大作业：实现因果推理算法

卜家梓 *521030910395* 冯超 *521030910413* 汪楠舸 *521030910415*

Abstract

本报告是 AI3610 课程《类脑智能》大作业“用 ColoredMNIST 数据集实现因果推理方法”的项目实验报告。项目小组成员为卜家梓、冯超、汪楠舸，小组成员分工注明在项目分工板块。

在训练深度神经网络的过程中，我们时常假设训练数据与测试数据具有近似的分布，即“独立同分布”准则；然而，模型在实际任务中接受的数据往往不满足这一条件，使其性能大幅下降。因此，我们需要设法提高模型在训练数据分布以外的其他分布上的泛化能力，即分布外泛化能力。

ColoredMNIST 是通过对 MNIST 数据集的标签进行二元化，并执行染色和一定概率的标签翻转操作得到的数据集。在 ColoredMNIST 中，训练集和测试集存在 Out-of-Distribution 现象，样本的分布具有较大的差异。图像的数字形状和颜色产生了虚假的因果关系，使得正常训练的神经网络在测试集上的表现很差，弱于随机初始化。

在本次大作业项目中，我们首先推导了图像颜色与数字形状、标签之间存在的因果关系和因果图，随后利用该因果图推导得到在干预输入图像 X 下标签 Y 的概率分布 $P(Y|do(X))$。随后，我们提出了**损失加权**、**模型加权**、**数据集分布调整**三种利用该分布表达式的方案，均能显著提高卷积神经网络在此数据集上的表现。

按照要求，我们将利用因果关系的三种训练方式也应用到脉冲神经网络模型、忆阻器神经网络模型上，取得了较好的泛化效果。

针对作业的拓展部分，我们总结并实现了四种额外的能够提高模型分布外泛化能力的方法：

1. Distributionally-Robust-Neural-Network.
2. Domain-Adversarial Neural Networks.
3. Invariant-Risk-Minimization.
4. Correct-N-Contrast.

最后，我们对分布外泛化的领域研究方向进行了展望，并针对报告环节中老师和同学们提出的问题进行了一定的改进与补充说明。

1. 项目分工

以下是本组在该项目中的分工表格，附上三位小组成员的签字确认：

表 1: 项目分工

项目内容	卜家梓	冯超	汪楠舸
前期探索	✓	✓	✓
文献调研	✓	✓	✓
因果图推导	✓	✓	✓
loss 加权训练	✓		✓
模型加权训练	✓		
数据集后门调整	✓		✓
SNN 训练	✓		
忆阻器网络训练	✓	✓	
DRO 算法实现	✓		
DANN 算法实现	✓		
IRM 算法实现	✓		✓
CNC 算法实现		✓	
参数调试实验	✓	✓	✓
Presentation	✓	✓	✓
报告撰写	✓	✓	✓
审稿与修改	✓	✓	✓

成员签字 1 _____

成员签字 2 _____

成员签字 3 _____

2. ColoredMNIST 数据集

2.1. 数据集构造

本项目中选取 ColoredMNIST 数据集作为研究对象，下面我们剖析该数据集的构造过程，从而对任务有更全面的把握。

首先，数据集构造函数加载了原版的 MNIST 数据集的训练集，依据其真实标签值 $target$，构造 ColoredMNIST 的二元标签 $binary\_label$：

$$binary\_label = \mathbf{1}(target \geq 5) \tag{1}$$

式中 $\mathbf{1}(\cdot)$ 为条件指示函数，条件满足时为 1，否则为 0。

然后，该 $binary\_label$ 有 25% 的概率会发生 0/1 翻转，作为最终的标签。设 p 为 $[0,1]$ 区间内随机采样的浮点数，则该过程可以表示为：

$$binary\_label = \begin{cases} binary\_label & 0 \leq p \leq 0.75 \\ 1 - binary\_label & 0.75 < p \leq 1 \end{cases}$$

经过随机翻转操作的 $binary\_label$ 将用于指导对应图像的染色。具体地，$binary\_label$ 为 0 的图像中的数字将被染成红色，反之将被染成绿色。我们用二元变量 $colored\_red$ 指示图片的颜色：

$$colored\_red = \mathbf{1}(binary\_label = 0) \tag{2}$$

在数据的划分上，ColoredMNIST 规定 MNIST 训练集的前 20000 张为训练集 1，第 20001~40000 张为训练集 2，剩余 20000 张为测试集。为了在数据集上构造不同的分布，ColoredMNIST 在三个数据集划分上选择了不同的概率阈值，并根据该概率阈值进行随机颜色翻转。

具体而言，三个数据集划分上的翻转概率阈值 $threshold$ 分别为：

$$threshold = \begin{cases} 0.2 & \text{in train1} \\ 0.1 & \text{in train2} \\ 0.9 & \text{in test} \end{cases}$$

执行颜色的随机红绿翻转：

$$colored\_red = \begin{cases} 1 - colored\_red & 0 \leq p < threshold \\ colored\_red & threshold \leq p \leq 1 \end{cases}$$

图1展示了 ColoredMNIST 构造的训练样本：

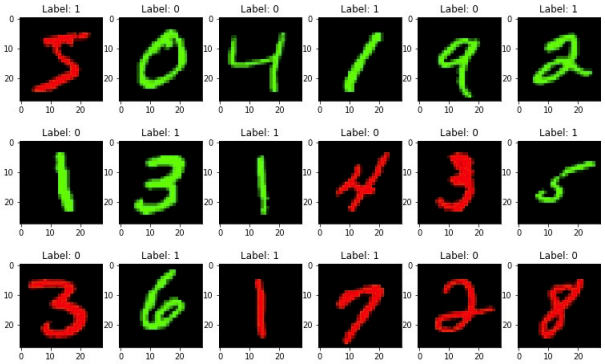

图 1: *ColoredMNIST Dataset*

值得注意的是，由于 25% 的标签经过翻转，神经网络在 ColoredMNIST 数据集上的理论最佳精度为 75%，在评估模型时应该关注实际精度与 75% 的差距绝对值而不是单纯考察精度的高低。

