Final Competition Report: Self-Supervised Learning with Momentum Contrast

Arthur Jinyue Guo (jg5505) * **Yaowei Zong (yz7413)** *

Abstract

This is the Final Competition report of Group 09. We adapted the MoCo structure to our competition. We encountered some issues during the development and eventually did not have enough time for large-epoch pretraining. We achieved an accuracy of 15.98% on the evaluation set. The result is not perfect, but we've done our best as a two people group.

1. Literature Review

Proposed by (He et al., 2019), the Momentum Contrast (MoCo) model achieved a high accuracy in popular self-supervised image recognition tasks.

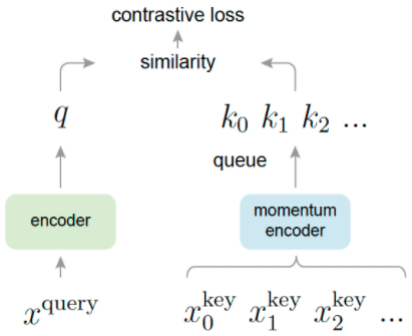

Figure 1. The structure of MoCo

The idea of MoCo is to use an dynamically updated memory bank of negative samples. The queue of negative samples is updated after each mini batch, the new images in the batch will be en-queued, and the oldest batch is de-queued.

The encoder for negative samples is dynamically updated as well. After each mini batch, the momentum encoder will keep a portion of itself, and add a portion of the query encoder. The key encoder will not be updated with gradient descent.

After the encoded latent of query image and key images are computed by the encoders, a similarity is calculated by simply the dot product of query and key. Then, the self-similarity and similarity of q and all k_n forms the contrastive loss. The paper used InfoNCE loss, which can be seen as a log-soft(arg)max of all dot products over the self-similarity. The equation of InfoNCE is:

$$L_q = -\log \frac{\exp(q \cdot k_+/\tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i/\tau)} \tag{1}$$

During training, the backprop only goes to the query encoder. The key encoder is only updated through the momentum process, with the value of momentum set as a hyperparameter.

The MoCo v2 paper by (Chen et al., 2020) introduced several tricks to further improve the performance, including MLP head during training, cosine learning rate schedule, and improved image augmentation with Gaussian Blur. These techniques, along with some hyperparameter changes, is said to be able to improve the performance by about 7% on ImageNet classification task.

The original MoCo repository was created for distributed training. We modified the code for single GPU training on GCP. We created a split batch norm class to imitate the behaviour of the distributed batch norm, and modified the batch shuffling function to work on single GPU.

2. Hyperparameters

We modified several hyperparameters according to the characteristics of our datasets. The original MoCo was pre-trained on IN-1B dataset with about 1 billion images, and the size of memory bank was 65536, which is about 0.007% of the entire dataset. We modified it to 2048, which is 0.004% of our unlabeled set.

Since we have a much smaller memory bank, we want to change the momentum for key encoder. The original paper used momentum of 0.999 to keep the consistency of the

*Equal contribution . Correspondence to: Arthur Jinyue Guo <jg5505@nyu.edu>, Yaowei Zong <yz7413@nyu.edu>.

memory bank. Since we now have a much smaller memory bank, we can let the key encoder update faster without worrying of the consistency. We've set the value to 0.99.

Additionally, we used a backbone model of resnet50, the MLP head and the cosine learning rate from MoCo v2.

3. Augmentation Issue

While modifying the model to fit the shape of our training set, we discovered the huge influence of image augmentation to the final result.

```
RandomResizedCrop(96, scale=(0.2, 1))
RandomApply([ColorJitter( .4, .4, .4,
            .1)], p=0.8)
RandomGrayscale(p=0.2)
RandomApply([GaussianBlur([.1, 2])],
            p=0.5)
RandomHorizontalFlip()
ToTensor()
Normalize()
```

The augmentation used for pre-training.

At first, we removed the `RandomResizedCrop` from the augmentation. We thought it would be useless since our images all have the same shape of 96×96. However, after 100 epochs of training, the model only achieved an accuracy of 2.1% on the validation set.

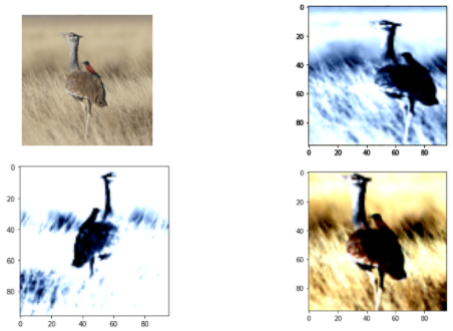

Figure 2. Up left: the original input image. Up right: augmented image with RandomResizedCrop. Bottom left: augmented without RandomResizedCrop. Bottom right: CenteredCrop used for evaluation

The problem of removing `RandomResizedCrop` is that the model will learn to classify the images by the position of the object. The model will quickly reach a high accuracy in pre-training, but actually fails to learn the features of each class, thus eventually fails in supervised training and evaluation.

