AlignMixup: Improving Representations By Interpolating Aligned Features

摘要

Mixup 是一种强大的数据增强方法，通过在输入或特征空间中对两个或多个样本进行插值，同时对对应的目标标签进行插值。然而，如何最优地对图像进行插值仍然未有明确定义。最近的 Mixup 方法尝试在一幅图像上叠加或裁剪拼贴两个或更多的对象，这需要仔细选择区域。Mixup 还与自动编码器（autoencoders）相关联，因为自动编码器通常会生成从一个图像连续变形为另一个图像的结果。然而，这样的图像通常质量较低。在本文中，我们从“变形”的角度重新审视 Mixup，并引入了 AlignMixup 方法，该方法在特征空间中几何对齐两幅图像。对齐的特征允许我们在两组特征之间进行插值，同时保留其中一组的位置信息。值得注意的是，这种方法保留了一幅图像的几何形状或姿态，同时保留了另一幅图像的外观或纹理。我们还证明，即使在 Mixup 训练中，自动编码器也可以在分类器从未见过解码图像的情况下改善表示学习。AlignMixup 在五个不同的基准上超过了当前最先进的 Mixup 方法。

关键词：数据增强；图像分类

1 引言

数据增强是一种有效的正则化方法，通过增加标注或非标注数据的数量和多样性，能够显著提高模型的泛化能力，并且几乎不增加额外的计算成本。因为同一个样本可以在多个训练周期中以不同的方式进行变换，从而丰富了模型的训练数据。然而，传统的数据增强方法通常仅对单一图像进行处理，并且限于进行保持标签一致性的变换（label-preserving transformations），这使得其在探索图像流形（image manifold）之外的可能性时存在一定局限性。因此，传统方法对于抗击记忆化训练数据和对抗样本的敏感性提升作用有限。

Mixup 方法通过对两个或多个样本进行混合操作，在输入空间或特征空间中进行插值，同时对目标标签进行插值。该方法能够使类别表示更加平滑，减少过度自信的错误预测，并在远离训练数据的区域内平滑决策边界。然而，Mixup 在输入空间中进行图像插值时，往往会导致生成不自然的图像。

近年来，新的 Mixup 方法尝试在输入空间中将不同图像中的多个对象合成一个新图像，从而更加高效地利用训练数据中的像素。然而，随机选择拼接区域和标签混合可能会误导分类器，导致模型学习到的特征不够有用。这也提出了一个关键问题：如何才能找到合适的图像插值方式?

研究表明，在网络的深层表示流形上进行插值，更容易得到真实的样本，因为这种插值方式能够平滑地遍历数据的底层流形，从而捕获两幅图像的显著特征。自动编码器通过解码混合后的潜在表示（latent codes）能够有效捕捉图像的语义对应关系。然而，这些方法通常计算开销较大，需要同时训练编码器、解码器和分类器，有时还需要加入对抗性判别器。此外，尽管这些方法在某些任务中取得了进展，但在大型数据集上的表现往往不如传统的输入空间 Mixup，因为生成的图像质量较差。

本文提出了一种新的视角：从变形（deformation）的角度出发，将其视为一种自然的图像插值方式，其中一幅图像可以连续地变形为另一幅图像。与传统方法不同，我们并不直接在输入空间中进行插值，也不局限于向量形式的潜在编码，也不解码图像。相反，我们提出了一种基于特征空间中显式语义对应关系的几何对齐方法。

具体来说，我们通过显式对齐两张图像的特征张量，生成软对应关系。这些特征张量可以看作是具有坐标的特征集合，因此一个集合中的每个特征都可以与另一个集合中的少量特征进行插值。通过选择保留一个集合的坐标或另一个集合的坐标，我们定义了一种非对称操作。最终得到的结果是一张图像连续地形变，而不是两张图像的简单叠加。观察这种非对称形变，我们发现插值后的图像能够保留其中一张图像的几何结构或姿态，同时呈现另一张图像的外观或纹理。

如图 1所示，我们的方法（AlignMixup）能够保留图像 2 的姿态和图像 1 的纹理，这与现有的 mixup 方法有所不同。需要注意的是，与流形 mixup 类似，我们不进行解码，因此无需担心生成图像的质量。

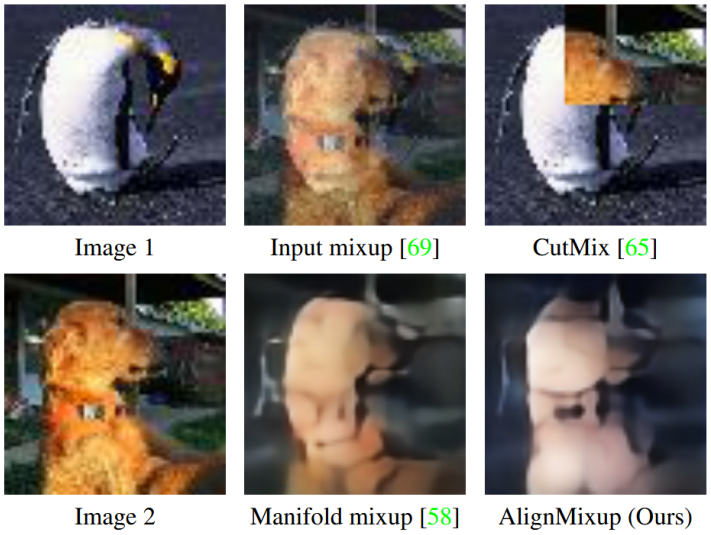


图 1. 不同的混合方法

2 相关工作

2.1 Mixup

[32] 及其他类似方法 [15] [26] 提出了通过线性插值两个样本来增强数据。虽然 [32] 在中间表示上应用了 Mixup，但 [28] 是最早提出该方法的，提出了流形 Mixup。在没有对齐的情况下，结果是图像 [32] 或特征 [28] 的叠加。为了消除“流形入侵”——即混合数据与真实

