分布外泛化的算法探索、总结与改进

上海交通大学　徐日晞

同组成员：罗镜天、陈楷润
本报告基本内容由本人撰写并共享，结果分析、个人探索和注解部分由个人独立完成。

摘要

深度神经网络通常采用独立同分布的假设进行训练，即假设测试数据分布与训练数据分布相似。然而，当用于实际任务时，这一假设并不成立，导致其性能显著下降。例如，对于猫狗二分类问题，如果训练集中所有狗都在草地上，所有的猫都在沙发上，而测试集反之，那么模型在没有测试集信息的情况下，很有可能会把沙发和猫联系在一起，在测试时将会把在沙发上的狗误认为是猫。理想的人工智能系统应尽可能在分布外（Out-of-Distribution）的情况下有较强的泛化能力。本文以彩色数字识别数据集为例，对常见的OoD算法进行探索、总结，并通过引入边缘信息提取等方式，对其中IRM、RSC等算法进行改进。

1　概述

在分布外（Out-of-Distribution, OoD）泛化问题中，要区分两种偏移的形式 [1]，一种是**多样性偏移**，另一种是**相关性偏移**。多样性偏移主要源于测试集中出现了训练集中概率极低的全新特征或风格。相关性偏移则通常是由于一个非因果的、虚假的特征与标签在训练环境中高度相关，而这种相关性在测试环境中减弱甚至反转，造成泛化的困难。比如经典的狗在草地上、猫在沙发上的问题。

本次实验主要关注多样性偏移的情况，基本的数据集1为 Colored MNIST 的变体：在三种训练环境中，数据分别为红色数字黑色背景、绿色数字黑色背景、白色数字黑色背景；在测试阶段，数据分别为黄色数字白色背景、蓝色数字白色背景。

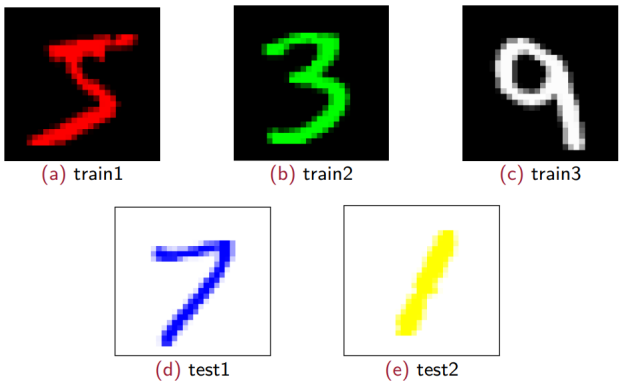


图 1: 数据集1

在初级阶段，我们先构建了一个LeNet模型，探索此模型在数据集1上的表现；在中级阶段，我们尝试对训练集进行增广，得到数据集2，并在此基础上探索LeNet的表现，给出可视化；在高级

阶段，我们保留原有的训练集，对测试集进行增广，得到数据集3，并在LeNet架构的基础上加入了IRM，RSC等常见的OoD算法，基于它们的实际表现做出分析和改进，并给出一些提高泛化性能的经验性方法。

2 初级阶段：LeNet的构建

2.1 LeNet的改进

LeNet1是第一个将卷积、权重共享、池化、全连接、反向传播这些关键机制统一成一个可训练、可部署的系统的网络，其网络结构如下所示。

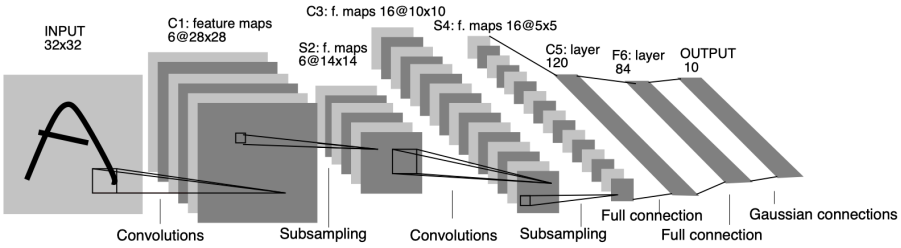


图 2: LeNet原始架构

在本次任务中，我们对经典的LeNet网络结构进行了如下调整和优化，以更好地适配输入图像和分类任务：

- 在每层线性运算（卷积层、全连接层）后使用ReLU激活函数，替换原始网络中的sigmoid函数，避免梯度消失等问题
- 模型输入由原始的单通道32 × 32的灰度图调整为本次任务中3通道28 × 28的RGB图像
- 模型输出类别数由传统MNIST数字识别的10个类别变成该二分类任务的2个类别

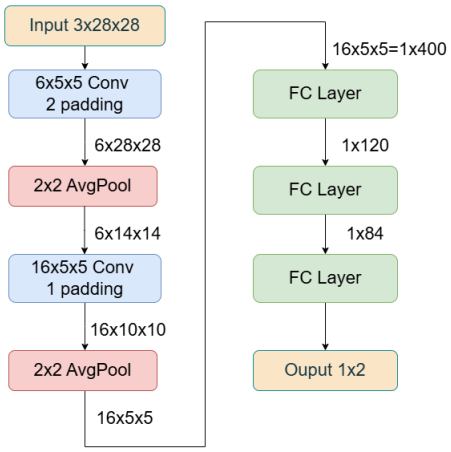


图 3: 调整优化后的LeNet架构

表 1: 基础LeNet在数据集1的训练与测试准确率对比

训练集	训练准确率 (%)	测试集1准确率 (%)	测试集2准确率 (%)
Train1	99.8	61.1	51.0
Train2	99.9	57.9	49.0
Train3	99.7	65.7	52.7

2.2 LeNet的训练和测试

结果分析：如上表所示，LeNet5在训练集上表现出了极高的准确率，但是在测试集上的准确率较低，具有比较大的泛化差距，出现了比较严重的过拟合。这也是我们探索OoD问题算法的意义。

3 中级阶段：数据的增广与分析

在本阶段，我们 **构建了一个叫RandomColor的nn.Module类，能够随机生成数字颜色和背景颜色，如果两个颜色过于接近则重新生成，得到训练集2a。将其与随机灰度化、随机裁剪并缩放、随机擦除等变换整合到一起，作用于训练集上，得到训练集2b**，它们和原来的测试集一起，组成数据集2，如图所示。