When we add `RandomResizedCrop` to pre-training and supervised training, and `CenteredCrop` to evaluation, the accuracy improves significantly. Unfortunately, it took us more than a week to figure out this problem, and we did not have enough time for large scale pre-training. We only finished 100 epochs of pre-training, and the result could be much better if we could reach 200 or even 400.

4. Labeling Request

To pick out bad images from the unlabeled dataset that our model struggled to learn, we modified the pretraining code to take a small sample image set of size 12,800. Such sampler is randomly selected from the unlabeled images and pretrained for 100 epochs each.

We repeated the above steps for 40 (512,000 total/12,800 sampler) times and reported the final accuracies, losses, and image indices. Then we sort the results by worst accuracies and losses, remove duplicate indices, pick out the top 12,800 images that give the worst pretraining results for the label request job.

After getting back the image labels, we add those images with the original training set to form a new training set. The new images are renamed following the original training set indices (25600 to 38399). Extra labels are appended to the original train label tensor (`train_label_tensor.pt`).

5. Result Analysis

Our final leaderboard submission achieved an accuracy of 15.98% on validation set. This is not high compared to other groups but is already a huge improvement compared to our second leaderboard submission (only 2.16%).

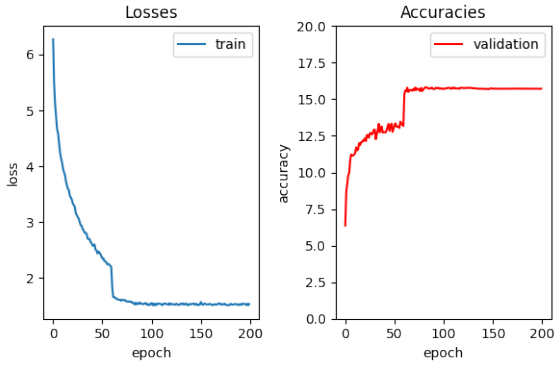

Figure 3. Training Set Losses and Accuracies

We can see from Figure 3 that there is a clear drop in loss and an increase in accuracy around epoch 60. This is because we adopted a scheduled learning rate method that multiplies the learning rate by 0.1 at each milestone. We set 4 milestones at $60, 80, 120, 160$, but we only noticed an improvement at the first milestone.

Figure 4 is a sample feature map of the bird image showed

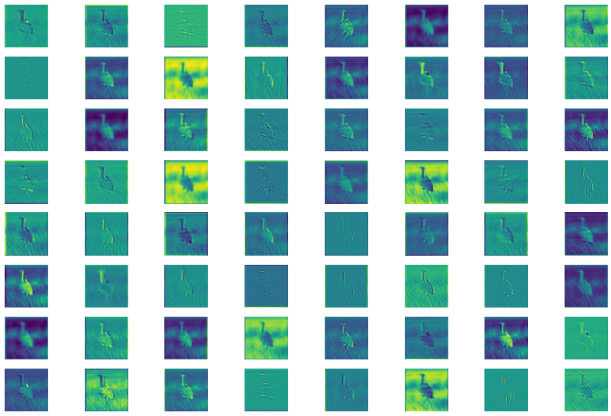

Figure 4. Feature Map after Convolutional Layer 1

above in Figure 2. We can see that our model picked up a few features of the bird such as the outlines of the object, color differences from the background, and feather of the bird.

The result for the model trained with extra labels was not improved (only 5.94%, a 9.83% decrease compared with the base model). We think the reason for the performance decrease is because the model has changed since the label request task and those extra labels are not even distributed hence the model is biased.

6. Future Improvements

Due to the changes in image augmentation, we didn't have much time to perform more experiments with our model. One big factor that affects the accuracy is the epoch number during the pretraining stage. We noticed that in the MoCo paper, changes the epochs from 200 to 800 for the unsupervised pretrain increased the ImageNet linear classification accuracy from 67.5% to 71.1% (Chen et al., 2020). The leading teams in the final leaderboard have very large epoch sizes (ranging from 600 to 1000 epochs). We can try larger epochs to refine our pretraining model.

Another factor we can improve is image augmentation. Since removing the `RandomResizedCrop` results in a significant drop in linear classification accuracy, and one change in MoCo v2 is the "aug+" which adopts blur and stronger color distortion from SimCLR, we can experiment with different image augmentations to find a combination that fits our data set better.

References

Chen, X., Fan, H., Girshick, R., and He, K. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.