Self-Supervised Learning

Tretext Tasks

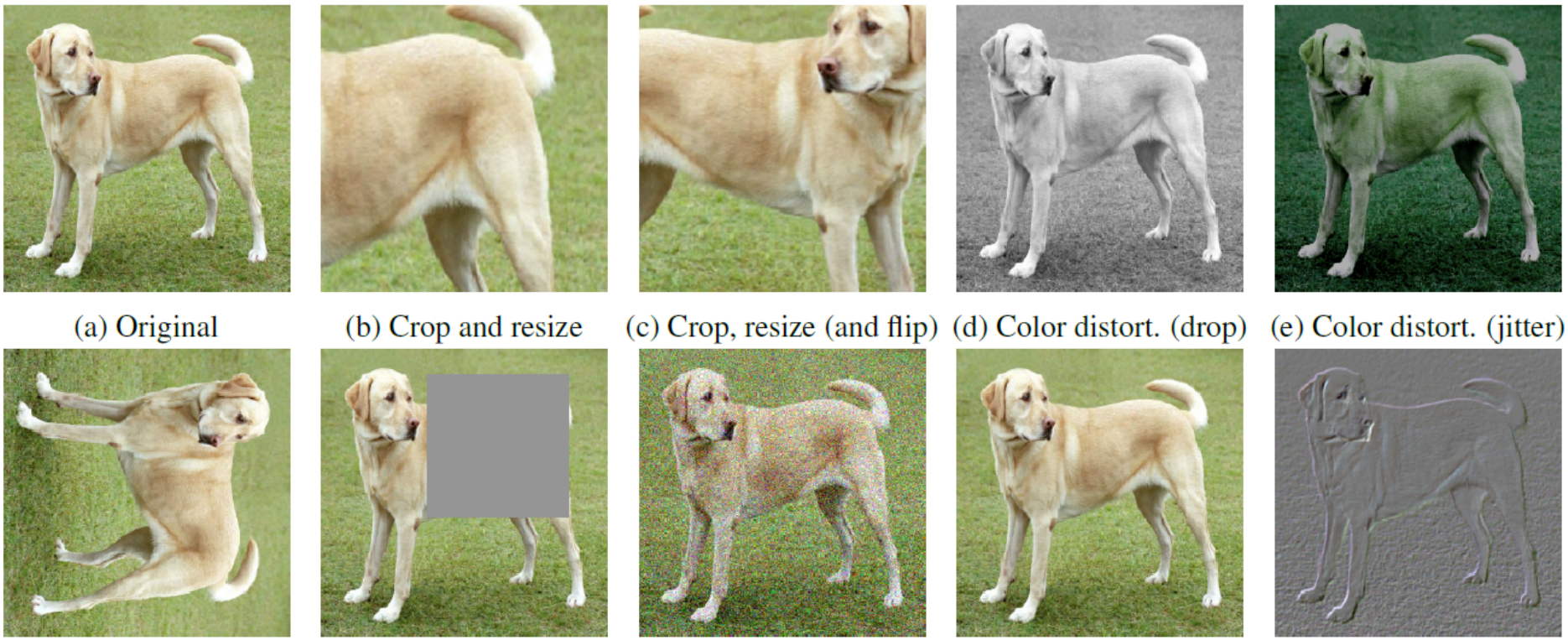
Klaudia Balcer

Advanced Data Mining

Contents

A **Sim**ple **C**ontrastive **L**earning of Visual **R**epresentations

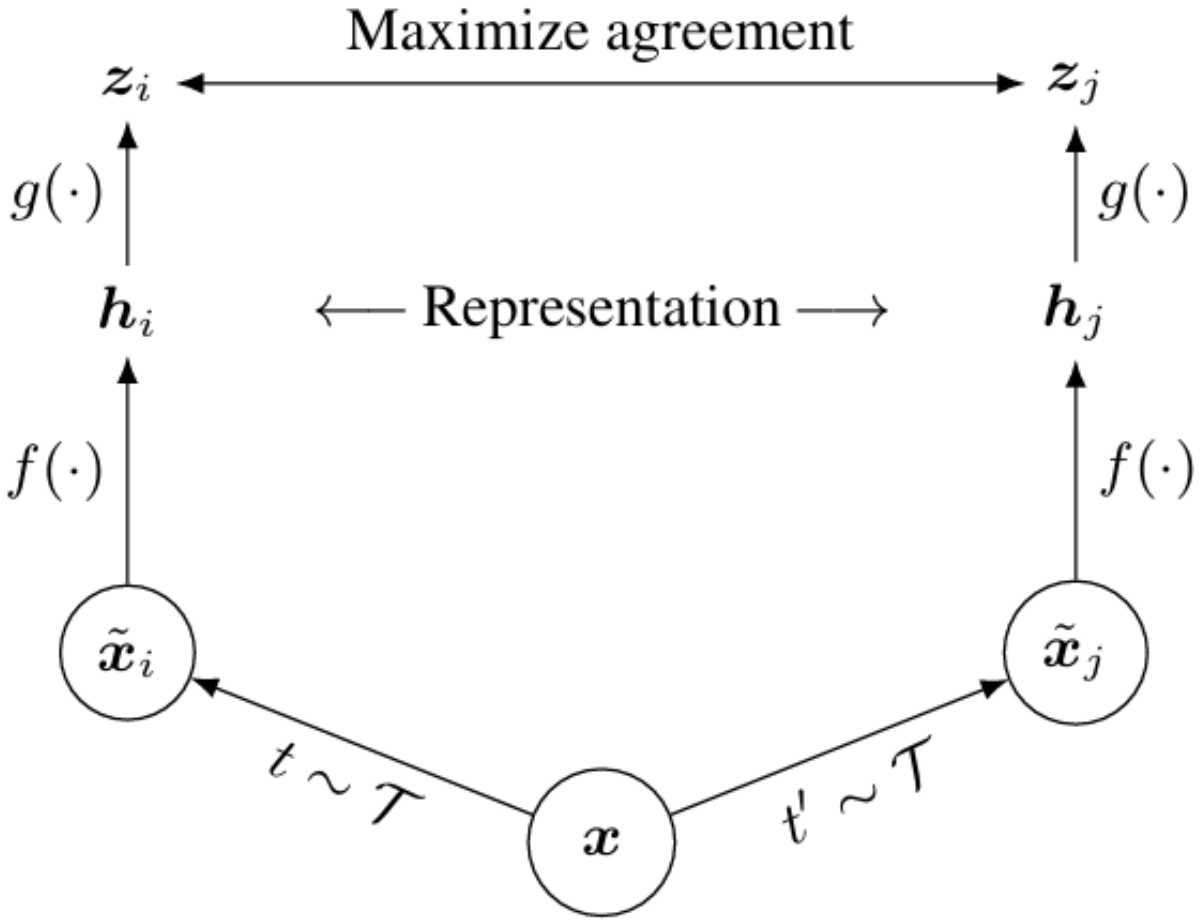
Image Augmentations



(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

(f) Rotate $\{90°, 180°, 270°\}$ (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering

SimCLR



https://dl.acm.org/doi/10.5555/3524938.3525087

SimCLR

- stochastic data augmentation t:
  - random cropping
  - random color distortions
  - Gaussian blur
- base encoder $f(.)$ – ResNet (https://arxiv.org/abs/1512.03385)
- projection head $g(.)$ – MLP
- contrastive loss:

