Momentum Contrast for Unsupervised Visual Representation Learning(CVPR 2020)

作者：Kaiming He Haoqi Fan Yuxin Wu Saining Xie Ross Girshick

复现人：林弋刚

摘要

MoCo 提出了用于无监督视觉表征学习的动量对比学习方法。从对比学习基于字典查找的角度出发，MoCo 结合队列和编码器动量更新的方法，构建了一个大型且一致的字典，从而促进对比无监督学习。MoCo 在 ImageNet 分类问题中提供了有竞争力的结果，且 MoCo 学习到的表征很好地迁移到下游任务。MoCo 在众多数据集上的 7 个检测/分割任务中胜过有监督的预训练模型。这表明在许多视觉任务中，无监督和有监督的表征学习之间的差距已经基本消除。

本次选取 MoCo 作为复现论文，使用 Mini ImageNet 作为数据集，结合多种数据增强方式，实现单 GPU 训练代码，并将 MoCo 训练得到的模型作为特征提取模型，迁移到下流的图像分类任务进行验证，最后在仅需 15 个 epochs 的情况下取得 54 的 top1 准确率和 81 的 top5 准确率。

关键词：对比学习 动量更新 无监督学习 视觉表征学习

1 引言

无监督的表示学习在自然语言处理中非常成功，但有监督的预训练在计算机视觉中仍然占主导地位。最近的研究表明，对比学习在无监督视觉表征学习取得很好成果，这些方法将字典中存储的特征向量作为负样本。较大的字典有助于更好地提取特征，于是 MoCo 使用队列维护一个字典，当前和最近几个批次的样本特征保留在队列中，允许字典大小远大于批量大小，从而实现字典大小与批量大小的解耦。此外 MoCo 使用动量更新的方法，保证队列中不同批次的样本所用的编码器尽量保持一致。

当代大数据图像领域较难获取标注好的数据集，所以无监督的视觉表征学习逐渐兴起，MoCo 解决了无监督视觉学习中的两个问题，即字典大小受限和大字典中样本所用编码器不一致。本次选取 MoCo 进行复现，考虑到研究方向涉及视频帧的表征提取。由于视频帧数量巨大，故无法标注，所以需要使用无监督学习进行表征提取，故挑选了较为经典的无监督表征提取模型 MoCo 进行复现。

2 相关工作

无监督学习方法一般涉及两个方面：预训练任务和损失函数。预训练任务表示所解决的任务不是真正感兴趣的，而是仅仅为了学到一个较好的数据表征。文中基于这两个方面的相关研究进行讨论。

2.1 损失函数

定义损失函数的常用方式是度量模型预测值与固定目标之间的差异，例如重构输入像素，图片着色，像素块匹配。其他替代方案一般有以下两类：

对比损失：衡量样本在一个表示空间中的相似性，而不是将输入匹配到一个固定的目标，在对比损失公式中，目标可以在训练过程中实时变化。

对抗损失：衡量概率分布之间的差异，经常结合生成类型的任务。近期有研究探索了用于表征学习的对抗方法。

2.2 预训练任务

不同的预训练任务一般会对应不同的损失函数。近期各种预训练任务被提出，如去噪自动编码器、跨通道自动编码器（着色）。一些预训练任务通过变换形成伪标注，如 InstDisc 经过图像变换生成了对应的正样本，该正样本相当于一个标注，有了伪标注即可找到合适的损失函数。

2.3 相近工作

InscDisc 使用 Memory Bank 作为字典，其优点在于字典大小没有限制，其缺点是由于字典中的样本特征是在不同的批次中编码得到的，所以使用的编码器不一样，会导致效果不好；

端到端的训练则直接使用每个 Batch 的样本作为字典，其优点是同一个 Batch 的特征都是由一个相同的编码器得到，有一致性，但是缺点是字典的大小收到 Batch Size 的制约，字典大小受限。

