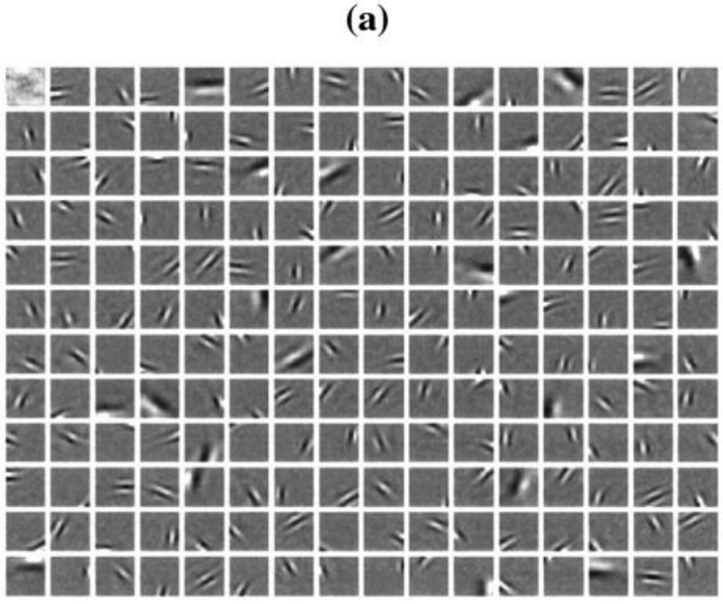
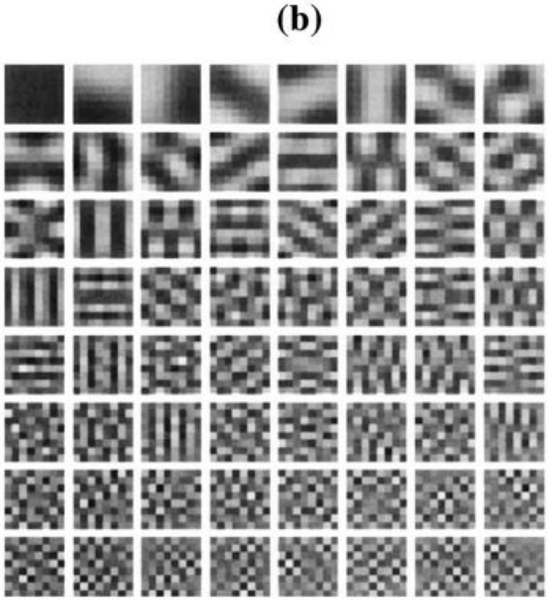


FIG. 3 Test cases. Representative training images are shown at the left and the resulting basis functions that were learned from these examples are shown at the right. In *a*, images were composed of sparse pixels: each pixel was activated independently according to an exponential distribution, $P(x) = e^{-|x|}/Z$. In *b*, images were composed similarly to *a*, except with gratings instead of pixels (that is, 'sparse pixels' in the Fourier domain). In *c*, images were composed of sparse, non-orthogonal Gabor functions with the method described by Field[12]. In all cases, the basis functions were initialized to random initial conditions. The learned basis functions successfully recover the sparse components from which the images were composed. The form of the sparseness cost function was $S(x) = -e^{-x^2}$, but other choices (see text) yield the same results.

