

SoftPatch: 基于噪声数据的无监督异常检测

Xi Jiang¹

jiangx2020@mail.sustech.edu.cn

Jianlin Liu²

jenningsliu@tencent.com

Jinbao Wang^{1*}

wangjb@sustech.edu.cn

Qian Nie²

stephennie@tencent.com

Kai Wu²

lloydwu@tencent.com

Yong Liu²

choasliu@tencent.com

Chengjie Wang²

jasoncjwang@tencent.com

Feng Zheng^{1*}

zhengf@sustech.edu.cn

¹南方科技大学, 计算机科学与工程系 ²腾讯
, 优图实验室

摘要

尽管主流的无监督异常检测 (AD) 算法在学术数据集上表现良好, 但由于其训练数据纯净的理想实验设定, 在实际应用中的性能受到限制。在现实世界的异常检测中, 使用含噪声数据进行训练是一个不可避免的问题, 但相关讨论却很少。本文首次考虑了图像感知异常检测中的标签级噪声问题。为解决此问题, 我们提出了一种基于记忆的无监督AD方法——SoftPatch, 该方法能在图像块级别高效去噪。在构建核心集之前, 我们利用噪声判别器生成异常值分数以进行块级噪声消除。这些分数随后被存储于记忆库中, 以软化异常检测边界。与现有方法相比, SoftPatch保持了对正常数据的强大建模能力, 并缓解了核心集中的过度自信问题。在多种噪声场景下的综合实验表明, SoftPatch在MVTecAD和BTAD基准测试中优于当前最先进的AD方法, 并在无噪声设定下与这些方法性能相当。

1 引言

仅凭无标注的正常图像检测异常是一个极具吸引力的课题, 尤其在工业应用中, 缺陷可能极其微小且难以收集。为解决此问题, 人们提出了无监督感官异常检测 (亦称协变量偏移检测[1; 2]), 并已得到广泛探索。近期的深度学习方法[3; 4; 5; 6; 7]通常将异常检测问题建模为单类别学习问题, 并借助计算机视觉技巧来提升感知能力——这些方法通过提供的洁净正常训练集来提取代表性特征。大多数先前的无监督异常检测方法需测量测试样本与标准数据集分布之间的距离, 以判断样本是否偏离标准数据集。尽管现有方法已取得优异性能, 但它们都依赖于洁净训练集进行特征提取。

*Corresponding Author

用于后续与异常特征进行比较的名义特征。对训练数据过于信任可能导致陷阱。如果标准正常数据集被噪声数据（即有缺陷的样本）污染，估计的边界将不可靠，对异常数据的分类准确率也会较低。总的来说，当前的无监督异常检测方法并非为噪声数据设计，且对其缺乏鲁棒性。

然而，在实际应用中，标准正常数据集中不可避免地会混入噪声，尤其是在工业制造领域，每日生产的大量产品更是如此。这些噪声通常源于固有的数据偏移或人为误判。同时，现有的无监督异常检测方法[8; 9; 10]因其对训练集进行详尽建模的策略，容易受到噪声数据的影响。如图1所示，噪声样本极易误导那些过度自信的异常检测算法，导致算法错误分类测试集中的类似异常样本，并产生错误定位。此外，针对含噪声数据的异常检测可发展为完全无监督的设置，相比以往无监督异常检测中默认训练集全部无缺陷的隐含监督信号，这种设置摒弃了该假设。这有助于拓展更多工业质检场景，例如无需数据过滤即可快速部署到新的生产线。

本文首先指出了研究异常检测（AD）中，尤其是在无监督感官AD中噪声数据问题的重要性。我们的解决方案受到近期先进方法PatchCore [8]的启发。PatchCore提出了一种方法，通过最近邻搜索对标准正常数据集的原始CNN特征进行子采样，并建立一个较小的核心集作为记忆库。然而，核心集的选择和分类过程容易受到污染数据的影响。为此，我们提出了一种基于图像块级别的选择策略，以剔除噪声样本中的噪声图像块。与传统的样本级去噪相比，该方法能够分离异常图像块，并利用噪声样本中的正常图像块构建核心集。具体而言，去噪算法为每个待选入核心集的图像块分配一个离群因子。基于图像块级别的去噪，我们提出了一种具有更强噪声鲁棒性的新型AD算法，命名为SoftPatch。考虑到噪声样本难以完全去除，SoftPatch利用离群因子对核心集样本进行重新加权。实验证明，图像块级别的去噪和核心集样本的重新加权能有效修正知识错位，并缓解推理过程中核心集的过度自信。在多种噪声场景下的广泛实验表明，SoftPatch在MVTec异常检测（MVTecAD）[11]基准测试中优于当前最先进的AD方法。同时，由于现有数据集中存在噪声，SoftPatch在原始BTAD [12]数据集上取得了最优结果。代码可在<https://github.com/TencentYoutuResearch/AnomalyDetection-SoftPatch>找到。

我们的主要贡献总结如下：

- 据我们所知，我们是首个专注于含噪声数据的图像传感异常检测的研究，这是一个更实际但鲜有探索的场景。现有的图像传感异常检测方法完全依赖训练集的洁净度，导致其在噪声干扰下性能下降。
- 我们提出了一种针对核心集记忆库的补丁级去噪策略，相比传统的样本级去噪，该策略本质上提高了数据利用率。基于此策略，我们应用了三个噪声判别器，通过结合核心集的重加权机制来增强模型的鲁棒性。
- 我们为含噪声数据的无监督异常检测设定了一个基线，该基线在额外噪声数据设置和无噪声的通用设置中均表现良好，为后续研究提供了新的视角。

2 相关工作

2.1 无监督异常检测

使用代理任务进行训练。当缺乏异常类别和形状信息时，代理任务（亦称自监督学习）是一种可行的解决方案。Sheynin等人[13]在多尺度生成模型后采用水平翻转、平移、旋转和灰度变换等转换操作来增强表征学习。Li等人[14]指出直接应用现有自监督任务对检测局部缺陷并非最优方案，并提出名为CutPaste的新型代理任务，该方法通过裁剪标准图像区块并粘贴来模拟异常样本。

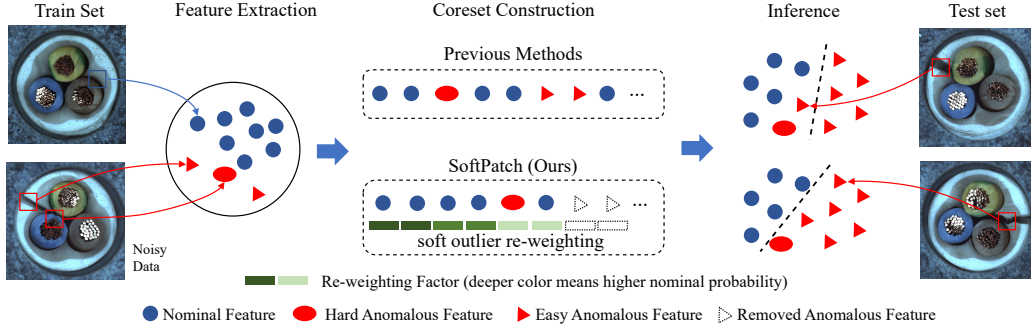


图1: SoftPatch示意图。与以往构建核心集时未考虑噪声数据负面影响的方法不同, SoftPatch通过清除易处理的噪声数据构建干净训练集, 并通过软重加权减轻难处理噪声数据的影响。

将其放回随机位置。类似地, DRAEM [15] 通过Perlin噪声合成异常。然而, 合成异常与真实异常之间不可避免的差异会干扰模型的判断标准, 并限制其泛化性能。异常之间的差距通常大于异常与正常之间的差距。这就是为什么被某些噪声样本欺骗的异常检测方法在处理其他类型的异常时仍能表现良好的原因。

不可知方法。包括知识蒸馏和图像重建, 不可知方法基于一个理论: 从未见过异常样本的模型在推理时, 对正常样本和异常样本的输入会表现出不同的行为。知识蒸馏在异常检测中被巧妙运用。Bergmann等人[16]提出, 异常补丁的表征在预训练的教师模型和学生模型之间存在差异, 学生模型需使用无异常训练集尽力模拟教师输出。基于此理论, Salehi等人[17]指出, 在蒸馏过程中考虑多个中间输出并使用更小的学生网络能获得更好效果。反向蒸馏[18]采用逆向流机制, 避免了相同滤波器造成的混淆, 并阻止异常扰动传播至结构类似于重建网络的学生模型。图像重建方法[7; 19; 20]利用以下假设: 在正常数据集上训练的重建网络无法重构异常部分。通过比较重建图像与原始图像的差异, 可获得高分辨率检测结果。然而, 所有不可知方法均需要较长的训练阶段, 这限制了其应用场景, 例如完全无监督学习中要求的快速部署假设。