$$\mathcal{L} = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)},$$

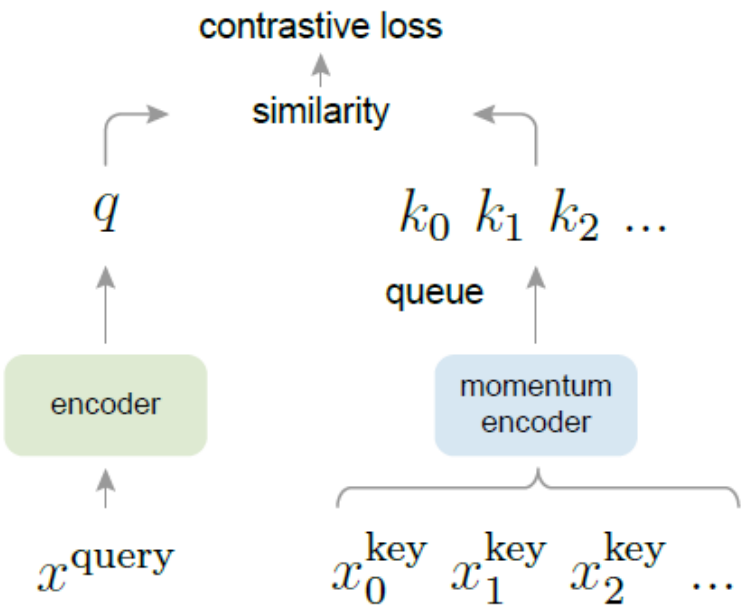
  where sim is cosine similarity and as negatives we treat both views of all other images;
  – NT-Xent (the normalized temperature-scaled cross entropy loss).
- huge batch size – from 256 to 8192, LARS optimizer, global batch normalization

Momentum Contrast

MoCo

- Contrastive learning as dynamic dictionary:
  - keys – instances from the data
  - values / queries – encoded images
- InfoNCE loss:

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+/\tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i/\tau)}$$



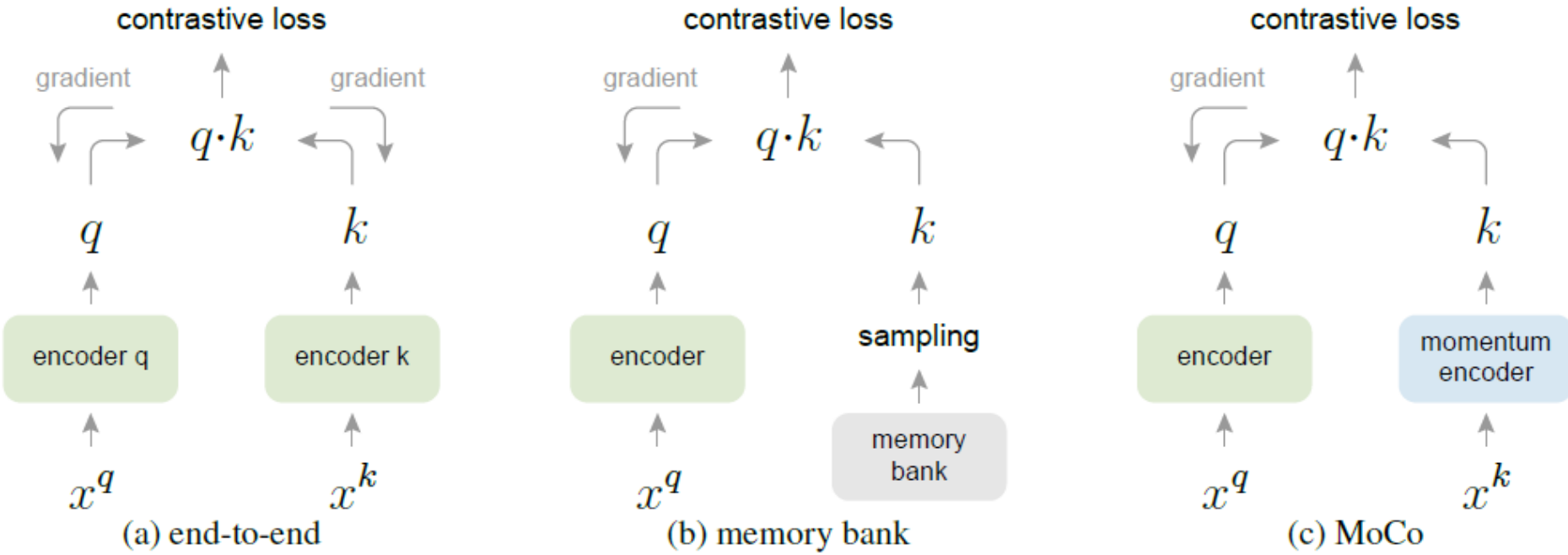
MoCo

- Queue:
  - consist of embeddings of images from **multiple mini-baches**
  - progressive replacement: when a new mini-batch is added, the oldest is removed
- Momentum encoder:
  - when we do have *keys* from outside the current mini-batch, we cannot do back-propagation on the key encoder
  - using the query encoder to the key encoder leads to rapid changes and inconsistency in the queued embeddings
  - we use a momentum update rule:

$$\theta_k \leftarrow m\theta_k + (1-m)\theta_q$$

  - a slowly evolving key encoder is a core to making use of a queue
- Look at the pseudo-code from the paper (page 4)