NATURE · VOL 381 · 13 JUNE 1996



Sparse codes



PCA components

Sparse Coding Fundamentals

D. Drix, V.V. Hafner and M. Schmuker / Neural Networks 131 (2020) 37–49

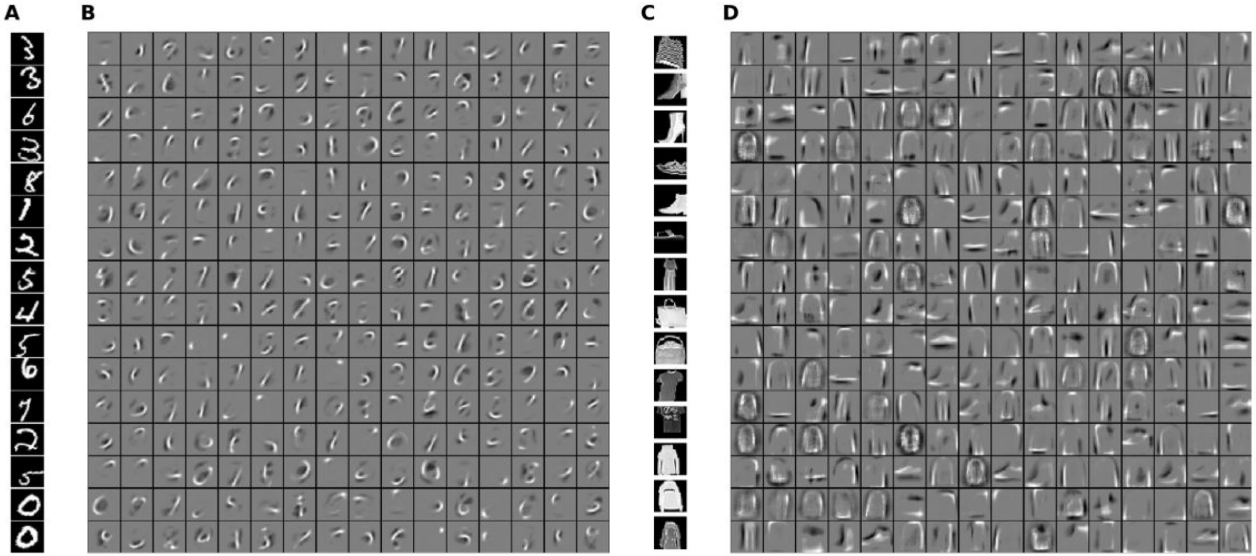


Fig. 3. The network learns independent components from the MNIST datasets. A, B: the network learns pen-stroke shapes from the MNIST dataset. A: sample input stimuli. Black corresponds to zero and white to one. B: receptive fields (weights) of a network with 256 neurons after training on 120,000 digits (28 × 28 pixels) with random distortions. Middle grey corresponds to zero, lighter pixels to excitatory weights, and darker pixels to inhibitory weights. C, D: the network learns the outlines and parts of the various items of clothing in the Fashion-MNIST dataset; for instance the neuron in the top-right corner of D responds to short sleeves. All other details are the same as for A and B.

Sparse Coding Fundamentals

through optimization we are trying to find the values that correspond to the minimum of given equation

one observed sample as vector (i.e. image)

coefficient to what extent we use this factor (scalar, learned for every sample)

one factor (vector, same for all dataset samples). learned for the whole dataset

$$argmin_{r_i^{(j)}, f_i} \sum_{j=1}^{n} \left\| \mathbf{x^{(j)}} - \sum_{i=1}^{k} r_i^{(j)} \mathbf{f_i} \right\|_2 + \lambda \underline{\sum_{i=1}^{k} \| r_i^{(j)} \|_0}$$

iterating over factors. result -- restored sample from sparse representation to original one

taking sum of all learned coefficients across all samples to force most of them = 0. All non-zeros are evaluated to 1 and increase total result.

subject to

$$\| f_i \|_2 \leq 1, \forall i = 1 \ldots k$$

optimization is done under constraint, that total L2 sum of all factors will be below 1. Otherwise it will explode to compensate r_i penalty