(a) 数据集2的训练集2a（只随机颜色）

(b) 数据集2的训练集2b（加入各类随机变换）

3.1 模型性能变化

表 2: 基础LeNet在数据集2的训练与测试准确率对比

训练集	训练准确率 (%)	测试集1准确率 (%)	测试集2准确率 (%)
Train2a	99.3	98.8	98.6
Train2b	92.83	97.53	97.07

结果分析：如表 2、图 5和图 6所示，在加入了随机变换后，模型的训练准确率和损失都没有仅颜色变换低，说明数据的增广迫使模型学习一个更本质和鲁棒的规律，这个训练过程损失下降的更慢，也更难达到一个极低的数值。但是它能有更好的泛化性能。

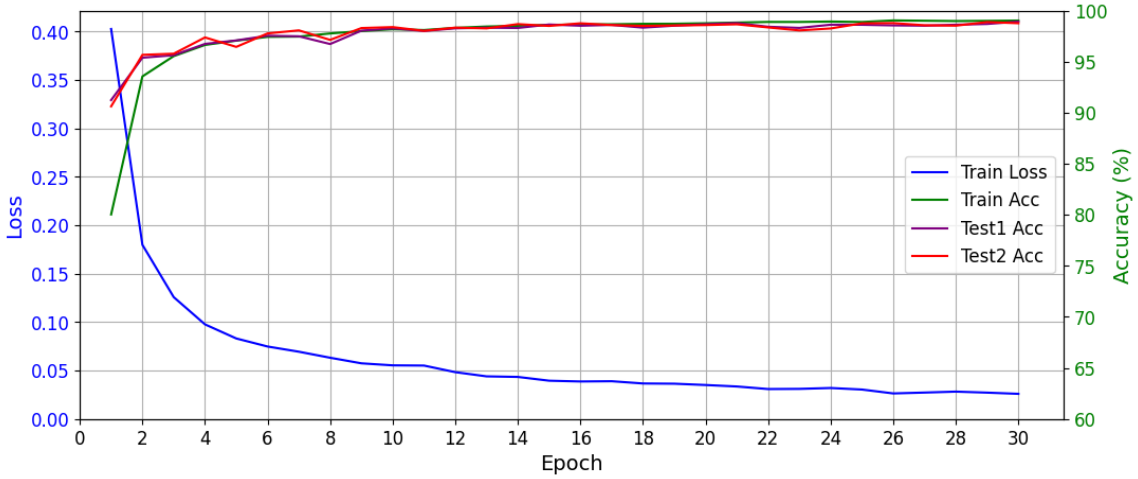


图 5: 用训练集2a训练过程的损失、准确率和测试准确率

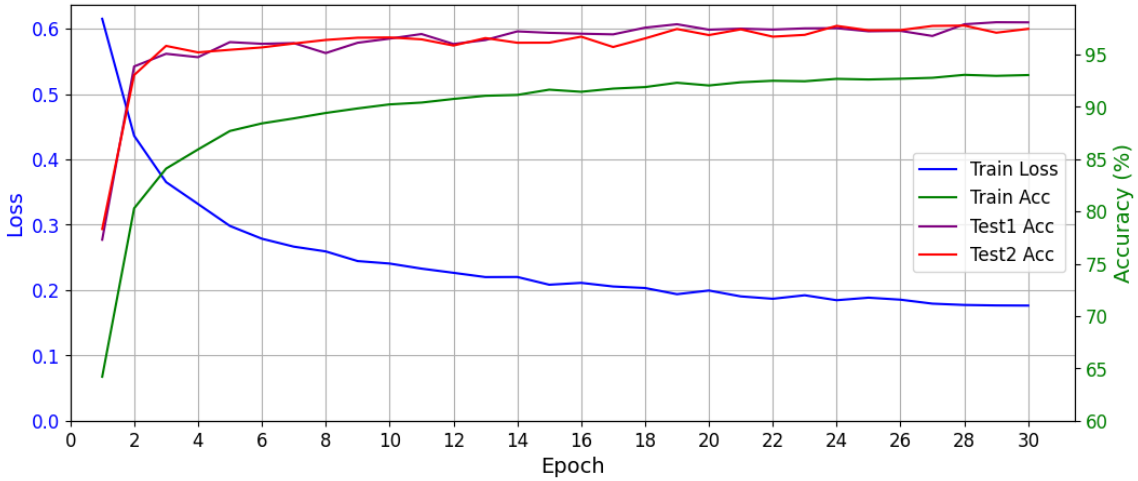


图 6: 用训练集2a训练过程的损失、准确率和测试准确率

3.2 参数敏感度分析

我们的调参遵循这样的原则，即先在更大的幅度上面进行大致调优，再缩小范围，在更小的范围内寻找最优的参数。

3.2.1 学习率

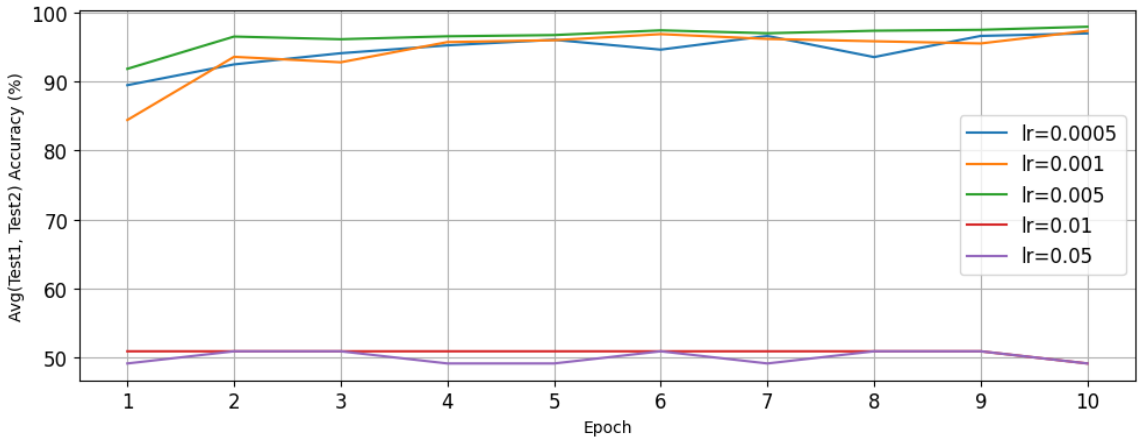


图 7: 不同学习率的测试准确率(1)

