Momentum Contrast for Unsupervised Visual Representation Learning

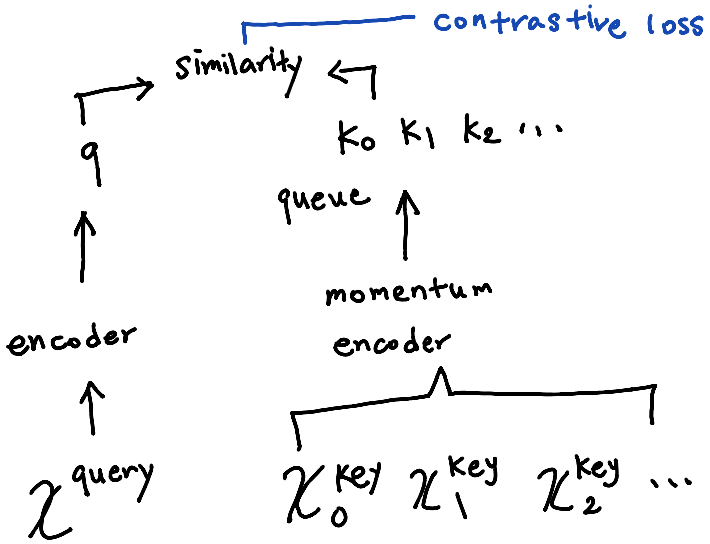
0. Abstract

- Momentum Contrast (MoCo) for unsupervised visual representation learning
- from a perspective on contrastive learning as dictionary look-up
- build a dynamic dictionary with a queue and a moving-averaged encoder

1. Introduction

- unsupervised visual representation learning related to the contrastive loss
- can be thought of as building dynamic dictionaries
- "keys" are represented by an encoder network
- unsupervised learning trains the encoder to perform dictionary look-up
- learning is formulated as minimizing a contrastive loss
- desirable to build dictionaries that are large and consistent as they evolve during training
- maintain the dictionary as a queue of data samples
- queue decouples the dictionary size from the mini-batch size
- a slowly progressing key encoder, implemented as a momentum-based moving average of the query encoder to maintain consistency

Figure 1. MoCo trains a visual representation encoder. The keys are encoded by a slowly progressing encoder, driven by a momentum update with the query encoder.



2. Related Work

- Unsupervised/self-supervised learning methods generally involve two aspects: pretext tasks and loss functions
- MoCo focuses on the loss function aspect

3. Method

3.1 Contrastive Learning as Dictionary Look-up

- Contrastive learning can be thought of as training an encoder for dictionary look-up task
- encoded query q
- a set of encoded samples $\{k_0, k_1, k_2, \dots\}$
- a single key in the dictionary that matches q (denoted $k+$)
- contrastive loss value is low when q is similar to $k+$, dissimilar to all other
- With similarity measured by dot product, called InfoNCE

$$L_q = -\log \frac{\exp(q \cdot k+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}$$

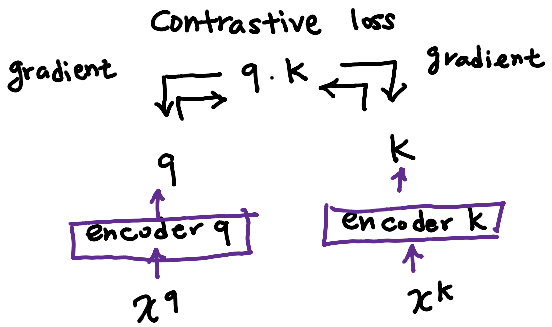
τ : temperature hyper-parameter

- sum over one positive and K negative samples
- the log loss of a (K+1)-way softmax-based classifier to classify q as k+
- query representation $q = f_q(x^q)$

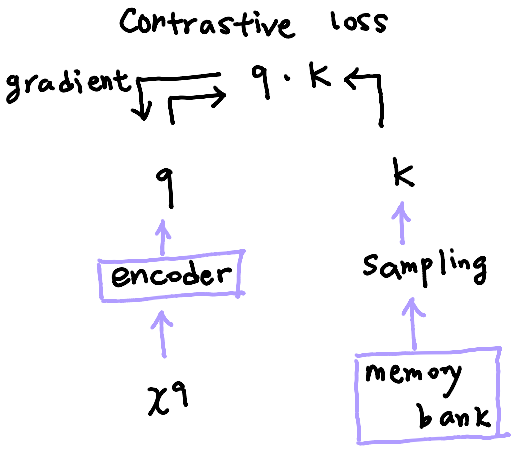
f_q : encoder network

x^q : query sample

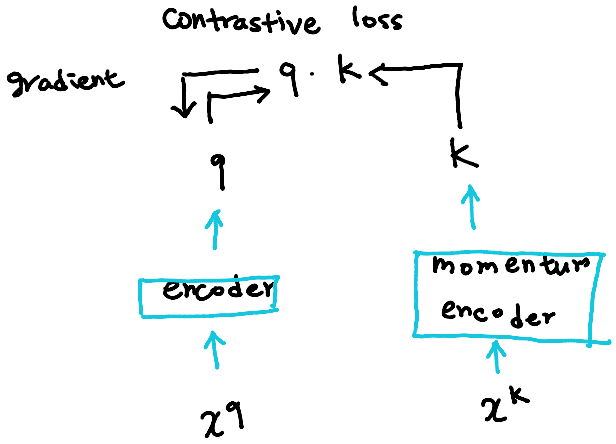
Figure 2. Conceptual comparison of three contrastive loss mechanisms
(a) encoders are updated end-to-end by back-propagation