特征建模。我们特指对提取器输出特征进行直接建模的方法, 包括分布估计[21; 22]、分布变换[23; 9]、预训练模型适配[24; 25]以及记忆存储[26; 8]。PaDiM[21]利用多元高斯分布来估计正常数据的补丁嵌入。在推理阶段, 不规则补丁的嵌入将处于分布之外。这是一种简单而高效的方法, 但高斯分布不足以应对更复杂的数据情况。因此, 为增强密度估计, DifferNet[23]和CFLOW[9]采用了基于多尺度表示的可逆归一化流。Hou等人[26]提出, 特征图上的划分粒度与模型对正常和异常样本的重建能力密切相关。因此, 他们将一个多尺度分块记忆库嵌入到自编码器网络中, 作为对过去数据的建模。PatchCore[8]是一种更显式但具有价值的基于记忆的方法, 它将下采样后的补丁特征存储在记忆库中, 并计算测试特征与核心集之间的最近邻距离作为异常分数。尽管PatchCore在典型设置中表现出色, 但它对训练集过于自信, 这导致了较差的噪声鲁棒性。

2.2 含噪声数据学习

噪声标签识别正成为监督学习的一个新兴课题, 但在无监督异常检测中却鲜有探索, 因为缺乏明确的标签。在分类任务中, 部分研究[27; 28]提出通过高置信度阈值过滤噪声伪标签数据。Li等人[29]则采用混合模型筛选噪声标签数据, 并以半监督方式进行训练。

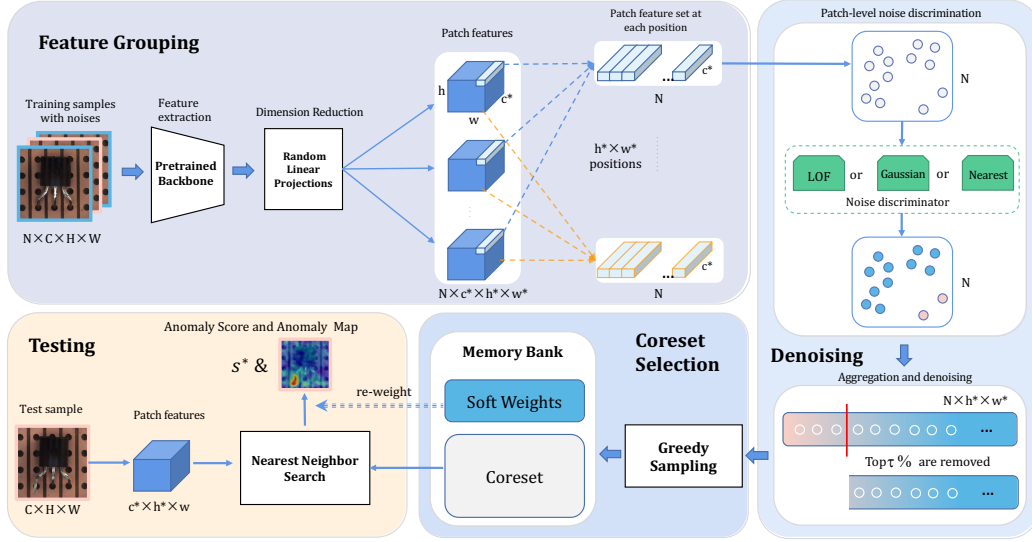


图2: 所提出方法的概览。在训练阶段, 通过噪声判别器在特征图的每个位置上以图像块级别区分噪声。图像块节点的颜色越深, 其为噪声块的概率越高。在为所有图像块获取异常值分数后, 移除异常值分数最高的前 $\{v^*\}$ %图像块。核心集是去噪后剩余图像块的子集。与其他方法不同, 我们的记忆库由核心集中的样本及其异常值分数组成, 这些分数以软权重形式存储。软权重将在推理过程中进一步用于重新加权异常分数。

孔等人[30]对有害训练样本进行了重新标注。在目标检测领域, 多增强[31]、师生网络[32]或对比学习[33]等方法被采用, 借助专家模型的知识来缓解噪声影响。然而, 当前的有噪声标签识别方法均依赖标注数据来协同修正噪声数据。相比之下, 我们的目标是以无监督方式提升模型的噪声鲁棒性, 无需引入人工标注。

尽管无监督异常检测领域已有一些关于模型鲁棒性的研究, 但由于“异常检测”这一术语的多义性, 它们的研究对象和任务与我们的工作存在区别。近期一项综述[34]探讨了30种异常检测算法的模型鲁棒性, 但该研究在标注错误设定中排除了无监督方法。Pang等人[35]利用连续帧中的信息处理视频异常检测, 无需人工标注数据; 而我们的工作则专注于单幅图像的异常检测。其他相关论文[36; 37; 38]致力于消除语义异常检测中的噪声与损坏数据。与语义异常检测不同, 我们聚焦于图像感知异常检测[1]——该领域近期备受关注且包含异常定位这一新任务。尽管现有方法[39; 40; 41; 42]以统一方式处理协变量偏移与语义偏移, 并通过通用流程增强模型鲁棒性, 但其基础架构相较于快速发展的感知异常检测方法仍显薄弱, 导致鲁棒性提升效果有限。图像感知异常检测中的噪声与正常数据更为相似, 这带来了更大的挑战。

3 所提出的方法

3.1 概述

基于补丁的无监督异常方法, 如PatchCore [8]和CFA [25], 包含三个主要流程: 特征提取、核心集选择与记忆库构建, 以及异常检测。其中一个重要假设是训练集仅包含正常图像, 且核心集应全面覆盖整个训练数据的分布。在测试过程中, 输入图像会直接在记忆库中搜索相似特征, 异常分数则基于其与最近邻补丁的差异度计算。若假设不成立, 搜索过程可能会失效。

干净的全覆盖记忆库包含噪声。因此，我们提出了SoftPatch，它在核心集构建前通过噪声判别器过滤噪声数据，并软化搜索过程以降低未过滤的困难噪声样本的权重。

针对样本级标签污染的通用去噪方法在图像感官异常检测中效果欠佳。在图像感官异常检测（以工业缺陷检测和医学图像分析为代表）中，异常通常仅占据图像的极小区域。在样本层面，噪声数据难以区分，但样本的内在偏差可能更为显著。因此，我们提出一种在特征空间工作的块级去噪策略，以更准确判断噪声块。首先，根据位置对采集的特征进行分组，以缩小噪声判别范围。随后，在核心集采样前插入块级噪声判别流程，该流程依据每个位置的特征分布生成噪声评分。由于含噪图像的大部分区域通常无异常，我们移除噪声块并保留其余部分以最大化数据利用率。同时，反映聚类行为的剩余去噪评分被用于在推理阶段缩放异常分数。算法的其他部分（如特征提取、降维、核心集采样和最近邻搜索）遵循基线方法PatchCore[8]。图2展示了SoftPatch的框架。

图像级去噪的目标是从 \mathcal{X} 中找到 \mathcal{X}_{noise} ，其中 $\mathcal{X} = \{x_i: i \in [1, N], x_i \in \mathbb{R}^{C \times H \times W}\}$ 表示训练图像（通道数 C ，高度 H ，宽度 W ）。遵循现有工作[8]的惯例，我们使用 $\phi_i \in \mathbb{R}^{c^* \times h^* \times w^*}$ 作为图像 $x_i \in \mathcal{X}$ 的特征图（通道数 c^* ，高度 h^* ，宽度 w^* ）， $\phi_i(h, w) \in \mathbb{R}^{c^*}$ 作为聚合特征图上位于 (h, w) 处、维度为 c 的补丁。

3.2 噪声判别性核心集选择

随着训练图像的增加，特征记忆可能变得极其庞大，难以通过整体统计来区分噪声。因此，我们按位置对所有特征进行分组，并计算它们的异常值分数。然后汇总所有分数以确定噪声块，之后我们仅移除得分最高的前 τ 百分位特征。我们总共应用了三种降噪方法。

3.2.1 最近邻

假设噪声样本 X_{noise} 的数量远少于干净样本 $X_{nominal}$ ，我们将最近邻距离设为基线[43]，其中距离较大表示异常值。给定一组图像， $\phi \in \mathbb{R}^{N \times c^* \times h^* \times w^*}$ 代表所有特征。每个图像块的最近邻距离 \mathcal{W}_i^{nn} 定义为：

$$\mathcal{W}_i^{nn}(h, w) = \min_{n \in [1, N], n \neq i} \|\phi_i(h, w) - \phi_n(h, w)\|_2, \quad (1)$$