数据冲突， [8] 进行了一些改进。与流形 Mixup 不同，AlignMixup 通过对齐后在深层的特征张量上进行插值。另一种选择是对随机图像区域进行非线性混合，例如从遮掩方形区域 [5] 到从一幅图像中剪切矩形区域并将其粘贴到另一幅图像上 [31]，以及使用任意区域的多种变体 [10] [24] [25]。与随机选择区域不同，显著性可以用于定位来自不同图像的对象，并将它们合成一幅图像 [16] [17] [21] [27]。有研究提出通过利用教师网络的知识，基于显著性混合图像 [4]。与将多个对象组合到一幅图像中不同，AlignMixup 尝试将一个对象变形为另一个对象。另一种选择是 Automix [33]，它使用 U-Net 代替自动编码器，在多个层级进行混合。该方法仅限于小数据集，并且相对于流形 Mixup [28] 提升有限。StyleMix 和 StyleCutMix [13] 在两幅图像之间插值内容和风格，使用 AdaIN [14] 风格迁移自动编码器网络。与之相比，AlignMixup 对齐特征张量并直接插值匹配特征，而不使用任何额外的网络。

2.2 Alignment

图像配准 [2] [19]、光流 [30]、语义对齐 [9] [11] 和图像检索 [23] 中曾使用过局部对应关系来进行特征张量的类内对齐。这里，我们主要使用类间对齐。在少样本学习中，查询图像与支持图像之间的局部对应关系对于查找注意力图至关重要，例如 CrossTransformers [6] 和 DeepEMD [32] 都利用了这一点。地球移动者距离（EMD） [22]，或 Wasserstein 度量，是最优传输 [29] 的一种实例，通过线性规划来解决。为了加速， [3] 通过 Sinkhorn 距离与熵正则化计算最优匹配。该距离广泛应用于生成模型中的分布之间 [7] [20]。EMD 曾用于输入空间中的 Mixup，例如 3D 点云的点 Mixup [1] 和用于图像的 OptTransMix [33]，这与我们的工作最为接近。然而，坐标对齐仅适用于背景干净的图像。我们则在特征空间中对齐张量，这是一种通用的方法。我们通过使用 Sinkhorn 距离来实现这一点，这比 EMD [3] 快上几个数量级。

3 本文方法

3.1 基本概念

问题表述

设 (x, y) 为一张图像，其中 $x \in X$ 为输入图像，$y \in Y$ 为其一热编码的类别标签。X 表示输入图像空间，$Y = [0,1]^k$ 表示类别标签空间，其中 k 为类别的数量。一个编码网络 $F : X \to \mathbb{R}^{c\times w\times h}$ 将图像 x 映射到特征张量 $A = F(x)$，其中 c 为通道数，$w \times h$ 为空间分辨率。一个分类器 $g : \mathbb{R}^{c\times w\times h} \to \mathbb{R}^k$ 将特征张量 A 映射到概率向量 $p = g(A)$，表示类别的概率。

Mixup

我们沿用了 [28] 中的方法，在网络的不同层中混合表示，尤其是靠近分类器的最深层。给定两组标注图像 (x, y)、$(x', y') \in \mathcal{X} \times \mathcal{Y}$，从 Beta 分布 $\mathrm{Beta}(\alpha, \alpha)$ 中随机抽取插值因子 $\lambda \in [0, 1]$。随后，我们通过标准 Mixup 操作对标签 y, y' 进行线性插值：

$$\mathrm{mix}_\lambda(y, y') := \lambda y + (1 - \lambda)y' \tag{1}$$

并使用以下通用公式对输入 x, x' 进行插值：

$$\mathrm{Mix}_\lambda^{f_1,f_2}(x, x') := f_2(\mathrm{mix}_\lambda(f_1(x), f_1(x'))) \tag{2}$$

其中 Mix_λ 是需要定义的 Mixup 操作符。该通用公式允许在输入或特征层 $f_2 \circ f_1$ 中进行插值，具体如下：

$$\text{input } (x): \quad f_1 := \text{id}, \quad f_2 := F \tag{3}$$

$$\text{feature } (A): \quad f_1 := F, \quad f_2 := \text{id} \tag{4}$$

其中 id 表示恒等映射。在公式 (3) 中，我们在公式 (2) 中将 Mix_λ 定义为标准 Mixup mix_λ（公式 (1)），类似于文献 [32] 的方法；而对于公式 (4)，我们按照第 3.2 节中讨论的方法定义 Mix_λ。

默认情况下，我们通过对混合样本的分类器输出以及相应的混合标签使用分类损失函数 L_c，对编码器网络和分类器进行训练：

$$L_c(g(\text{Mix}_\lambda^{f_1,f_2}(x, x')), \text{mix}_\lambda(y, y')) \tag{5}$$

其中 $L_c(p, y) := -\sum_{i=1}^{k} y_i \log p_i$ 是标准交叉熵损失。有关更多选项（如使用自编码器结构）的探讨，请参考第 4 节。

3.2 对齐特征张量的插值

Alignment

对齐是指在插值之前找到图像元素之间的几何对应关系。特征张量非常适合这一任务，因为它的空间分辨率较低，从而减少了优化成本，并且允许进行语义对应，因为靠近分类器的特征通常较小。值得注意的是，我们并不试图将两个或更多的物体组合成一张图像，而是将两个物体对齐，然后插值成一个图像。我们不对输入图像的物体结构做任何假设，也不使用真实的对应关系。我们的特征张量对齐基于最优传输理论 [29]，特别是基于 Sinkhorn 距离（SD）[3]。

设 $A := F(x)$，$A' := F(x')$ 为图像 x 和 x' 的特征张量。我们通过展开空间维度将它们重塑为 $c \times r$ 的矩阵 A 和 A'，其中 $r := h \times w$。然后，矩阵 A 和 A' 的每一列 $a_j, a'_j \in \mathbb{R}^c$ 都对应于原图像 x 和 x' 中某个空间位置的特征向量。

设 M 为 $r \times r$ 的成本矩阵，其中元素

$$m_{ij} := \|a_i - a'_j\|^2 \tag{6}$$

表示这些特征向量之间的成对距离。我们希望找到一个传输计划，即一个 $r \times r$ 的矩阵 $P \in U_r$，其中：

$$U_r := \{P \in \mathbb{R}^{+r \times r} : P1 = P^T 1 = \frac{1}{r}\} \tag{7}$$

并且 1 是 $\mathbb{R}^r$ 中的全 1 向量。也就是说，P 是非负的，并且每行和每列的和为 $1/r$，表示 A 和 A_0 空间位置的联合概率，边缘分布为均匀分布。我们选择 P 以最小化其特征的成对距离的期望值，目标函数如下：

$$P^* = \arg\min_{P \in U_r} \langle P, M \rangle - \epsilon H(P) \tag{8}$$

其中，$H(P) := \sum_{ij} p_{ij} \log p_{ij}$ 是矩阵 P 的熵，$\langle \cdot, \cdot \rangle$ 是 Frobenius 内积，ϵ 是正则化系数。最优解 P^* 是唯一的，可以通过形成 $r \times r$ 的相似矩阵 $\frac{M}{\epsilon}$，然后应用 Sinkhorn-Knopp 算法进行求解，即通过迭代地归一化行和列。