MoCo



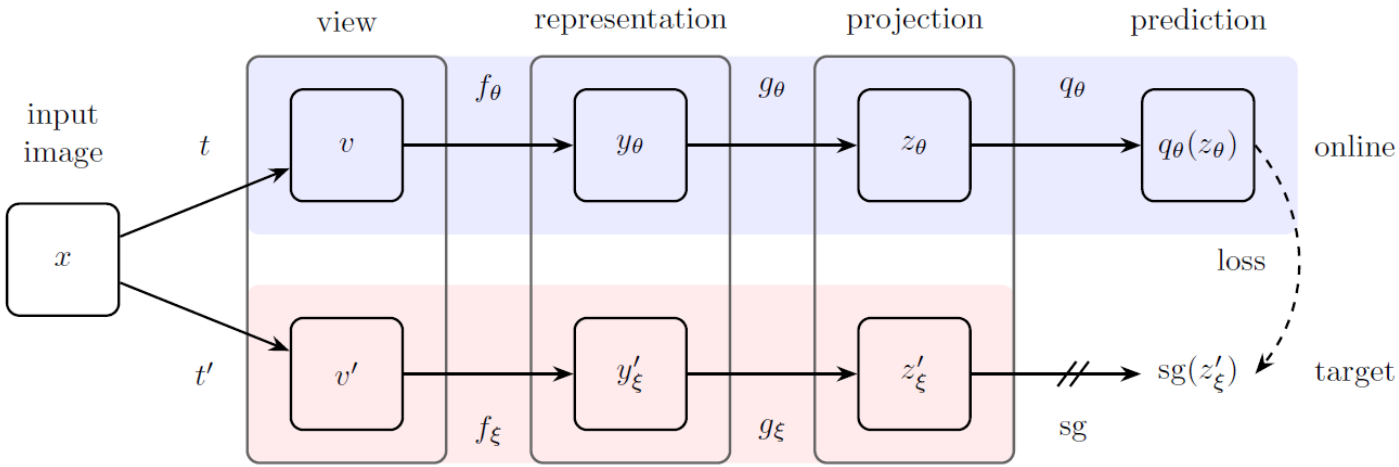
Technical details: ResNet encoder, temperature $\tau = 0.07$, augmentations: a 224×224-pixel crop is taken from a randomly resized image, and then undergoes random color jittering, random horizontal flip, and random grayscale conversion, all available in PyTorch' s torchvision package, per device batch normalization (details in the paper), SGD optimizer.

https://openaccess.thecvf.com/content_CVPR_2020/papers/He_Momentum_Contrast_for_Unsupervised_Visual_Representation_Learning_CVPR_2020_pa

Bootstrap your own Latent

BYOL

- experimental fining: when learning to mimic a random network, we can obtain much better representations than a random initialization (18.8% vs 1.4% on classification downstream task)
- two neural networks (online and target networks) that interact and learn from each other
- BYOL trains its online network to predict the target network' s representation of another augmented view of the same image
- the effectiveness of BYOL depends less on the augmentations used than on previous approaches



https://arxiv.org/pdf/2006.07733

BYOL

- online network is defined by a set of weights θ and is comprised of three stages: an encoder f_θ, a projector g_θ and a predictor q_θ,
- The target network has the same architecture as the online network, but uses a different set of weights ξ
- we update the target network with moving average (momentum encoder):

$$\xi \leftarrow \tau\xi + (1-\tau)\xi$$

- the prediction is made to map one representation to the other
- loss function (one summand, second summand with augmentations swap):