2.2. 因果关系初步观察

回顾构造数据集的全过程，我们不难发现，原 MNIST 图像的信息（$binary\_label$ 等）一定程度上决定了染色情况（$colored\_red$），即使它们都在一定概率下经过了翻转。**然而，我们能得出这样的结论是因为我们在构造数据集过程中处于上帝视角。在实际任务中，我们仅能知道数据集满足一定的分布，而具体的构造过程是不可见的**。所以，我们不能简单地从数据集构造中总结因果关系。

现在，让我们从实际任务的角度出发。观察 ColoredMNIST 数据集，我们注意到图像具有红绿两种颜色，即环境因素；同时图像本身具有信息，即手写数字的形状；此外，每张图像还对应至一个唯一的二元标签。在下一版块中，我们将着重讨论利用以上变量对因果图进行建模，并推导图像信息和二元标签间的因果关系。

3. 因果图的建立

3.1. 变量间的因果关系，因果图

基于上一版块的分析，我们得知了任务中共有三种变量，分别是环境因素（color）、图像字形（image）以及二元标签（label）。我们设环境因素为 E，图像字形为 X，对应的二元标签为 Y。

首先，环境因素一定程度上是图像字形的诱因，这是因为模型可能将颜色和字形之间关联起来产生虚假因果关系（spurious correlation），所以因果图中含有有向边 (E, X)。

在本任务中，图像字形与二元标签之间存在因果关系是任务能够进行的基础假设，故因果图中应含有有向边 (X, Y)。

在环境因素颜色的影响下，模型将会学习到图像颜色与二元标签的因果关联，因此因果图中还应含有有向边 (E, Y)。

明确了上述因果关系后，我们可以建立如图2所示的因果图。

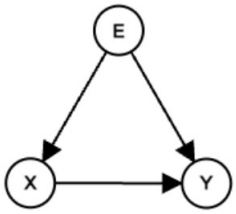

图 2: *Cause-effect Graph*

观察因果图得知，E 与 X、Y 构成分叉结构，X 与 Y 之间存在因果关联。其中，关联流可以通过 $X \to Y$ 的路径在 X 和 Y 之间流动，也可以通过 $X \leftarrow E \to Y$ 的路径在 X 和 Y 之间流动。容易得到，**前者是因果关联，后者是非因果关联，并且是一条后门路径**。

3.2. 推导 $P(Y|do(X))$ 的表达式

接下来，我们实施后门调整，通过干预图像信息 X，阻塞这一条后门路径，只保留 X 和 Y 之间的因果关联。

对图像信息 X 进行干预，即 $do(X)$，此时我们再画出干预分布下的因果图，如图3所示。相较于联合分布的因果图图2，它移除了所有指向干预节点 X 的边，即有向边 (E, X) 被删去。

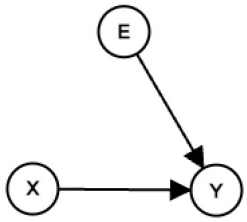

图 3: *Cause-effect Graph(do(X))*

至此，我们成功阻塞了因果图中的后门路径。

在联合分布的因果图中，我们可以计算：

$$P(Y, E|do(X)) = P(Y|X, E) \cdot P(E) \cdot P(X = X_0|E) \tag{3}$$

式中 X_0 为 X 的干预值。而我们已经知道，被干预结点的条件概率分布 $P(X = X_0|E) = 1$，故上式可以简化为：

$$P(Y, E|do(X)) = P(Y|X, E) \cdot P(E) \tag{4}$$

这等价于在干预分布的因果图中计算。也正是因为 $P(X = X_0|E) = 1$，我们才能够忽略有向边 (E, X)。

式4两边对环境因素 E 遍历求和可得 Y 的边缘分布 $P(Y|do(X))$：

$$P(Y|do(X)) = \sum_E P(Y|X, E) \cdot P(E) \tag{5}$$

式5也是作业第一题中需要我们推导的结果。

4. 因果图辅助卷积神经网络训练

4.1. Baseline：直接训练 CNN

我们直接在 ColoredMNIST 数据集上训练自定义的卷积神经网络 MyModel，训练结果如表2所示（我们给出的数据均为最佳表现）。

表 2: *baseline*

Method	**Epochs**	**Accuracy(%)**
CNN	50	35.395

由于网络学习到虚假的因果关系，其表现弱于随机初始化网络的表现（50%），符合猜想。

4.2. 因果关系解读

式5给出了图片信息 X 与二元标签 Y 之间因果关联的表达式。该式指出，在建模 X 与 Y 的关系时，我们需考虑环境因素 E 的每一种取值下 X 与 Y 的关联，并将它们按照 E 的各种取值比例进行加权求和。

依照上述思路，我们提出了三种建模式5中因果关联的方法：

1. **双 $loss$ 加权** $(bi-loss)$：训练单一模型。在训练模型时，我们制作每个 $batch$ 经过颜色翻转的 $batch_i$，分别通过网络，将所得的 $loss$ 和 $loss_i$ 按照按照训练集中两种颜色的比例加权求和，用于网络的训练。
2. **双模型加权** $(bi-model)$：训练两个模型。预先将训练集拆分为仅含有红色图像的 $train_r$ 和仅含有绿色图像的 $train_g$，分别用于两个模型的训练；在测试时，将两个模型的输出按照训练集比例加权求和作为最终输出。
3. **数据集分布调整**：我们通过对比模型可以拟合的 $P(Y|X)$ 表达式和与颜色无关的 $P(Y|do(X))$ 表达式，发现了只需要调整数据集分布使得两式近似相等即可使得模型在训练集上学习到接近 $P(Y|do(X))$ 的效果。

