Exploring Simple Siamese Representation Learning阅读笔记

Facebook AI研究院Xinlei Chen与Kaiming He近期的一篇无监督学习又一大作。

习惯看到大神的文章，就拿下来读一读，跟随大神的脚步，修炼AI。

一、论文摘要

在**无监督视觉表达学习**中，**孪生网络**已经成为很多模型的通用架构。这些模型最大化同一图像的两个增广后版本的相似性，使其避免**“崩溃解（collapsing solutions）**”的问题。在这篇论文中，作者报道了令人惊奇的实验结果，简单的孪生网络在没有如下结构的情况下可以学习到非常有意义的表示：**（i）negative sampler pairs(负样本对)，（ii）large bacth(非常大的batch)，（iii）momentum encoder(动量编码器)**。实验发现对于loss函数与架构来说"崩溃"的情况确实存在，但是**"stop-gradient"**的操作对于**防止崩溃**有很大的作用。针对"**stop-gradient**"的含义，作者提供了一个假设，并采用概念证明来验证这个想法。“SimSam”方法在ImageNet以及一些其他的任务中实现了竞争性的结果。我们希望这种简单的baseline会激发人们重新考虑孪生网络在无监督学习中的作用。

SimSiam的主要框架：

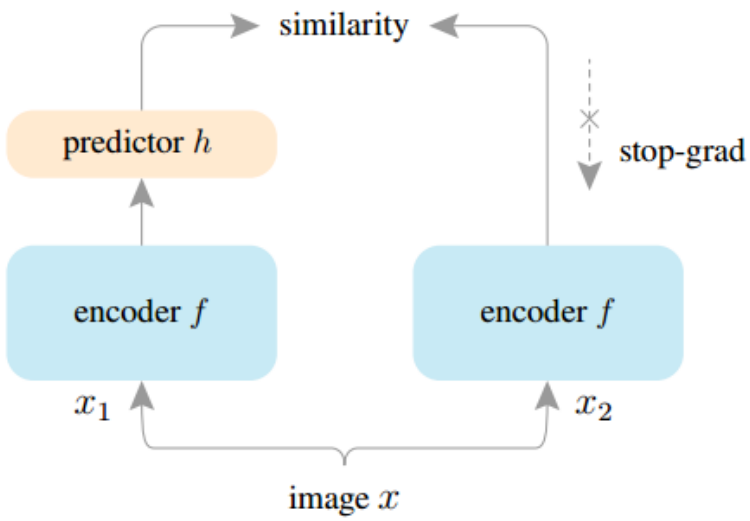


Figure 1. **SimSiam architecture**. Two augmented views of one image are processed by the same encoder network f (a backbone plus a projection MLP). Then a prediction MLP h is applied on one side, and a stop-gradient operation is applied on the other side. The model maximizes the similarity between both sides. It uses neither negative pairs nor a momentum encoder.

二、 论文主要思想

论文提出了一个简单的无监督网络架构，基本网络结构是一个孪生网络，不采用常用的孪生网络的一些基本的方法：**i）negative sampler pairs(负样本对)，（ii）large bacth(非常大的batch)，（iii）momentum encoder(动量编码器)**，也能够得到较好的效果。

