Self - Supervised Representation Learning

Anishka Vaitla (22160) , Krishna Eyunni(22234), Sanyat Fale(22204), Varivashya Poladi(22152)
https://github.com/SanyatFale/Term-Paper

Abstract

This paper provides a succinct exposition of the Momentum Contrast (MoCo) methodology. We conduct experimental investigations in the realm of image classification, employing the CIFAR-10 dataset to scrutinize the MoCo framework under diverse parameter configurations. Our experiments and findings demonstrate the consistent superiority of the MoCo model over supervised learning counterparts across varying experimental conditions.

1 Introduction

Prior to the advent of MoCo, while unsupervised representation learning had demonstrated notable success in natural language processing (NLP), supervised representation learning methods had exhibited superior performance in computer vision tasks. The emergence of MoCo addresses the gap between unsupervised and supervised learning.

2 Contrastive learning and Loss Function

Contrastive Learning is a technique that enhances the performance of vision tasks by contrasting samples against each other to learn common attributes between data classes and set apart different data clases from another. Contrastive learning benefits from larger batch sizes and longer training. This effect is observed as larger batch sizes and longer trainings provide more negative examples, facilitating convergence, thus improving results.

A loss function is a method for evaluating how well an algorithm models a dataset. Contrastive losses measure the similarities of sample pairs in a representation space. Instead of matching an input to a fixed target, in contrastive loss formulations the target can vary on-the-fly during training and can be defined in terms of the data representation computed by a network.

3 Method of MoCo

Momentum Contrast (MoCo) is a mechanism designed to construct dynamic dictionaries for contrastive learning. The dictionary "keys" are randomly sampled from the data and represented by an encoder network. Consider an encoded query q, and a set of encoded samples k0, k1, k2... that are the keys of a dictionary. Assume that there is a single key in the dictionary that q matches. A contrastive loss function is one whose value is low when q is similar to its positive key (denoted by k+) and dissimilar to the remaining keys (considered negative keys for q). The loss function used in MoCo is called InfoNCE, wherein the the contrastive loss function is given as below:

$$Lq = -\log \frac{exp(q.k_+)/\tau}{\sum_{i=0}^{K} exp(q.k_i)/\tau}$$

where τ is a temperature hyper-parameter. The sum is over one positive and K negative samples. Intuitively, this loss is the log loss of a (K+1)-way softmax-based classifier that tries to classify q as k_+.

Preprint. Under review.

MoCo maintains the dictionary as a queue of data samples. MoCo acheives a large and consistent dictionary. The encoded representations of the current mini-batch are enqueued, and the oldest are dequeued. The key encoder evolves during training, with the dictionary keys originating from the preceding several mini-batches. It employs a slowly progressing key encoder, implemented as a momentum-based moving average of the query encoder, ensuring the consistency of the dictionary keys. The query encoder is updated via back-propagation using gradients. A momentum update is employed for the key encoder. Formally, denoting the parameters of the key encoder f_k as θ_k and those of the query encoder f_q as θ_q, we update θ_k by:

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

Here, m ∈ [0, 1) is a momentum coefficient. Only the parameters θ_q are updated by back-propagation. The momentum update ensures that θ_k evolves more smoothly than θ_q. Experimental results indicate that a relatively large momentum (e.g., m = 0.999) performs significantly better than a smaller value (e.g., m = 0.9), suggesting that a slowly evolving key encoder is crucial for effective utilization of a queue.

4 Experiments

The algorithm used in MoCo has some key parameters such as batch size, number of epochs, etc., based on which the accuracy of the self-supervised learning model is expected to vary. The encoder convolutional network used for our MoCo model has 2 convolutional layers and 1 fully connected layer. The standard convolutional net for baseline has 2 convolutional layers, 3 fully connected layers.

We study unsupervised training performed in **CIFAR-10** dataset.

We vary the following parameters to study their effect on the accuracy of the MoCo model and report the findings, one at a time. The original parameters were TrainingDataSize:TestingDataSize = 1:5, Augmentations = Aug0 (mentioned below), NumberOfEpochs = 20, QueueSize = 4096, VectorDimension = 128 and BatchSize = 100.

Ratio of sizes of the training and the testing datasets

The CIFAR-10 dataset consisting of 60000 images from 10 different classes is divided into the following ratios: 10:90 (6000 images in the training dataset and 54000 images in the testing dataset), 20:80, 30:70, 40:60, 50:50 in this order. The total size of the dataset used (both training and testing datasets combined) remains constant. We compare the accuracies of the unsupervised model with the corresponding accuracies of the supervised model for each ratio of the sizes of the training and the testing datasets.

We observe the following:

The accuracy of the unsupervised learning model consistently increases with the size of the training dataset and the consequent decrease in the size of the testing dataset.

The accuracy of the supervised learning model also increases with the increase in the size of the training dataset.

The accuracies of the supervised learning model are less than the corresponding accuracies of the unsupervised learning model until the ratio of the sizes of the training and testing datasets is 50:50.

As the size of the testing dataset increases, the gap between the accuracies of the supervised model and the unsupervised model reduces in general. In our experiment, we observe that this is true until the ratio becomes 40:60. We observe that the gap increases for the ratio 50:50 as compared to the ratio 40:60.

Based on the above observations, we infer that the larger the size of the training dataset, the higher the accuracy of the unsupervised model. Furthermore, the unsupervised model performs better than the supervised model to a certain extent. We hypothesize that as the size of the training dataset increases, the supervised model may fare better than the unsupervised model.