我们首先计算距离，然后在批次维度（邻居）中取最小值作为 \mathcal{W}^{nn} 。先前的方法[8; 25]已证明，来自预训练网络的最小特征距离可作为区分异常的指标。该方法能识别明显的异常值，但受不同簇分布不均的影响，其中某些簇可能具有较大的内部距离，导致被错误地视为噪声数据。为了平等对待所有簇，我们提出了另一种多变量高斯方法，用于计算异常值分数，避免不同簇密度差异的干扰。

3.2.2 多元高斯分布

在高斯分布的正则化作用下，所有干净图像的特征可以被平等对待。为了动态地将高斯分布应用于图像特征，我们为每个图像块 $\phi_i(h, w)$ 在批次维度上计算内点概率，类似于3.2.1节的做法。多元高斯分布 $\mathcal{N}(\mu_{h,w}, \Sigma_{h,w})$ 可以表示为： $\mu_{h,w}$ 是 $\phi_i(h, w)$ 的批次均值，样本协方差 $\Sigma_{h,w}$ 的计算公式为：

$$\Sigma_{h,w} = \frac{1}{N-1} \sum_{n=1}^N (\phi_n(h, w) - \mu_{h,w})(\phi_n(h, w) - \mu_{h,w})^T + \epsilon I, \quad (2)$$

其中正则化项 ϵI 使 $\Sigma_{h,w}$ 满秩且可逆[21]。最终，通过估计的多变量高斯分布 $\mathcal{N}(\mu_{h,w}, \Sigma_{h,w})$ 。需注意，我们并未在计算时排除 x_i

计算 \mathcal{N} ，因此所有样本可以共享一个联合分布，这比逐个进行计算的计算量要小得多。马哈拉诺比斯距离被计算为每个补丁的噪声幅度 $\mathcal{W}_i^{avg}(h, w)$ ：

$$\mathcal{W}_i^{avg}(h, w) = \sqrt{(\phi_i(h, w) - \mu_{(h, w)})^T \Sigma_{h, w}^{-1} (\phi_i(h, w) - \mu_{(h, w)})}. \quad (3)$$

高马氏距离意味着高异常值分数。尽管高斯分布能够归一化并捕捉图像特征的本质，但小的特征簇可能会被大的特征簇所淹没。在批次中存在一个显著特征簇和一个小簇的情况下，小簇可能会超出计算出的 $\mathcal{N}(\mu_{h, w}, \Sigma_{h, w})$ 的1-、2-或3- Σ 范围，从而被错误地分类为异常值。在分析了上述两种方法后，我们需要一种能够满足以下条件的方法：1. 平等对待所有图像特征；2. 平等对待大簇和小簇；3. 适用于高维计算。

3.2.3 局部离群因子 (LOF)

LOF[44]是一种基于局部密度的离群点检测器，主要用于电子商务中的犯罪活动检测。受LOF启发，我们可以解决3.2.2节中提到的三个问题：1. 计算每个簇的相对密度可以归一化不同密度的簇；2. 使用局部k-距离作为度量以缓解大簇的压倒性影响；3. 将距离建模为归一化特征距离可应用于高维补丁特征。因此，首先计算基于k-距离的绝对局部可达密度 $lrd_i(h, w)$ 为：

$$lrd_i(h, w) = 1 / \left(\frac{\sum_{b \in \mathcal{N}_k(\phi_i(h, w))} dist_k^{reach}(\phi_i(h, w), \phi_b(h, w))}{|\mathcal{N}_k(\phi_i(h, w))|} \right), \quad (4)$$

$$dist_k^{reach}(\phi_i(h, w), \phi_b(h, w)) = \max(dist_k(\phi_b(h, w)), d(\phi_i(h, w), \phi_b(h, w))), \quad (5)$$

其中 $d(\phi_i(h, w), \phi_b(h, w))$ 是L2范数， $dist_k(\phi_i(h, w))$ 是第k个邻居的距离， $\mathcal{N}_k(\phi_i(h, w))$ 是 $\phi_i(h, w)$ 的k个最近邻集合， $|\mathcal{N}_k(\phi_i(h, w))|$ 是集合的数量，在没有重复邻居时通常等于k。通过每个图像块的局部可达密度，大簇的压倒性影响被大幅降低。为了将局部密度归一化为相对密度以平等对待所有簇，图像i的相对密度 \mathcal{W}_i^{LOF} 定义如下：

$$\mathcal{W}_i^{LOF}(h, w) = \frac{\sum_{b \in \mathcal{N}_k(\phi_i(h, w))} lrd_b(h, w)}{|\mathcal{N}_k(\phi_i(h, w))| \cdot lrd_i(h, w)}. \quad (6)$$

$\mathcal{W}_i^{LOF}(h, w)$ 是邻域相对于补丁自身的相对密度，并表示为补丁的内点置信度。我们的实验发现，上述三种降噪方法在构建核心集前的数据预选中均有帮助，而LOF提供了最佳性能。然而，在可视化清理后的训练集后，我们发现与正常样本相似的困难噪声样本仍隐藏在数据集中。为了进一步减轻噪声数据的影响，我们提出了一种软重加权方法，可根据异常值得分降低噪声样本的权重。

3.3 基于SoftPatch的异常检测

除了构建核心集外，所有选定图像块的异常因子均以软权重形式存储于记忆库中。借助如图2所示的去噪后图像块级记忆库 \mathcal{M} ，可通过图像块级最近邻搜索为测试样本 $x_i \in \mathcal{X}^{test}$ 计算图像级异常分数 $s \in \mathbb{R}$ 。将测试样本的图像块特征集合记为 $\mathcal{P}(x_i)$ ，对于每个图像块 $p_{h, w} \in \mathcal{P}_{x_i}$ ，其最近邻搜索可通过以下公式表示：

$$m^* = \arg \min_{m \in \mathcal{M}} \|p - m\|_2. \quad (7)$$

在最近邻搜索之后，测试图像块与其在 \mathcal{M} 中的最近邻可以配对为 (p, m^*) 。对于每个图像块 $p_{i, j} \in \mathcal{P}_{x_i}$ ，块级异常分数通过以下公式计算：

$$s_{h, w} = \mathcal{W}_{m^*} \|p_{h, w} - m^*\|_2, \quad (8)$$

其中 \mathcal{W}_{m^*} 是由一个噪声判别器计算出的软权重。图像级异常分数通过寻找最大的软权重重新加权的补丁级异常分数获得： $s^* = \max_{(h, w)} s_{h, w}$

与直接平等考虑各补丁的PatchCore不同，SoftPatch通过噪声判别器根据噪声水平软化异常分数。软权重，即局部离群因子，已经考虑了最近节点周围的局部关系。因此，可以实现与PatchCore类似的效果，但具有更强的噪声鲁棒性和更少的搜索次数。根据图像级异常分数，样本被分类为正常样本或异常样本。

4 实验

4.1 实验细节

数据集。我们的实验主要在MVTecAD和BTAD基准测试[11; 12]上进行。MVTecAD包含15个类别，总计3629张训练图像和1725张测试图像；BTAD包含3个类别，共1799张图像。不同类别的工业生产场景构成了全面挑战，例如物体或纹理类型以及是否存在旋转。由于MVTecAD的每个类别都划分为仅包含正常图像的训练集和同时包含正常与异常样本的测试集，为创建带噪声的训练集，我们从测试集中随机抽取异常图像，并将其与原有训练图像混合。请注意，与无噪声情况相比，训练集中原始正常样本数量保持不变。在此设置(*No overlap*)下，注入的异常样本将不参与评估，这更符合实际应用场景。我们还构建了另一种设置(*Overlap*)，其中注入的异常样本也包含在测试集中，以证明外观相似的缺陷会严重恶化基于噪声数据训练的异常检测器性能的风险。同时，重叠样本也用于测试我们算法的离群点检测性能。通过控制注入训练集的负样本比例，我们获得了多个具有不同噪声比例的新数据集，命名为MVTecAD-noise- n ，其中 n 指代噪声比例。对于BTAD，我们直接使用原始数据划分。