3 本文方法

3.1 本文方法概述

MoCo 的结构如图 1，定义了一个待查询图片 x^{query}，和一个队列 x_0^{key},x_1^{key}... 在该队列中，一般包含了单个的正样本和多个负样本。通过对比损失来学习特征表示。MoCo 解决了 Memory Bank 中存在的问题。Memory Bank 虽然可以保留足够多的样本进行 negative 采样，但是由于 Memory Bank 容量很大，导致了采样的特征具有不一致性，其中存储的图像特征都是过去的 encoder 编码的特征。

鉴于此，MoCo 使用一个队列来存储和采样负样本，队列中存储多个近期用于训练的 batch 的特征向量。队列容量要远小于 Memory Bank，但可以远大于 batch 的容量。本文提出了将队列作为一个动态的，而非静态的 Memory Bank。队列在不断的进行更新，新的训练 batch 入队列后，最老的训练 batch 出队列。这里入队列的并不是图像本身，而是图像特征。

训练时，待查询样本记为 x^q，经过 encoder 网络 f_q 进行编码得到 $q = f_q(x^q)$ 。随后从队列中采样了 $K+1$ 个样本作为 key。这些 key 是用不同的 encoder 网络 f_k 进行编码得到的。由于 encoder 的变化非常缓慢，因此虽然 $k^0, k^1, ...k^K$ 是通过不同 encoder 编码的，编码器导致的差异会非常小。具体的，本文使用一种 Momentum 更新的方式来更新，其参数是对 Query Encoder 的平滑拷贝：

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

其中 $m = 0.999$，表示 θ_k 呈现一种缓慢的变化。

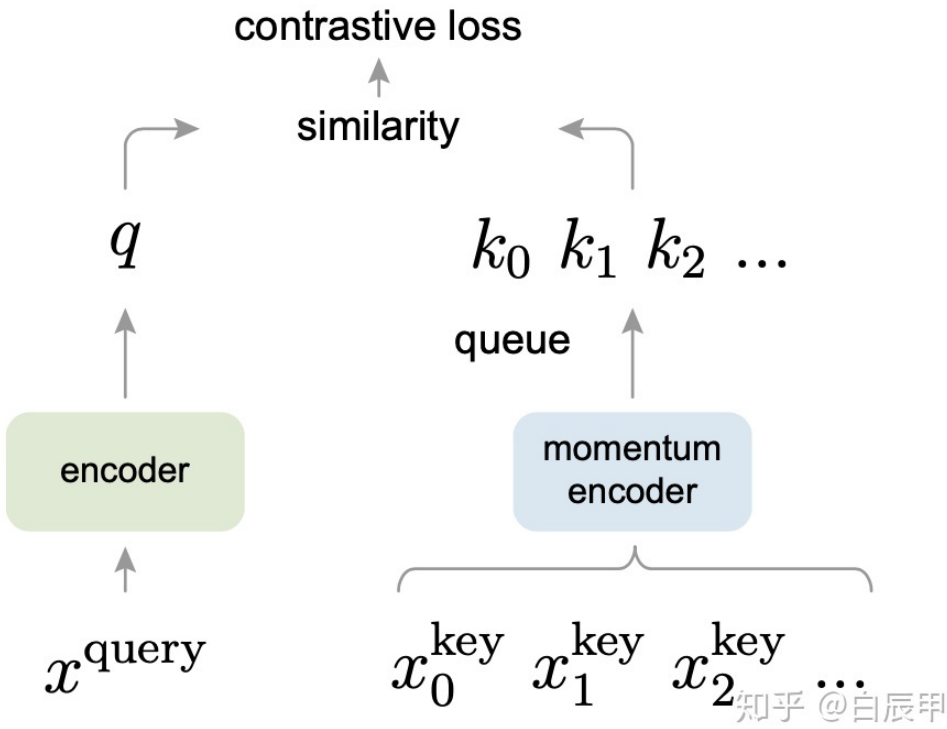

图 1: 方法示意图

3.2 损失函数定义

假设其中只有一个正样本 k_+，其余均为负样本，则根据 InfoNCE 损失，可以将其表示为：

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

此处的正样本可以根据简单的数据增强得到，τ 是温度系数，用来调节更加注重简单的图像还是困难的图像。

4 复现细节

4.1 与已有开源代码对比

使用 Mini ImageNet 数据集进行训练，并实现对应的数据预处理。MoCo 原文使用 ImageNet 作为数据集，本次为了探究较少的数据量对无监督表征学习的效果有无影响以及影响有多大，使用了 Mini ImageNet 进行训练。

