1 目标任务

设计一套针对图片拼接的神经网络，并尝试能否将其作为一种预训练来辅助作业二中的图片分类任务。

2 图片拼接数据集（CIFAR-10 PERMutation）

在 CIFAR-10 数据集中，如图 1 将一张图片拆分为左上、右上、左下、右下（对应位置标签 $0, 1, 2, 3$）四个等大子图，并随机排列。目标为重排的正确位置标签。可以将此图片拼接问题看成分类问题，类别就是位置。

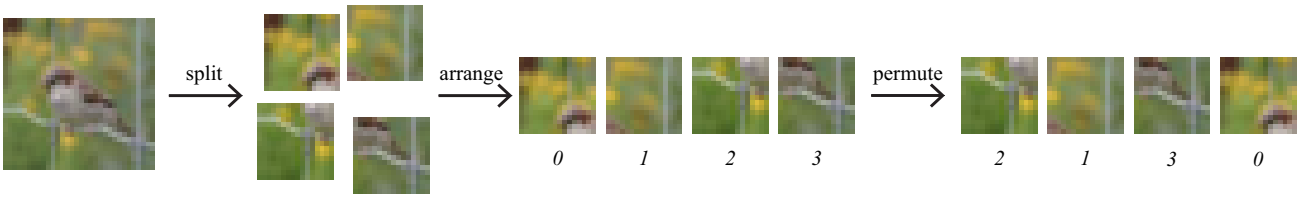


Figure 1: Illustration of splitting one image into 4 patches and permuting them, as the dataset does.

3 CIFAR-10 PERM 数据集图片拼接

3.1 网络结构

参考 DeepPermNet [1] 的模型范式，我采用了平行卷积特征提取、多子图特征聚合、双随机矩阵生成的流程。

卷积提取网络的架构与作业二中的 Simple Classifier 前几层类似，称为 Simple Extractor。特别地，Simple Extractor 进行了尺寸不变 padding 来保留对拼接可能更重要的边缘信息。而后续则使用与 DeepPermNet 中各维度都相同的线性层作为特征聚合器（Aggregator）。其中，第一层线性层将数据维度增大，从而快速，乃至带来冗余地融合多子图信息。最后通过 Sinkhorn [2, 3] 算法进行归一化得到双随机矩阵（每行每列的和固定为 1），作为预测值输出。具体模型结构和运行管线如图 2。