通过一个**"stop-gradient"**的操作对于**崩溃**的防止起到了至关重要的作用。zuoyong

以下几种方法的对比：

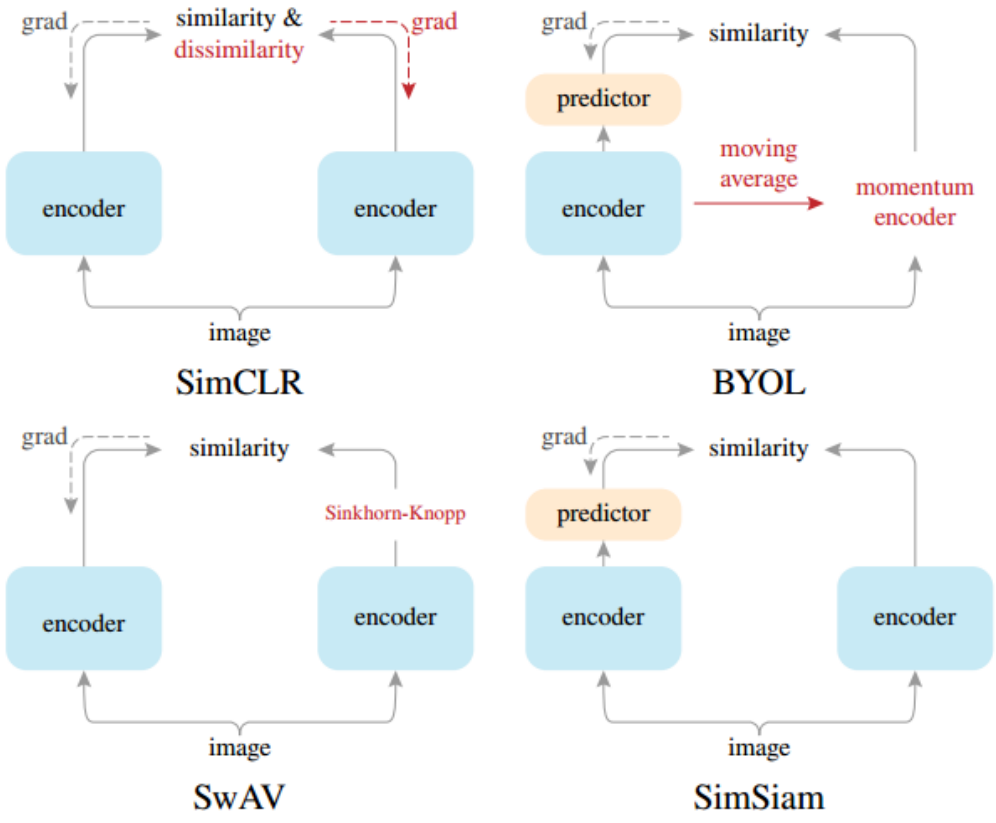


Figure 3. **Comparison on Siamese architectures**. The encoder includes all layers that can be shared between both branches. The dash lines indicate the gradient propagation flow. In BYOL, SwAV, and SimSiam, the lack of a dash line implies stop-gradient, and their symmetrization is not illustrated for simplicity. The components in red are those missing in SimSiam.

本文提出的简单孪生网络的基本结构就是：

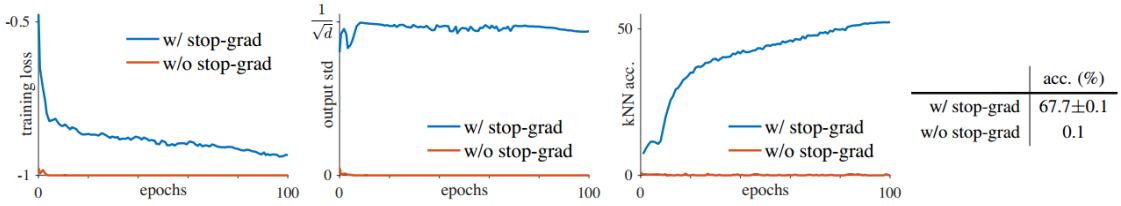
- **没有negative样本对的SimCLR**
- **没有在线聚类的SwAV**
- **没有动量编码器的BYOL**

算法伪代码（非常简单）：

Algorithm 1 SimSiam Pseudocode, PyTorch-like

```
# f: backbone + projection mlp
# h: prediction mlp

for x in loader: # load a minibatch x with n samples
   x1, x2 = aug(x), aug(x) # random augmentation
   z1, z2 = f(x1), f(x2) # projections, n-by-d
   p1, p2 = h(z1), h(z2) # predictions, n-by-d

   L = D(p1, z2)/2 + D(p2, z1)/2 # loss

   L.backward() # back-propagate
   update(f, h) # SGD update

def D(p, z): # negative cosine similarity
   z = z.detach() # stop gradient

   p = normalize(p, dim=1) # l2-normalize
   z = normalize(z, dim=1) # l2-normalize
   return -(p*z).sum(dim=1).mean()
```

其中主要的部分就是加入了**"stop-gradient"**，这个部分作者通过实验证明，这个操作对于防止"崩溃解"有非常重要的作用。



	acc. (%)
w/ stop-grad	67.7±0.1
w/o stop-grad	0.1

Figure 2. **SimSiam with *vs*. without stop-gradient**. **Left plot**: training loss. Without stop-gradient it degenerates immediately. **Middle plot**: the per-channel std of the ℓ_2-normalized output, plotted as the averaged std over all channels. **Right plot**: validation accuracy of a kNN classifier [36] as a monitor of progress. **Table**: ImageNet linear evaluation ("w/ stop-grad" is mean±std over 5 trials).

这里的实验曲线证明，**"stop-gradient"**对于防止"崩溃解"有很重要的作用。从上边实验曲线可以看出，在没有`Stop-gradient`时，优化器迅速找了了一个退化解并达到了最小可能损失-1（这里采用余弦相似的和作为损失）。为证实上述退化解是"崩溃"导致的，作者研究了输出的规范化结果的标准差。如果输出"崩溃"到了常数向量，那么其每个通道的标准差应当是0，上图中间的实验曲线说明这个问题。

上图最右侧的KNN分类准确率更能说明在没有`Stop-gradient`的情况下确实存在着"崩溃解"的问题，使用`Stop-gradient`能够有效的环节这个问题。

文中的实验还对于predictor MLP的作用分析，以及BN层的作用的影响分析。给出了一些数学推导结果。

论文地址：https://arxiv.org/abs/2011.10566

专栏所有文章请点击下列文章列表查看：

知乎专栏：小哲AI专栏文章分类索引跳转查看

AI研习社专栏：小哲AI专栏文章分类索引