表1：在带噪声的MVTecAD数据集上的异常检测性能。结果在MVTecAD-noise-0.1上进行评估。*Overlap*表示注入的异常图像包含在测试集中。PaDiM*使用ResNet18作为骨干网络。PatchCore-random使用1%随机子采样器替代默认的贪婪子采样器。*Gap*行展示了噪声场景与正常场景之间的性能差距。

| Noise=0.1 | No overlap | | | | | | Overlap | | | |
|------------|--------------|--------------|--------------|-------------------|--------------------|---------------|--------------|-----------|------------------|---------------|
| Category | PaDiM | CFLOW | PatchCore | SoftPatch-nearest | SoftPatch-gaussian | SoftPatch-lof | PaDiM* | PatchCore | PatchCore-random | SoftPatch-lof |
| bottle | 0.994 | 0.998 | 1.000 | 1.000 | 0.997 | 0.937 | 1.000 | 0.692 | 0.998 | 1.000 |
| cable | 0.873 | 0.925 | 0.982 | 0.935 | 0.952 | 0.995 | 0.680 | 0.756 | 0.920 | 0.994 |
| capsule | 0.920 | 0.947 | 0.976 | 0.916 | 0.662 | 0.963 | 0.796 | 0.783 | 0.779 | 0.955 |
| carpet | 0.999 | 0.961 | 0.996 | 0.995 | 0.999 | 0.991 | 0.890 | 0.681 | 0.973 | 0.993 |
| grid | 0.966 | 0.891 | 0.971 | 0.972 | 0.997 | 0.968 | 0.674 | 0.526 | 0.793 | 0.969 |
| hazelnut | 0.956 | 1.000 | 0.998 | 1.000 | 1.000 | 1.000 | 0.543 | 0.441 | 0.998 | 1.000 |
| leather | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 0.964 | 0.739 | 1.000 | 1.000 |
| metal_nut | 0.987 | 0.959 | 0.999 | 0.994 | 0.997 | 0.999 | 0.820 | 0.765 | 0.969 | 1.000 |
| pill | 0.918 | 0.929 | 0.975 | 0.921 | 0.873 | 0.963 | 0.722 | 0.770 | 0.874 | 0.955 |
| screw | 0.838 | 0.784 | 0.966 | 0.862 | 0.475 | 0.960 | 0.567 | 0.710 | 0.462 | 0.923 |
| tile | 0.977 | 0.991 | 0.985 | 0.996 | 0.997 | 0.993 | 0.830 | 0.716 | 1.000 | 0.981 |
| toothbrush | 0.927 | 0.906 | 0.997 | 1.000 | 0.997 | 0.997 | 0.700 | 0.800 | 0.797 | 0.994 |
| transistor | 0.953 | 0.896 | 0.953 | 1.000 | 0.992 | 0.990 | 0.471 | 0.491 | 0.943 | 0.999 |
| wood | 0.991 | 0.972 | 0.984 | 0.984 | 0.997 | 0.987 | 0.831 | 0.579 | 0.980 | 0.986 |
| zipper | 0.852 | 0.928 | 0.981 | 0.976 | 0.979 | 0.978 | 0.679 | 0.792 | 0.950 | 0.974 |
| Average | 0.943 | 0.939 | 0.984 | 0.970 | 0.927 | 0.986 | 0.740 | 0.683 | 0.896 | 0.982 |
| Gap | -0.007 | -0.03 | -0.008 | +0.002 | -0.001 | 0.0 | -0.151 | -0.309 | -0.015 | -0.004 |

评估指标。我们在MVTecAD中为每个类别报告图像级和像素级的AUROC，并取平均值得到平均图像/像素级AUROC。为了表示噪声鲁棒性，还展示了无噪声数据与含噪声数据之间的性能差距。除非另有说明，我们的方法SoftPatch指的是第3.2.3节中使用LOF的SoftPatch-LOF。

实现细节。我们在噪声场景中测试了三种最先进的异常检测算法：PatchCore [8]、PaDim [21] 和 CFLOW [9]，并遵循其主要设置。在无特殊说明的情况下，特征提取器的主干网络为 *Wide-ResNet50*，PatchCore 和 SoftPatch 的核心集采样比例为 10%。对于 MVTecAD 图像，我们仅使用 256×256 分辨率并进行中心裁剪。

将它们调整为 224×224 并进行归一化处理。对于BTAD数据集，我们使用 512×512 的分辨率。我们为每个类别训练一个独立的模型。请注意，与许多根据噪声比例（这在现实中是不可知的）设置超参数的方法不同，我们将SoftPatch中的阈值 τ 和 $LOF-K$ 在所有噪声场景和类别中分别固定为0.15和6。超参数的影响在消融实验中进行研究。我们所有的实验均在Nvidia V100 GPU上运行，并重复三次以报告平均结果。

4.2 含噪声条件下的异常检测性能

表2：在带噪声的MVTecAD数据集上的异常定位性能。结果基于MVTecAD-noise-0.1进行评估。

| Noise=0.1 | No overlap | | | | | | Overlap | | | |
|-----------|------------|--------|-----------|-------------------|--------------------|---------------|---------|-----------|------------------|---------------|
| Category | PaDiM | CFLOW | PatchCore | SoftPatch-nearest | SoftPatch-gaussian | SoftPatch-lof | PaDiM* | PatchCore | PatchCore-random | SoftPatch-lof |
| Average | 0.972 | 0.969 | 0.956 | 0.971 | 0.977 | 0.979 | 0.955 | 0.654 | 0.951 | 0.969 |
| Gap | -0.007 | -0.006 | -0.025 | -0.008 | -0.001 | -0.002 | -0.013 | -0.327 | -0.021 | -0.012 |

在MVTecAD上的实验。如表1和表2所示，当向训练集中添加10%的异常样本以破坏数据时，所有现有方法均出现不同程度的性能下降，尽管在*No overlap*设置下并未造成灾难性影响。与其他方法相比，无论使用哪种噪声判别器，所提出的SoftPatch在异常检测和定位方面都表现出更强的抗噪声数据鲁棒性。在SoftPatch的三种变体中，SoftPatch-lof实现了最佳的综合性能，具有最高的准确性和最强的鲁棒性。有趣的是，PaDiM[21]、CFLOW[9]和SoftPatch-gaussian的性能下降幅度明显小于PatchCore，这表明将特征建模为高斯分布确实有助于去噪。然而，将每个空间位置的特征分布建模为单一高斯分布无法处理未对齐的图像，例如MVTecAD中的*screw*类别，这解释了这些类别上较差的表现（参见螺丝类别的数据行）。另一方面，PatchCore的贪婪采样策略是一把双刃剑，既能提高特征空间覆盖率，也对噪声更加敏感。这就是为什么在PatchCore中使用随机采样虽然性能有所妥协，但更具鲁棒性的原因（参见PatchCore 1%-随机列）。SoftPatch-nearest在未对齐情况下表现略好，但由于未考虑特征分布，导致其整体性能较差。

表3：BTAD上无额外噪声的异常检测性能。最佳结果以粗体显示，次佳结果以下划线标示。最后一列列出了测试集中异常样本的数量。

| Category | SPADE | P-SVDD | PatchCore | PaDiM | SoftPatch(ours) | Anomaly samples |
|----------|--------------|--------|--------------|--------------|-----------------|-----------------|
| 01 | 0.914 | 0.957 | 1.000 | 1.000 | <u>0.999</u> | 50 |
| 02 | 0.714 | 0.721 | <u>0.871</u> | <u>0.871</u> | 0.934 | 200 |
| 03 | 0.999 | 0.821 | 0.999 | 0.971 | <u>0.997</u> | 41 |
| Mean | 0.876 | 0.833 | <u>0.957</u> | 0.947 | 0.977 | - |

在BTAD上的实验。我们还在另一个数据集BTAD上将SoftPatch与其他SOTA方法进行了比较。令人惊讶的是，即使在原始无额外噪声的设置下，SoftPatch也取得了新的SOTA结果（表3）。通过检查所有训练样本，我们发现类别BTAD-02的训练集中已存在许多噪声样本（通常为细小划痕），这更符合我们的设定，并进一步证明了我们方法的必要性。噪声图像详见附录A.6（表8）。此外，BTAD-02包含更多具有相似外观异常的异常样本。在BTAD-02类别中，我们的方法相比其他方法取得了显著提升。即使人为添加噪声，SoftPatch仍能保持领先性能（附录A.8）。

性能趋势。为了探究不同方法如何随着噪声水平的增加而变化，我们在MVTecAD-noise-{0 ~ 0.15}上进行了进一步的实验。所提出方法的结果如图3所示。在*No overlap*设置下，随着噪声比例的增加，PatchCore的像素级AUROC下降高达3.7%。性能随着噪声比例上升而下降。相反，尽管默认性能略逊于PatchCore（约0.006