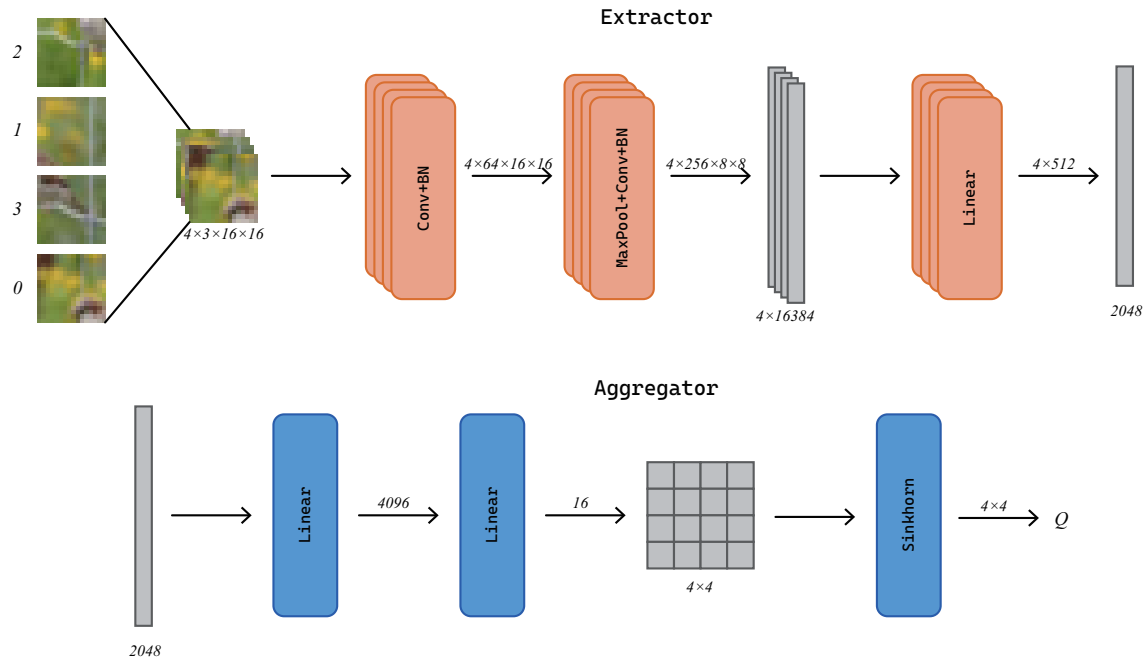


Figure 2: Pipeline and architecture of the method.

全部实验结果详见各 `checkpoint` 目录。

3.2 训练目标

如前所述，本任务可以看作是分类任务，分类的对象是子图，类别是子图的位置。此时的直观想法就是对每张子图计算交叉熵损失进行加和。但实际上，我们计算得到的双随机矩阵 Q 不仅表示每个子图所属位置的概率分布（从每行看），还可以理解为每个位置应放子图的概率分布（从每列看）。换言之，双随机矩阵表达了子图和位置两个随机变量的联合分布。因此，对于训练的目标还需要进一步讨论。

对于联合分布 $p(X,Y)$，有联合熵（joint entropy）

$$H(X,Y) = -\sum_x \sum_y p(x,y) \ln p(x,y),$$

而此时代入真实概率分布 $p(X,Y)$ 和预测概率分布 $q(X,Y)$，就能得到联合分布的交叉熵

$$H(p,q) = \mathbf{E}_p \left[-\ln q(x,y)\right] = -\sum_x \sum_y p(x,y) \ln q(x,y).$$

回顾对每张子图计算交叉熵损失并求和，有

$$H(p(x,\cdot), q(x,\cdot)) = -\sum_y p(x,y) \ln q(x,y),$$

$$\sum_x H(p(x,\cdot), q(x,\cdot)) = -\sum_x \sum_y p(x,y) \ln q(x,y) = H(p,q),$$

因此，在双随机矩阵引入的（或假设的）概率先验下，直接计算每张子图的交叉熵损失并进行加和，就等价于计算联合分布的交叉熵，是符合概率学的损失函数定义方式。

特别地，如果不进行和为一的归一化，损失函数的惩罚也可以引起正确预测“概率”的增长，得到较好的训练结果。在使用梯度性质不佳的归一化导致结果较差时，可以尝试放弃概率可解释性，直接使用未归一化的输出进行损失计算。此时得到的损失只具有训练过程中的相对意义。

4 使用预训练 Simple Extractor 的图片分类

由于图片尺寸和输出尺寸有不同，且模型的前部能够提取具有泛用能力的特征，而后部往往倾向于整合任务相关的特征，我选择只取用 Simple Extractor 的预训练参数，并对提取得到的特征使用新训练的神经网络进行处理分析。

4.1 网络结构

我选择了两种代表性的结构思路进行设计，第一种与作业三的网络结构更相似，第二种则更类似作业二的网络结构。

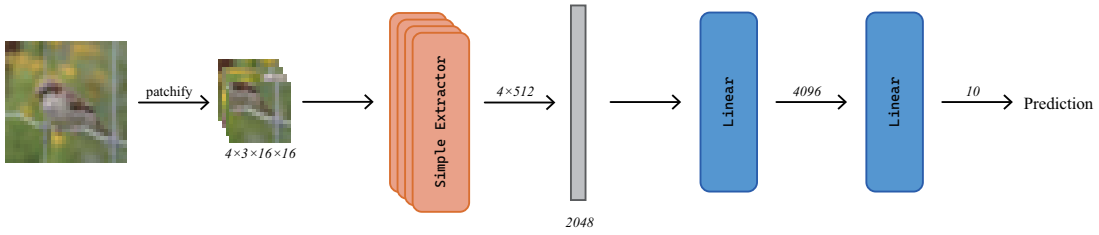
4.1.1 Patched Classifier

一种直接的想法是沿用作业三中的子图结构，从而保证预训练参数与数据之间的适应性，结构见图3(a)。这种结构不能完整地学习图片的全局信息，且在特征整合上也设计得较为简单，因此在分类任务中表现并不好。

