





































































































深度学习导论作业4

数据集说明

- **CIFAR-10**是一个经典的图像分类数据集，该数据集共有60000张彩色图像，包含50000张训练集和10000张测试集数据，这些图像的大小是32*32，分为10个类，每类6000张图。

airplane										
automobile										
bird										
cat										
deer										
dog										
frog										
horse										
ship										
truck										

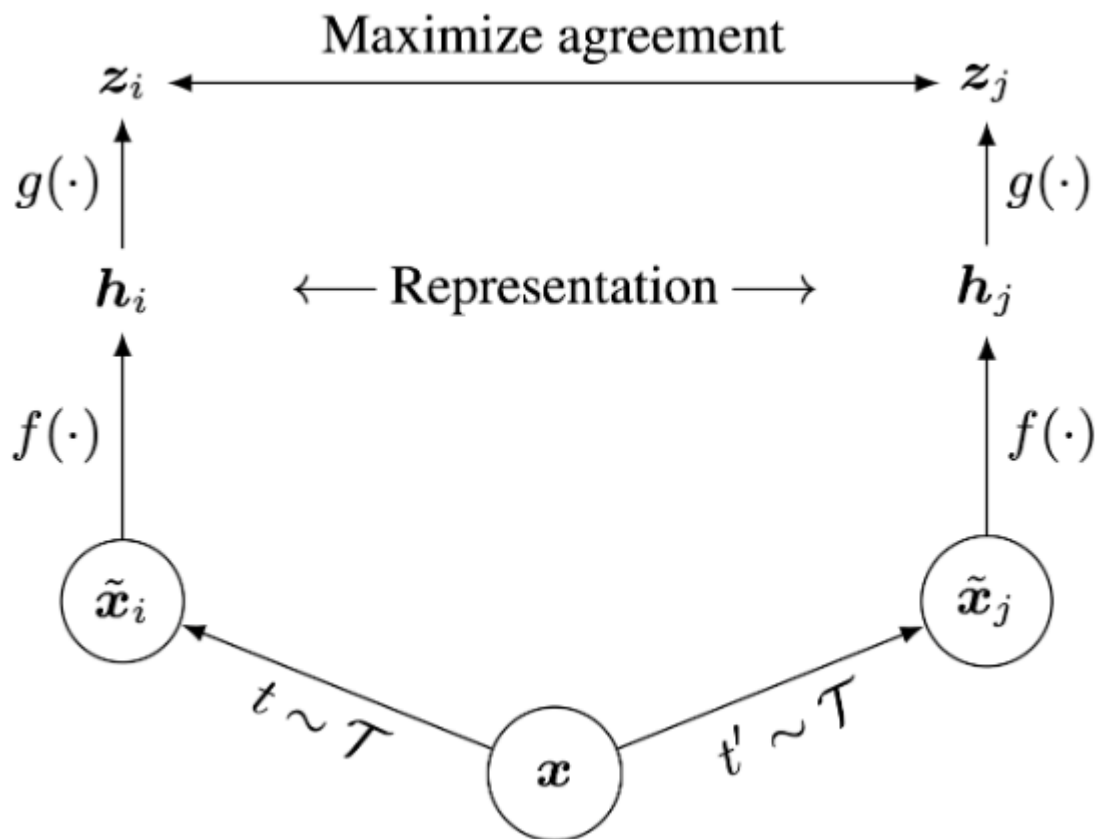
考虑到同学们的硬件条件，本次实验给出的示例dataloader.py中选取了10% CIFAR-10数据用于自监督预训练，同学们可根据自己的硬件条件在合理的运行时间里对采样比例进行调整。

本次实验的目的为利用对比式自监督学习框架（如SimCLR），在CIFAR-10数据集子集上进行训练，并进一步在图像分类任务上进行微调，实现图像分类，分析采用不同数据增强策略等影响。

实验步骤

以SimCLR框架为例下面给出一实验步骤示例，

对于给出的image x , SimCLR用两种不同的数据增强方式 t 和 t' 来生成图像的正样本对 \tilde{x}_i 和 \tilde{x}_j , f 是一个基本编码器网络，从增强数据样本中提取表示向量（representation） h_i 和 h_j , 然后通过一个小型的投影头 g 将向量映射到对比损失应用的空间。对比损失的目标是最大化最终向量 $z_i = g(h_i)$ 和 $z_j = g(h_j)$ 之间的一致性。