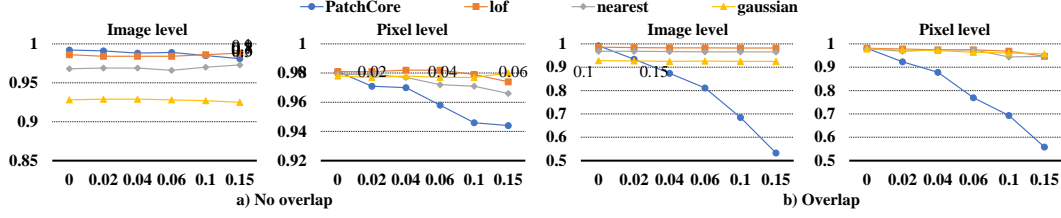


图3：噪声训练下异常检测性能的对比。no overlap表示注入的异常图像已从测试集中移除，而overlap则未移除。

并且图像级和像素级AUROC均无下降），所提出的SoftPatch-lof性能下降速度要慢得多，这证明了其更好的去噪能力。至于SoftPatch-nearest和SoftPatch-gaussian，它们也更具鲁棒性，但基础性能较差（参见噪声比例=0时的图3）。图5中核心集的可视化结果同样表明，随机采样虽能避免采样异常值，却无法充分建模正常样本。与上述讨论一致，在Overlap设置下，随着噪声增加，PatchCore的性能出现灾难性下滑（图像与像素级AUROC降幅最高达40%）。这一结果符合预期，因为PatchCore采用贪心策略进行核心集采样，会偏向特征空间中的异常值。SoftPatch-lof始终优于其他方法，且随着噪声水平上升未出现显著性能下降。附录A.3（图6和图7）提供了更多与其他方法的对比。实验结果表明，由于MVTecAD数据集中缺陷外观差异极大，这一风险被掩盖了。在这种情况下，即使有异常特征被误加入核心集，它们在测试时也不太可能被检索到。然而风险依然存在，当测试中出现类似缺陷时将被触发。

更多实验可见于附录，例如图像级与块级去噪的比较（附录A.4）、计算分析（附录A.5）以及增强重叠设置（附录A.7）。

表4：软权重的消融研究。在MVTecAD上的性能得分为Image/pixel-level AUROC。

| | | No overlap | | Overlap | |
|---------------------|-------------|--------------|--------------|--------------|--------------|
| Noise discriminator | Soft weight | Image level | Pixel level | Image level | Pixel level |
| None | | 0.985 | 0.946 | 0.685 | 0.693 |
| Gaussian | | 0.927 | 0.977 | 0.925 | 0.961 |
| Gaussian | ✓ | 0.922 | 0.974 | 0.924 | 0.965 |
| Nearest | | 0.970 | 0.971 | 0.966 | 0.944 |
| Nearest | ✓ | 0.972 | 0.978 | 0.968 | 0.958 |
| LOF | | 0.985 | 0.984 | 0.984 | 0.963 |
| LOF | ✓ | 0.986 | 0.979 | 0.982 | 0.969 |

4.3 消融研究

4.3.1 所提出模块的有效性

我们通过从流程中移除两个提出的模块——噪声判别器和软权重，验证了它们的有效性。如表4所示，噪声判别器在像素级AUROC方面显著提升了噪声鲁棒性。在噪声判别器的三种决策选择中，LOF在鲁棒性和容量之间取得了最佳平衡，从而在所有设置下实现了最大的性能提升。我们通过可视化不同方法的采样核心集进一步分析了中间结果，这表明SoftPatch-LOF采样的异常特征远少于基线（见图5）。软权重与噪声判别器结合使用，以进一步提升最终结果。在SoftPatch-Nearest中使用软权重时，我们仅观察到微小的改进。我们推测其他两种噪声判别器已经对噪声数据具有较好的鲁棒性。

Table 5: Image/pixel-level AUROC result for different $LOF-K$ on two settings.

| K | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|------------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|-------------|
| Overlap | 0.983 /0.955 | 0.982/0.951 | 0.983 /0.959 | 0.982/ 0.975 | 0.981/0.973 | 0.982/0.968 | 0.980/0.968 |
| No overlap | 0.985 /0.972 | 0.985 /0.975 | 0.984/0.977 | 0.984/0.982 | 0.985 /0.980 | 0.984/ 0.983 | 0.981/0.982 |

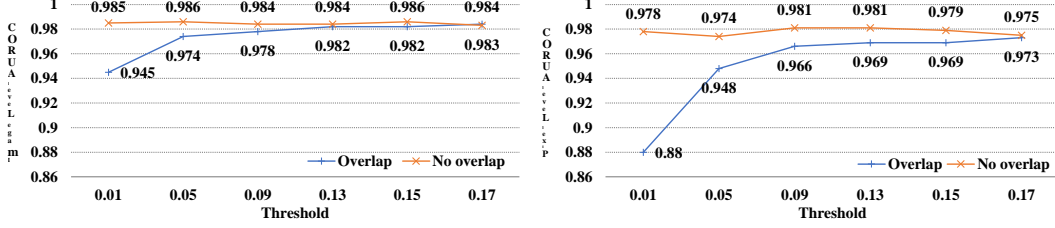


图4: SoftPatch-LOF中阈值 τ 的性能趋势。结果在MVTecAD-noise-0.1数据集上进行评估。

4.3.2 参数选择

为了探究两个参数（ $LOF-k$ 和阈值 τ ）对最终性能的影响，我们在我们的方法上进行了参数搜索。如表5所示，当 $LOF-k$ 大于5时，我们的方法取得了更好的性能，这表明只要 $LOF-k$ 不过小或过大，我们的方法对其不敏感。如果 $LOF-k$ 过小，则因考虑的邻近点过少而无法准确估计局部密度；相反，过大的 $LOF-k$ 可能导致不理想的跨聚类连接，从而无法捕捉真实的数据分布。

阈值 τ 指的是构建核心集时被剔除的补丁特征比例。图4显示在 *Overlap* 设置下，AUROC 随阈值 τ 增大呈上升趋势，这符合预期，因为更高的阈值意味着更激进的去噪策略。在 *Overlap* 设置中，误采样的特征是性能急剧下降的直接原因，因此更激进的去噪能显著改善结果。然而在 *No Overlap* 设置下，噪声特征的影响相对不明显。尽管最佳 $LOF-k$ 和阈值 τ 会随类别和噪声水平变化，我们在所有场景中均简单采用固定值 6 和 0.15。

5 结论

本文强调了研究无监督异常检测中噪声数据问题的实际价值。通过在先前任务中引入一种新颖的噪声设置，我们测试了现有方法及SoftPatch的性能。对于现有方法，尽管未针对噪声设置进行适配，其中部分方法在某些场景中仅出现轻微的性能下降。然而，对于其他方法或在其他场景中，性能下降可能更为显著甚至具有破坏性。就所提出的SoftPatch而言，它在所有噪声设置中均表现出稳定的性能，优于其他方法。

工业检测系统是计算机视觉的重要应用，需要具备良好的鲁棒性。注入训练集的噪声打破了训练样本均为正常的朴素假设，也让模型提前接触到异常数据的分布。未来需要更多关于含噪声数据的无监督异常检测研究。

致谢

本工作得到国家自然科学基金项目（批准号：61972188、62122035和62206122）的资助。

参考文献

- [1] 杨景康, 周恺阳, 李逸轩, 刘子纬. 广义分布外检测: 综述. *arXiv preprint arXiv:2110.11334*, 2021.

- [2] 陶贤, 龚新意, 张鑫, 闫少华, Chandranath Adak. 工业图像无监督异常定位的深度学习研究综述. *IEEE Transactions on Instrumentation and Measurement*, 2022。
- [3] 杨明辉, 吴鹏, 刘静, 冯辉. Memseg: 一种利用差异性与共性进行图像表面缺陷检测的半监督方法. *arXiv preprint arXiv:2205.00908*, 2022。
- [4] Vitjan Zavrtanik, Matej Kristan, Danijel Skocaj. DSR - 一种用于表面异常检测的双子空间重投影网络。 *CoRR*, abs/2208.01521, 2022。
- [5] Choubo Ding, Guansong Pang, and Chunhua Shen. 捕捉灰天鹅与黑天鹅: 开放集监督异常检测。发表于 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 第7388–7398页, 2022年。
- [6] Nicolae-Cătălin Ristea, Neelu Madan, Radu Tudor Ionescu, Kamal Nasrollahi, Fahad Shahbaz Khan, Thomas B Moeslund, Mubarak Shah. 用于异常检测的自监督预测卷积注意力块。收录于 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 第13576–13586页, 2022年。
- [7] 严旭东, 张淮东, 徐雪森, 胡晓伟, 衡鹏安. 从正常样本中学习语义上下文用于无监督异常检测。于 *Proceedings of the AAAI Conference on Artificial Intelligence*, 第35卷, 第3110–3118页, 2021年。
- [8] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox 和 Peter Gehler. 迈向工业异常检测的完全召回。 *arXiv preprint arXiv:2106.08265*, 2021年。
- [9] Denis Gudovskiy, Shun Ishizaka 和 Kazuki Kozuka. Cflow-ad: 通过条件归一化流进行定位的实时无监督异常检测。发表于 *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 第98–107页, 2022年。
- [10] 叶铮, 王翔, 邓睿, 包天鹏, 赵瑞, 吴立伟. 聚焦你的分布: 用于异常检测与定位的从粗到细非对比学习。于 *2022 IEEE International Conference on Multimedia and Expo (ICME)*, 第1–6页. IEEE, 2022。
- [11] Paul Bergmann, Michael Fauser, David Sattlegger 和 Carsten Steger. Mvtec AD——一个用于无监督异常检测的综合性真实世界数据集。收录于 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 页码 9592–9600, 2019年。
- [12] Pankaj Mishra, Riccardo Verk, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. VT-ADL: 一种用于图像异常检测与定位的视觉Transformer网络。发表于 *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, 第01–06页. IEEE, 2021。
- [13] Shelly Sheynin, Sagie Benaim 和 Lior Wolf. 一种用于少样本异常检测的层次化变换判别生成模型。发表于 *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 第8495–8504页, 2021年10月。
- [14] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon 和 Tomas Pfister. CutPaste: 用于异常检测与定位的自监督学习。收录于 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 第9664–9674页, 2021年。
- [15] Vitjan Zavrtanik, Matej Kristan 和 Danijel Skočaj. Draem——一种用于表面异常检测的判别性训练重建嵌入方法。收录于 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 第8330–8339页, 2021年。
- [16] Paul Bergmann, Michael Fauser, David Sattlegger 和 Carsten Steger. 无监督学生: 基于判别性潜在嵌入的学生-教师异常检测。发表于 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 第4183–4192页, 2020年。
- [17] Mohammadreza Salehi, Niousha Sadjadi, Soroosh Baselizadeh, Mohammad H Rohban 和 Hamid R Rabi-ee. 用于异常检测的多分辨率知识蒸馏。载于 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 第14902–14912页, 2021年。
- [18] 邓寒秋, 李星宇. 基于单类嵌入反向蒸馏的异常检测. *arXiv preprint arXiv:2201.10703*, 2022。
- [19] 康周, 肖宇霆, 杨建龙, 程俊, 刘文, 罗伟新, 顾再旺, 刘江, 高盛华. 利用P-Net编码结构-纹理关系用于视网膜图像异常检测。于 *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, 第360–377页. Springer, 2020。