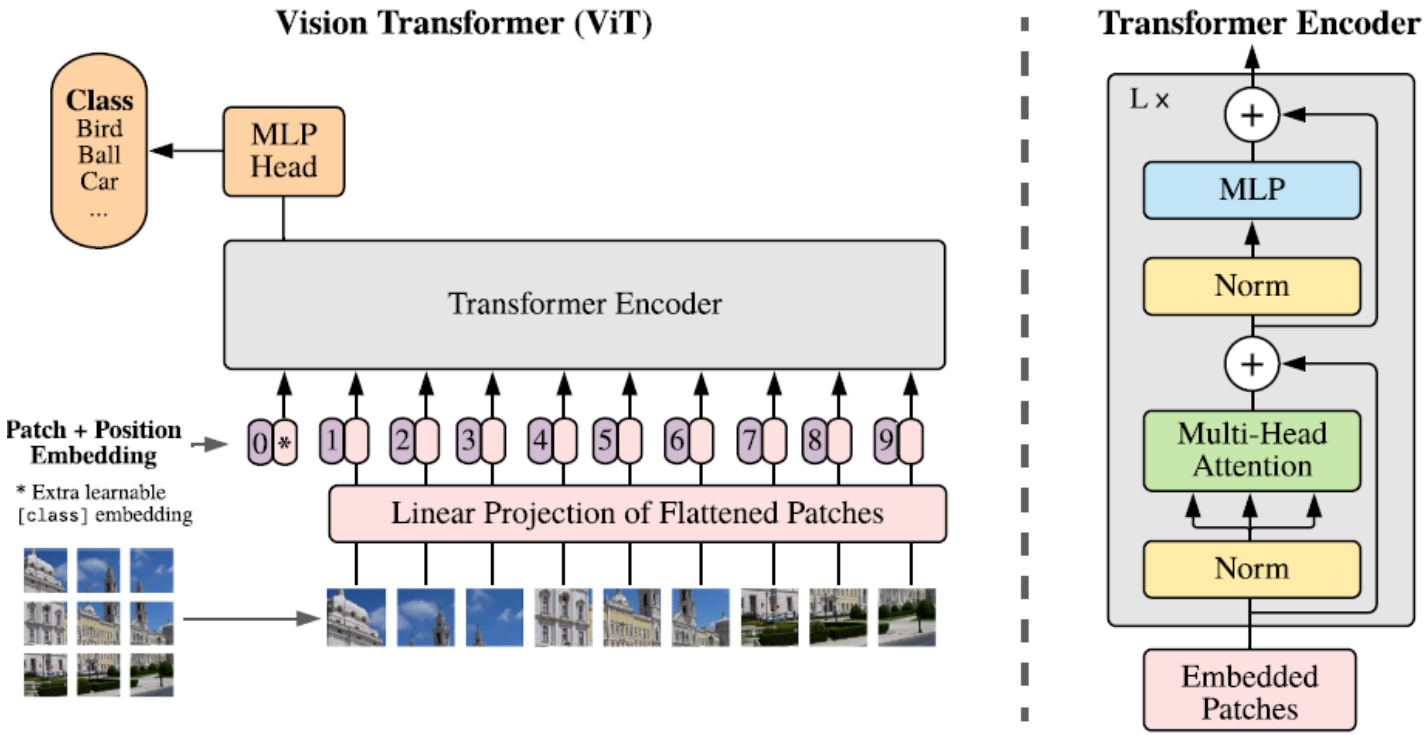
$$\mathcal{L}_{\theta,\xi} \triangleq \left\| \overline{q_\theta}\left(z_\theta\right) - \bar{z}'_\xi \right\|_2^2 = 2 - 2 \cdot \frac{\left\langle q_\theta\left(z_\theta\right), z'_\xi \right\rangle}{\left\| q_\theta\left(z_\theta\right) \right\|_2 \cdot \left\| z'_\xi \right\|_2}$$

- online network is optimized using a gradient based optimizer (LARS)
- architecture: ResNet50 (convnet)