下面我们详细介绍这三种方法的实现，具体代码细节可以参考我们的提交文件。

4.3. 双 $loss$ 加权

4.3.1. 训练细节

我们对训练过程进行更改。首先，我们将训练集的每个 $batch$ 整体进行颜色翻转，该过程可以通过交换 $batch$ 中每张图像的红绿颜色通道实现：

$$batch_i = \texttt{Color\_Reverse}(batch) \tag{6}$$

将两者含有的图像 img 和 img_i（经过反转）分别通过模型，得到对应的输出 $output$ 和 $output_i$：

$$output = model(img) \tag{7}$$

$$output_i = model(img_i) \tag{8}$$

$batch$ 和 $batch_i$ 含有的 $target$ 是一致的，与上面得到的输出计算两种 $loss$，损失函数选用交叉熵 $\texttt{CrossEntropy}(\cdot)$，分别记为 $loss$ 与 $loss_i$：

$$loss = \texttt{CrossEntropy}(output, target) \tag{9}$$

$$loss_i = \texttt{CrossEntropy}(output_i, target) \tag{10}$$

按照式5原理，我们应该将同一张图像的红色版本的损失 $loss_r$ 和绿色版本的损失 $loss_g$ 按照训练集中两种颜色的比例加权，即：

$$loss = ratio_r \cdot loss_r + ratio_g \cdot loss_g \tag{11}$$

通过遍历数据集，我们发现 ColoredMNIST 数据集中红色图像占比与绿色图像占比基本相同，因此式11可以改写为：

$$\begin{aligned} loss &= ratio_r \cdot loss_r + ratio_g \cdot loss_g \\ &= 0.5 \cdot (loss_r + loss_g) \\ &= 0.5 \cdot (loss + loss_i) \end{aligned}$$

于是我们可以利用之前求得的 $loss$ 和 $loss_i$ 计算最终的 $loss$ 值，反向传播用于模型的训练。

在双 $loss$ 加权训练中，我们采用 $batch\_size = 64$，$lr = 0.0001$ 训练配置。

4.3.2. 实验结果展示

表3展示了双 *loss* 加权方法与直接训练模型在 ColoredMNIST 上的精度对比。

表 3: *bi-loss VS baseline*

Method	**Epochs**	**Accuracy(%)**
CNN (baseline)	50	35.395
CNN + bi-loss	30	**73.495**

双 *loss* 加权训练方法显著改善了模型性能，已经比较接近理想水平 75%，并且能有效加速模型的收敛。

4.4. 双模型加权

4.4.1. 训练细节

我们将训练集提前划分为仅含有红色图像的 $train_r$ 和仅含有绿色图像的 $train_g$，将两个子训练集按元素配对进行同步，分别用于训练专用于分类红色图像的 $model_r$ 和专用于分类绿色图像的 $model_g$。这个过程可以表示为：

$$train = train_r \cup train_g \tag{12}$$

$$model_r.\texttt{fit}(train_r) \tag{13}$$

$$model_g.\texttt{fit}(train_g) \tag{14}$$

进行测试时，对于测试集中的一张图像 img，我们将其分别通过两个模型，计算其输出 $output_r$ 和 $output_g$：

$$output_r = model_r(img) \tag{15}$$

$$output_g = model_g(img) \tag{16}$$

将两模型的输出按照训练集中两种颜色的比例进行加权，作为最终的预测结果：

$$\begin{aligned} output &= ratio_r \cdot output_r + ratio_g \cdot output_g \\ &= 0.5 \cdot (output_r + output_g) \end{aligned}$$

与双 *loss* 加权类似，这里的颜色比例也可以使用 0.5 来近似，达到简化计算的目的。

在双模型加权训练中，我们采用 $batch\_size = 64$，$lr = 0.001$ 的训练配置。

4.4.2. 实验结果展示

表4展示了双模型加权方法与直接训练模型在 ColoredMNIST 上的精度对比。

表 4: *bi-model VS baseline*

Method	**Epochs**	**Accuracy(%)**
CNN (baseline)	50	35.395
CNN + bi-model	30	**66.213**

双模型加权方法也能够明显地帮助模型提高在训练集分布之外的泛化能力，最终表现略低于双 *loss* 加权方法。在收敛速度上，双模型加权与双 *loss* 加权基本相同。

4.5. 数据集分布调整

4.5.1. 分布调整

让我们先回顾依据后门准则得到的式5：

$$P(Y|do(X)) = \sum_E P(Y|X,E) \cdot P(E)$$

由于 $do(X)$ 阻塞了因果图中的后门路径 $X \leftarrow E \rightarrow Y$，分布 $P(Y|do(X))$ 与环境因素颜色 E 的分布是无关的，也即 $P(Y|do(X))$ 表示的 X 与 Y 之间关联对于颜色分布不同的训练集、测试集均适用，这也是我们尝试拟合 $P(Y|do(X))$ 的原因。

对于数据集 (X, Y)，我们直接训练神经网络将会得到 $P(Y|X)$ 的拟合，在因果图图2中，根据局部马尔科夫假设，我们可以得到：

$$\begin{aligned} P(Y|X) &= \sum_E P(Y,E|X) \\ &= \sum_E P(Y|X,E) \cdot P(E|X) \end{aligned}$$

对比该式与式5，我们可以发现两式只有最后

一个因子有差异，如果我们可以对训练集上的数据分布做适当的调整，使得训练集上 $P(E|X)$ 分布与 $P(E)$ 一致，那么就能使得模型在训练集上拟合到的 $P(Y|X)$ 近似目标分布 $P(Y|do(X))$。

通过提前遍历数据集，我们已经知道 $P(E)$ 近似为一个均匀分布，即红色图像的数量与绿色图像的数量基本相同。在此基础上，我们只需将 $P(E|X)$ 也调整至均匀分布即可。

具体的，我们需要保证数据集中每一张图像都存在一张红色版本和一张绿色版本，这是容易实现的，可以采取和双 *loss* 加权相似的染色方法。换言之，**数据集分布调整方法和双 *loss* 加权方法本质上是统一的**。

此外，我们还探索发现，当我们保证训练集上 $P(Y|E)$ 是均匀分布时，即每个标签中均有红色图像数目等于绿色图像数目时，也能达到和上述数据集调整类似的效果，并且这种调整**不需要加入染色图像的额外信息，仅仅是对训练集图像进行重复**。其理论依据我们还在持续探索中。