- [20] 梁宇飞, 张江宁, 赵士玮, 吴润泽, 刘勇, 潘书文。面向无监督异常检测的全频段通道选择表征。 *arXiv preprint arXiv:2203.00259*, 2022。
- [21] Thomas Defard, Aleksandr Setkov, Angelique Loesch 与 Romaric Audigier。Padim: 一种用于异常检测与定位的补丁分布建模框架。收录于 *International Conference on Pattern Recognition*, 第475–489页。Springer, 2021。
- [22] Jihun Yi 和 Sungroh Yoon。Patch SVDD: 用于异常检测与分割的块级SVDD。发表于 *Proceedings of the Asian Conference on Computer Vision*, 2020年。
- [23] Marco Rudolph, Bastian Wandt 与 Bodo Rosenhahn。相似却不同: 基于标准化流的半监督缺陷检测。收录于 *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 第1907–1916页, 2021年。
- [24] Tal Reiss, Niv Cohen, Liron Bergman 和 Yedid Hoshen。Panda: 为异常检测与分割适配预训练特征。于 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 第2806–2814页, 2021年。
- [25] Sungwook Lee, Seunghyun Lee 和 Byung Cheol Song。面向目标异常定位的基于耦合超球体的特征自适应方法。 *arXiv preprint arXiv:2206.04325*, 2022。
- [26] 侯金磊, 张莹莹, 钟乔勇, 谢迪, 濮世亮, 周虹。分而治之: 学习块级记忆用于无监督异常检测。于 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 第8791–8800页, 2021年。
- [27] Zijian Hu, Zhengyu Yang, Xuefeng Hu, and Ram Nevatia。SIMPLE: 用于半监督分类的简单伪标签利用。载于 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 第15099–15108页, 2021年6月。
- [28] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, 以及 Chun-Liang Li。Fixmatch: 通过一致性和置信度简化半监督学习。 *Advances in Neural Information Processing Systems*, 33:596–608, 2020。
- [29] Junnan Li, Richard Socher 与 Steven C. Hoi。Dividemix: 将带噪标签学习视为半监督学习。 *arXiv preprint arXiv:2002.07394*, 2020。
- [30] Shuming Kong, Yanyan Shen, and Linpeng Huang。通过基于影响力的数据重标注解决训练偏差问题。发表于 *International Conference on Learning Representations*, 2021年。
- [31] 徐梦德、张政、胡涵、王建峰、王丽娟、韦方云、白翔、刘子成。基于软教师的端到端半监督目标检测。发表于 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 第3060–3069页, 2021年。
- [32] Yen-Cheng Liu, Chih-Yao Ma, Zijian He, Chia-Wen Kuo, Kan Chen, Peizhao Zhang, Bichen Wu, Zsolt Kira, and Peter Vajda。半监督目标检测的无偏教师。 *arXiv preprint arXiv:2102.09480*, 2021。
- [33] 杨帆、吴凯、张书一、蒋冠男、刘勇、郑锋、张伟、王成杰、曾龙。类感知对比半监督学习。 *arXiv preprint arXiv:2203.02261*, 2022。
- [34] 宋乔, 胡西阳, 黄海亮, 蒋明琪, 赵越。Adbench: 异常检测基准测试。 *arXiv preprint arXiv:2206.09426*, 2022。
- [35] 庞冠松、严成、沈春华、Anton van den Hengel 与白雪。基于自训练深度序数回归的端到端视频异常检测。载于 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 第12173–12182页, 2020年。
- [36] Boyang Liu, Ding Wang, Kaixiang Lin, Pang-Ning Tan, and Jiayu Zhou。RCA: 一种用于异常检测的深度协作自编码器方法。发表于 *IJCAI*, 第1505–1511页, 2021年。
- [37] 周冲和Randy C Paffenroth。基于鲁棒深度自编码器的异常检测。发表于 *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 665–674页, 2017年。
- [38] 吴双、赵靖宇和田广健。理解并缓解深度异常检测中的数据污染: 一种基于核函数的方法。载于 *IJCAI*, 第2319–2325页, 2022年。

[39] Jinsung Yoon, Kihyuk Sohn, Chun-Liang Li, Serkan O Arik, Chen-Yu Lee, 与 Tomas Pfister。自监督、精炼、重复：改进无监督异常检测。2021年。[40] Chen Qiu, Aodong Li, Marius Kloft, Maja Rudolph, 与 Stephan Mandt。针对含噪声数据的异常检测的潜在异常暴露。 *arXiv preprint arXiv:2202.08088*, 2022年。[41] Antoine Cordier, Benjamin Missaoui, 与 Pierre Gutierrez。使用预训练网络进行全无监督视觉检测的数据精炼。 *arXiv preprint arXiv:2202.12759*, 2022年。[42] Yuanhong Chen, Yu Tian, Guansong Pang, 与 Gustavo Carneiro。通过插值高斯描述符进行深度单类分类。收录于 *Proceedings of the AAAI Conference on Artificial Intelligence*, 第36卷, 第383–392页, 2022年。[43] Leif E Peterson。K-最近邻。 *Scholarpedia*, 4(2):1883, 2009年。[44] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, 与 Jörg Sander。LOF：识别基于密度的局部异常值。收录于 *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 第93–104页, 2000年。

附录

A.1 核心集可视化

图5展示了内存库中核心集的可视化效果。PatchCore保留了过多噪声特征，这些特征明显属于异常值。尽管通过随机采样替代了贪婪采样，PatchCore避免了大部分噪声特征，但在模型训练集上表现欠佳，且仍受部分噪声干扰。SoftPatch的核心集则既洁净又分布分散。我们的核心集保留了部分来自噪声样本的特征，因为我们相信异常图像同样包含大量正常图像块。因此，符合正态分布的特征被保留下来，以增强模型的感知能力。

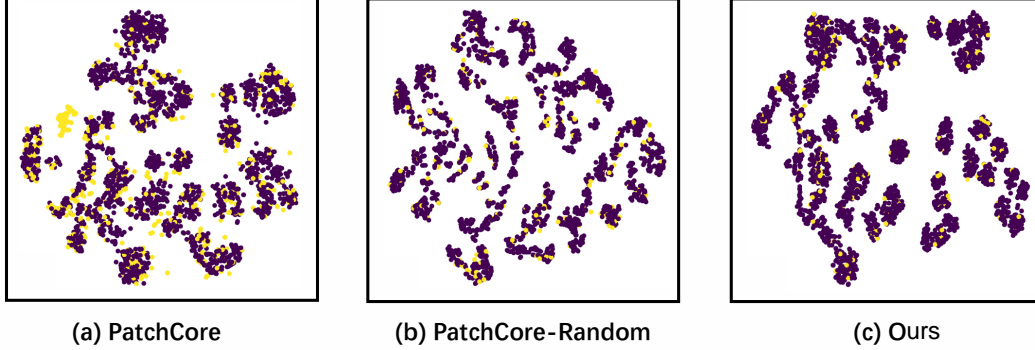


图5：相同噪声训练集下各AD方法的核心对比，MVTecAD-Pill数据集噪声等级0.1。我们使用t-SNE进行降维可视化。黄色圆点代表噪声样本的局部特征，紫色圆点表示正常样本。与其他两种方法相比，SoftPatch清除了噪声局部特征并正确建模了正常数据。