插值

赋值矩阵 $R := r^{P^*}$ 是一个双随机的 $r \times r$ 矩阵，其元素 r_{ij} 表示矩阵 A 中的列 a_i 与矩阵 A' 中的列 a'_j 之间的对应概率。因此，我们通过以下方式对齐矩阵 A 和 A'：

$$\widetilde{A} := A' R^T \tag{9}$$

$$\widetilde{A}' := AR \tag{10}$$

这里，矩阵 $\widetilde{A}$ 中的列 $\widetilde{a}_i$ 是矩阵 A' 列的凸组合，这些列与矩阵 A 中的同一列 a_i 对应。我们通过扩展空间维度，将 $\widetilde{A}$ 重塑为 $c \times w \times h$ 的张量 $\widetilde{A}$，并称 $\widetilde{A}$ 为对齐后的 A 对应于 A'。然后，我们在 A 和原始特征张量 A' 之间进行插值：

$$\mathrm{mix}_\lambda(A, \widetilde{A}) \tag{11}$$

如图 2所示，$\widetilde{A}$ 在几何上接近 A。由于 $\widetilde{A}$ 与 A' 之间存在明确的对应关系，同时又在几何上与 A 接近，因此它适合与 A 进行插值。对称地，我们也可以将 A' 对齐到 A，并在 $\widetilde{A}'$ 和 A' 之间进行插值：

$$\mathrm{mix}_\lambda(A', \widetilde{A}') \tag{12}$$

当对齐后的特征张量进行混合时，我们在公式（2）中定义的 Mix_λ 操作将 (A, A') 映射到（11）或（12），并根据需要随机选择其中之一。

图 2描述了特征张量的对齐与插值过程。代价矩阵 M 包含张量 A 和 A' 中特征向量之间的成对距离。通过在相似度矩阵 $e^{-M/\tau}$ 上应用 Sinkhorn-Knopp [18] 算法，得到分配矩阵 R。根据 R，将 A 对齐到 A'，得到 $\tilde{A}$。接着，我们在 A 和 $\tilde{A}$ 之间进行插值。对称地，我们也可以将 A' 对齐到 A，并在 A' 和 $\tilde{A}'$ 之间进行插值。左侧展示了 A 和 A'（16 个二维点的简化示例），右侧为半透明的参考图。

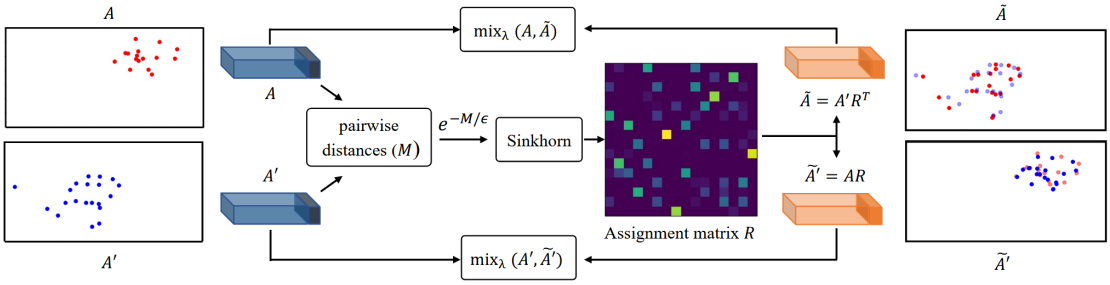


图 2. 特征张量对齐和插值

图 3. 可视化对齐

3.3 可视化与讨论

解码器

我们使用解码器来研究有无特征对齐的生成图像。设 $f : \mathbb{R}^{c\times w\times h} \to \mathbb{R}^d$ 为将张量 A 映射到嵌入空间的全连接层。我们使用 $f \circ F$ 作为编码器，解码器 $D : \mathbb{R}^d \to X$ 将嵌入 $e = f(A)$ 映射回图像空间，重建图像 $\hat{x} = D(e)$。该自动编码器仅使用未混合的清晰图像进行训练，通过重建损失 L_r 来优化：$L_r(x, x') = \|x - x'\|^2$ 。我们使用生成的图像仅用于可视化目的，但在第 4 节的 AlignMixup 训练中也可选择性使用。

对比

对于不同的 $\lambda \in [0, 1]$，我们使用（11）或（12）对特征张量 A 和 A_0 进行插值，生成新的图像，并通过解码器 D 解码嵌入。图 3展示了生成的图像，对齐的可视化。对于不同的 $\lambda \in [0, 1]$，我们在没有对齐（上图）或已对齐的特征张量（下图）情况下，对两个图像 x 和 x' 的特征张量 A 和 A' 进行插值，并通过解码器 D 解码生成新的图像。(a)、(c) 中我们将 A 对齐到 A' 并与（11）进行混合。(b)、(d) 中我们将 A' 对齐到 A 并与（12）进行混合。仅为示意：在训练过程中，分类器并未看到解码后的图像。当将 A 对齐到 A_0 并使用（11）进行混合时，当 $\lambda = 0$ 时，生成的图像保留了 x 的姿态和 x_0 的纹理。特别是在图 3(a) 中，当 x 为“企鹅”而 x_0 为“狗”时，生成的图像保留了企鹅的姿态，同时狗的纹理与企鹅的身体对齐。类似地，在图 3(c) 中，金鱼的纹理与鹳鸟的纹理对齐，而鹳鸟的姿态被保留。反之，如图 3(b) 和图 3(d) 所示，当将 A_0 对齐到 A 并使用（12）进行混合时，当 $\lambda = 0$ 时，生成的图像保留了 x_0 的姿态和 x 的纹理。与此相对，未经对齐的特征生成的图像则更像是简单的重叠。

通过在训练过程中随机选择多个 $\lambda \in [0, 1]$，我们可以生成大量样本，捕捉来自一张图像的纹理和来自另一张图像的姿态。这使得模型能够探索图像流形之外的空间，从而提高其泛化能力，并在多个基准任务中取得更好的表现，如第 5 节讨论的那样。