使用了不同的数据增强方法。本次复现在原有的两种数据增强的基础上，新增了一种数据增强方法。

实现单卡 GPU 训练版本。原有代码仅支持多卡训练，本次在多卡训练代码的基础上实现了单卡 GPu 训练的代码，多一种选择，可根据实际需求选择使用单卡或多卡版本。

使用不同的预训练模型和训练规则验证 MoCo 用于图像分类下游任务的效果。使用了复现时使用 Mini ImageNet 作为数据集进行训练得到的 MoCo 预训练模型、原作者提供的使用 ImageNet 作为数据集训练得到的 MoCo 预训练模型、无预训练模型分别进行图像分类的训练，并分析仅微调 ResNet50 最后的 fc 层的参数和训练、微调 ResNet 所有层的参数这两种训练规则的效果。Mini ImageNet 是 ImageNet 的子集，而 ResNet50 使用 ImageNet 进行训练，所以官方给出的 ResNet 预训练模型已经针对图像分类任务学会了我们的数据集特征，所以用其和 MoCo 对比效果没有意义，在实验中不纳入对比。

算法伪代码如下：

Procedure 1 Pseudocode of MoCo in a PyTorch-like style.

f_q, f_k : encoder networks for query and key
queue : dictionary as a queue of K keys (C × K)
m : momentum
t : temperature
$f_k.params = f_q.params$
for *x* ***in*** *loader* **do**
    $x_q = augument(x)$
    $x_k = augument(x)$
    $q = f_q(x_q)$
    $k = f_k(x_k)$
    $logits_{positive} = batch\ matrix\ mul(q.view(N,1,C), k.view(N,C,1))$ # negative logits : NxK
    $logits_{negative} = matrix\ mul(q.view(N,C), queue.view(C,K))$ # logits : Nx(1 + K)
    $logits = cat(logits_{positive}, logits_{negative}], dim = 1)$ # contrastive loss, Eqn.(1)
    $labels = zeros(N)$ # positives are the0 − th
    $loss = CrossEntropyLoss(logits/t, labels)$ # SGD update : query network
    $loss.backward()$
    $update(f_q.params)$ # momentum update : key network
    $f_k.params = m * f_k.params + (1 - m) * f_q.params$ # update dictionary
    $enqueue(queue, k)$ # enqueue the current minibatch
    $dequeue(queue)$
end