在数据集分布调整训练中，我们采用 $batch\_size = 32$，$lr = 0.0001$ 的训练配置。

4.5.2. 实验结果展示

上述第一种数据集调整方法本质上与双 *loss* 加权方法一致，表5展示第二种数据集调整方法与直接训练模型在 ColoredMNIST 上的精度对比。

表 5: *adjust VS baseline*

Method	Epochs	Accuracy(%)
CNN (baseline)	50	35.395
CNN + adjust	30	**72.205**

数据集调整在未对正常训练过程进行更改的前提下改善了模型的泛化性能和收敛速度，但可能需要严谨的理论证明和大量实践依据作为指导。

5. 因果图辅助脉冲神经网络训练

5.1. 脉冲神经网络简介

脉冲神经网络（Spiking Neural Networks，SNNs）的灵感来源于生物神经系统中神经元的工作方式。在脉冲神经网络中，神经元的活动被建模为离散的脉冲事件，通常称为“脉冲”或“尖峰”。神经元通过在空间和时间上的连接来相互通信，而信息传递则通过脉冲的传播来完成。每个脉冲表示神经元的电活动在时间上的瞬时变化，这种方式更符合生物神经系统中神经元之间的通信方式。

5.2. 实现细节

在此处我们对实验的细节进行简要阐述，具体的细节请参见我们的代码实现。在这个实验里，我们仅仅将原先的 CNN 网络改变为 SNN 网络，其他的保持不变，而我们 SNN 网络的具体结构是基于官网上提供的基础 SNN 网络进行改良的。

5.3. 实验结果展示与分析

将上述几种方法中的 CNN 网络改变为 SNN 网络后，在保持其他超参数保持不变的情况下进行实验，具体结果如表6所示。

表 6: *SNN*

Method	Epochs	Accuracy(%)
SNN (baseline)	50	30.990
SNN + bi-loss	50	**70.625**
SNN + bi-model	50	63.087
SNN + adjust	30	68.932

观察上述实验结果，我们分析得到以下结果：

首先，SNN 的整体效果与 CNN 相比略有下滑，收敛速度也相对减慢，这可能是由于脉冲神经网络的离散脉冲信号和时序性质等网络内在因素导致的。另外，在进行离散事件的处理时，SNN 可能会引入信息的丢失和模糊性，从而影响了模型的精度。相较之下，CNN 在连续的激活值上进行操作，更适合处理连续性的数据，尤其在图像处

理等任务上表现出色。

但我们也不难看出，我们的优化方法在 SNN 上也产生了较好的效果，并且对整体效果的下滑有一定的抑制作用。bi-loss 的优化方法在 SNN 网络中再次取得了最佳的效果，说明此方法具有较强的普适性和鲁棒性。

6. 因果图辅助忆阻器神经网络训练

6.1. 忆阻器综述

6.1.1. 忆阻器简介

忆阻器，全称记忆电阻器 (*Memristor*)。它是表示磁通与电荷关系的电路器件。忆阻器也有很多存储优点：比如尺寸小、速度快、低功耗、与 CMOS（互补式金氧半导体）工艺可兼容等。

6.1.2. 忆阻器模拟实验

此处用到的对忆阻器的模拟基于 memtorch 实现。memtorch 是基于忆阻深度学习系统的仿真框架。memtorch 的具体使用方法我们在小作业中已经习得，即将预训练的 CNN 模型转换为 MNN 模型，此处我们将前述三种方法训练得到的 CNN 模型分别转换为 MNN 模型并测试其表现。

6.2. 实验细节说明

在此处我们对实验的细节进行简要阐述，具体的细节请参见我们的代码实现。我们分别利用在前文中实现 CNN 获得的对应 model，转化为 memtorch 模型后进行测试。为体现差异性，我们此处首先使用理想模拟，以获取标准 MNN 在对应 model 上与 CNN 实现的差异。其次，我们针对其中综合表现效果最好的模型进行非理想测试，探究不同因素对实验的影响结果。

6.3. 实验结果展示与分析

6.3.1. MNN 理想模拟

我们在理想情况下对 MNN 进行模拟，得到了表7所示的实验结果。观察表7中数据，我们得到

表 7: *MNN*

Method	**Epochs**	**Accuracy(%)**
MNN (baseline)	50	60.365
MNN + bi-loss	30	**73.128**
MNN + bi-model	30	71.845
MNN + adjust	30	50.050

以下结论：

首先，MNN 在 bi-loss 和 bi-model 的方法上保持相对稳定，说明忆阻器的实现是相对稳定的。

其次，MNN 在 baseline 上出现了较大的异常上升，在 adjust 上的性能有大幅的下降。在重复多次实验排除偶然误差的情况下，我们考虑可能是忆阻器对数据的模拟过程与神经网络的实现过程存在一定差异，最终导致一些在神经网络上有较大成效的方式，性能会有一定下滑。

baseline 虚高的原因可能是因为原始 CNN（采用 CNN baseline 的训练结果）并没有良好学习到具体特征，导致忆阻器模型生成的输出大多在 0.5 上下浮动，忆阻器精度较低产生误判，导致忆阻器获得的 baseline 数据呈现虚高现象。

6.3.2. 对 *bi-loss* 进行非理想模拟

我们对效果最好的 bi-loss 模型进行非理想模拟探究，实验结果见表8。扩展实验的代码比较琐碎，我们已经上传至 `https://jbox.sjtu.edu.cn/l/81ZD59`，不再随作业项目一起提交。

表 8: *MNN* 非理想模拟

Method	**Epochs**	**Accuracy(%)**
理想状态	30	73.128
设备故障	30	68.715
设备损耗	30	53.468

我们发现，在非理想的情况下 MNN 的准确率会有所下降，这同样符合我们的猜想。其中设备的损耗导致的性能比起偶发性的故障更加严重，造成了接近 20% 的性能下滑。

7. 提高部分前瞻

在本项目中，我们在完成了基础部分所有要求的前提下，总结并实现了四种额外的提高方法：

1. **DRO**（Distributionally Robust Stochastic Optimization[1]）
2. **DANN**（Domain-Adversarial Neural Networks[2]）
3. **IRM**（Invariant Risk Minimization[3]）
4. **CNC**（Correct-N-Contrast[4]）

在这四种方法中，仅有 **DRO 需要明确知晓 spurious feature 是颜色 color，而其余三种方法 DANN、IRM、CNC 并不需要了解这一层面的信息**，具有更强的泛用性。