4 复现细节

4.1 与已有开源代码对比

原论文有开源代码，代码可在 https://github.com/shashankvkt/AlignMixup_CVPR22.git 下载。在复现过程中，先运行了源代码验证模型的有效性，然后在源代码基础上对数据增强、模型架构、Mixup 策略以及自监督学习方法这几个方面进行了探索，以期在提升模型性能的同时增强其泛化能力和鲁棒性。

在数据增强方面，对原有的训练数据预处理进行了扩展，在原有的 RandomCrop、RandomHorizontalFlip 基础上加入了新的图像增强操作，以此来以增加训练数据的多样性，提升模型对不同环境和条件下数据变化的鲁棒性。在数据预处理阶段引入了 ColorJitter，通过随机变化图像的亮度、对比度、饱和度和色调，从而模拟了不同光照条件下的图像变换，这为模型提供了更加丰富的训练数据；还添加了 RandomRotation，以在训练过程中模拟不同视角的变化，通过随机旋转图像来模拟不同视角下的变化，增加模型对图像几何变换的适应性。这些数据增强方法的结合，期望能够使得模型在面对多样化的输入数据时表现出更强的稳定性与泛化能力。

在模型架构的选择上，在原有的 ResNet 和 WideResNet 模型进行扩展，可选择更强大的网络架构，如 Vision Transformer（ViT）和 Swin Transformer。这些 Transformer 架构相较于传统卷积神经网络（CNN），具有更强的全局特征捕获能力和更深的上下文建模能力，特别适合处理复杂的图像任务，将这些架构引入到实验中，可比较它们与传统 CNN 架构在各数据集上的性能差异。同时，还可以引入轻量化网络架构，如 MobileNetV2，在保证计算效率的同时可优化模型的参数量和计算量，尤其适用于计算资源有限的场景。这一部分的探索可为模型架构的选择提供更多的可能性，可揭示在不同硬件环境下的最佳实践。

在 Mixup 策略的改进方面，可以进一步探索结合 CutMix 和 Manifold Mixup 等其他插值方法的可能性。可以 0.5 的概率随机选择使用 AlignMixup 或 CutMix，尝试结合不同的插值方法来提升模型的鲁棒性。CutMix 通过切割图像并将其部分区域替换，能够进一步增强模型对局部特征的学习，而 Manifold Mixup 则通过在高维特征空间中进行插值，增强了模型的正则化效果。通过这种策略，希望能够优化 Mixup 的表现，探索不同增强策略对训练过程的影响，从而提升模型的泛化能力。

还可以将自监督学习方法，如对比学习（SimCLR、BYOL 等），引入到训练过程中。这些方法通过最大化相似样本之间的距离，最小化不同样本之间的距离，帮助模型学习到更具判别性的特征。通过将对比学习损失与传统的分类损失相结合，期望能提升模型对无监督信息的学习能力，从而提高其在数据稀缺场景下的表现。这一部分的探索将有助于理解自监督学习在图像分类任务中的潜力，特别是在处理复杂图像或低标注数据的场景下。

4.2 实验环境搭建

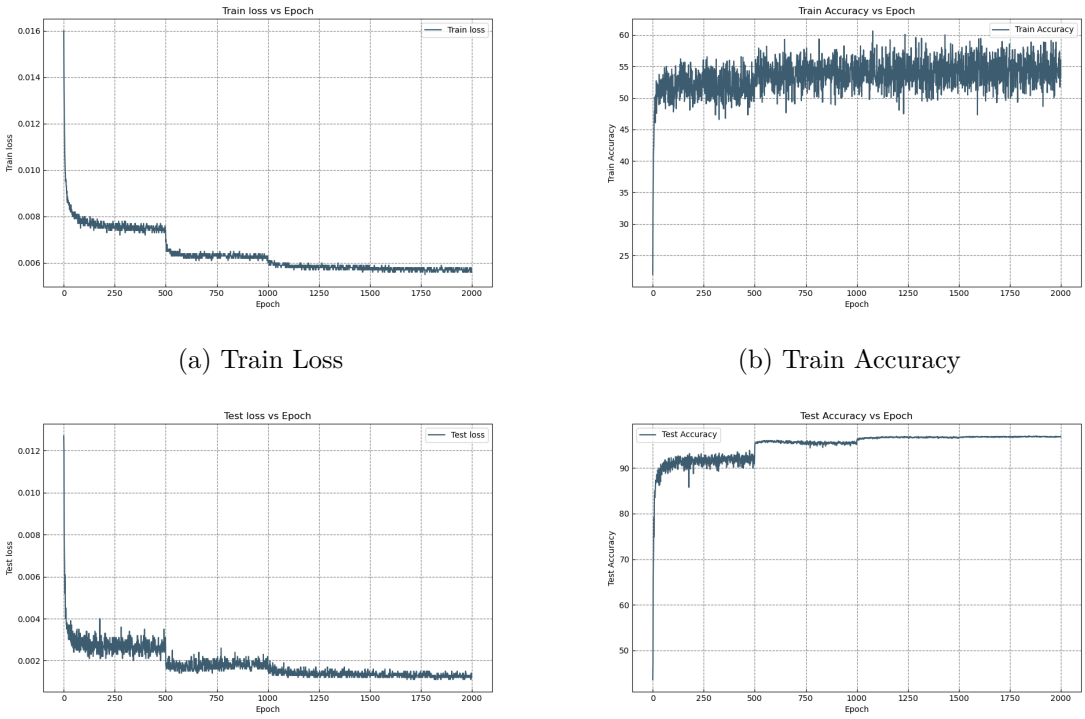
实验环境搭建在 Linux 服务器上进行，硬件配置采用 NVIDIA A6000 显卡，以确保高效的计算性能。软件环境方面，Python 版本为 3.8.11，深度学习框架使用 PyTorch 1.10.1 和 TorchVision 0.11.2 版本。依赖库包括 NumPy 1.21.0、CUDA 11.3.1 和 cuDNN 8.2.0.53-11.3 版本，确保在 GPU 加速下的良好兼容性。此外，安装了 tar 1.34 和 py-virtualenv 16.7.6 等