4.2 实验环境搭建

Pytorch 1.12.1

4.3 创新点

使用更小的数据集并实现相应的数据预处理，验证小数据集对无监督学习的效果影响。

结合了多种数据增强方法，实现更多数据增强的选择。

实现了单 GPU 版本，完善代码。

对多个预训练模型的表征提取效果进行验证，同时验证只微调 fc 层和微调整个模型的效果差异。

5 实验结果分析

综合以下实验结果进行分析：

训练时 (K+1)way 的 top1 与 top5 准确率

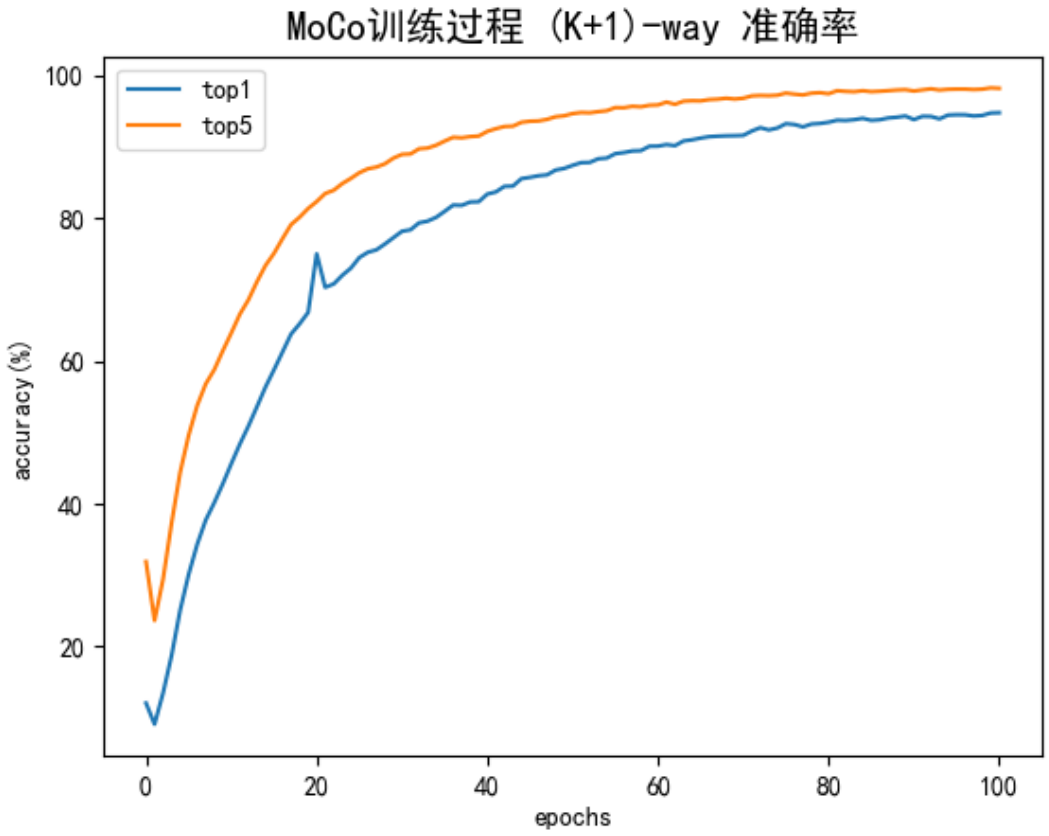

图 2: MoCo 训练时 (K+1)way 准确率

只微调 fc 层，我们的 MoCo 预训练模型、官方的 MoCo 预训练模型和 random init. 预训练模型分别用于图像分类下游任务时 top1 准确率的变化：