SPARSE CODING

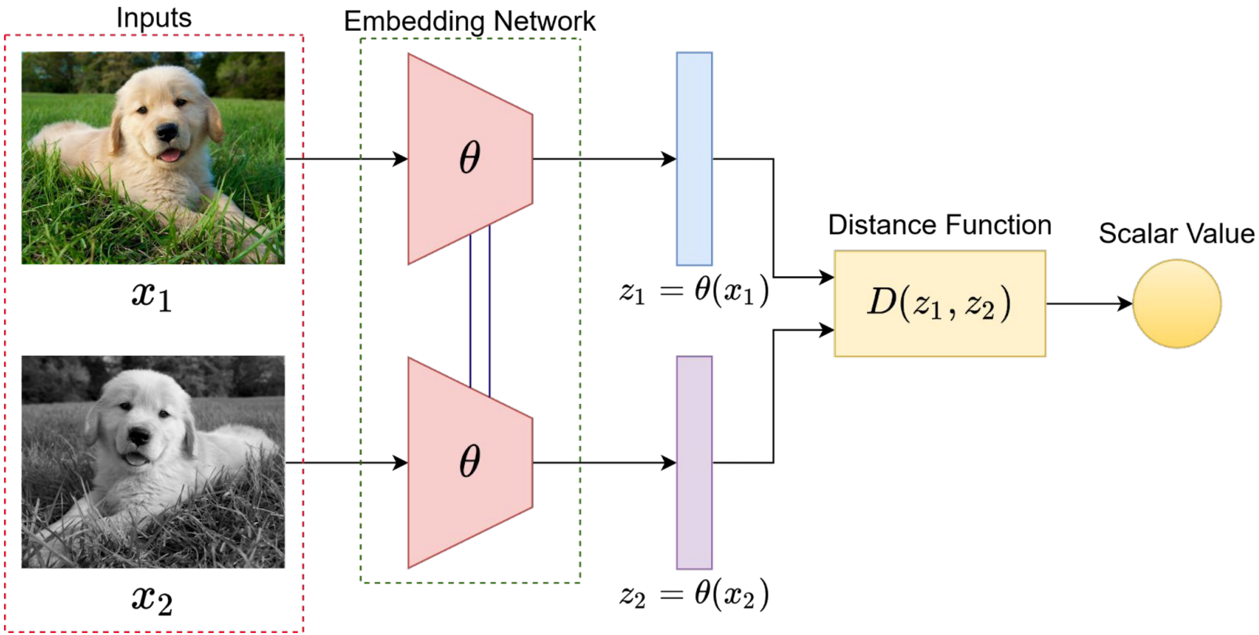
Topics: sparse coding

- For each $\mathbf{x}^{(t)}$ find a latent representation $\mathbf{h}^{(t)}$ such that:
  - it is sparse: the vector $\mathbf{h}^{(t)}$ has many zeros
  - we can reconstruct the original input $\mathbf{x}^{(t)}$ as well as possible
- More formally:

$$\min_{\mathbf{D}} \frac{1}{T}\sum_{t=1}^{T} \min_{\mathbf{h}^{(t)}} \frac{1}{2} \overbrace{||\mathbf{x}^{(t)} - \underbrace{\mathbf{D}\,\mathbf{h}^{(t)}}_{\text{reconstruction } \widehat{\mathbf{x}}^{(t)}}||_2^2}^{\text{reconstruction error}} + \underbrace{\lambda}_{\text{reconstruction vs. sparsity control}} \overbrace{||\mathbf{h}^{(t)}||_1}^{\text{sparsity penalty}}$$

  - we also constrain the columns of $\mathbf{D}$ to be of norm 1
    - otherwise, $\mathbf{D}$ could grow big while $\mathbf{h}^{(t)}$ becomes small to satisfy the prior
  - sometimes the columns are constrained to be no greater than 1

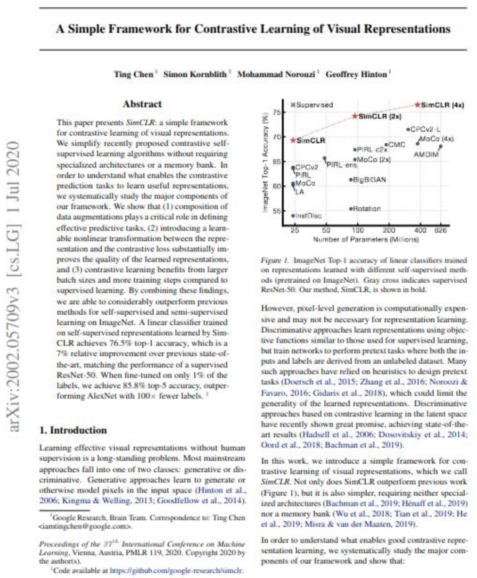
Self-Supervised Learning



SimCLR

Barlow Twins

MoCo



arXiv:2002.05709v3 [cs.LG] 1 Jul 2020

A Simple Framework for Contrastive Learning of Visual Representations

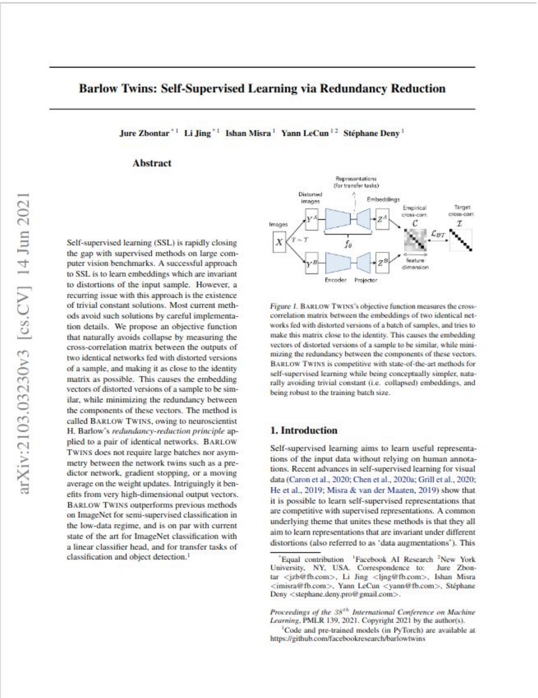
Ting Chen [1] Simon Kornblith [1] Mohammad Norouzi [1] Geoffrey Hinton [1]