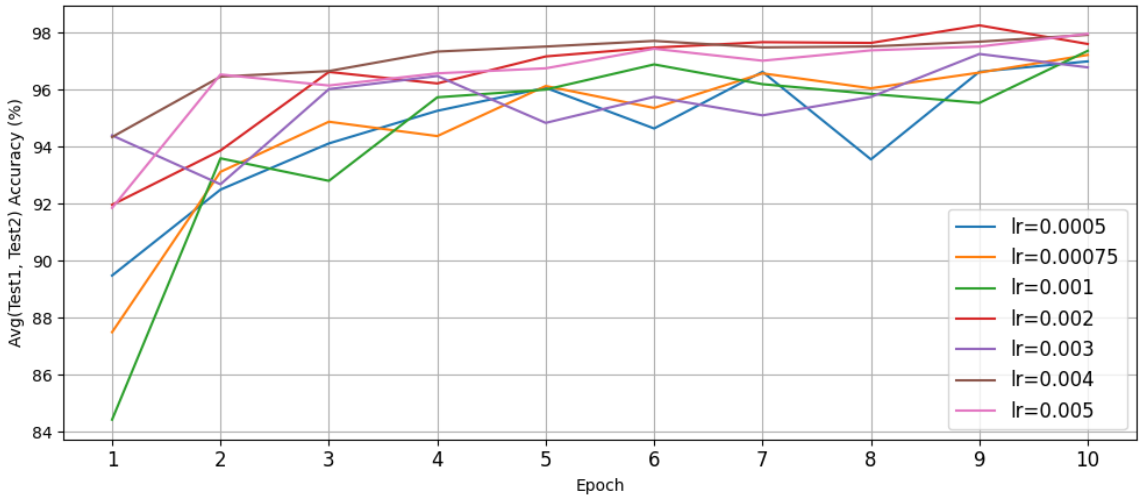


图 8: 不同学习率的测试准确率(2)

结果分析：在学习率小于0.01时，模型的准确率能保持在较高的位置，但是一旦大于等于0.01，模型的准确率急剧下降，甚至退化到了随机猜测的水平，这可以解释为更新步长太大，使模型在最优解附近“来回横跳”，无法落到最优。因此我们要设置合适的学习率，并结合学习率调度器等方法，动态调整学习率。

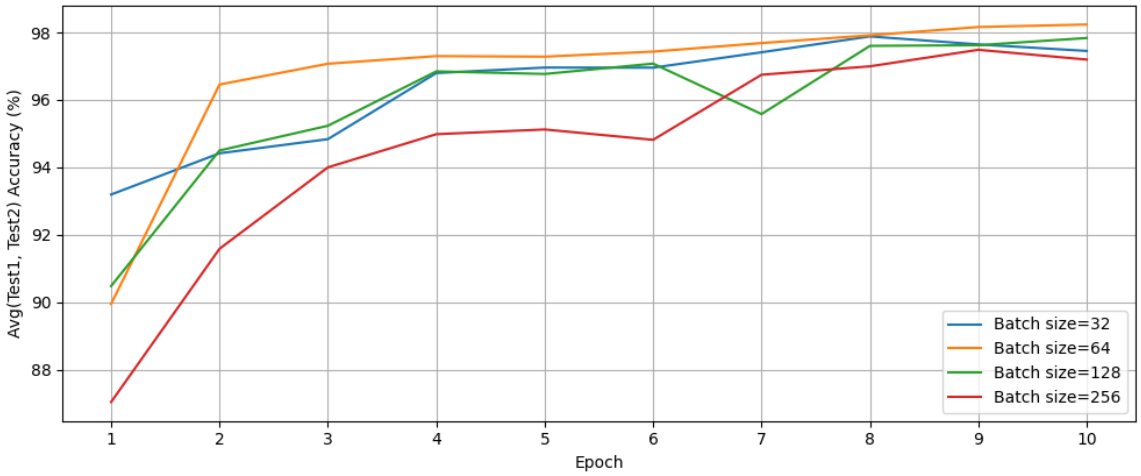


图 9: 不同批次大小的测试准确率

3.2.2 批次大小

结果分析：同学问的比较多为什么批次大小为256时准确率提升得比较慢。这是由于我们采用随机梯度下降，每一个batch更新一次梯度，更大的批次大小意味着更小的更新频率，更慢的优化过程，同时，也不利于跳出局部最优找到全局最优。

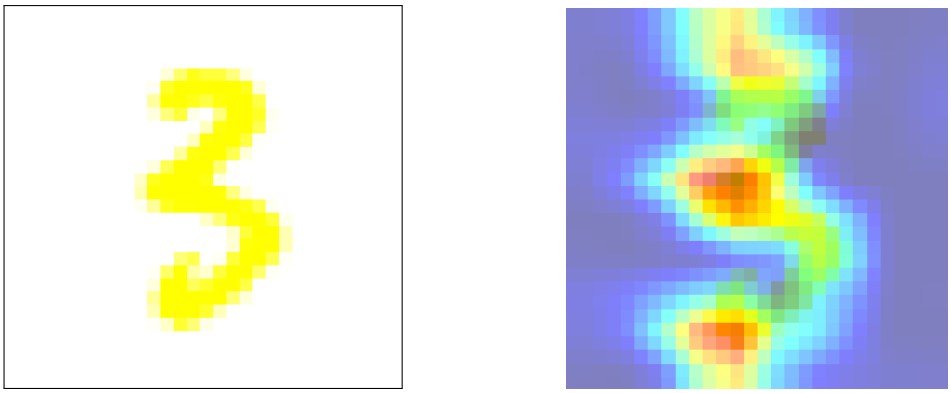
3.3 注意力分析

Grad-CAM（Gradient-weighted Class Activation Mapping）是一种基于**梯度**的可视化方法，用于定位 CNN 模型在进行分类判断时关注的图像区域。现对其原理进行简要说明：

1. Feature map 是卷积层输出的多通道二维激活图，反映输入图像在该层提取的各种**局部特征**。在本实验中，我们选择了模型的最后一层卷积输出作为 Grad-CAM 的目标层。
2. Grad-CAM利用目标类别对每个feature map通道的梯度信息，计算每个通道的权重（代表该通道对该类别的重要程度）。
3. 将每个 feature map 通道乘以对应权重，逐元素相加后得到一个二维加权和图，即为Grad-CAM的热力图，它融合了所有 feature map 通道的重要信息，反映了模型在空间位置上对目标类别的**关注度**。