A.2 实验结果详情

表6：所有类别的异常定位性能详情。结果在MVTecAD-noise-0.1数据集上进行评估。

| Noise=0.1 | No overlap | | | | | | Overlap | | | |
|------------|------------|--------|-----------|-------------------|--------------------|---------------|---------|-----------|------------------|---------------|
| Category | PaDiM | CFLOW | PatchCore | SoftPatch-nearest | SoftPatch-gaussian | SoftPatch-lof | PaDiM* | PatchCore | PatchCore-random | SoftPatch-lof |
| bottle | 0.986 | 0.984 | 0.987 | 0.987 | 0.986 | 0.987 | 0.981 | 0.714 | 0.979 | 0.975 |
| cable | 0.916 | 0.958 | 0.843 | 0.915 | 0.981 | 0.983 | 0.946 | 0.670 | 0.969 | 0.971 |
| capsule | 0.986 | 0.985 | 0.986 | 0.988 | 0.977 | 0.990 | 0.984 | 0.883 | 0.984 | 0.989 |
| carpet | 0.992 | 0.989 | 0.992 | 0.992 | 0.993 | 0.992 | 0.980 | 0.765 | 0.951 | 0.989 |
| grid | 0.974 | 0.947 | 0.991 | 0.990 | 0.989 | 0.990 | 0.879 | 0.482 | 0.882 | 0.974 |
| hazelnut | 0.987 | 0.991 | 0.990 | 0.990 | 0.991 | 0.990 | 0.978 | 0.418 | 0.957 | 0.924 |
| leather | 0.994 | 0.994 | 0.991 | 0.994 | 0.994 | 0.993 | 0.992 | 0.683 | 0.987 | 0.993 |
| metal_nut | 0.933 | 0.956 | 0.842 | 0.894 | 0.964 | 0.984 | 0.911 | 0.779 | 0.938 | 0.983 |
| pill | 0.956 | 0.983 | 0.971 | 0.974 | 0.972 | 0.981 | 0.960 | 0.608 | 0.971 | 0.976 |
| screw | 0.989 | 0.977 | 0.995 | 0.991 | 0.969 | 0.994 | 0.974 | 0.745 | 0.953 | 0.969 |
| tile | 0.956 | 0.953 | 0.953 | 0.960 | 0.962 | 0.954 | 0.921 | 0.700 | 0.919 | 0.954 |
| toothbrush | 0.991 | 0.988 | 0.989 | 0.988 | 0.988 | 0.985 | 0.954 | 0.692 | 0.984 | 0.985 |
| transistor | 0.960 | 0.887 | 0.847 | 0.965 | 0.954 | 0.942 | 0.939 | 0.317 | 0.914 | 0.936 |
| wood | 0.973 | 0.964 | 0.969 | 0.947 | 0.946 | 0.939 | 0.946 | 0.522 | 0.896 | 0.929 |
| zipper | 0.986 | 0.978 | 0.986 | 0.989 | 0.988 | 0.988 | 0.978 | 0.823 | 0.975 | 0.986 |
| Average | 0.972 | 0.969 | 0.956 | 0.971 | 0.977 | 0.979 | 0.955 | 0.654 | 0.951 | 0.969 |
| Gap | -0.007 | -0.006 | -0.025 | -0.008 | -0.001 | -0.002 | -0.013 | -0.327 | -0.021 | -0.012 |

A.3 噪声中的性能趋势

图6和图7展示了SOTA AD方法和SoftPatch在不同噪声场景下的性能趋势。由于对训练数据的过度自信以及贪婪子采样算法，PatchCore的性能随噪声增加下降最为明显。相比之下，CFLOW和PaDiM

同样受到噪声影响，但幅度较小。SoftPatch在所有噪声水平下均保持稳定的性能表现。遗憾的是，在无噪声场景中SoftPatch略弱于PatchCore，这可能源于过于保守的阈值设置。

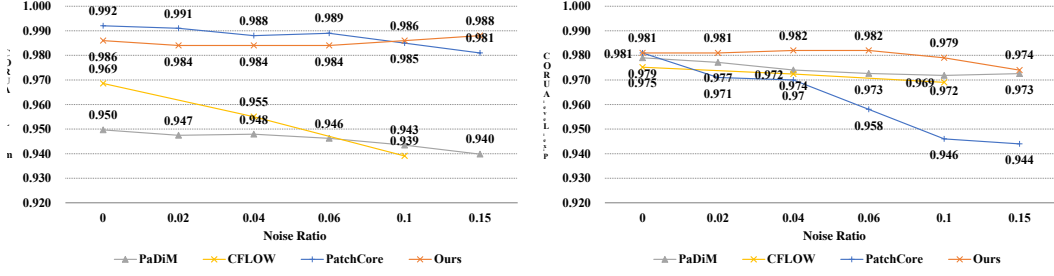


图6: 不同无重叠噪声水平下的性能。

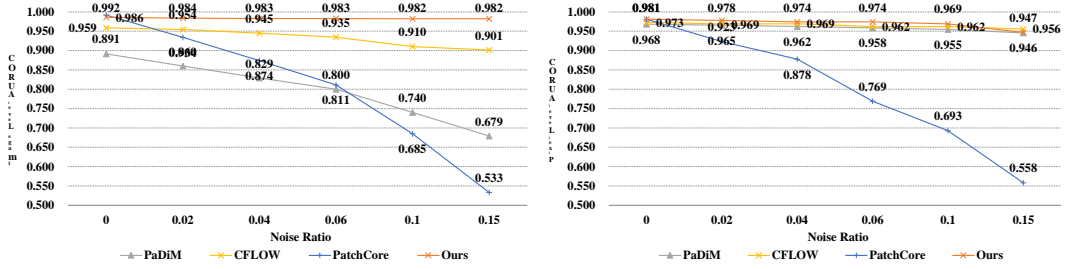


图7: 不同重叠噪声水平下的性能。

A.4 图像级去噪与补丁级去噪

一种消除噪声数据的简单策略是在训练前删除异常样本，这是一个无监督的离群点检测任务。然而，现有离群点检测方法效果不佳，因为异常图像与正常图像之间的距离远小于不同类别图像之间的距离。同时，我们发现某些异常检测方法也能识别训练集中的离群点。借鉴[41]的思路，我们采用PaDiM*作为图像级去噪方法，该方法兼顾了成本与效果。PaDiM*是PaDiM的简化版本，使用ResNet18作为主干网络以提升计算速度。PaDiM*对所有训练样本进行评分，随后根据阈值剔除高离群值的样本。表7和表8的对比表明，图像级去噪显著提升了现有SOTA异常检测方法在噪声场景下的性能，但与SoftPatch相比仍存在差距。

A.5 计算分析

根据理论分析，SoftPatch所需的运行时间并不比PatchCore多。PatchCore中贪婪采样过程的复杂度为 $\mathcal{O}(N^2 h^2 w^2)$ ，这是最耗时的部分。而SoftPatch-LOF中噪声判别过程的复杂度为 $\mathcal{O}(N^2 h w)$ ，因为特征已预先分组。因此通过 $\mathcal{O}(N^2 h w + N^2 h^2 w^2) = \mathcal{O}(N^2 h^2 w^2)$ ，SoftPatch的计算复杂度与PatchCore相当。实际上，SoftPatch会更快，因为它将部分补丁作为噪声移除。

排除数据加载时间，SoftPatch与PatchCore在其余时间开销上的对比如图9所示。本实验使用的GPU为RTX TITAN 24G。两者在训练和测试阶段花费的时间几乎相同，这表明我们的块级去噪方法并未带来不可接受的开销。相反，图像级去噪会显著增加训练时间。