在接下来的几个版块中，我们会详细介绍这些方法的原理与实现。

8. Distributionally Robust Stochastic Optimization

8.1. 方法简介

对于一般的机器学习分类任务，模型需要根据一个给定的有限训练集去做出分类决策，即模型只能依据有限的训练样本对数据的真实分布做出估计，因此模型的性能往往受限于训练集样本的数量和多样性。

分布鲁棒优化（Distributionally Robust Optimization，DRO）考虑现有样本组成的分布作为球心，ϵ 为半径的一个 wasserstein 空间球体内的所有分布。真实分布理论上不会偏离训练集分布很远，因此在 ϵ 值合理的前提下，可以认为数据的真实分布也被包含在这个空间球体内。那么，我们只需要保证模型在整个空间球体内效果最差的分布下得到最低的风险，就可以保证真实分布下模型性能的下界，即提供一个 lower bound。

Group DRO 考虑将训练集数据划分为若干组来模拟 wasserstein 球体内的各种分布，最小化模型在这些训练组上的最差情况损失（worst case loss），能够一定程度上避免模型因学习到 spurious correlations 而在特定训练组上产生高 loss 现象。

后续的研究还表明，向 Group DRO 算法中添加正则项能够在进一步提高模型在最差训练组（worst-group）上的测试准确率的同时，维持较高的平均测试准确率。

8.2. 训练细节

根据 Group DRO 的流程，我们首先将训练集按照颜色 color 和标签 target 划分为 4 个 group：红 & 0，红 & 1，绿 & 0，绿 & 1。这 4 个训练组依次标号为 0,1,2,3。

执行伪代码1所示的训练算法。

Algorithm 1 Group DRO on ColoredMNIST

Initialization: $\lambda_2 \leftarrow 0.001, \eta_q \leftarrow 0.001, q \leftarrow [\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}], epoch \leftarrow 0, model$
repeat
  $epoch \leftarrow epoch + 1$
  Randomly choose a group index i.
  Randomly sample 1 sample from $group[i]$.
  Train $model$ with the sample and obtain $loss$, calculate $l_2 = \sum_{p \in model} p^2$.
  Update $loss \leftarrow loss + \lambda_2 \cdot l_2$.
  Update $q[i] \leftarrow q[i] \cdot e^{\eta_q \cdot loss}, q \leftarrow \frac{q}{\sum_j q_j}$.
  Update $lr \leftarrow 0.001 \cdot q[i]$.
until $epoch = MAXSampleNumber$
Output: Distributionally robust model $model$

根据该算法，模型在从各个训练组中随机采样训练的同时，动态地更新这些训练组的权重 q 和训练的学习率 lr，引导模型更加关注在表现较差的训练组上的学习，这符合 Group DRO 的核心思想，即最小化在最差训练组上的损失。

在 DRO 训练中，我们采用 $batch\_size = 32$，初始 $lr = 0.001$ 的训练配置。

8.3. 实验结果展示

表9展示了 DRO 算法在 ColoredMNIST 上的精度表现。

DRO 算法展现出了极高的训练效率，仅仅训练 10000 个样本（尚不足一个训练集的样本数）后，便可以达到相当高的精度。

表 9: *DRO*

Method	Sample Number	Accuracy(%)
DRO	10000	73.310

9. Domain-Adversarial Neural Networks

9.1. 方法简介

9.1.1. 域适应

域适应（Domain Adaption[5][6]）是迁移学习中一个重要分支，它的目标是把具有不同分布的源域（Source Domain）数据和目标域（Target Domain）数据映射到同一个特征空间。

因此，我们在源域数据上训练得到的分类器，也可以用于目标域数据的分类。

9.1.2. 域迁移对抗神经网络

将源域数据和目标域数据映射到同一个特征空间，关键步骤是**从源域数据和目标域数据中提取共同的特征**。传统的域适应算法一般选取固定的某些特征，现在我们更多地选用神经网络帮助我们提取更高阶的特征。

域迁移对抗神经网络（Domain-Adversarial Neural Networks，DANN）首次将对抗学习的思想融入迁移学习中，该网络结构能够提取源域与目标域之间“可供迁移的特征”，这些特征满足：

1. 域不变性（Domain-invariance）：对于某一数据提取的特征，我们无法根据该特征判断原数据属于原域还是目标域。

2. 可分性（Discriminativeness）：利用这些特征，我们可以较好地在源域上完成数据分类任务，即特征具有足够的代表性。

根据这两个原则，DANN 的损失应包含两类：**源域分类损失，域判别损失**。

图4展示了 DANN 的网络结构。

DANN 的主要结构包含以下三个部分：

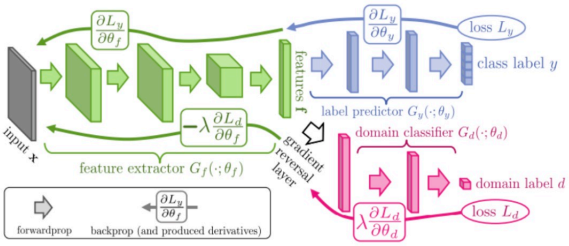

图 4: *DANN*

1. 特征提取器（Feature Extractor）：图4中绿色结构，用于提取源域数据和目标域数据中的可迁移特征。

2. 源域分类器（Label Predictor）：图4中蓝色结构，根据提取的特征对源域数据进行分类。

3. 域判别器（Domain Classifier）：图4中红色结构，根据提取的特征对数据的来源进行分类，即判别该数据属于源域还是目标域。

DANN 的核心思想类似对抗生成网络 GAN[7]，在训练源域分类器的同时设法“欺骗”域判别器，尽可能混淆它对于数据来源的判断。

特征提取器和源域分类器构成一个正常的前馈神经网络；在特征提取器和域判别器之间，DANN 引入了一个梯度反转层（gradient reverse layer, GRL）。在训练过程中，对于来自源域的数据，网络将会**最小化它们的源域分类误差**；对于来自源域和目标域的所有数据，由于梯度反转层的存在，网络将会**最大化它们的域判别误差**。