4 高级部分：OoD算法的复现、分析与改进

- **因果学习、不变学习：**核心思想是**“寻找不变的本质”**，致力于挖掘变量间稳定不变的因果关系，学习一种在所有已知环境中都表现“不变”的特征表示或预测模型。它们往往需要划分不同环境的数据。其中，不变风险最小化（Invariant Risk Minimization, IRM） [2]是里程碑式的工作。此外，领域无关表示学习（如对抗训练方法DANN [3]）也属此类，其目标是使模型无法从特征中分辨出数据的来源领域。



(a) 原始图片　　(b) Grad-CAM

图 10: Grad-CAM可视化

- **数据增强与合成：**核心思想是**“见多识广”**，通过对数据加入噪声，增强模型的泛化性，或者通过创造更多样、更困难的数据，让模型对未见过的变化具备更强的抵抗力。
- **自挑战：**核心思想是**“自我挑战，防止懈怠”**，模型在训练时倾向于依赖最容易的特征（捷径）。这类方法通过某种机制，在训练中“人为制造困难”，阻止模型过度依赖少数几个特征。其中，表征自挑战 (Representation Self-Challenging, RSC) [4]是该流派的典型代表。

此外，还有稳定学习、异质性感知学习等方法。在一篇OoD算法的综述 [5]中作了全面的总结。我们主要对RSC和IRM这两个算法进行了复现、分析与改进。

4.1 RSC算法

RSC是一种针对泛化能力的自挑战正则化方法。它借鉴了dropout的思想，但不同于随机丢弃，它是有选择地丢弃那些与最高梯度相关联的信息，并迫使模型利用剩余的信息进行预测，使其在下一轮迭代中寻找其他特征来弥补被抑制的信息，从而逐步学习到与虚假相关性无关的特征表示。**RSC主要在前向传播过程起作用，不需要提前划分不同环境**。

4.1.1 模型假设

假设模型由一个特征提取器 $\Phi(\mathbf{x};\theta_\Phi)$ 和一个分类器 $w(\mathbf{z};\theta_w)$ 组成，其中 $\mathbf{z}=\Phi(\mathbf{x};\theta_\Phi)$ 是输入 $\mathbf{x}$ 提取的特征(features)，$\mathbf{a}=c(\mathbf{z};\theta_w)$是模型的输出(outputs)。模型的总参数集为 $\theta=\{\theta_\Phi,\theta_w\}$。为方便我们假设批次大小(batch_size)为1。

4.1.2 计算梯度

对于给定输入 $\mathbf{x}_i$ 和真实标签 y_i，首先$\mathbf{x}_i$经过特征提取器得到特征 $\mathbf{z}_i$，维度为 $C\times H\times W$。对$\mathbf{z}_i$继续前向传播得到输出$\mathbf{a}_i$，维度为2。对y_i独热编码得到维度为2的向量$\mathbf{y}'$. 将$\mathbf{a}_i$和$\mathbf{y}'$相乘后对$\mathbf{z}_i$求偏导，得到

$$\mathbf{g}_\mathbf{z}=\frac{\partial(\mathbf{a}_i\odot\mathbf{y}')}{\partial\mathbf{z}} \tag{1}$$

4.1.3 重要性度量

我们从通道(channel)和空间(spatial)两个维度来度量特征的重要性。

- **通道重要性**: 通道重要性 $\mathbf{v} \in \mathbb{R}^C$ 代表了每个通道对预测的平均贡献，通过对每个通道的梯度值在空间维度上求平均范数得到：

$$(\mathbf{v})_c = \frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} |\mathbf{g}_{c,h,w}| \tag{2}$$

- **空间重要性**: 空间重要性 $\mathbf{s} \in \mathbb{R}^{H \times W}$ 的计算方式为：首先对梯度张量取绝对值，然后在通道维度上求平均值。对于每一个样本，空间位置 (h, w) 的重要性值由下式给出：

$$(\mathbf{s})_{h,w} = \frac{1}{C} \sum_{c=1}^{C} |\mathbf{g}_{c,h,w}| \tag{3}$$

此操作生成一个形状为 (H, W) 的空间图。

值得一提的是，源代码在实现空间重要性图时借鉴了**Grad-CAM** [6] 的思想，通过使用通道梯度的平均值作为权重，对原始特征图进行加权求和，得到一个空间重要性热图 $(\mathbf{s})_{h,w} \in \mathbb{R}^{H \times W}$：

$$(\mathbf{s})_{h,w} = \mathrm{ReLU}\left(\sum_{c=1}^{C} \left(\frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} \mathbf{g}_{c,h,w} \right) \cdot \mathbf{z}_{c,:,:} \right) \tag{4}$$

4.1.4 掩码生成

在得到重要性度量后，算法以等概率随机选择一种抑制策略：通道抑制或空间抑制。设丢弃率(drop_rate)为 ρ。

如果选择通道抑制：找到通道重要性 $\mathbf{v}_c$ 中值最大的前 ρ 比例的通道索引。生成一个通道掩码 $\mathbf{M}_{\text{chan}} \in \{0,1\}^{C \times 1 \times 1}$，在这些重要通道上置为0，其余为1。最终的挑战掩码 $\mathbf{M}_{\text{chal}} = \mathbf{M}_{\text{chan}}$。

如果选择空间抑制：找到空间重要性 $\mathbf{s}$ 中值最大的前 ρ 比例的空间位置索引。生成一个空间掩码 $\mathbf{M}_{\text{spat}} \in \{0,1\}^{1 \times H \times W}$，在这些重要位置上置为0，其余为1。最终的挑战掩码 $\mathbf{M}_{\text{chal}} = \mathbf{M}_{\text{spat}}$。

4.1.5 置信度引导的掩码修正

为了防止过度抑制对分类至关重要的特征，RSC引入了一种修正机制。计算原始预测置信度 $p(y_{\text{true}}|\mathbf{z})$ 和使用挑战后特征 $\mathbf{z}_{\text{chal}} = \mathbf{z} \odot \mathbf{M}_{\text{chal}}$ 得到的预测置信度 $p(y_{\text{true}}|\mathbf{z}_{\text{chal}})$。置信度的下降量为 $\Delta p = p(y_{\text{true}}|\mathbf{z}) - p(y_{\text{true}}|\mathbf{z}_{\text{chal}})$。对于置信度下降不显著（即 $\Delta p < \tau$，其中 τ 为一个阈值）的样本，我们对其掩码进行修正，使其值更接近于1（即减少抑制），从而保留关键特征。修正后的掩码记为 $\mathbf{M}'_{\text{chal}}$。