Augmentations

We use different sequences of augmentations. The labels we use for the various augmentations we use are as follows:

Aug0: Crop and Resize, Color distort (drop), Color distort (jitter), Rotation
Aug1: Color distort (jitter), Gaussian blur, Horizontal flip, Color distort (drop)
Aug2: Crop and Resize, Sobel filtering, Color distort (jitter), Rotate
Aug3: Crop and Resize, Sobel filtering, Gaussian blur, Color distort (drop)
Aug4: Horizontal flip, Gaussian blur, Rotate, Color distort (jitter), Crop and Resize
Aug5: Horizontal flip, Gaussian blur, Rotate, Color distort (jitter), Crop and Resize
Aug6: Color distort (jitter), Color distort (drop), Horizontal flip, Crop and Resize

We observe that the highest accuracy is obtained by using Aug0 and the least accuracy is obtained by using Aug1. Aug0 and Aug6 consist of the same types of augmentations, but in a different order. We observe that the accuracies corresponding to both of them are different, but not drastically so. We hypothesize that if different orders of particular sets of augmentations are chosen, the accuracies might change drastically. As reported in the SimCLR paper, it is critical to compose Crop and Resize with Color distort (jitter) as a potential issue with images is that when only random cropping is applied, most patches from an image share a similar color distribution. We observe that Aug1 is the only augmentation we use without this combination and consequently Aug1 has the least accuracy.

The augmentations imposed are crucial to the performance of the model.

Number of epochs in the MoCo encoder

We vary the number of epochs during the training of the MoCo encoder to 20, 30, 40 and 50. We observe that there is no significant pattern in the accuracies of the unsupervised model with respect to the number of epochs.

Size of the queue

We set the size of the queue of the dictionary to 16, 1024 and 4096.We observe that the accuracies of the model increase slightly with the increase in the size of queue.

Dimensions of the representation vectors of images

We change the dimensions of the representation vectors of images to 8, 64 and 128. We do not observe any significant change in the accuracies with a change in the dimensions of the representation vectors of images.

Size of the mini-batch

We vary the size of the mini-batch to 100, 200, 300 and 400. We observe no visible trend between the size of the mini-batch and the accuracy of the unsupervised model. We observe that accuracies do not change significantly.

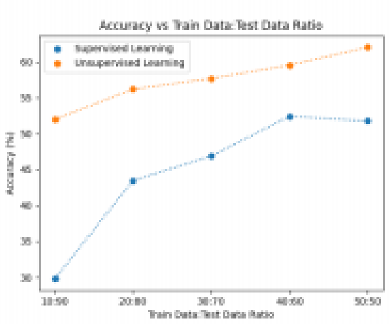

Figure 1: Dataset Ratios

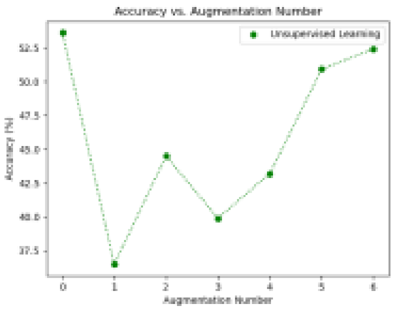

Figure 2: Augmentations

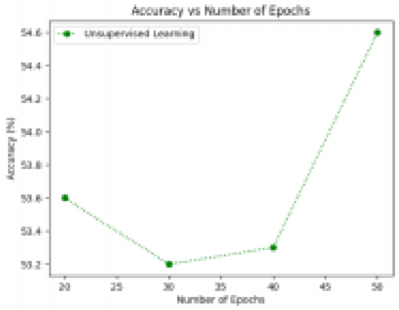

Figure 3: No. of epochs

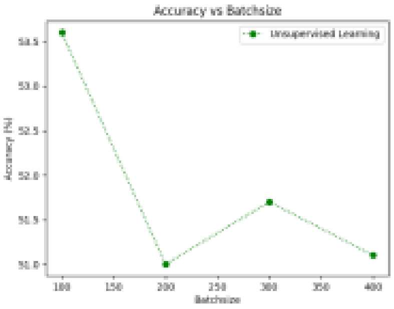

Figure 4: Batch sizes

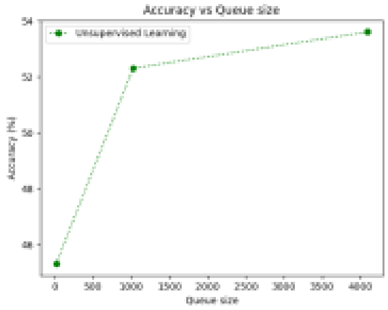

Figure 5: Queue sizes

5 Conclusion

We observe that MoCo has shown positive results in unsupervised learning. The performance of MoCo in image classification is better than the supervised counterpart for the CIFAR-10 dataset.

References

[1] Zhirong Wu, Yuanjun Xiong, Stella Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance dis-crimination

[2] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimension- ality reduction by learning an invariant mapping. In CVPR, 2006.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations.

[4] Sumit Chopra, Raia Hadsell, Yann LeCun. Learning a Similarity Metric Discriminatively, with Application to Face Verification

[5] Prannay Khosla, Piotr Teterwak , Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot,, Ce Liu, Dilip Krishnan. Supervised Contrastive Learning

[6]Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, Ross GirshickMomentum. Contrast for Unsupervised Visual Representation Learning

[7]Yonglong Tian, Dilip Krishnan, Phillip Isola. Contrastive Multiview Coding

[8] Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, Dipendra Misra. Investigating the Role of Negatives in Contrastive Representation Learning

[9] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, Phillip Isola. What Makes for Good Views for Contrastive Learning?