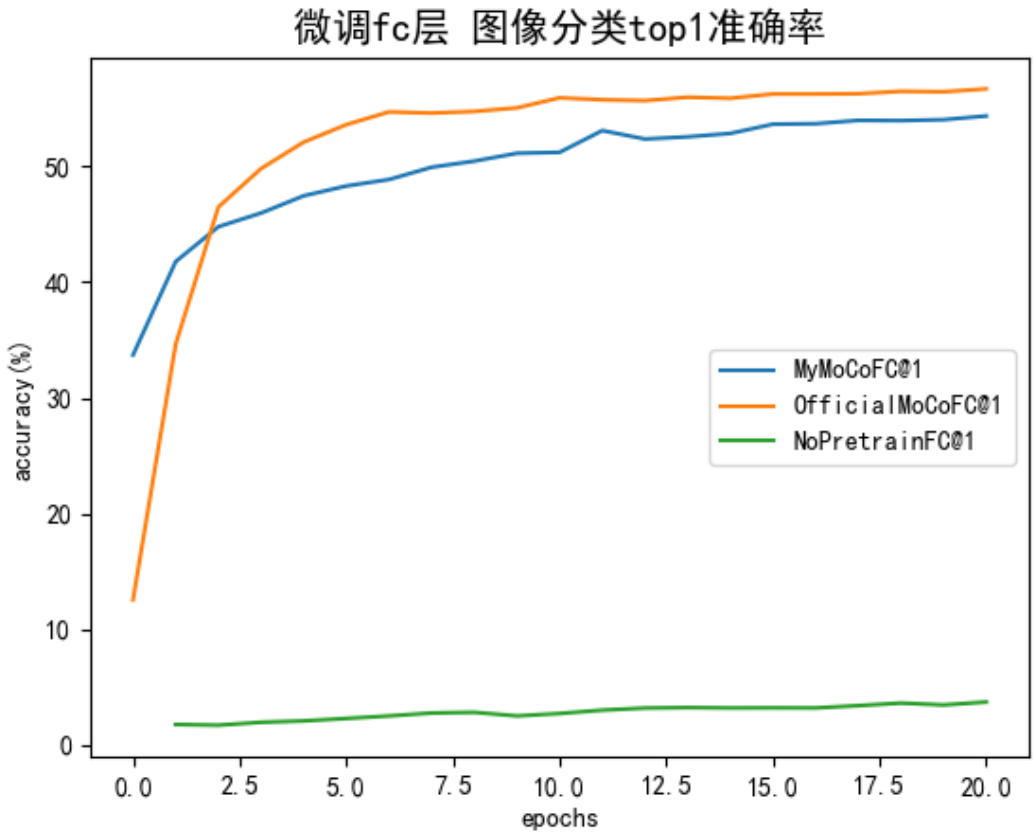

图 3: 微调 fc 层图像分类 top1 准确率

只微调 fc 层，我们的 MoCo 预训练模型、官方的 MoCo 预训练模型和 random init. 预训练模型分别用于图像分类下游任务时 top5 准确率的变化：

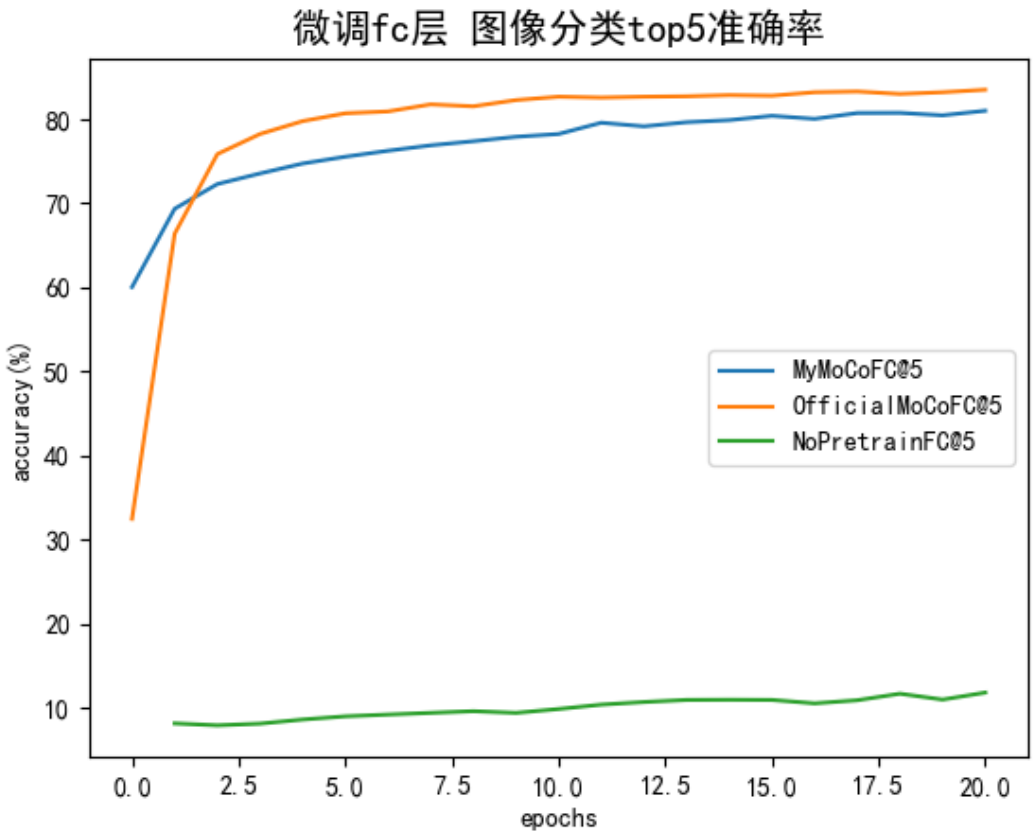

图 4: 微调 fc 层图像分类 top5 准确率

微调所有层，我们的 MoCo 预训练模型、官方的 MoCo 预训练模型和 random init. 预训练模型分别用于图像分类下游任务时 top1 准确率的变化：