(b) key representations are sampled from a memory bank



(c) MoCo: encodes the new key by a momentum-updated encoder



3.2 Momentum Contrast

1) Dictionary as a queue

- reuse the encode keys from the immediate preceding mini-batches
- dictionary size can be much larger than typical mini-batch size
- current mini-batch enqueued, oldest mini-batch dequeued (consistency)

2) Momentum update

- using a queue makes it intractable to update the key encoder by back-propagation
- denoting f_k parameters as θ_k and f_q as θ_q, update θ_k by

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

$m \in [0,1)$: momentum coefficient

* only θ_q are updated by back-propagation

* θ_k evolve more smoothly than θ_q

- large momentum works much better

Algorithm 1. Pseudo-code of MoCo ← queue: dictionary as a queue of K keys (C x K)

```
f_k.params = f_q.params #initialize
for x in loader: # load a mini-batch N with N samples
    x_q = aug(x)  # a randomly augmented version
    x_k = aug(x)  # another randomly augmented version

    q = f_q.forward(x_q)  queries: N x C
    k = f_k.forward(x_k)  keys: N x C
    k = k.detach()  no gradient to keys

    positive logits: N x 1
    l_pos = bmm(q.view(N,1,C), k.view(N,C,1))
    negative logits: N x K
    l_neg = mm(q.view(N,C), queue.view(C,K))
    logits: N x (1+K)
    logits = cat([l_pos, l_neg], dim=1)

    contrastive loss
    # positives are the 0-th
    labels = zeros(N)
    loss = CrossEntropyLoss(logits/t, labels)

    loss.backward()
    update(f_q.params)
    f_k.params = m*f_k.params + (1-m)*f_q.params

    #update the dictionary
    enqueue(queue, k) enqueue the current mini-batch
    dequeue(queue) dequeue the earliest mini-batch
```

3.3 Pretext Task

- instance discrimination task
- adopt a ResNet as the encoder (128-D)
- this output vector is normalized by its L2-norm
- temperature set as 0.07
- data augmentation setting: 224x224-pixel crop is taken from a randomly resized image, and then undergoes random color jittering, random horizontal flip, and random grayscale conversion
- Shuffling BN: replace encoder with shuffling BN

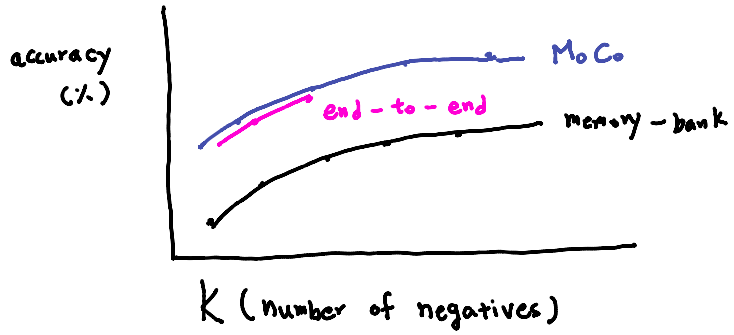
4. Experiments

- ImageNet-1M(IN-1M): count the image number as classes are not exploited by unsupervised learning
- Instagram-1B(IG-1B): long-tailed, unbalanced distribution of real-world data
- Training: SGD as our optimizer. weight decay is 0.0001 and the SGD momentum is 0.9. For IN-1M, mini-batch size of 256 in 8 GPUs. train for 200 epochs. For IG-1B, mini-batch size of 1024 in 64f GPUs, taking up to 6 days.

4.1 Linear Classification Protocol

- perform unsupervised pre-training on IN-1M
- freeze the features and train a supervised linear classifier (a fully-connected layer followed by softmax)
- train this classifier on the global average pooling features of a ResNet, for 100 epochs

Figure 3. Comparison of three contrastive loss mechanisms



4.2 Transferring Features

- main goal of unsupervised learning is to learn features that are transferrable
- feature normalization: system for a downstream task often has hyper-parameters selected for supervised pre-training, adopt feature normalization during fine-tuning, fine-tune with BN that is trained instead of freezing it by an affine layer
- schedule: in our fine-tuning, MoCo uses the same schedule as the ImageNet supervised counterpart
- MoCO is better than its ImageNet supervised counterpart in all metrics