Appendix A. Learning Methods of Self-Supervised learning

In this appendix, we review the four main categories of self-supervised approaches for feature extraction, differing in the way pairs of similar/dissimilar images are defined and exploited for training the model:

Contrastive learning: in this framework, the feature representation is learned by exploiting both positive (similar) and negative (dissimilar) pairs of images, where the former pairs are typically obtained by applying different transformations on the same image. An example in this category is SIMCLR [37] that generates two views of each image by two independent applications of several data augmentation techniques on the image. Positive pairs are then all pairs of views originating from the same image, while negative pairs are pairs of views coming from different images. All views are passed through a ResNet backbone followed by a projection MLP head to obtain vectors of features that are then compared using a contrastive loss that takes into account both positive and negative pairs. Variants of this model include CPC [75], AMDIM [76], and MoCo [9].

Non-contrastive learning: the main difference with respect to contrastive learning is that non-contrastive learning only relies on positive pairs to learn the representation of an image. For example, BYOL [10] creates two views of a given image similarly as SIMCLR and passes each of them into a different network. The first network, called the teacher, has its parameters computed as an exponential moving average of the weights of the second network, called the student. The student network parameters are trained using a loss that enforces the similarity of the feature vectors predicted by the two networks from the two views of the same image, while vectors of dissimilar images are never compared.

Clustering: in this framework, the supervision signal for training the representation is obtained by computing the similarity of the image features to cluster prototypes, which avoids the costly consideration of all pairwise comparisons between images. For example, SwAV [38] computes a code vector from an image by matching its predicted features to (trainable) prototype vectors and then train the model so that it allows to predict the code corresponding to a view of an image from the representation of another view of the same image.

Generative learning: this approach learns a generative model that can reproduce an image from a transformed view of this image. This idea is explored for example in the cdpath21-gan method [24] that extract features from the latent representation of an adversarial auto-encoder trained to generate an image from a central crop of this image. The intuition behind this method is that by learning to expand a central crop, the model will focus on the important features and therefore lead to a good representation.