Abstract

This paper presents *SimCLR*: a simple framework for contrastive learning of visual representations. We simplify recently proposed contrastive self-supervised learning algorithms without requiring specialized architectures or a memory bank. In order to understand what enables the contrastive prediction tasks to learn useful representations, we systematically study the major components of our framework. We show that (1) composition of data augmentations plays a critical role in defining effective predictive tasks, (2) introducing a learnable nonlinear transformation between the representation and the contrastive loss substantially improves the quality of the learned representations, and (3) contrastive learning benefits from larger batch sizes and more training steps compared to supervised learning. By combining these findings, we are able to considerably outperform previous methods for self-supervised and semi-supervised learning on ImageNet. A linear classifier trained on self-supervised representations learned by SimCLR achieves 76.5% top-1 accuracy, which is a 7% relative improvement over previous state-of-the-art, matching the performance of a supervised ResNet-50. When fine-tuned on only 1% of the labels, we achieve 85.8% top-5 accuracy, outperforming AlexNet with 100× fewer labels. [1]

Figure 1. ImageNet Top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Gray cross indicates supervised ResNet-50. Our method, SimCLR, is shown in bold.

1. Introduction



arXiv:2103.03230v3 [cs.CV] 14 Jun 2021

Barlow Twins: Self-Supervised Learning via Redundancy Reduction

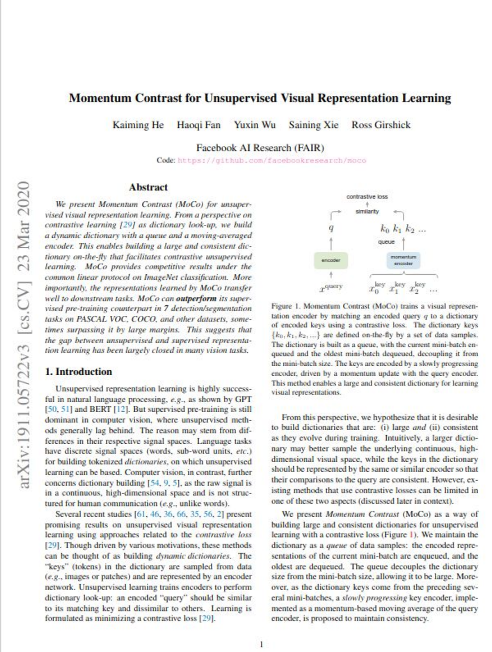
Jure Zbontar [*1] Li Jing [*1] Ishan Misra [1] Yann LeCun [1,2] Stéphane Deny [1]

Abstract

Self-supervised learning (SSL) is rapidly closing the gap with supervised methods on large computer vision benchmarks. A successful approach to SSL is to learn embeddings which are invariant to distortions of the input sample. However, a recurring issue with this approach is the existence of trivial constant solutions. Most current methods avoid such solutions by careful implementation details. We propose an objective function that naturally avoids collapse by measuring the cross-correlation matrix between the outputs of two identical networks fed with distorted versions of a sample, and making it as close to the identity matrix as possible. This causes the embedding vectors of distorted versions of a sample to be similar, while minimizing the redundancy between the components of these vectors. The method is called BARLOW TWINS, owing to neuroscientist H. Barlow's *redundancy-reduction principle* applied to a pair of identical networks. BARLOW TWINS does not require large batches nor asymmetry between the network twins such as a predictor network, gradient stopping, or a moving average on the weight updates. Intriguingly it benefits from very high-dimensional output vectors. BARLOW TWINS outperforms previous methods on ImageNet for semi-supervised classification in the low-data regime, and is on par with current state of the art for ImageNet classification with a linear classifier head, and for transfer tasks of classification and object detection. [1]

Figure 1. BARLOW TWINS's objective function measures the cross-correlation matrix between the embeddings of two identical networks fed with distorted versions of a batch of samples, and tries to make this matrix close to the identity. This causes the embedding vectors of distorted versions of a sample to be similar, while minimizing the redundancy between the components of these vectors. BARLOW TWINS is competitive with state-of-the-art methods for self-supervised learning while being conceptually simpler, naturally avoiding trivial constant (i.e. collapsed) embeddings, and being robust to the training batch size.