DANN 能够在尽可能使得数据来源不可分的前提下优化源域数据的分类精度。在源域分类器和域判别器的“对抗”训练下，特征提取器能够提取到在两个域上都存在的更精确的可迁移特征。

具体的，我们如果设置源域为原版的 MNIST 训练集，目标域为 ColoredMNIST 训练集，那么我们希望 DANN 能够提取的**可迁移特征即为手写数字图像的字形**，图像的颜色则作为不可迁移特征在训练过程中被 DANN 滤除。

9.2. 训练细节

值得注意的是，在进行训练前，我们需要将原版 MNIST 训练集的标签 $target$ 预处理成二元标签 $binary\_label$ 的形式，其处理方式类似于 ColoredMNIST 数据集获取 $binary\_label$ 的方法，但不需要经过标签翻转，具体内容读者可以参见本报告第一版块，此处不再赘述。

在训练过程中，我们还动态地控制梯度反转层的反转率 α。设当前 $epoch$ 的训练集遍历进度为 idx，总训练轮数为 num_epochs，训练集长度为 l，则每轮训练我们做如下计算：

$$p = \frac{idx + epoch \cdot l}{num\_epochs \cdot l} \tag{17}$$

$$\alpha = \frac{2}{1 + e^{-10 \cdot p}} - 1 \tag{18}$$

我们定义 DANN 的损失函数如下：

$$loss = loss^s_{cls} + loss^s_{dom} + loss^t_{dom} \tag{19}$$

式中 $loss^s_{cls}$ 为源域数据的分类损失，$loss^s_{dom}$ 为源域数据的域判别损失，$loss^t_{dom}$ 为目标域数据的域判别损失。以上三种损失均采用交叉熵损失。

在 DANN 训练中，我们采用 $batch\_size = 32$，$lr = 0.001$ 的训练配置。

9.3. 实验结果展示

表11展示了 DANN 在 ColoredMNIST 上的精度表现。

表 10: *DANN*

Method	**Epochs**	**Accuracy(%)**
DANN	30	74.561

DANN 的性能非常优良，达到了本次大作业项目中最优秀的水平，训练效率上也不逊色于其他方法，迁移学习的性能优势可见一斑。

10. Invariant Risk Minimization

10.1. 方法简介

不变风险最小化（Invariant Risk Minimization，IRM）的目标是通过最小化在不同环境之间变化的风险来训练模型，而不是仅仅关注整体的平均风险。IRM 的关键思想是通过引入环境不变性来减少模型对于环境特定信息的依赖，从而提高其泛化性能。具体而言，IRM 通过将模型对于不同环境的输出变化与输入的变化解耦，强制模型学习对于环境差异不敏感的特征。这使得模型更能够在新的环境中保持稳健性，并减轻环境间的分布偏移对模型性能的影响。

输入的数据特征可分为两种，不变特征（invariant feature，后记为 X_i）和虚假特征（spurious feature，后记为 X_s）。不变特征在数据环境变换时保持不变，因此与数据标签的相关性非常稳定；而虚假特征通常与环境相关，环境和数据标签的相关性是不稳定的。由此可以看出，$P(Y|X_i)$ 是稳定的，$P(Y|X_s)$ 是不稳定的，而我们的目标是学习一个仅仅依赖于 X_i 的机器学习模型。

假设我们的数据来自不同的环境 $\{e_1, e_2 \cdots, e_E\}$，而我们希望学习一个仅依赖于 X_i 的数据表示，即有 $P^e(Y|\Phi(X)) = P(Y|\Phi(X))$。此时，如果我们寻找一个分类器 ω 对 $P(Y|\Phi(X))$ 最优，则该分类器对每一个子环境都是最优的。将该目标表示为标准优化问题如下：

$$\min_{\omega, \Phi} \sum_e R^e(\omega, \Phi) \tag{20}$$

$$s.t. \quad \omega \in argmin_{\omega^e} R^e(\omega^e, \Phi) \tag{21}$$

由于我们的目标是寻找一个分类器 ω 使得它在每个环境内都同时最优，而如果分类器 ω 在某个环境下是最优的，那么 loss 对它的导数就是 0，如果导数不为零，那我们就需要给予相应的惩罚。根据这个思路，我们可以得到以下优化问题：

$$\min_{\omega, \Phi} \sum_e R^e(\omega, \Phi) + \lambda ||\nabla_\omega R^e(\omega, \Phi)||_2^2 \tag{22}$$

其中的第一项保证了模型的预测能力，第二

项保证了模型的预测不变性，λ 是预测能力和预测不变性之间的正则化平衡。

10.2. 训练细节

当使用随机梯度下降的小批量估计目标时，可以获得梯度范数平方的无偏估计如下：

$$\sum_{k=1}^{b}[\nabla_{\omega|\omega=1.0}\mathcal{L}(\omega\cdot\phi(X_k^{e,i}),Y_k^{e,i}) \cdot\nabla_{\omega|\omega=1.0}\mathcal{L}(\omega\cdot\phi(X_k^{e,j}),Y_k^{e,j})] \tag{23}$$

其中 $(X^{e,i},Y^{e,i})$ 和 $(X^{e,j},Y^{e,j})$ 是环境 e 中大小为 b 的两个随机批次数据，$\mathcal{L}$ 是某损失函数（这里使用交叉熵损失函数）。

具体训练算法如伪代码2所示。

Algorithm 2 Invariant Risk Minimization

Initialization: $model, \lambda \leftarrow 2, epoch \leftarrow 0, w \leftarrow 1, trainloader1(T_1), trainloader2(T_2)$
repeat
  $epoch \leftarrow epoch + 1$
  for data pair $(x_i, y_i), (x_j, y_j) \in (T_1, T_2)$ **do**
    Selete data pair from trainloader1 and trainloader2
    Calculate $\mathcal{L}_1$ and $\mathcal{L}_2$ with $\mathcal{L}(\omega \cdot \phi(X_k^{e,i}), Y_k^{e,i})$
    Update $\mathcal{L}_{error} \leftarrow (L1 + L2)/2.$
    Update $\mathcal{L}_{penalty} \leftarrow \nabla_\omega L1 \cdot \nabla_\omega L2.$
    Update loss $\leftarrow \mathcal{L}_{error} + \lambda \cdot \mathcal{L}_{penalty}.$
  end for
until $epoch = MaxEpoch$
Output: Feature Representation Model $model$

