CS482/682 Final Project Report Group

Unsupervised Image Classfication with Momentum Contrast (MoCo)

Yuxuan Chen (ychen349), Zixu Chen (zchen73), Yaxi Hu (yhu53), Jiya Zhang (jzhan215)

1 Introduction

Background As the amount of data increases exponentially, it is impractical to label each of the data manually. Hence, the new approaches of using self-supervised and unsupervised learning, which do not require manual data annotation, have gradually become the next promising research area. In this report, we are going to mainly focus on applying a state-of-the-art Momentum Contrast (MoCo [4]) approach to classify images in the CIFAR10 [6] dataset.

MoCo Momentum Contrast (MoCo) for unsupervised visual representation learning, builds a dynamic dictionary with a queue and a moving-averaged encoder as dictionary look-up from a perspective on contrastive learning. MoCo has been proven to be competitive on ImageNet classfication tasks, and outperforms its supervised pre-training counterpart in 7 detection/segmentation tasks.

Related Work Recently, there are multiple papers focusing on unsupervised image classification (e.g. MoCo [4], CMC [12], InsDis [13]), while there are notable works in the past as well, such as Jigsaw [9], exemplar network [2], and ensemble learning [3].

2 Methods

Dataset We mainly focus on the CIFAR10 [6] dataset. In addition, we have also tried the tiny imagenet [7] dataset, which is a subset of the gigantic imagenet [1] dataset. The reason why we end up not using the tiny imagenet is that it does not provide labels for the test set. Though we have successfully trained our model with the tiny imagenet dataset, we cannot test our results. In addition, we have also tested a simplified implementation of MoCo [10] on MNIST [8] dataset and produced descent results (but in the report we only focus on the CIFAR10 results).

Setup, Training and Evaluation Our MoCo [4] model implementation is based on an open-source implementation of the model on GitHub [11]. Since the model is originally designed for imagenet [1], we have to change the code to make it fit for CIFAR10 [6]. We try two different configurations. The first is to just resizing the CIFAR10 images to 224x224 (size of imagenet images), and the second is to modify the preprocessing part of data transformation and adjusting the input size of the ResNet-50 [5] to let the model directly fit the CIFAR10 images.

MoCo uses a different loss function from most other techniques, namely InfoNCE. It can be expressed as: $L_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}$ where τ is a temperature hyper-parameter. This contrastive loss function has a low value when q is similar to its positive key k_+ and dissimilar to all other keys (considered negative keys for q). It serves as an unsupervised objective function for training the encoder networks that represent the queries and keys.

The core of the approach is to maintain the dictionary as a queue of data samples. The samples in the dictionary are progressively replaced. The current mini-batch is enqueued to the dictionary, and the oldest mini-batch in the queue is removed. The dictionary can always represent a sampled subset of all data, while the outdated encoded keys of oldest mini-batch can be removed.

Using a queue can make the dictionary large, but it also makes it intractable to update the key encoder by back-propagation (the gradient should propagate to all samples in the queue). We denote the parameters of f_k as θ_k and those of f_q as θ_q, we update θ_k by: $\theta_q \leftarrow m\theta_k + (1-m)\theta_q$ where $0 \leq m < 1$ is a momentum coefficient.The momentum update makes θ_k evolve more smoothly than θ_q. By doing this, keys encoded by different encoders can have very small difference.

In the pre-training stage, among AlexNet, ResNet50, and double half-ResNet50, we choose to use the normal ResNet50 as our auto-encoder as it performs relatively well while not requiring too many resources. We set up 64 as the batch size, 0.03 as the learning rate, 0.1 as the learning rate decay rate, and NCE loss as the loss function. We trained our network for 20 epochs on CIFAR10.

For training and evaluating classifier, we applied our saved pre-trained model from the last stage with different learning rate of 0.1, 0.001, and 30 (the optimal learning rate claimed by the MoCo authors).

3 Results

We evaluate the model's performance based on training loss, top-1 test accuracy, and top-5 test accuracy. The ”ep” refers to the number of episode that our ResNet-50 auto-encoder is trained in the pre-training stage. The ”lr” means the learning rate. Lastly, the ”diff config” indicates the second tweaking approach we mentioned above.

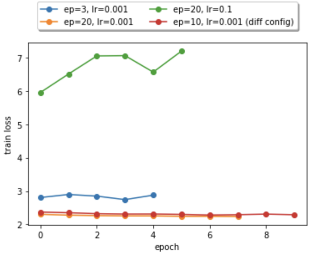

Figure 1: train loss

From the results, we can see that more epochs in the pre-training stage helps improving test accuracy a lot. With only 3 epochs, the model is basically randomly guessing the output. In addition, since CIFAR10 is a much smaller dataset than imagenet, the claimed optimal learning rate of 30 obviously does not fit in our case (its training loss result is not shown because it's too high: around 2000). We need a learning rate of 0.001 to perform optimally. Moreover, the second approach of tweaking the model does not help improve accuracy or loss. The performance is basically on par with the first approach.

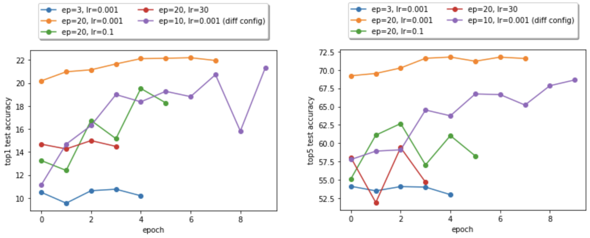

Figure 2: top 1 test accuracy, top 5 test accuracy

4 Discussion

One apparent deficiency of our result is that the top-1 test accuracy is only slightly over 20 percent and top-5 test accuracy is slightly over 70 percent in the optimal setup of our model. For a relatively simple dataset like CIFAR10, this result is much lower than other SOTA results.

We propose several reasons for that: 1. We don't train enough epochs. We only train for 20 epochs, while the MoCo authors trained 200 epochs with 64 GPUs and took 72 hours. We have not tuned the hyperparameters well enough. We notice that even by changing the normalization mean and standard deviation or the random cropping value in data transformation, the loss will become keep increasing. There could be a lot of space for us to tune to make the model fit for CIFAR10. 3. Most importantly, since the training loss of the pre-training stage is pretty low (slightly above 1), we doubt that there are something we miss in the training and evaluation of the classifier. It could be that we miss to adjust the input size somewhere and therefore the model has not been fully adapted to use the CIFAR10 dataset.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[2] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in neural information processing systems*, pages 766–774, 2014.

[3] Naassih Gopee. Classifying cifar-10 images using unsupervised feature & ensemble learning.

[4] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The cifar-10 dataset. *online: http://www. cs. toronto. edu/kriz/cifar. html*, 55, 2014.

[7] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 2015.

[8] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

[9] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016.

[10] peisuke. Reproduction of momentum contrast for unsupervised visual representation learning. *https://github.com/peisuke/MomentumContrast.pytorch*, 2019.

[11] Yonglong Tian. Pytorch implementation of ”contrastive multiview coding”, ”momentum contrast for unsupervised visual representation learning”, and ”unsupervised feature learning via non-parametric instance-level discrimination”. *https://github.com/HobbitLong/CMC*, 2019.

[12] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019.

[13] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.