Self-**di**stillation with **no** Labels

ViT – recap

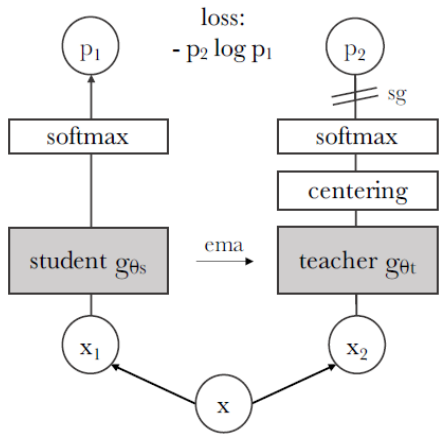
- show their power on large datasets (14M-300M images)
- applying a standard Transformer directly to images, with the fewest possible modifications



https://openreview.net/pdf?id=YicbFdNTTy

DINO

- One of the components of the success of Transformers in NLP was self-supervised pretraining; ViT is trained in a supervised manner



- Both networks have the same architecture but different parameters.
- We apply a stop-gradient (sg) operator on the teacher to propagate gradients only through the student.
- The teacher parameters are updated with an exponential moving average (ema) of the student parameters.

DINO

- Knowledge distillation without any true labels (with soft pseudo-labels), and the teacher is built during the training, somewhat related to co-distillation (similar to mutual training)
- *Global* augmentations (containing over 50% of the image), *Local* augmentations (containing less than 50% of the image), V – set of all views

$$\min_{\theta_s} \sum_{x \in \{x_1^g, x_2^g\}} \sum_{\substack{x' \in V \\ x' \neq x}} H(P_t(x), P_s(x'))$$

- teacher is updated with students weights in momentum encoder:

$$\theta_t \leftarrow \lambda \theta_t + (1 - \lambda)\theta_s$$

with λ from cosine schedule from 0.996 to 1, no batch-normalization

- avoiding collapsing: sharpening (temperature in softmax) and centering:

$$g_t(x) \leftarrow g_t(x) + c$$

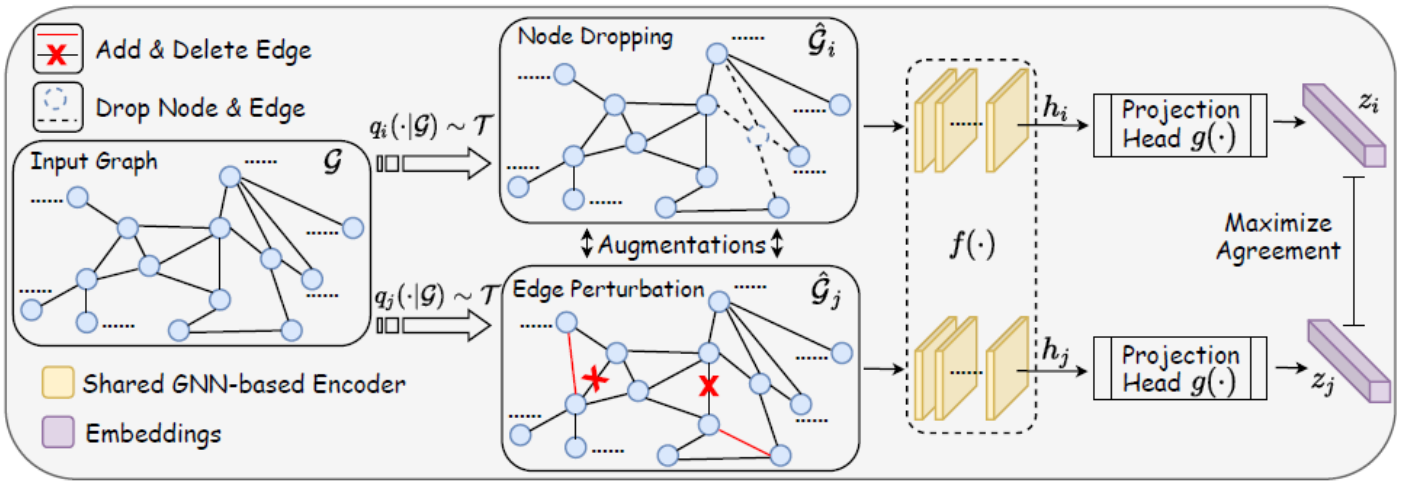
$$c \leftarrow mc = (1 - m)\frac{1}{B}\sum_{i=1}^{B} g_{\theta_t}(x_i)$$