值得注意的是，这里的 trainloader1 和 trainloader2 数据来自不同的环境，因此算法中将其配对计算预测不变性损失。

在 IRM 训练中，我们采用 $batch\_size = 64$，$lr = 0.001$ 的训练配置。

10.3. 实验结果展示

表11展示了 IRM 在 ColoredMNIST 上的精度表现。

IRM 的性能同样非常优秀，但是相比其他进阶算法收敛速度相对较慢。

表 11: *IRM*

Method	Epochs	Accuracy(%)
IRM	30	73.425

10.4. 优化与改进

我们的优化主要在对正则化平衡参数 λ 的选择上。经过多次实验与观察，我们发现 loss 的不变性风险项（前述第二项）在整个 loss 中的比重不断减小，因此理论上 λ 在训练过程中应逐步增大。

在前面的算法中，我们的 λ 是以迭代次数的平方速度增长的，我们设想是否可以找到一个更好的调节方式，使模型预测能力和预测不变性之间能够始终保持较好的平衡。

首先我们尝试了线性增长和指数增长，效果并没有明显的提升。接着，我们希望通过 loss 本身的数据去调节 λ 的值，即设置 λ 为两种 loss 的比值，但是在实验中我们发现，loss 的比值虽然整体上的趋势是逐渐变大，但是真正的 loss 比值存在非常大的不确定性，导致训练过程中的鲁棒性很低。

最后，我们仍然选择了原先的调节方式，但是我们总结得到，虽然 λ 的调节方式可能不会对最终的模型性能有很大的影响，但是更合理的调节方式能使训练过程更加稳健，可能会拥有更快的收敛速度和更强的鲁棒性。

11. Correct-N-Contrast

11.1. 方法简介

对于常规的机器学习方法，数据被虚假的关联 (spurious correlations) 所强烈影响。为了规避这种影响，我们观察到在最差组（正确率最低的组）中，神经网络的最差组准确度与其输出（即最后一个隐藏层的输出）对真实标签的依赖程度密切相关，而与虚假属性无关。理论而言，我们可以通过量化对齐这一属性可以显著提升正确率。而且这一性质在任何的虚假属性中关系均存在。

n 类正确比对 (Correct-n-Contrast) 是一种两

阶段对比学习方法，用于更好地对齐表示并提高对虚假相关性的鲁棒性。其关键思想是使用对比学习将相同类别但具有不同虚假属性的样本的表示“推在一起”，同时将具有相同虚假属性但不同类别的样本的表示“拉开”。CNC 改善了类内对齐性，同时保持了类间可分性，鼓励模型依赖于能够预测类标签但不涉及虚假属性的特征。

CNC 首先使用一个经过正则化的 ERM 模型推断虚假属性，该模型经过训练可以预测类标签。我们使用这些预测通过对比学习和新颖的采样策略训练另一个稳健的模型。我们随机选择一个锚点，选择与相同类别但具有不同 ERM 预测的样本作为“推在一起”的困难正例，以及具有相同 ERM 预测但不同类别的样本作为“拉开”的困难负例。这鼓励忽略虚假差异并学习锚点与正例之间的类别特定相似性，反之亦然，鼓励忽略虚假相似性并学习锚点与负例之间的类别特定差异。因此，CNC 通过对比学习纠正了 ERM 模型学到的虚假相关性。

Algorithm 3 Correct-n-Contrast (CNC)

Require: Training dataset (X, Y), # positives M, # negatives N, learning rate η, epochs K.

Stage 1: Inferring pseudo group labels

Train ERM model $f_{\hat{\theta}}$ on (X, Y); save $\hat{y}_i := f_{\hat{\theta}}(x_i)$.

Stage 2: Supervised contrastive learning

Initialize "robust" model f_θ.
for $k = 1$ to K **do**
  for anchor $(x, y) \in (X, Y)$ **do**
    Let $\hat{y} := f_{\hat{\theta}}(x)$ be the ERM model prediction of x.
    Get M positives $\{(x + m, y + m)\}$, where $y + m = y$ and $\hat{y} + m \neq \hat{y}$, for $m = 1, \ldots, M$.
    Get N negatives $\{(x_n^-, y_n^-)\}$, where $y_n^- \neq y$ and $\hat{y}_n^- = \hat{y}$, for $n = 1, \ldots, N$.
    Update f_θ by $\theta \leftarrow \theta - \eta \cdot \nabla \hat{L}(f_\theta; x, y)$ with anchor, M positives, and N negatives.
  end for
end for

Output: Return final model f_θ from Stage 2.

11.2. 训练细节

根据对应的思路，我们可以先训练得到一个 baseline 模型，在这个基础上得到一个鲁棒性更好的最终模型。

训练算法如伪代码3所示。

其中 loss 计算式如下：

$$\hat{L}(f_\theta; x, y) = \lambda \hat{L}_{\text{sup}}^{con}(f_{\text{enc}}; x, y) + (1-\lambda)\hat{L}_{\text{cross}}(f_\theta; x, y) \tag{24}$$

这里，$\hat{L}_{\text{sup con}}(f_{\text{enc}}, x, y)$ 表示 x 及其正负样本的监督对比损失 $\hat{L}_{\text{cross}}(f_\theta, x, y)$ 表示在 x、M 个正样本和 N 个负样本上的平均交叉熵损失；$\lambda \in [0, 1]$ 是一个平衡超参数。

监督对比损失如下：

$$\mathcal{L}_{\text{sup}}^{con}(x, f_{\text{enc}}) = \mathop{\mathbb{E}}_{z, \{z_m^+\}_{m=1}, \{z_n^-\}_{n=1}} \left[-\log \frac{\exp(z^T z_m^+/\tau)}{\sum_{m=1}^{M} \exp(z^T z_m^+/\tau)} + \sum_{n=1}^{N} \exp(z^T z_n^-/\tau) \right] \tag{25}$$