表7: 图像级去噪与块级去噪的异常检测性能。PaDiM*+PaDiM*、PaDiM*+CFLOW和PaDiM*+PatchCore是采用图像级去噪的异常检测方法。PaDiM*应用于图像级去噪时, 我们采用与SoftPatch相同的阈值 (0.15) 作为噪声比例。我们也尝试了更严格的阈值0.1, 但效果更差。所有结果均在MVTecAD-noise-0.1数据集上采用重叠评估方式得出。

| Category | PaDiM* | PaDiM*+ PaDiM* | CFLOW | PaDiM*+ CFLOW | PatchCore | PaDiM*(threshold -0.1)+PatchCore | PaDiM*+ PatchCore | SoftPatch-lof |
|------------|--------|-------------------|--------------|------------------|-----------|-------------------------------------|----------------------|---------------|
| bottle | 0.937 | 0.994 | 1.000 | 1.000 | 0.692 | 0.984 | 1.000 | 1.000 |
| cable | 0.680 | 0.741 | 0.916 | 0.841 | 0.756 | 0.890 | 0.888 | 0.994 |
| capsule | 0.796 | 0.854 | 0.945 | 0.939 | 0.783 | 0.892 | 0.909 | 0.955 |
| carpet | 0.890 | 0.937 | 0.960 | 0.950 | 0.681 | 0.963 | 0.974 | 0.993 |
| grid | 0.674 | 0.765 | 0.799 | 0.830 | 0.526 | 0.850 | 0.870 | 0.969 |
| hazelnut | 0.543 | 0.725 | 0.999 | 0.990 | 0.441 | 0.871 | 0.929 | 1.000 |
| leather | 0.964 | 0.979 | 0.996 | 1.000 | 0.739 | 0.957 | 0.989 | 1.000 |
| metal_nut | 0.820 | 0.949 | 0.957 | 0.986 | 0.765 | 0.965 | 0.977 | 1.000 |
| pill | 0.722 | 0.745 | 0.897 | 0.924 | 0.770 | 0.898 | 0.913 | 0.955 |
| screw | 0.567 | 0.542 | 0.570 | 0.639 | 0.710 | 0.916 | 0.907 | 0.923 |
| tile | 0.830 | 0.906 | 0.980 | 0.981 | 0.716 | 0.939 | 0.957 | 0.981 |
| toothbrush | 0.700 | 0.869 | 0.878 | 0.928 | 0.800 | 0.981 | 0.997 | 0.994 |
| transistor | 0.471 | 0.770 | 0.872 | 0.788 | 0.491 | 0.777 | 0.825 | 0.999 |
| wood | 0.831 | 0.966 | 0.954 | 0.970 | 0.579 | 0.943 | 0.976 | 0.986 |
| zipper | 0.679 | 0.678 | 0.931 | 0.873 | 0.792 | 0.909 | 0.914 | 0.974 |
| Average | 0.740 | 0.828 | 0.910 | 0.909 | 0.683 | 0.916 | 0.935 | 0.982 |

表8: 图像级去噪与块级去噪的异常定位性能。

| Category | PaDiM* | PaDiM*+ PaDiM* | CFLOW | PaDiM*+ CFLOW | PatchCore | PaDiM*(threshold -0.1)+PatchCore | PaDiM*+ PatchCore | SoftPatch-lof |
|------------|--------------|-------------------|-------|------------------|-----------|-------------------------------------|----------------------|---------------|
| bottle | 0.981 | 0.983 | 0.984 | 0.986 | 0.714 | 0.984 | 0.985 | 0.975 |
| cable | 0.946 | 0.954 | 0.950 | 0.956 | 0.670 | 0.738 | 0.739 | 0.971 |
| capsule | 0.984 | 0.982 | 0.986 | 0.985 | 0.883 | 0.851 | 0.876 | 0.989 |
| carpet | 0.980 | 0.984 | 0.986 | 0.988 | 0.765 | 0.960 | 0.988 | 0.989 |
| grid | 0.879 | 0.876 | 0.961 | 0.948 | 0.482 | 0.797 | 0.818 | 0.974 |
| hazelnut | 0.978 | 0.977 | 0.982 | 0.987 | 0.418 | 0.798 | 0.825 | 0.924 |
| leather | 0.992 | 0.993 | 0.993 | 0.995 | 0.683 | 0.966 | 0.979 | 0.993 |
| metal_nut | 0.911 | 0.968 | 0.960 | 0.984 | 0.779 | 0.784 | 0.834 | 0.983 |
| pill | 0.960 | 0.956 | 0.976 | 0.984 | 0.608 | 0.706 | 0.713 | 0.976 |
| screw | 0.974 | 0.968 | 0.973 | 0.970 | 0.745 | 0.887 | 0.889 | 0.969 |
| tile | 0.921 | 0.927 | 0.945 | 0.946 | 0.700 | 0.924 | 0.968 | 0.954 |
| toothbrush | 0.954 | 0.986 | 0.984 | 0.983 | 0.692 | 0.977 | 0.986 | 0.985 |
| transistor | 0.939 | 0.965 | 0.834 | 0.908 | 0.317 | 0.932 | 0.945 | 0.936 |
| wood | 0.946 | 0.947 | 0.934 | 0.943 | 0.522 | 0.800 | 0.918 | 0.929 |
| zipper | 0.978 | 0.973 | 0.979 | 0.967 | 0.823 | 0.875 | 0.878 | 0.986 |
| Average | 0.955 | 0.963 | 0.962 | 0.969 | 0.654 | 0.865 | 0.889 | 0.969 |

A.6 现有数据集中的噪声

尽管现有研究数据集组织良好, 但部分异常样本仍存在误分类情况。图8展示了两个广泛使用的数据集中正常集合内的异常样本。在实际生产数据中, 噪声干扰将更为严重。

A.7 增强重叠设置下的性能

我们进行了另一项实验, 在训练集中对重叠图像进行增强, 使其与测试集中的图像有所不同。我们尝试了不同程度的外观变化和

表9: MVTecAD上各类别的平均训练和推理时间。时间单位为秒。

| | Training time | Inference time |
|------------------|---------------|----------------|
| SoftPatch-LOF | 21.2958 | 15.6146 |
| PatchCore | 21.3869 | 15.8763 |
| PaDiM*+PatchCore | 74.2912 | 15.5386 |

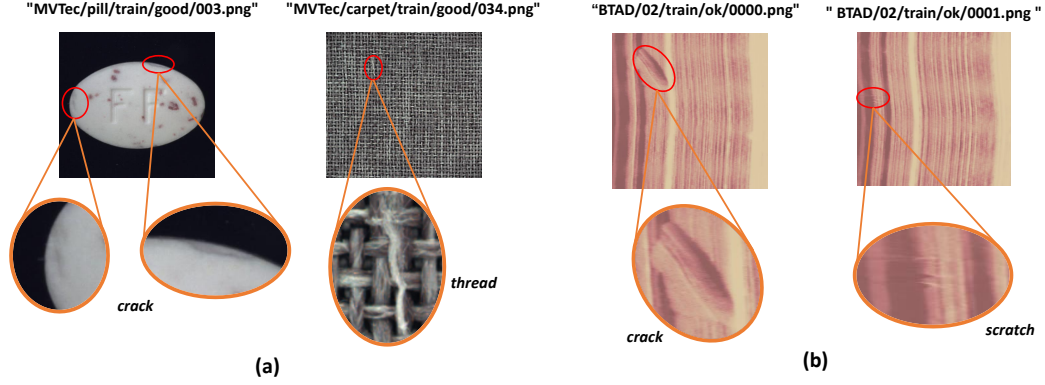


图 8: (a) MVTecAD 数据集与 (b) BTAD 数据集中的噪声示例。

结构增强。表10的结果表明，尽管PatchCore的性能有所提升，但当重叠样本经过变换时，我们的方法仍展现出更好的鲁棒性。

表10: 在增强 $\{v^*\}$ 设置下MVTecAD的性能表现。

| Setting | Overlap with gaussian noise | Overlap with noise and blur | Overlap with rotation | Overlap with affine transformation |
|--------------|-----------------------------|-----------------------------|-----------------------|------------------------------------|
| Method | PatchCore / Ours | PatchCore / Ours | PatchCore / Ours | PatchCore / Ours |
| Detection | 0.760 / 0.984 | 0.848 / 0.984 | 0.950 / 0.984 | 0.933 / 0.984 |
| Localization | 0.790 / 0.969 | 0.864 / 0.970 | 0.924 / 0.978 | 0.915 / 0.978 |

A.8 在带噪声的BTAD上的性能

性能比较见表11和12。由于BTAD-03类别的异常样本数量不足以满足噪声样本的要求，我们尝试了另外两种方法。

表11: BTAD-noise-0.1上的异常检测性能。

| Noise = 0.1 | No overlap | | Overlap | |
|-------------|--------------|---------------|-----------|---------------|
| Category | PatchCore | SoftPatch-LOF | PatchCore | SoftPatch-LOF |
| 01 | 1.000 | 1.000 | 0.522 | 1.000 |
| 02 | 0.860 | 0.922 | 0.738 | 0.912 |
| Mean | 0.930 | 0.961 | 0.630 | 0.956 |

表12: BTAD-noise-0.1上的异常定位性能。

| Noise = 0.1 | No overlap | | Overlap | |
|-------------|------------|---------------|-----------|---------------|
| Category | PatchCore | SoftPatch-LOF | PatchCore | SoftPatch-LOF |
| 01 | 0.982 | 0.999 | 0.319 | 0.815 |
| 02 | 0.949 | 0.953 | 0.754 | 0.936 |
| Mean | 0.966 | 0.976 | 0.536 | 0.875 |