数据增强

首先是实现数据增强部分，如示例所示，给出了几种不同的数据增强方式，其中12两种操作的代码已经实现好以供参考，可以自己进行增删测试不同的数据增强结果。

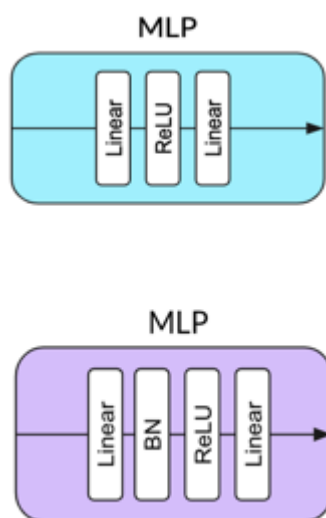
```
def get_augmentations(normalize=True):
    """
    定义SimCLR数据增强
    参数:
        normalize: 是否添加Normalize
    返回:
        transforms.Compose(transform_list): 数据增强操作
    """
    transform_list = [
        # 1. 随机调整大小并裁剪到32x32
        transforms.RandomResizedCrop(size=32),
        # 2. 以0.5的概率水平翻转图像
        transforms.RandomHorizontalFlip(p=0.5),
        # 3. 以0.8的概率应用颜色抖动
        # 4. 以0.2的概率转换为灰度图
        transforms.ToTensor()
    ]
    if normalize:
        transform_list.append(transforms.Normalize( #cifar-10数据集的均值和标准差
            (0.4914, 0.4822, 0.4465),
            (0.2023, 0.1994, 0.2010)
        ))
```

```
return transforms.Compose(transform_list)
```

Base Encoder and Projection Head

接下来的步骤是将基础编码器和投影头应用于增强样本 \tilde{x}_i 和 \tilde{x}_j ，SimCLR中使用了ResNet系列来作为base encoder，可以考虑直接调用如ResNet18作为模型骨架/尝试别的框架或自己实现。注意，在直接调用如ResNet18作为模型骨架时要去掉最后的分类层。

投影头（Projection Head）是一个小型神经网络，用于映射表示向量 h_i 和 h_j 到应用对比损失的空间。在SimCLR中发现，使用非线性投影头可改善其前一层的表现质量。具体来说，他们使用了一个具有一个隐藏层的MLP作为投影头，可以参考带BatchNorm的实现/不带BatchNorm如：



对比损失函数公式

一个包含 N 张训练图像的小批量数据经过数据增强后，共生成 $2N$ 个增强样本。对于每一对正样本 (i, j) （即来自同一原始图像的两个增强样本），对比损失函数的目标是最大化向量 z_i 和 z_j 之间的一致性。

总损失 L 的计算公式为：

$$\mathcal{L}_{NT-Xent} = -\frac{1}{2N} \sum_{i=1}^N \log \frac{\exp(\frac{z_i \cdot z_j}{\tau})}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\frac{z_i \cdot z_k}{\tau})}$$

训练 SimCLR model

接下来，在训练集上训练自监督模型，保存模型权重。其流程如下：

Algorithm 1 SimCLR's main learning algorithm.

```
input: batch size  $N$ , constant  $\tau$ , structure of  $f, g, \mathcal{T}$ .  
for sampled minibatch  $\{\mathbf{x}_k\}_{k=1}^N$  do  
  for all  $k \in \{1, \dots, N\}$  do  
    draw two augmentation functions  $t \sim \mathcal{T}, t' \sim \mathcal{T}$   
    # the first augmentation  
     $\tilde{\mathbf{x}}_{2k-1} = t(\mathbf{x}_k)$   
     $\mathbf{h}_{2k-1} = f(\tilde{\mathbf{x}}_{2k-1})$  # representation  
     $\mathbf{z}_{2k-1} = g(\mathbf{h}_{2k-1})$  # projection  
    # the second augmentation  
     $\tilde{\mathbf{x}}_{2k} = t'(\mathbf{x}_k)$   
     $\mathbf{h}_{2k} = f(\tilde{\mathbf{x}}_{2k})$  # representation  
     $\mathbf{z}_{2k} = g(\mathbf{h}_{2k})$  # projection  
  end for  
  for all  $i \in \{1, \dots, 2N\}$  and  $j \in \{1, \dots, 2N\}$  do  
     $s_{i,j} = \mathbf{z}_i^\top \mathbf{z}_j / (\|\mathbf{z}_i\| \|\mathbf{z}_j\|)$  # pairwise similarity  
  end for  
  define  $\ell(i, j)$  as  $\ell(i, j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbf{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$   
   $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^N [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$   
  update networks  $f$  and  $g$  to minimize  $\mathcal{L}$   
end for  
return encoder network  $f(\cdot)$ , and throw away  $g(\cdot)$ 
```

构建一线性分类器对自监督模型进行评估

接下来利用训练好的自监督模型，将projection head从 SimCLR 模型中移除，并接上一线性层对其进行微调，以完成简单的分类任务。注意，线性层之前的所有层都应该被冻结。

实验分析（任选两点完成即可）

- 在本实验中，不强制要求用SimCLR完成，可以尝试MoCo系列/BYOL模型等对比式自监督，对比它们结果的差异
- 分析采取不同数据增强视图的结果差异
- 分析采用不同的base encoder效果
- 分析实验中超参数的影响，如Loss中的温度系数，batch_size大小（影响自监督中的负样本数量）对于自监督训练的影响
- 分析Projection head的结构对于训练效果的影响（带不带BN？如果不用Projection head的效果怎么样？）
- 对比直接使用一基准分类器对于数据集进行分类和使用自监督+微调基准分类器对于数据集进行分类性能的差异
- 你可以想到的其他分析方向