Graph Contrastive Learning with Augmentations

GraphCL

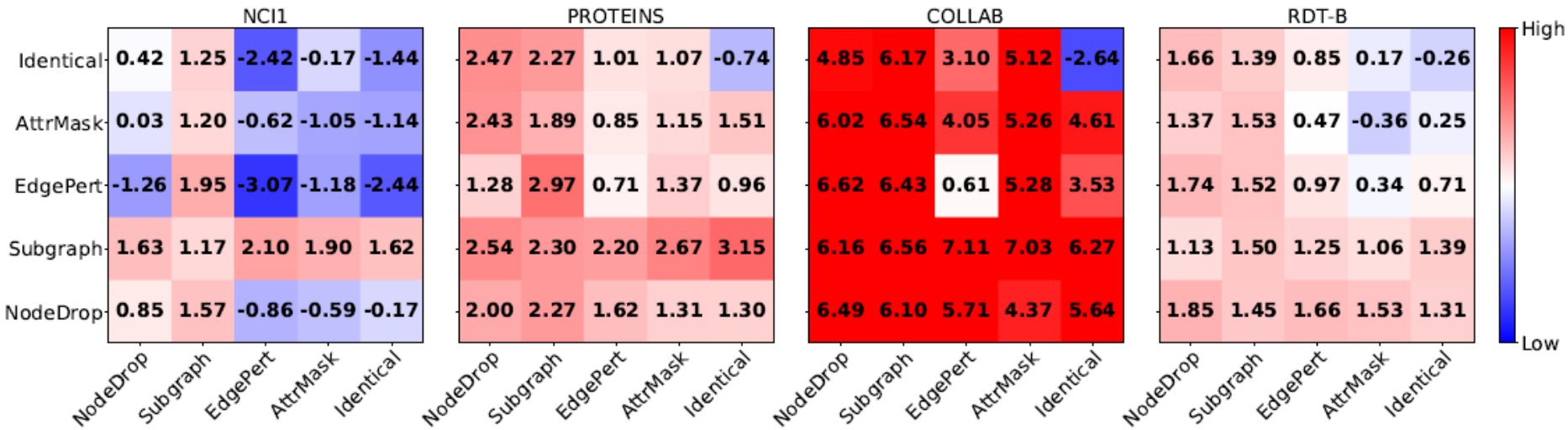
- Augmentations: node dropout, edge perturbation, attribute masking, subgraph selection (via random walk). The selection depends on the domain.
- NT-Xent loss:

$$\ell_n = -\log \frac{\exp\left(\text{sim}\left(\boldsymbol{z}_{n,i}, \boldsymbol{z}_{n,j}\right)/\tau\right)}{\sum_{n'=1,n'\neq n}^{N} \exp\left(\text{sim}\left(\boldsymbol{z}_{n,i}, \boldsymbol{z}_{n',j}\right)/\tau\right)}$$