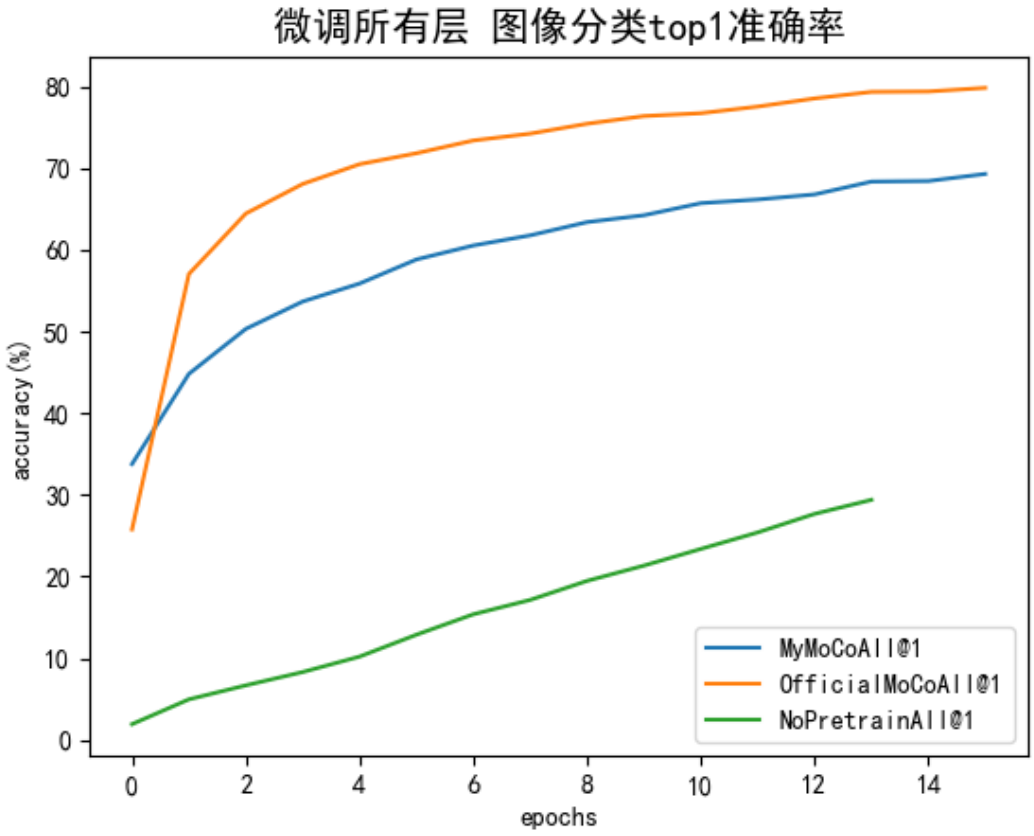

图 5: 微调所有层图像分类 top1 准确率

只微调 fc 层，我们的 MoCo 预训练模型、官方的 MoCo 预训练模型和 random init. 预训练模型分别用于图像分类下游任务时 top5 准确率的变化：