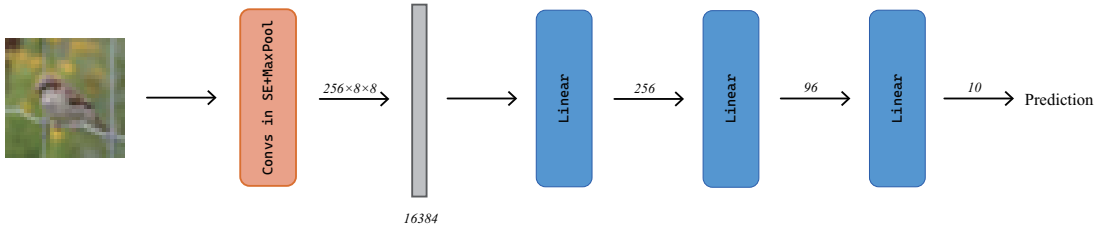
4.1.2 Two-stage Classifier

另一种思路是直接将 Simple Extractor 前几层卷积层作为全图的卷积层使用，之后使用线性层直到最终输出预测，结构见图3(b)。这种方法的可行性在于尽管图片大小不同，卷积核相对于整张图片的尺度是在两个任务中不变的，因此原先用于处理子图的网络同样能用于处理整张图片。值得一提的是，在本结构中，对提取得到的特征没有采取升维操作，而是仿照 Simple Classifier 的网络结构设计了 $256-96-10$ 的逐步降维线性层。

全部实验结果详见各 `checkpoint` 目录。



(a) Patched Classifier.



(b) Two-stage Classifier. SE is short for Simple Extractor.

Figure 3: Architectures of classifiers with pretrained Simple Extractor.

4.2 对抗过拟合

训练过程中出现明显过拟合，亦可使用数据增强和 weight decay 来增加模型的鲁棒性、泛化性，甚至使得模型在未增强的测试集上的准确率高于训练准确率。

5 实验结果

实验结果如表 1, 2 所示。有趣的是，与 DeepPermNet [1] 的结果不同，在本实验的设置下，不使用计算略复杂的 Sinkhorn 反而既提升了训练速度又提高了模型表现。在不必输出预测概率时，或许这是更好的选择。

	With Sinkhorn			Without Sinkhorn								
Model	Pure	WD1	C+F.3	Pure	WD.3	WD.5	WD1	WD3	WD5	WD6	WD7	WD10
Train Loss	0.7691	0.7865	0.8177	0.0038	0.0097	0.0140	0.0254	0.0675	0.1133	0.1348	0.1574	0.2268
Loss	0.8628	0.8731	0.8160	0.5168	0.4064	0.3741	0.3314	0.2809	0.2689	0.2691	0.2721	0.3273
Acc.	0.8802	0.8773	0.9317	0.8880	0.8883	0.8884	0.8921	0.8965	0.8996	0.9002	0.8994	0.8785

Table 1: Result of our image concatenator in aid of anti-overfitting methods. WDX, C, and F.X stand for weight decay of $Xe-3$, random crop, and random flip with rate $0.X$, respectively. Train losses, test losses, and test accuracies are reported.

	From Scratch		Pretrained		Finetune	
Model	Two-stage	Patched	Two-stage	Patched	Two-stage	Patched
Loss						
Acc.						

Table 2: Result of classifiers with Simple Extractor on CIFAR-10. Weights from .

6 训练细节

学习率 $3e-4$ 衰减 0.8。不使用 weight decay。batch size 1024，共训练 $5,000,000$ 迭代。从第 $50,000$ 次起每 $200,000$ 次迭代学习率衰减。数据增强不启用；若启用，方式为随机翻转和随机裁剪。

对分类任务：学习率 $1e-2$ 衰减 0.5。weight decay $1e-3$。

全部实验结果详见各 `checkpoint` 目录。