1. Introduction



arXiv:1911.05722v3 [cs.CV] 23 Mar 2020

Momentum Contrast for Unsupervised Visual Representation Learning

Kaiming He Haoqi Fan Yuxin Wu Saining Xie Ross Girshick

Facebook AI Research (FAIR)

Code: https://github.com/facebookresearch/moco

Abstract

We present Momentum Contrast (MoCo) for unsupervised visual representation learning. From a perspective on contrastive learning [29] as dictionary look-up, we build a dynamic dictionary with a queue and a moving-averaged encoder. This enables building a large and consistent dictionary on-the-fly that facilitates contrastive unsupervised learning. MoCo provides competitive results under the common linear protocol on ImageNet classification. More importantly, the representations learned by MoCo transfer well to downstream tasks. MoCo can ***outperform*** *its supervised pre-training counterpart in* ***7*** *detection/segmentation tasks on PASCAL VOC, COCO, and other datasets, sometimes surpassing it by large margins. This suggests that the gap between unsupervised and supervised representation learning has been largely closed in many vision tasks.*

Figure 1. Momentum Contrast (MoCo) trains a visual representation encoder by matching an encoded query q to a dictionary of encoded keys using a contrastive loss. The dictionary keys $\{k_0, k_1, k_2, ...\}$ are defined on-the-fly by a set of data samples. The dictionary is built as a queue, with the current mini-batch enqueued and the oldest mini-batch dequeued, decoupling it from the mini-batch size. The keys are encoded by a slowly progressing encoder, driven by a momentum update with the query encoder. This method enables a large and consistent dictionary for learning visual representations.

1. Introduction

To-Do : Leer!

SimCLR

A Simple Framework for Contrastive Learning of Visual Representations

Ting Chen[1] Simon Kornblith[1] Mohammad Norouzi[1] Geoffrey Hinton[1]

Abstract

This paper presents *SimCLR*: a simple framework for contrastive learning of visual representations. We simplify recently proposed contrastive self-supervised learning algorithms without requiring specialized architectures or a memory bank. In order to understand what enables the contrastive prediction tasks to learn useful representations, we systematically study the major components of our framework. We show that (1) composition of data augmentations plays a critical role in defining effective predictive tasks, (2) introducing a learnable nonlinear transformation between the representation and the contrastive loss substantially improves the quality of the learned representations, and (3) contrastive learning benefits from larger batch sizes and more training steps compared to supervised learning. By combining these findings, we are able to considerably outperform previous methods for self-supervised and semi-supervised learning on ImageNet. A linear classifier trained on self-supervised representations learned by SimCLR achieves 76.5% top-1 accuracy, which is a 7% relative improvement over previous state-of-the-art, matching the performance of a supervised ResNet-50. When fine-tuned on only 1% of the labels, we achieve 85.8% top-5 accuracy, outperforming AlexNet with 100× fewer labels. [1]

1. Introduction

Learning effective visual representations without human supervision is a long-standing problem. Most mainstream approaches fall into one of two classes: generative or discriminative. Generative approaches learn to generate or otherwise model pixels in the input space (Hinton et al., 2006; Kingma & Welling, 2013; Goodfellow et al., 2014).

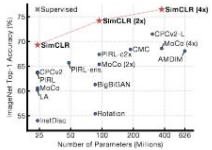


Figure 1. ImageNet Top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Gray cross indicates supervised ResNet-50. Our method, SimCLR, is shown in bold.

However, pixel-level generation is computationally expensive and may not be necessary for representation learning. Discriminative approaches learn representations using objective functions similar to those used for supervised learning, but train networks to perform pretext tasks where both the inputs and labels are derived from an unlabeled dataset. Many such approaches have relied on heuristics to design pretext tasks (Doersch et al., 2015; Zhang et al., 2016; Noroozi & Favaro, 2016; Gidaris et al., 2018), which could limit the generality of the learned representations. Discriminative approaches based on contrastive learning in the latent space have recently shown great promise, achieving state-of-the-art results (Hadsell et al., 2006; Dosovitskiy et al., 2014; Oord et al., 2018; Bachman et al., 2019).