4.1.6 与掩码相乘并再次前向传播

将生成的掩码应用于原始特征图上：

$$\mathbf{z}_{\text{chal}} = \mathbf{z} \odot \mathbf{M}'_{\text{chal}} \tag{5}$$

再对得到的$\mathbf{z}_{\text{chal}}$通过分类器，得到

$$\mathbf{a}' = w(\mathbf{z}_{\text{chal}}; \theta_w) \tag{6}$$

这就完成了前向传播。其余过程不变。

4.2 IRM算法

IRM的核心思想是学习一个特征提取器 Φ，使得基于该表征构建的最优分类器 $\mathbf{w}$ 在所有训练环境中都保持不变。IRM主要解决的是**相关性偏移**的问题，举一个经典的例子：一个用于识别“牛”的模型，可能会在训练数据中学到一个虚假的相关性，即“牛”总是出现在绿色的草地上。IRM 的目标就是找到一种特征提取器，它能够丢弃背景颜色（虚假特征），而专注于“牛”本身的内在的不变特征。不同于RSC，**IRM主要在计算损失时起作用，并且需要划分不同的训练环境。**

4.2.1 优化原则

IRM 原则被形式化地定义为一个约束优化问题。给定一个环境的集合$\mathcal{E}_{tr}$，它旨在寻找一个特征提取器 Φ，使得存在一个分类器 w，它在所有环境下都有最优的分类。

其目标函数可以写作：

$$\min_{\substack{\Phi:\mathcal{X}\to\mathcal{H}\\ w:\mathcal{H}\to\mathcal{Y}}} \sum_{e\in\mathcal{E}_{tr}} R^e(w\circ\Phi) \quad \text{s.t.} \quad w\in\arg\min_{\bar{w}:\mathcal{H}\to\mathcal{Y}} R^e(\bar{w}\circ\Phi), \forall e\in\mathcal{E}_{tr}. \tag{7}$$

其中：$R^e(\cdot)$ 表示在环境 e 中的期望损失$R^e(f)=\mathbb{E}_{(X,Y)\sim P^e}[\ell(f(X),Y)]$，其中 ℓ 是损失函数。这是无法直接计算的，实践中用经验风险$R^e(f)=\frac{1}{n}\sum_{i=1}^{n}\ell(f(x^{(i)},y^{(i)}))$来估计。

这等价于最小化：

$$L_{\text{IRM}}(\Phi,w)=\sum_{e\in\mathcal{E}_{tr}} R^e(w\circ\Phi)+\lambda\cdot\mathbb{D}(w,\Phi,e) \tag{8}$$

其中，$\mathbb{D}(w,\Phi,e)$衡量w与最优分类器的距离。

如果用最小二乘法求最小距离，可以证明$\mathbb{D}(w,\Phi,e)$等价于损失梯度的平方；并且可以证明取定$w'=1.0$，只优化Φ，不影响优化结果。于是得到了IRMv1。

4.2.2 IRMv1

IRMv1 的目标函数如下：

$$\min_{\Phi,w}\sum_{e\in\mathcal{E}_{tr}} R^e(w\circ\Phi)+\lambda\cdot\left\|\nabla_{w'}R^e(w'\circ\Phi)\Big|_{w'=1.0}\right\|^2 \tag{9}$$

我们可以将这个公式分解为两部分：

- $\sum_{e\in\mathcal{E}_{tr}} R^e(w\circ\Phi)$: 这是标准的**经验风险最小化**（ERM）项。它的作用是确保模型能够很好地拟合训练数据。
- $\lambda\cdot\left\|\nabla_{w'}R^e(w'\circ\Phi)\Big|_{w'=1.0}\right\|^2$: 这是**不变性惩罚项**（penalty），在 $w'=1.0$ 处的梯度为零，是 $w'=1.0$ 成为最优解的一个必要条件。

4.3 算法的复现

基于以上的理论，我们参照源代码，将这两个算法应用到我们的数据集中。我们构建了数据集3：训练集与数据集1相同，均为背景均为黑色，三个环境的数字颜色分别为红色、蓝色和白色；测试集我们使用了随机着色，随机改变测试图片中的背景和数字颜色。

我们先用数据集1（未增广）的测试集作为验证集，选取了最好的超参数，然后在数据集3上进行测试。

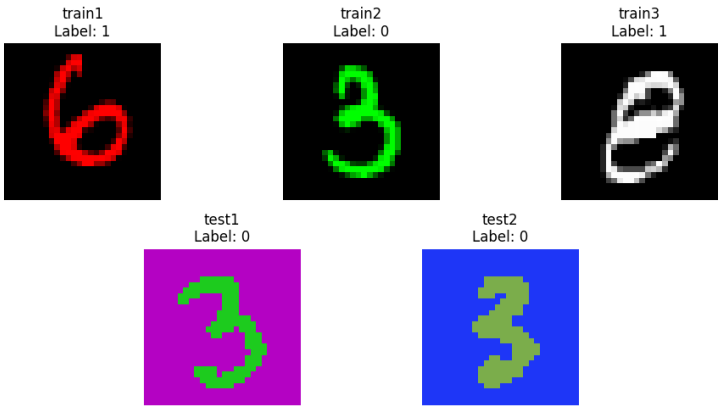


图 11: 数据集3的可视化

4.3.1 复现过程中的困难

在复现的过程中，主要遇到如下的困难：

- **模型架构问题**

  RSC 的源代码使用的模型架构是 ResNet18RSC，输入是 224x224 的图像，经过一系列的卷积和步长为2的池化/卷积层后，在进入最后的全局平均池化层 (avgpool) 之前，特征图的尺寸正好是 7x7。但是我们数据集中的数据是28x28的，必须要修改模型的结构。但是调整卷积核、步长之后进行训练效果并不理想：无论怎么调整超参数（包括动态调整、warmup等技巧），训练准确率能达到98%以上，但是验证准确率始终卡在49%-50%的区间，这跟随机猜测没有任何区别！