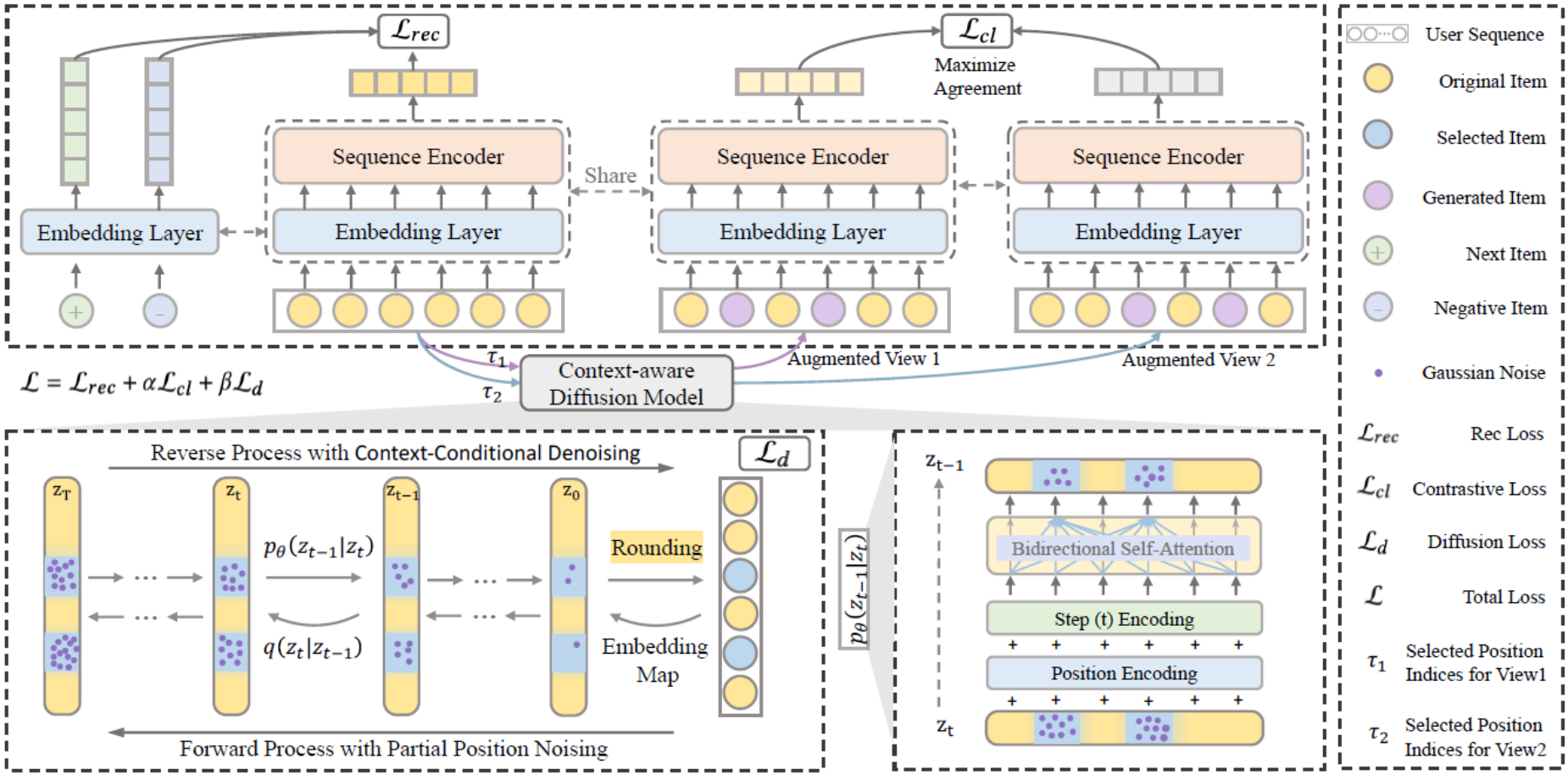
https://proceedings.nips.cc/paper/2020/file/3fe230348e9a12c13120749e3f9fa4cd-Paper.pdf

GraphCL


NCI1
PROTEINS
COLLAB
RDT-B
Identical 0.42 1.25 -2.42 -0.17 -1.44
AttrMask 0.03 1.20 -0.62 -1.05 -1.14
EdgePert -1.26 1.95 -3.07 -1.18 -2.44
Subgraph 1.63 1.17 2.10 1.90 1.62
NodeDrop 0.85 1.57 -0.86 -0.59 -0.17
2.47 2.27 1.01 1.07 -0.74
2.43 1.89 0.85 1.15 1.51
1.28 2.97 0.71 1.37 0.96
2.54 2.30 2.20 2.67 3.15
2.00 2.27 1.62 1.31 1.30
4.85 6.17 3.10 5.12 -2.64
6.02 6.54 4.05 5.26 4.61
6.62 6.43 0.61 5.28 3.53
6.16 6.56 7.11 7.03 6.27
6.49 6.10 5.71 4.37 5.64
1.66 1.39 0.85 0.17 -0.26
1.37 1.53 0.47 -0.36 0.25
1.74 1.52 0.97 0.34 0.71
1.13 1.50 1.25 1.06 1.39
1.85 1.45 1.66 1.53 1.31
NodeDrop Subgraph EdgePert AttrMask Identical
High
Low

Context-aware Diffusion-based Contrastive Learning for Sequential Recommendation

CaDiRec



https://arxiv.org/html/2405.09369v5