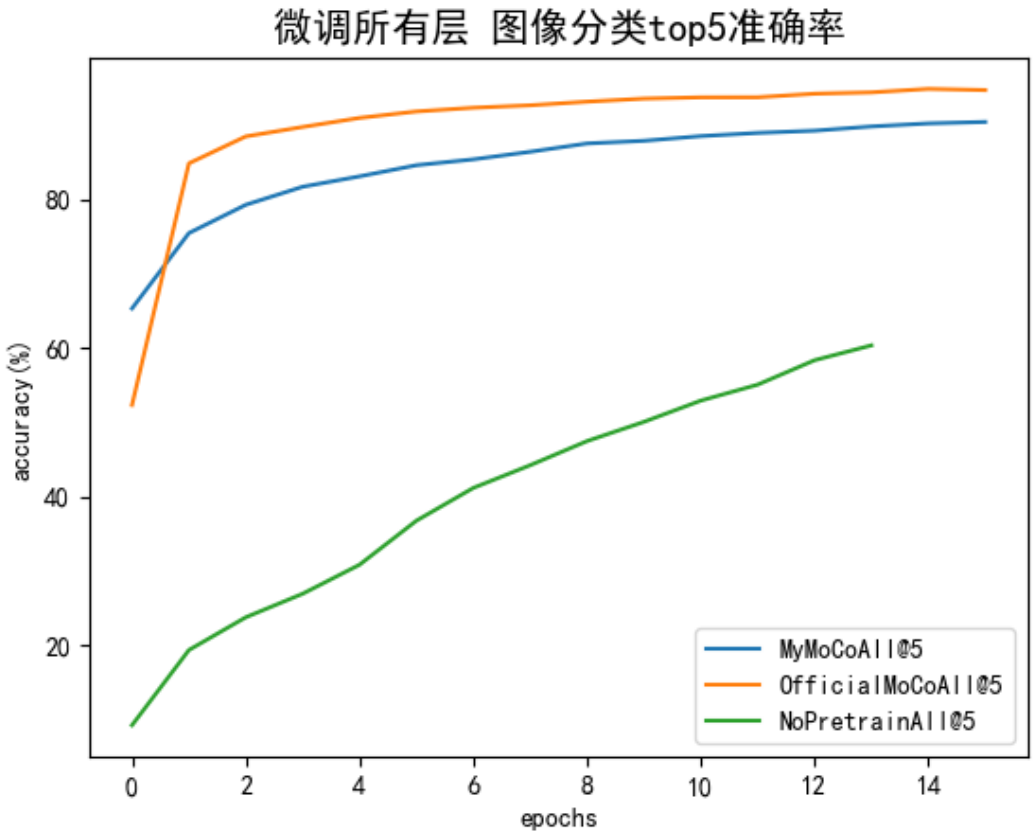

图 6: 微调所有层图像分类 top5 准确率

可以看到，四种情况下，我们的 MoCo 预训练模型的效果都没有官方给出的 MoCo 模型好，官方的 MoCo 模型使用了更大的数据集进行训练，而他的效果更好，也从侧面验证了无监督学习需要大量的图片，才能训练出一个能够较好提取特征的模型。

另外，只能微调 fc 层时可以看到 random init. 几乎无法训练，而微调所有时，则准确率还能正常上升，这是因为只训练 fc 层的参数太少了，拟合不了。

对比只微调 fc 层和微调所有层，可以发现我们的模型和官方的模型差距在仅微调 fc 层时较小，可以说明我们模型的特征提取效果还不错；而在微调所有层时该差距被拉大，这说明我们的 MoCo 模型不仅在特征提取能力上有不足，在特征提取模型的分布上也没有官方的好。所以为了用无监督学习训练一个效果好，且分布也更好的图像特征提取模型，一个优质的大数据集是有必要的。

6 总结与展望

本次复现首先阅读了 MoCo 原论文，调研了文中提到的相关工作，在读懂论文的前提下阅读并理解了作者的源代码，并在数据集、数据增强、单 GPU 训练方面改进了代码，并且使用不同的预训练模型和不同的训练规则，在图像分类下游任务中验证了 MoCo 的表征学习效果。

由于机器受限，本次复现训练时，batch size 无法调到更大，更大的 batch size 能对结果产生多大的影响，在后面解决机器的限制后，我会多做一次实验；同时可以注意到，本次使用 Mini ImageNet 训练的 MoCo 模型用于 Mini ImageNet 图像分类时，其效果较好，但在其他和 ImageNet 无关的数据集上是否能展现一样好的效果，或者能否将在一个数据集上训练得到的表征提取模型用于其他的数据集(作者已验证可以)，后续在时间允许的情况下，我将会做实验进行验证。

由于时间关系，本次复现未能将 MoCo 应用在其他下游任务或其他具体场景中，以此来验证 MoCo 的效果。由于目前的研究方向与视频帧的特征提取有关，所以未来将会尝试把 MoCo 应用到视频帧的特征提取中。