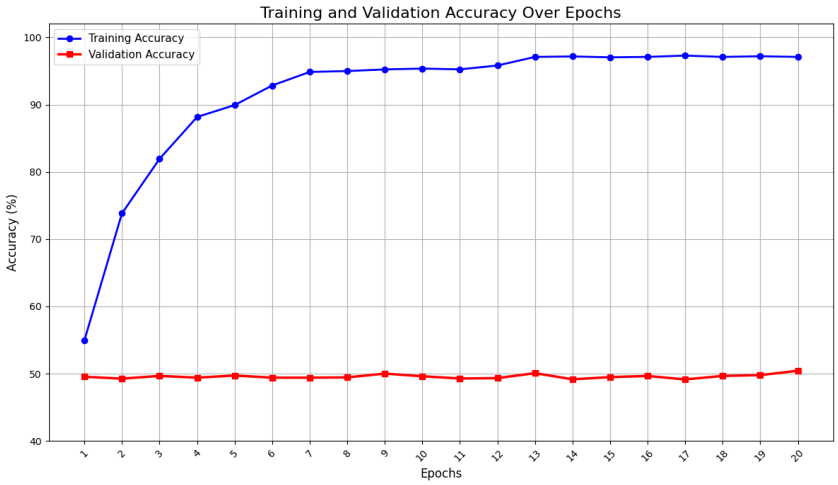


图 12: 在改进的ResNet18RSC下的准确率

如图 12所示，在强大的ResNet18RSC下模型出现了过拟合，无论如何调整惩罚项也无济于事。为此，我们放弃了该架构，回归基础的LeNet架构。

- **验证准确率不佳，甚至不如基本ERM模型**

  如图 13所示，使用RSC或IRM算法训练的模型在多轮测试的准确率均不如最基本的模型。

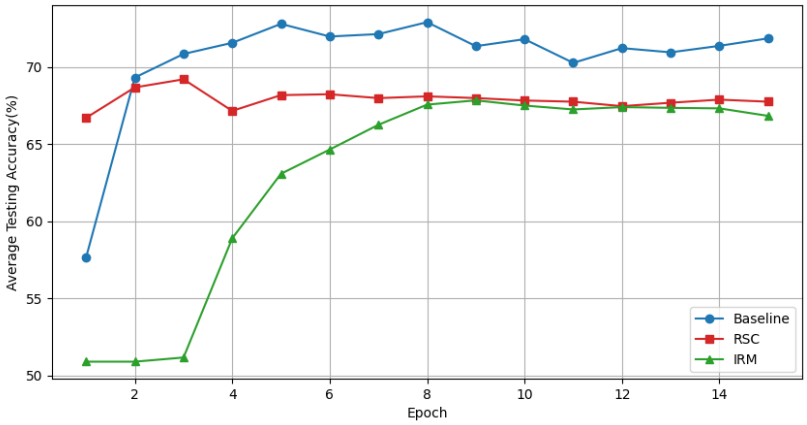


图 13: RSC、IRM和baseline的准确率对比

4.4 算法和模型的改进

基于前面的探索，我们给出改进的算法：

- **在输入层和中间层对每一个样本的每一个通道引入归一化**

  对于任意一个样本 $n \in \{1, \ldots, N\}$ 和任意一个通道 $c \in \{1, \ldots, C\}$，我们考虑其对应的二维切片 $X_{n,c} \in \mathbb{R}^{H \times W}$：

$$\text{均值:} \quad \mu_{n,c} = \frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} x_{n,c,h,w} \tag{10}$$

$$\text{方差:} \quad \sigma_{n,c}^2 = \frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} (x_{n,c,h,w} - \mu_{n,c})^2 \tag{11}$$

  注意：这里的统计量下标是 n, c，表明它们对于每个样本和每个通道都是独立的。总共有 $N \times C$ 组独立的均值和方差。使用上述统计量对每个像素进行归一化，并进行缩放和平移。

$$x'_{n,c,h,w} = \gamma_c \left(\frac{x_{n,c,h,w} - \mu_{n,c}}{\sqrt{\sigma_{n,c}^2 + \epsilon}} \right) + \beta_c \tag{12}$$

  其中，ϵ 是一个防止除以零的极小数。γ_c 和 β_c 是可学习的仿射变换参数，每个通道拥有一对，共 C 对。

  对改进的解释：如果输入的一批图像中存在“风格”差异，对每一张图片的特征归一化会将这些统计特性（如均值和方差）拉到一个标准化的水平，从而“抹掉”这些图像各自的风格。在更深的层次，特征图代表的可能是纹理、边缘组合等更抽象的信息，我们认为在这里继续进行风格归一化，能鼓励模型学习更本质的形状和结构，而不是特定的背景和外观。

- **引入边缘信息提取**

  在标准的CNN中，显式地引入一个并行的边缘检测分支，并将提取到的边缘特征与网络学习的高级语义特征进行有效融合。当然这不是我们原创的，而是计算机视觉领域的一个经典做法，我们把它融入到了这里。

  这包含两个并行的信息处理流：

1. **语义特征流 (Semantic Flow)**: 输入图像 X 首先通过一个标准的CNN$\mathcal{F}_{feat}$，提取高级语义特征 $\mathbf{F}_{feat}$。

$$\mathbf{F}_{feat} = \mathcal{F}_{feat}(X) \in \mathbb{R}^{N \times C_{feat} \times H' \times W'} \tag{13}$$

2. **边缘特征流 (Edge Flow)**: 同时，我们使用固定的、不可训练的Sobel算子（通过卷积层实现）对输入图像 X 进行处理，以计算其梯度模长，从而得到边缘强度图 $\mathbf{E}$。

$$\mathbf{E} = \sqrt{(X * \mathbf{K}_x)^2 + (X * \mathbf{K}_y)^2 + \epsilon} \tag{14}$$

   其中 $\mathbf{K}_x$ 和 $\mathbf{K}_y$ 分别为水平和垂直方向的Sobel卷积核。

为融合这两种异构信息，我们首先通过双线性插值将边缘图 $\mathbf{E}$ 的空间尺寸调整为与 $\mathbf{F}_{feat}$ 一致，记为 $\mathbf{E}_{resized}$。随后，在通道维度上将两者拼接，并通过一个可学习的融合层 $\mathcal{F}_{fuse}$（由 Conv-InstanceNorm-ReLU 组成）进行处理，得到最终的增强特征 $\mathbf{F}_{fused}$。

$$\mathbf{F}_{fused} = \mathcal{F}_{fuse}(\text{Concat}[\mathbf{F}_{feat}, \mathbf{E}_{resized}]) \tag{15}$$