工具，以支持环境的虚拟化和依赖管理。该环境配置保证了实验的顺利进行，并充分发挥了硬件性能，最大限度地提升了计算效率。

5 实验结果分析

5.1 复现结果呈现

在 CIFAR-10 数据集上使用 PreActResNet18（R-18）作为基础框架进行复现，训练次数为 2000 轮。



(a) Train Loss

(b) Train Accuracy

(c) Test Loss

(d) Test Accuracy

图 4. Paper Reproduction

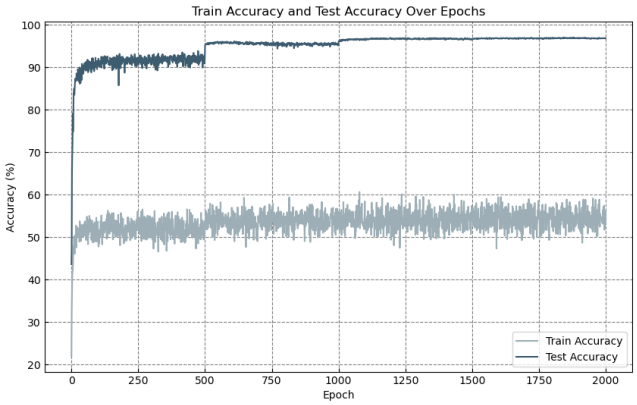


图 5. 训练精度与测试精度

5.2 图像分类与鲁棒性

在 CIFAR-10 和 CIFAR-100 数据集上使用 PreActResNet18（R-18）和 WRN16-8(WRN) 作为基础架构。基于 Manifold Mixup [28] 的实验设置，复现了部分当前最先进（SOTA）的 Mixup 方法，所有方法均基于作者提供的官方代码。部分框架因为其实验设置与本文的不同且无公开代码可用没有进行比较。使用 top-1 错误率（%）作为评估指标，验证 AlignMixup 在图像分类和对抗性鲁棒性（针对 FGSM 和 PGD 攻击）的效果。

图像分类

如表 1 所示，AlignMixup 和 AlignMixup/AE 的性能达到或超越了 SOTA 方法，尤其是在大规模数据集上取得了最低的 top-1 错误率。在 CIFAR-10 数据集上，AlignMixup 和 AlignMixup/AE 的表现与 Co-Mixup 和 PuzzleMix 相当（基于 R-18 和 WRN16-8）。在 CIFAR-100 数据集上，AlignMixup 分别比 StyleCutMix 和 Manifold mixup 高出 1.05% 和 0.46%（基于 R-18 和 WRN16-8）。在 TI 数据集上，AlignMixup 比 Co-Mixup 提高了 2.72%（基于 R-18）。从表 2 中可以看到，AlignMixup/AE 在 ImageNet 上比 PuzzleMix 提高了 2.41%。虽然 SOTA 方法在 ImageNet 上相较基准的总体改进仅约为 2%，AlignMixup/AE 进一步提升了 2.5%。

表 1. CIFAR-10/100 和 TI (TinyImagenet) 上的图像分类 top-1 错误 (%)

DATASET	CIFAR-10		CIFAR-100		TI
NETWORK	R-18	W16-8	R-18	W16-8	R-18
Baseline	5.19	5.11	23.24	20.63	43.40*
Input	4.03	3.98	20.21	19.88	43.48*
CutMix	3.27	3.54	19.37	19.71	43.11*
Manifold	2.95	3.56	19.80	19.23	40.76*
PuzzleMix	2.93	**2.99**	20.01	19.25	36.52*
Co-Mixup	2.89	3.04	19.81	19.57	35.85*
SaliencyMix	2.99	3.53	19.69	19.59	34.81
StyleMix	3.76	3.89	20.04	20.45	36.13
StyleCutMix	3.06	3.12	19.34	19.28	34.49
AlignMixup (ours)	2.95	3.09	18.29	18.77	33.13
AlignMixup/AE (ours)	**2.83**	3.15	**17.82**	**18.09**	**32.73**
Gain	+0.06	-0.10	+1.52	+1.14	+1.76

计算复杂度

表 2 展示了 AlignMixup 在 ImageNet 上与基准及 SOTA 方法相比的训练计算复杂度，使用 NVIDIA RTX 2080 Ti GPU 衡量参数数量和每批处理时间（毫秒/批次）。AlignMixup 的计算开销与 Manifold mixup 基本相同，但准确率提高了 1.82%。相比之下，SOTA 方法如 Co-Mixup 和 PuzzleMix 的计算开销分别比 AlignMixup 高 1.8 倍和 2.3 倍，但准确率平均低 0.6%。AlignMixup/AE 在 AlignMixup 的基础上进一步将准确率提高了 1.85%。需要注意的是，AlignMixup/AE 的参数数量增加了 40%，主要源于残差解码器，而该解码器仅在五分之一的情况下（非 mixup 的干净样本）被使用。在推理过程中，所有方法的计算复杂度是相同的。

表 2. 使用 Resnet-50 对 ImageNet 进行 300 个 epoch 的图像分类 top-1 错误 (%) 和计算分析

METHOD	PARAM.	MSEC/BATCH	TOP-1 ERROR (%)
Baseline	25M	418	23.68
Input†	25M	436	22.58
CutMix†	25M	427	21.40
Manifold†	25M	441	22.50
PuzzleMix†	25M	846	21.24
Co-Mixup*	25M	1022	–
SaliencyMix*	25M	462	21.26
StyleMix*	25M	828	–
StyleCutMix*	25M	912	–
AlignMixup (ours)	25M	450	20.68
AlignMixup/AE (ours)	35M	688	**18.83**
Gain			+2.41

挑战

从表 1 可以看出，AlignMixup 在 CIFAR-10 和 CIFAR-100 数据集上达到了 SOTA 的 top-1 错误率。这些结果是在 2000 个训练周期下计算得出的，遵循 [28] 的实验设置，其最佳性能也出现在 2000 个周期时。而部分基准 mixup 方法在 300 个周期时表现最佳，但并未从较长的训练时间中获益。与这些在图像空间中执行 mixup 的方法不同，Manifold mixup 和 AlignMixup 在特征空间中执行 mixup。这种特性使得网络需要更长时间训练以学习有意义的表示。对于 AlignMixup，由于在比 Manifold mixup 更深的层次上进行特征混合，这一挑战更加显著。实验数据显示，当训练周期从 300 增加到 2000 时，Manifold mixup 的 top-1 错误率从 21.64% 降至 19.80%，而 AlignMixup 的 top-1 错误率从 21.38% 降至 18.29%。