在这里，$\tau > 0$ 是一个标量温度超参数。最小化式25导致在表示空间中，z 距离 z^+ 比 z^- 更近。

在正式训练中 λ 取 0.5, τ 取 0.1,M,N 采用任取 64 个数据，分别获得获得 M,N 的值。

在 CNC 训练中，我们采用 $batch\_size = 16$，$lr = 0.0001$ 的训练配置。

11.3. 实验结果展示

表12展示了 CNC 算法在 ColoredMNIST 上的精度表现。

表 12: *CNC*

Method	**Epochs**	**Accuracy(%)**
CNC	30	71.142

事实上，只需要 1 个 epoch 即可达到最好的精度，之后训练会发生过拟合，正确率波动下降。

12. 反馈及改进

本版块为课堂展示 Q&A，我们针对同学们在展示课上提出的问题进行解答。

1. **Q**：DANN 方法的源域 MNIST 在输入 DANN 模型时将颜色通道进行了堆叠操作，是否可以认为这里利用了 spurious feature 是颜色的信息?

   A：堆叠颜色通道仅仅是为了保证 MNIST 数据集在输入上能与 ColoredMNIST 数据集对齐，实际上并没有利用到 spurious feature 是颜色的信息。如果将 spurious feature 更换为其他类型（例如一个不易发现的图案），DANN 方法依旧是可行的。

2. **Q**：DANN 方法需要用到干净数据集 MNIST，是否引入了额外信息?

   A：的确，DANN 方法的进行需要干净数据集，可以认为是额外信息。不过，在医学和自动驾驶等需要模型具备较强 OOD 泛化能力的领域上，干净的数据集是能够获取也是有必要获取的，DANN 仍然具备实用价值。

3. **Q**：DANN 方法中似乎源域数据集 MNIST 与目标域数据集 ColoredMNIST 是严格对齐的，输入的两张图像仅有颜色区别，这样做是不是不太严谨?

   A：源域和目标域的训练集 dataloader 均经过 shuffle 操作，并不存在严格对齐的情况。我们在代码中也将这个部分放到了更醒目的位置。

由于报告篇幅有限，难以对所有的问题进行解答，如果您对我们的项目存在疑问，欢迎与本组成员进行联系。

此外，我们对 IRM 方法进行了一些深入探究与讨论，具体内容详见前文 IRM 版块，我们已经将补充的部分标注为蓝色。

13. 项目总结及领域简要展望

13.1. 项目总结

在本次大作业项目中，我们对课堂上所学习的因果推理知识在 ColoredMNIST 数据集上进行实际应用，巩固了对课堂知识的理解。在完成基础部分的前提下，我们还总结出了四种额外的提高方法，均取得了较好结果。

ColoredMNIST 是一个 spurious feature 相对容易察觉的数据集，因此我们可以通过显式的后门调整来解决 spurious correlation。然而，在实际任务中，spurious feature 往往是较为隐蔽的，人类的肉眼难以直接察觉，所以不需要知晓 spurious feature 具体信息的 DANN、IRM、CNC 等方法具有更高的泛用价值。

经过前期调研，我们发现 CLIP[8] 分类器、Dual Network 等方法也能够完成 ColoredMNIST 分类任务，限于工程量以及报告篇幅，我们并没有实现。

表13展示了我们在本项目中实现的所有方法在 ColoredMNIST 数据集上的表现（在 MNN 部分，I 代表 Ideal，即理想情况；DF 代表 Device Fault，即设备故障；DER 代表 Device Endurance and Retention，即设备损耗）。

13.2. OOD 领域简要展望

我们已经知道，传统机器学习模型往往会因为域分布差距（domain distribution gaps）而恶化，难以满足工业界和学术界的要求，提高模型在未见的测试域上的泛化能力是一个经久不衰的研究方向。

目前已经有许多得到长足发展的与泛化相关的研究领域，例如领域适应（domain adaptation）、元学习（meta-learning）、迁移学习（transfer learning）、协变量偏移（covariate shift）等。近年来，域泛化（Domain generalization，DG）也受到了广泛的关注 [9]，最近的 OOD 算法主要可分为无监督的表征学习、有监督的模型学习和优化算法。时至今日，模型库技术 [10]（Model Zoo）、基于动态数据分布的表征学习 [11] 等新方法或新视角也

表 13: *All Our Methods on ColoredMNIST*

Method	Epochs	Accuracy(%)
CNN (baseline)	50	35.395
CNN + bi-loss	30	73.495
CNN + bi-model	30	66.213
CNN + adjust	30	72.205
SNN (baseline)	50	30.990
SNN + bi-loss	50	70.625
SNN + bi-model	50	63.087
SNN + adjust	30	68.932
MNN(I) (baseline)	50	60.365
MNN(I) + bi-loss	30	73.128
MNN(I) + bi-model	30	71.845
MNN(I) + adjust	30	50.050
MNN(DF) + bi-loss	30	68.715
MNN(DER) + bi-loss	30	53.468
DRO	/	73.310
DANN	30	**74.561**
IRM	30	73.425
CNC	30	71.142

被陆续应用于 OOD 问题的解决。

14. References

[1] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

[2] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of machine learning research*, 17(59):1–35, 2016.

[3] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant Risk Minimization, March 2020.

[4] Michael Zhang, Nimit S. Sohoni, Hongyang R. Zhang, Chelsea Finn, and Christopher Ré. Correct-N-Contrast: A Contrastive Approach for Improving Robustness to Spurious Correlations, March 2022.

[5] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79:151–175, 2010.

[6] Ievgen Redko, Emilie Morvant, Amaury Habrard, Marc Sebban, and Younes Bennani. *Advances in domain adaptation theory*. Elsevier, 2019.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[8] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[9] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip Yu. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[10] Qishi Dong, Awais Muhammad, Fengwei Zhou, Chuanlong Xie, Tianyang Hu, Yongxin Yang, Sung-Ho Bae, and Zhenguo Li. Zood: Exploiting model zoo for out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 35:31583–31598, 2022.

[11] Wang Lu, Jindong Wang, Xinwei Sun, Yiqiang Chen, and Xing Xie. Out-of-distribution representation learning for time series classification. In *The Eleventh International Conference on Learning Representations*, 2022.