结果的可视化如图 14所示，需要注意我们并不是把图片灰度化了，我们只是取了 R, G, B 三个边缘图在该位置的强度的最大值作为边缘强度**供可视化**。

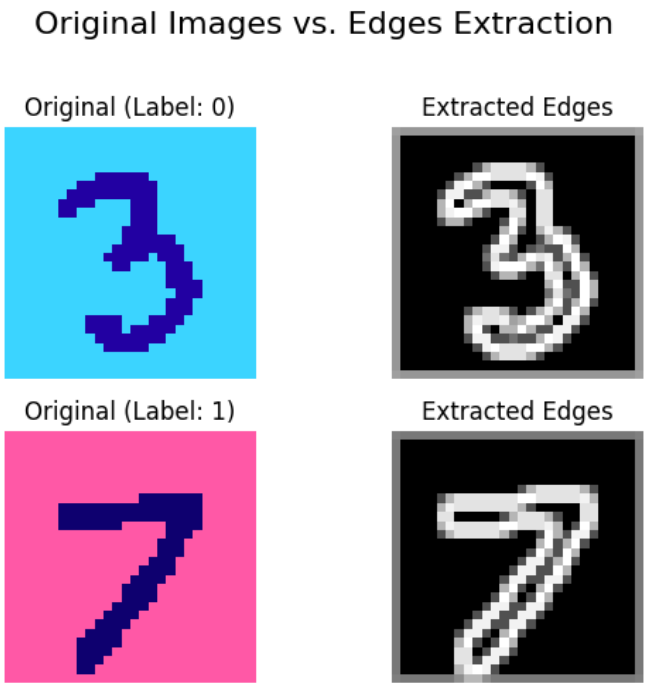


图 14: 边缘信息提取可视化

- **通道丢弃和空间丢弃相结合**

  只进行空间丢弃能让模型避免依赖特定的**局部特征**，进行通道丢弃则能让模型避免依赖**全局的、与风格相关**的特征。源代码采用随机数的方式选择丢弃方法，我们将其改进为两者结合的形式，对两者加权。

$$\mathbf{M}_{\text{chal}} = \alpha \mathbf{M}_{\text{spat}} + (1 - \alpha)\mathbf{M}_{\text{chan}} \tag{16}$$

经过多次试验，我们决定将α硬编码为0.3。

对改进的解释：依照本实验数据集的实际，模型可能会更多地依赖与风格相关的特征，而不是过分依赖局部特征，因此，应有侧重地使用通道丢弃，让它具有更大的权重，而不是单纯地以50%概率随机选取一种丢弃方法。

4.5 改进算法的实验

我们用数据集3对我们的算法进行了实验。其中，我们在每一组都引入了样本归一化，baseline组不使用任何的OoD算法，IRM组使用基本的IRMv1算法，RSC组使用基本的RSC算法，IRM+RSC组在前向传播过程加入RSC的正则化，并在目标函数中加入了IRM的惩罚项，RSC improve组在使用通道丢弃和空间丢弃相结合的RSC的同时，引入了边缘信息的提取。我们在每一轮都对模型进行了测试（仅测试，不参与调参），画出了测试准确率和训练轮数的关系图，如图 15所示。

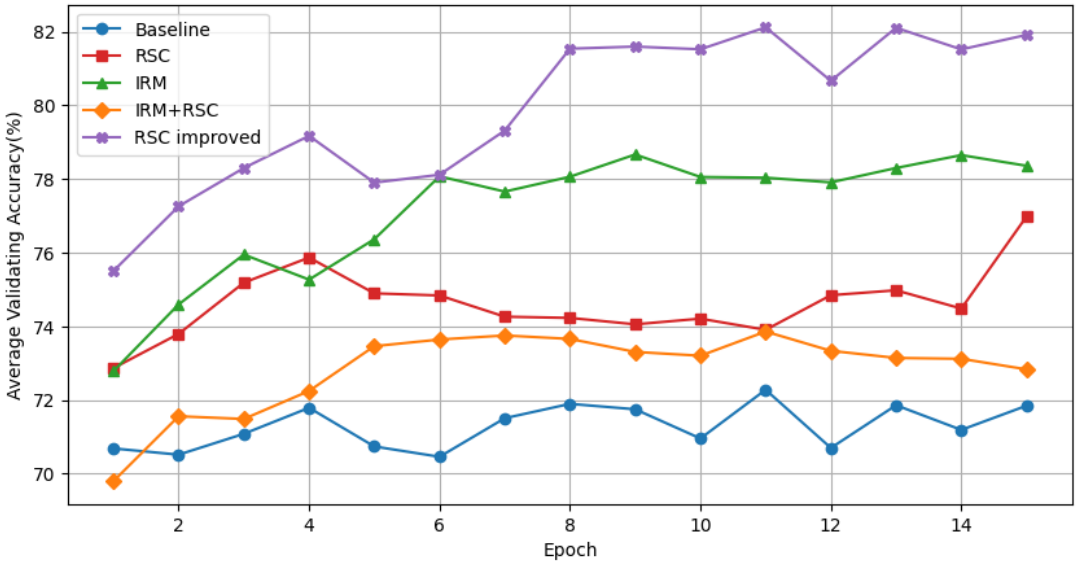


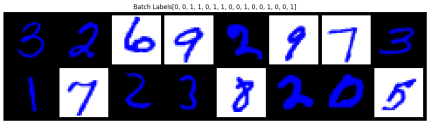
图 15: 不同算法的准确率对比

可以看到，我们改进的算法准确率可以去到80%以上，且使用OoD算法的组准确率均比baseline高，说明了架构和算法的有效性。

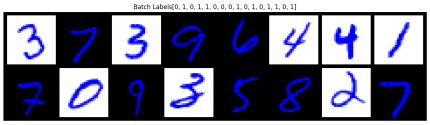
5 本人更多的探索与交流后的注解

5.1 对RSC在相关性偏移数据集上有效性的验证

前面我们通过优化网络结构和对RSC进行优化，验证了RSC在多样性偏移数据集上的效果。为了探究RSC在相关性偏移数据上的效果，我又构建了数据集4：所有数字颜色均为蓝色，训练集，标签为1的数字，背景色有95%的概率为黑色，5%的概率为白色，标签为0的反之；测试集，标签为1的数字，背景色全部为白色，标签为0的反之，可视化如图 16所示。这个数据集模仿猫狗问题，让背景颜色和标签相关联，目的是探究模型是否会学到错误的关联。 不改变LeNet5的结构，同时用ERM和RSC训练，效果如图 17和表 3所示。