对抗性鲁棒性：FGSM 和 PGD 攻击

根据 [17] 的评估协议，我们对 FGSM 使用 $8/255\,\ell_\infty$ ϵ-ball，对 PGD 使用 $4/255\,\ell_\infty$ ϵ-ball，步长为 2/255。我们在 CIFAR-10 和 CIFAR-100 上复现了 FGSM 和 PGD 的竞争方法结果；在 TI 数据集上复现了基准、Input、Manifold、CutMix 和 PuzzleMix 的 FGSM 结果 [17]，并复现了 SaliencyMix、StyleMix 和 StyleCutMix 的结果。如表 3 所示，AlignMixup 在鲁棒性方面比 SOTA 方法更优。在 CIFAR-10 的图像分类中，AlignMixup 与 PuzzleMix 和 Co-Mixup 表现相当，但在 FGSM 攻击的鲁棒性方面，AlignMixup 分别比 Co-Mixup 和 PuzzleMix 提高了 5.36% 和 2.28%。此外，在 TI 数据集上的 FGSM 攻击和 CIFAR-100 数据集上的更强 PGD 攻击下，AlignMixup 也表现出显著的鲁棒性提升。

5.3 Overconfidence

深度神经网络在远离训练数据的错误预测中往往表现得过于自信，而 Mixup 能够帮助缓解这一问题。评估改进的两个标准基准是：检测分布外数据（Out-of-distribution, OOD）的能力和校准能力，即准确率与置信度之间的差异。

表 3. 对 FGSM 和 PGD 攻击的鲁棒性

ATTACK	FGSM					PGD			
DATASET	CIFAR-10		CIFAR-100		TI	CIFAR-10		CIFAR-100	
NETWORK	R-18	W16-8	R-18	W16-8	R-18	R-18	W16-8	R-18	W16-8
Baseline	89.41	88.02	87.12	72.81	91.85	99.99	99.94	99.97	99.99
Input	78.42	79.21	81.30	67.33	88.68	99.77	99.43	99.96	99.37
CutMix	77.72	78.33	86.96	60.16	88.68	99.82	98.10	98.67	97.98
Manifold	77.63	76.11	80.29	56.45	89.25	97.22	98.49	99.66	98.43
PuzzleMix	57.11	60.73	78.70	57.77	83.91	97.73	97.00	96.42	95.28
Co-Mixup	60.19	58.93	77.61	56.59	–	97.59	96.19	95.35	94.23
SaliencyMix	57.43	68.10	77.79	58.10	81.16	97.51	97.04	95.68	93.76
StyleMix	79.54	71.05	80.54	67.94	84.93	98.23	97.46	98.39	98.24
StyleCutMix	58.79	56.12	77.49	56.83	80.59	97.87	96.70	91.88	93.78
AlignMixup (ours)	54.83	56.20	**74.18**	**55.05**	**78.83**	**95.42**	96.71	**90.40**	**92.16**
AlignMixup/AE (ours)	**52.13**	**54.86**	76.40	55.44	78.98	97.16	**95.32**	91.69	92.23
Gain	+4.98	+1.26	+3.31	+1.40	+1.76	+1.80	+0.87	+1.48	+1.60

分布外检测

根据文献 [12]，分布内数据（In-distribution, ID）指从网络训练分布中抽取的测试样本，而从任何其他分布中抽取的样本则是分布外数据（OOD）。在推理过程中，面对混合的 ID 和 OOD 样本，网络通过 Softmax 为已知类别分配概率。若最大类别概率低于某一阈值，则样本被分类为 OOD，否则为 ID。一个校准良好的网络应该能够为 ID 样本分配更高的概率，而为 OOD 样本分配更低的概率，从而更容易区分这两类分布。

我们基于 CIFAR-100 数据集（子节 4.1 中讨论）使用 R-18 训练的网络，将 AlignMixup 与 SOTA 方法进行比较。在推理过程中，ID 样本为 CIFAR-100 的测试图像，而 OOD 样本为 LSUN（crop）、iSUN 和 Tiny-ImageNet（crop）的测试图像，其中 crop 表示 OOD 样本被中心裁剪为 32×32 以匹配 ID 图像的分辨率 [31]。按照 [12]，我们使用 0.5 的阈值测量检测准确率（Detection Accuracy, Det Acc），并计算 ROC 曲线下面积（Area under ROC curve, AuROC）和精确率-召回率曲线下面积（Area under Precision-Recall curve, AuPR）。

如表 4 所示，AlignMixup 在所有指标上均大幅优于 SOTA 方法，表明其在减少过度自信预测方面效果更好。此外，我们还观察到，Input Mixup 的效果不如基线网络，这与文献 [31] 的结论一致。更多结果见补充材料。

5.4 弱监督目标定位（WSOL）

WSOL 的目标是在训练时仅使用类别标签而非边界框来定位目标物体。WSOL 通过提取视觉判别线索，指导分类器关注图像中的显著区域。我们使用与图像分类相同的过程训练 AlignMixup。在推理阶段，使用 CAM 计算显著图，将其通过 0.15 的阈值二值化，并取掩码的边界框。我们使用在 ImageNet 上预训练的 VGG-GAP 和 ResNet-50 ，并在 CUB200-2011 上进行微调。

使用 top-1 定位准确率（IoU 阈值为 0.5）和最大边界框准确率（Maximal Box Accuracy, MaxBoxAcc-v2）将 AlignMixup 与以下方法进行比较：不使用 Mixup 的基线 CAM、Input Mixup、CutOut 和 CutMix。如表 5 所示，在 top-1 定位准确率上，AlignMixup 使用 VGG-GAP 分别比 Input Mixup、CutOut 和 CutMix 提高了 11.4%、7.3% 和 0.6%，而使用 ResNet-50 分