In this work, we introduce a simple framework for contrastive learning of visual representations, which we call *SimCLR*. Not only does SimCLR outperform previous work (Figure 1), but it is also simpler, requiring neither specialized architectures (Bachman et al., 2019; Hénaff et al., 2019) nor a memory bank (Wu et al., 2018; Tian et al., 2019; He et al., 2019; Misra & van der Maaten, 2019).

In order to understand what enables good contrastive representation learning, we systematically study the major components of our framework and show that:

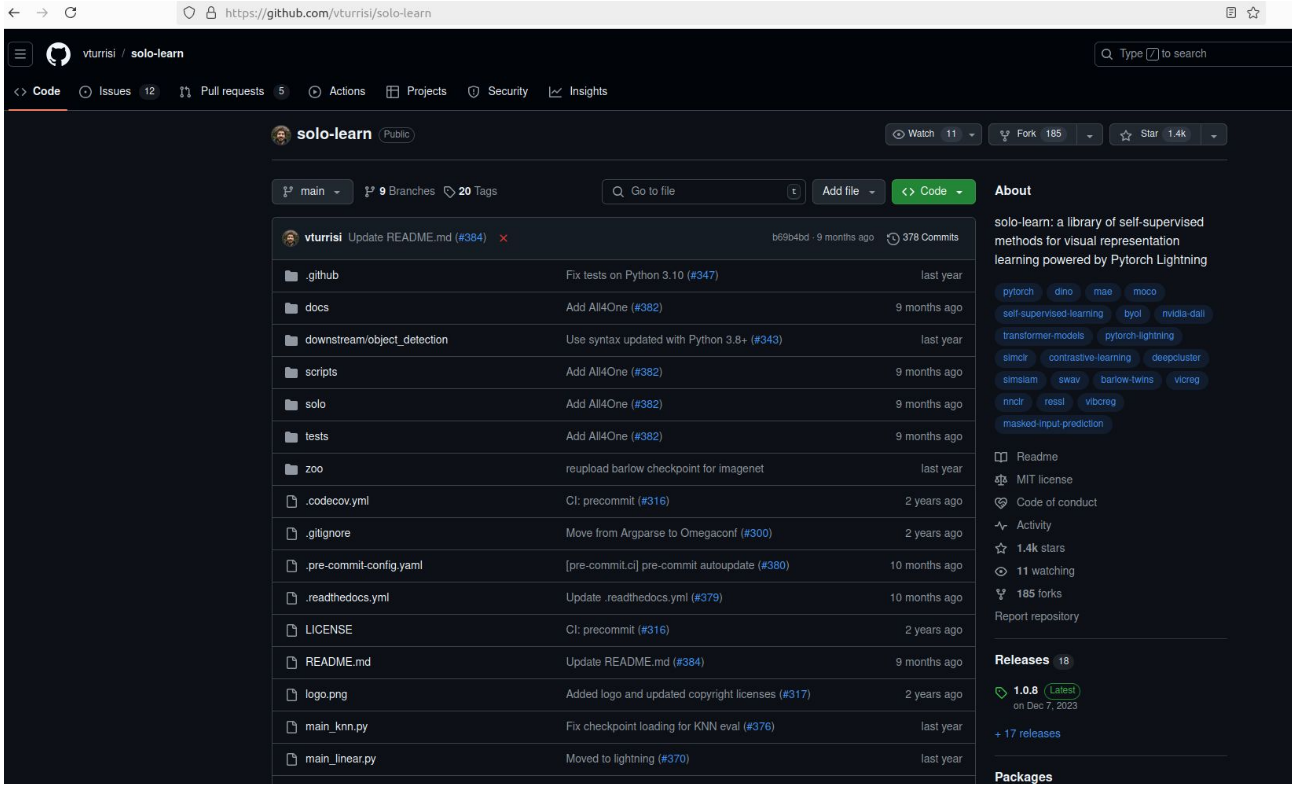
[1]Google Research, Brain Team. Correspondence to: Ting Chen <iamtingchen@google.com>.

Proceedings of the 37th International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020. Copyright 2020 by the author(s).

[1]Code available at https://github.com/google-research/simclr.

SimCLR

How do we later do Supervised Learning?



Librería de Self-Supervised Learning : Solo-Learn