(a) 训练集



(b) 测试集

图 16: 数据集4可视化

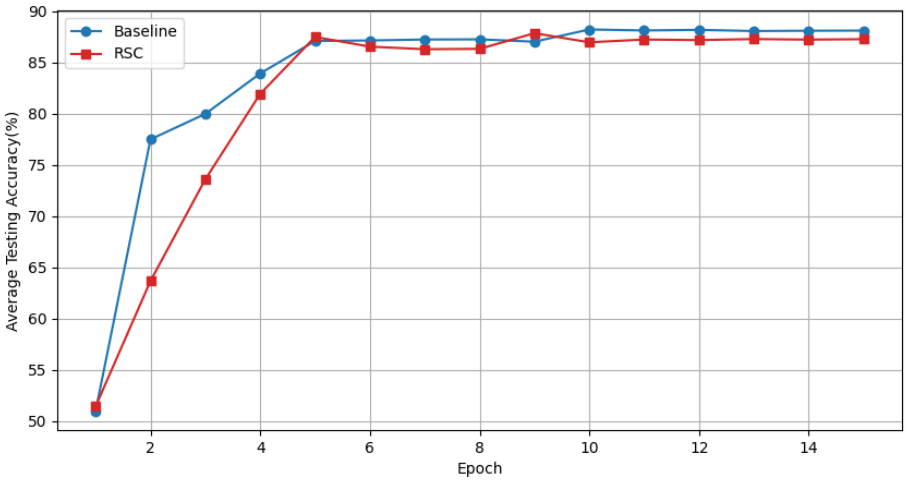


图 17: ERM和RSC在数据集4的效果对比

表 3: 不同模型在数据集4的训练与测试准确率对比

模型	训练准确率 (%)	测试准确率 (%)
Baseline	99.4	**88.1**
RSC	99.3	87.2

可以看到，在这个专门构建的数据集上，RSC的表现并不由于传统的ERM甚至略低于后者。这说明**不同OoD算法有其不同的应用场景，RSC在处理多样性偏移数据时效果会更好**。

5.2 基于特征解耦的算法尝试

我了解了特征解耦的方法，并尝试把IRM算法融入其中，具体过程如下：第一阶段为预训练，其核心是训练一个多目标变分自编码器（VAE）[7]。

主要训练三个模型：

- 编码器：要提取两个特征，**内容**和**样式**，内容就是数字形状这些在不同环境下保持不变的本质特征，样式就是背景色数字颜色这些非本质特征。输入图片，输出内容的均值、对数方差；样式的均值，对数方差。
- 解码器：负责复原图片。输入一个内容和样式的实例，要求它把图片复原出来，并返回。
- 分类器：只输入内容，要求输出类别。
- reparameterize函数：输入均值、对数方差，返回一个实例。

此阶段的损失函数整合了三部分：

- 重构损失，确保编码后的信息足以恢复原始图像，保证信息保真度；
- KL散度损失，作为正则化项，促使隐空间形成连续且规整的结构，赋予模型生成能力；这个不是非常理解，但是了解到它可以由编码器输出的分布和标准分布的距离衡量。
- 主任务分类损失

第二阶段为微调，其目标是在预训练模型的基础上，注入不变性约束以最大化OOD泛化性能。此阶段加载第一阶段训练好的权重，并采用一个极小的学习率。训练引入了基于IRM的损失。

通过最小化损失，该算法直接惩罚模型在不同环境（如不同颜色）下的性能不一致性，从而强迫编码器输出内容表示进一步提纯，最终学习到一个对风格变化鲁棒、仅依赖内容进行决策的泛化模型。

实验发现，预训练后的模型在数据集3的测试准确率可以达到70%左右，但是训练过程非常不稳定；微调效果不显著，甚至对模型起负面作用。这也启示我，强行拼接不同算法，即使它们理论上可以同时使用，实际效果可能会是$1+1<2$。

5.3 交流后的一些注解

- **关于RSC+IRM的合理性**

  我们实现了RSC+IRM结合的算法。因为IRM作用在损失函数，目的是“提取共同特征”，需要不同环境；RSC作用在前向传播过程，目的是“剔除主要特征”，对是否需要不同环境没有强制要求。因此两者是并行不悖的且我们也实现了它们的结合。

  但是我们验证发现，有的时候结合的效果略优于单一算法的效果，但没有显著的优势，且稍微修改超参数，结果可能会反转。这也是印证我前面所说的，**生硬地拼接两个算法，就好比你同时使用Dropout层和在损失函数加上L2/L1正则化，未必能有叠加的效果**。

- **关于图10：注意力的可视化**

  并不是一直都“注意力这么集中”，我们是选取了特定通道，并且选取了效果比较好的一组图片，其他图片的结果未必这么好，甚至有些注意到的是背景。但是为什么测试准确率还是这么高，其中的机制有待进一步研究。

- **关于RSC**

  RSC一直在剔除主要特征，会不会把本质的特征也剔除了？我的理解是有可能，这也是设置掩码修正（参见4.1.5）的意义。同时，我们可以通过动态调整drop_rate，如一种衰减的方式，避免这种情况的发生。

- **关于batch_size**

  有同学对敏感度分析关于批次大小的图产生疑问，我已经在原文中给出解释。

5.4 其他同学一些有意思的想法

谢卓航、魏新萌组，在RSC算法中对掩码生成（参见4.1.4）进行修改，如不抑制时乘上大于1的系数而不是1，抑制时也不是全变为0而是变成一个比较小的数，她们在数据集1的效果可以去到90%。其实这个我也尝试过，但是是在数据集3，效果不是很明显，但是是个不错的思路。

柴金成等，在IRM算法的损失函数中加入了L_2正则化项，可以使训练过程更稳定。

References

[1] Nanyang Ye et al. “OoD-Bench: Quantifying and Understanding Two Dimensions of Out-of-Distribution Generalization”. In: (2022).

[2] Martin Arjovsky et al. “Invariant Risk Minimization”. In: abs/1907.02893 (2019).

[3] Yaroslav Ganin et al. “Domain-Adversarial Training of Neural Networks”. In: (2017), pp. 189–209.

[4] Zeyi Huang et al. “Self-challenging Improves Cross-Domain Generalization.” In: (), pp. 124–140.

[5] Han Yu et al. “Towards Out-Of-Distribution Generalization: A Survey”. In: abs/2108.13624 (2021).

[6] Ramprasaath R. Selvaraju et al. “Grad-CAM: Visual Explanations from Deep Networks Via Gradient-Based Localization”. In: 128 (2019), pp. 336–359.

[7] Irina Higgins et al. “Beta-Vae: Learning Basic Visual Concepts with a Constrained Variational Framework.” In: 2016.