表 4. 使用 PreActResnet18 进行分布外检测

TASK	OUT-OF-DISTRIBUTION DETECTION											
DATASET	LSUN (CROP)				ISUN				TI (CROP)			
METRIC	DET ACC	AUROC	AUPR (ID)	AUPR (OOD)	DET ACC	AUROC	AUPR (ID)	AUPR (OOD)	DET ACC	AUROC	AUPR (ID)	AUPR (OOD)
Baseline	54.0	47.1	54.5	45.6	66.5	72.3	74.5	69.2	61.2	64.8	67.8	60.6
Input	57.5	59.3	61.4	55.2	59.6	63.0	60.2	63.4	58.7	62.8	63.0	62.1
Cutmix	63.8	63.1	61.9	63.4	67.0	76.3	81.0	77.7	70.4	84.3	87.1	80.6
Manifold	58.9	60.3	57.8	59.5	64.7	73.1	80.7	76.0	67.4	69.9	69.3	70.5
PuzzleMix	64.3	69.1	80.6	73.7	73.9	77.2	79.3	71.1	71.8	76.2	78.2	81.9
Co-Mixup	70.4	75.6	82.3	70.3	68.6	80.1	82.5	75.4	71.5	84.8	86.1	80.5
SaliencyMix	68.5	79.7	82.2	64.4	65.6	76.9	78.3	79.8	73.3	83.7	87.0	82.0
StyleMix	62.3	64.2	70.9	63.9	61.6	68.4	67.6	60.3	67.8	73.9	71.5	78.4
StyleCutMix	70.8	78.6	83.7	74.9	70.6	82.4	83.7	76.5	75.3	82.6	82.9	78.4
AlignMixup (ours)	74.2	79.9	84.1	75.1	72.8	83.2	84.1	80.3	77.2	85.0	87.8	85.0
AlignMixup/AE (ours)	**76.9**	**83.5**	**86.7**	**79.4**	**75.6**	**84.1**	**85.9**	**81.7**	**79.7**	**88.0**	**89.7**	**85.7**
Gain	+6.1	+3.8	+3.0	+4.5	+1.7	+1.7	+2.2	+1.9	+4.4	+3.2	+2.6	+3.8

别提高了 6.9%、3.8% 和 1.4%。此外，在 MaxBoxAcc-v2 指标上，AlignMixup 使用 VGG-GAP 和 ResNet-50 分别比 CutMix 提高了 1.2% 和 0.6%。AlignMixup 还优于专门用于 WSOL 的方法 ACoL 和 ADL，这些方法专注于学习空间分散的表示。更多的定性定位结果见补充材料。

表 5. CUB200-2011 上的弱监督对象定位

METRIC	TOP-1 LOC.		MAXBOXACC-V2	
NETWORK	VGG-GAP	RESNET-50	VGG-GAP	RESNET-50
ACoL	45.9	–	57.4	–
ADL	52.4	–	61.3	58.4
Baseline CAM	37.1	49.4	59.0	59.7
Input	41.7	49.3	57.1	60.6
CutMix	52.5	54.8	62.6	64.8
AlignMixup (ours)	**53.1**	**56.2**	**63.8**	**65.4**
Gain	+0.6	+1.4	+1.2	+0.6

5.5 消融实验（Ablation Study）

所有消融实验均基于 CIFAR-100 数据集，使用 R-18 作为编码器 F，特征张量 A 的大小为 $512 \times 4 \times 4$。我们研究了在不同层（x 和 A）进行混合的效果，是否在混合之前对 A 进行对齐，以及在不同自动编码器架构中使用解码器 D 的效果。我们报告 top-1 准确率（%）。所有结果见表 6。关于 Sinkhorn-Knopp 算法迭代次数影响的消融实验结果总结在补充材料中。

层

我们研究了混合层的选择（不考虑特征对齐）。根据公式 (2)，我们可以在两层中的任意一层进行混合，分别表示为 $\{x, A\}$。为了探讨更多情况，我们在 AlignMixup 的编码器中引入了额外的一层 f，并对其输出进行混合，该层作为 AlignMixup/AE 的潜在空间。f 可以是一个全连接（FC）层，输出向量 $e \in \mathbb{R}^{512}$，也可以是一个卷积层（核大小为 2×2，步幅为 2），输出 $128 \times 2 \times 2$ 的张量 E。在这两种情况下，Mixup 均可由公式 (2) 表示，其中 $f_1 := f \circ F$，

表 6. 在 CIFAR-100 上使用 R-18 进行消融

METHOD/ARCH	LAYERS	UNALIGNED	ALIGNED
Baseline		76.76	-
Manifold		80.20	-
StyleCutMix		80.66	-
AlignMixup	{x, e}	80.81	-
	{A}	79.07	80.28
	{e}	78.71	-
	{x, A}	80.34	81.71
	{x, A, E}	80.46	81.36
	{x, A, e}	80.33	**81.92**
AlignMixup/AE	{x, e}	81.92	-
	{A}	79.39	81.04
	{e}	79.49	-
	{x, A}	81.78	81.85
	{x, E}	80.80	81.54
	{x, A, e}	81.61	**82.18**
AlignMixup/AE	{x, $A_{2\times2}$, e}	81.47	81.20
	{x, $A_{4\times4}$, e}	81.61	82.18
	{x, $A_{8\times8}$, e}	80.49	**82.20**
AlignMixup/VAE	{x, μ, σ)}	81.81	-
	{x, A}	81.35	81.85
	{x, (M, Σ)}	80.45	81.10
	{x, A, (μ, σ)}	81.00	**81.89**

$f_2 := id$。混合层的选择为 $\{x, A, e\}$ 或 $\{x, A, E\}$。在没有解码器的 AlignMixup 中，不同未对齐层集的选择中，$\{x, e\}$ 的分类准确率最高。此外，AlignMixup/AE 在所有层的选择中均优于基线和表现最好的竞争对手 StyleCutMix，即使未对齐时也是如此，这说明使用解码器是有益的。

张量对齐

我们研究了在混合之前是否对特征张量进行对齐的影响，分别使用标准 Mixup（公式 (2)）和公式 (11)、(12)。需要注意的是，当 e 为向量时，我们不对齐。在 AlignMixup 中，我们观察到在混合之前对张量 A 和 E 进行对齐能显著提高分类准确率。此外，我们观察到对 e 额外使用一个 FC 层带来的改进很小（从 81.71% 提高到 81.92%），这意味着主要的改进来自对齐。总体而言，当混合 $\{x, A, e\}$ 且对齐 A 时，AlignMixup/AE 表现最好，比 StyleCutMix 提高了 1.52%。

对齐分辨率

在 AlignMixup/AE 的最佳设置下，我们研究了在不同空间分辨率对齐 A 的效果。默认设置为 4×4（记为 $A_{4\times4}$）。我们还实验了通过平均池化得到的 2×2（$A_{2\times2}$）和通过移除最后一个卷积层的下采样得到的 8×8（$A_{8\times8}$）。8×8 的准确率仅比 4×4 略高 0.02%，但计算代价更高。因此，我们选择 4×4 作为默认设置。相比之下，在 2×2 对齐的效果甚至不如完全不对齐，这可能是由于软对应关系通过平均导致信息丢失。

自动编码器架构

我们比较了 AlignMixup 的两种自动编码器架构：普通自动编码器（AlignMixup/AE）和变分自动编码器（AlignMixup/VAE）。后者用两个向量 $\mu, \sigma \in \mathbb{R}^{512}$ 代替 e，分别表示均值和标准差。此外，我们还研究了 $128 \times 2 \times 2$ 的张量，记为 M, Σ，其中两个变量同时进行混合。对于 AlignMixup 和 AlignMixup/AE，我们研究了不同层的组合，是否对齐。在所有三种架构中，当混合 $\{x, A, e\}$ 时表现最佳。对齐在所有三种架构中均有一致的提升。AlignMixup 和 AlignMixup/VAE 均不及 AlignMixup/AE，但它们的最佳设置仍优于基线和 StyleCutMix。

6 总结与展望

输入和潜在表示的组合 Mixup 被展示为一种简单且有效的成对数据增强方法。在大规模数据集上，该方法的优势尤为显著，特别是在减少过度自信的预测（如分布外数据检测）方面表现突出。特征张量的插值能够显著提升性能，尤其是在对齐操作后。该工作的贡献在于图像空间中的“良好”手工插值与潜在空间中的“完全学习”插值之间进行了折衷。一个挑战是，在不牺牲速度和简洁性的前提下，推动后者的发展，这将直接影响该方法的广泛适用性。

参考文献

[1] Yunlu Chen, Vincent Tao Hu, Efstratios Gavves, Thomas Mensink, Pascal Mettes, Pengwan Yang, and Cees GM Snoek. Pointmixup: Augmentation for point clouds. In *ECCV*, 2020.

[2] Christopher B Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. Universal correspondence network. In *NeurIPS*, 2016.

[3] Marco Cuturi. Sinkhorn distances: lightspeed computation of optimal transport. In *NeurIPS*, 2013.

[4] Ali Dabouei, Sobhan Soleymani, Fariborz Taherkhani, and Nasser M Nasrabadi. Supermix: Supervising the mixing data augmentation. In *CVPR*, 2021.

[5] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[6] Carl Doersch, Ankush Gupta, and Andrew Zisserman. Crosstransformers: spatially-aware few-shot transfer. In *NeurIPS*, 2020.

[7] Aude Genevay, Gabriel Peyré, and Marco Cuturi. Learning generative models with sinkhorn divergences. In *AISTATS*, 2018.

[8] Hongyu Guo, Yongyi Mao, and Richong Zhang. Mixup as locally linear out-of-manifold regularization. In *AAAI*, 2019.

[9] Kai Han, Rafael S Rezende, Bumsub Ham, Kwan-Yee K Wong, Minsu Cho, Cordelia Schmid, and Jean Ponce. Scnet: Learning semantic correspondence. In *ICCV*, 2017.

[10] Ethan Harris, Antonia Marcu, Matthew Painter, Mahesan Niranjan, Adam Pru¨gel Bennett, and Jonathon Hare. Fmix: Enhancing mixed sample data augmentation. *arXiv preprint arXiv:2002.12047*, 2020.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[12] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[13] Minui Hong, Jinwoo Choi, and Gunhee Kim. Stylemix: Separating content and style for enhanced data augmentation. In *CVPR*, 2021.

[14] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017.

[15] Hiroshi Inoue. Data augmentation by pairing samples for images classification. *arXiv preprint arXiv:1801.02929*, 2018.

[16] Jang-Hyun Kim, Wonho Choo, Hosan Jeong, and Hyun Oh Song. Co-mixup: Saliency guided joint mixup with supermodular diversity. In *ICLR*, 2021.

[17] Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In *ICML*, 2020.

[18] Philip A. Knight. The sinkhorn-knopp algorithm: convergence and applications. *SIAM Journal on Matrix Analysis and Applications*, 2008.

[19] Jonathan Long, Ning Zhang, and Trevor Darrell. Do convnets learn correspondence? In *NIPS*, 2014.

[20] Giorgio Patrini, Rianne van den Berg, Patrick Forre, Marcello Carioni, Samarth Bhargav, Max Welling, Tim Genewein, and Frank Nielsen. Sinkhorn autoencoders. In *Uncertainty in Artificial Intelligence*, 2020.

[21] Jie Qin, Jiemin Fang, Qian Zhang, Wenyu Liu, Xingang Wang, and Xinggang Wang. Resizemix: Mixing data with preserved object information and true labels. *arXiv preprint arXiv:2012.11101*, 2020.

[22] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover' s distance as a metric for image retrieval. *IJCV*, 2000.

[23] Oriane Siméoni, Yannis Avrithis, and Ondrej Chum. Local features and visual words emerge in activations. In *CVPR*, 2019.

[24] Cecilia Summers and Michael J Dinneen. Improved mixed-example data augmentation. In *WACV*, 2019.

[25] Ryo Takahashi, Takashi Matsubara, and Kuniaki Uehara. Ricap: Random image cropping and patching data augmentation for deep cnns. In *ACML*, 2018.

[26] Yuji Tokozume, Yoshitaka Ushiku, and Tatsuya Harada. Learning from between-class examples for deep sound recognition. In *ICLR*, 2018.

[27] A F M Uddin, Mst. Monira, Wheemyung Shin, TaeChoong Chung, and Sung-Ho Bae. Saliencymix: A saliency guided data augmentation strategy for better regularization. In *ICML*, 2021.

[28] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, 2019.

[29] Cédric Villani. *Optimal transport: old and new*. Springer Science & Business Media, 2008.

[30] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. Deepflow: Large displacement optical flow with deep matching. In *ICCV*, 2013.

[31] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019.

[32] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover' s distance and structured classifiers. *CVPR*, 2020.

[33] Jianchao Zhu, Liangliang Shi, Junchi Yan, and Hongyuan Zha. Automix: Mixup networks for sample interpolation via cooperative barycenter learning. In *ECCV*, 2020.