Hierarchical Gaussian Mixture Normalizing Flow Modeling for Unified Anomaly Detection

Xincheng Yao[1], Ruoqi Li[1], Zefeng Qian[1], Lu Wang[3], and Chongyang Zhang[1,2]*

[1] School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University
[2] MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
[3] School of Intelligent Manufacturing, Wuxi Vocational College of Science and Technology
{i-Dover, nilponi, zefeng_qian, sunny_zhang}@sjtu.edu.cn[1], wlulu0327@163.com[3]

Abstract. Unified anomaly detection (AD) is one of the most valuable challenges for anomaly detection, where one unified model is trained with normal samples from multiple classes with the objective to detect anomalies in these classes. For such a challenging task, popular normalizing flow (NF) based AD methods may fall into a "homogeneous mapping" issue, where the NF-based AD models are biased to generate similar latent representations for both normal and abnormal features, and thereby lead to a high missing rate of anomalies. In this paper, we propose a novel **H**ierarchical **G**aussian mixture normalizing flow modeling method for accomplishing unified **A**nomaly **D**etection, which we call HGAD. Our HGAD consists of two key components: inter-class Gaussian mixture modeling and intra-class mixed class centers learning. Compared to the previous NF-based AD methods, the hierarchical Gaussian mixture modeling approach can bring stronger representation capability to the latent space of normalizing flows. In this way, we can avoid mapping different class distributions into the same single Gaussian prior, thus effectively avoiding or mitigating the "homogeneous mapping" issue. We further indicate that the more distinguishable different class centers, the more conducive to avoiding the bias issue. Thus, we further propose a mutual information maximization loss for better structuring the latent feature space. We evaluate our method on four real-world AD benchmarks, where we can significantly improve the previous NF-based AD methods and also outperform the SOTA unified AD methods. The code will be available at `https://github.com/xcyao00/HGAD`.

1 Introduction

Anomaly detection has received increasingly wide attentions and applications in different scenarios, such as industrial defect detection [4, 10, 20, 32, 35, 36], video surveillance [1, 26], medical lesion detection [28, 39], and road anomaly

* Corresponding Author.

detection [7, 30]. Considering the highly scarce anomalies and diverse normal classes, most previous AD studies have mainly devoted to unsupervised one-class learning, *i.e.*, learning one specific AD model by only utilizing one-class normal samples and then detecting anomalies in this class[4]. However, such a one-for-one paradigm would require more human labor, time, and computation costs when training and testing on many product categories, and also underperform when the one normal class has large intra-class diversity.

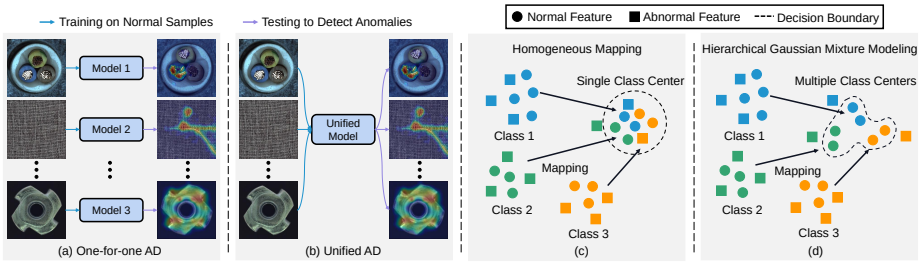

Fig. 1: Anomaly detection task settings. We aim to implement one unified AD model (b). (c) Mapping all input features to the same latent class center may induce the "homogeneous mapping" issue. (d) We propose a hierarchical Gaussian mixture modeling method for more effectively capturing the complex multi-class distribution.

In this work, we aim to tackle a more practical task: unified anomaly detection. As shown in Fig. 1b, one unified model is trained with normal samples from multiple classes, and the objective is to detect anomalies for all these classes without any fine-tuning. Nonetheless, solving such a task is quite challenging. Currently, there are two reconstruction-based AD methods for tackling the challenging unified AD task, UniAD [35] and PMAD [34]. But the reconstruction-based methods may fall into the "identical shortcut reconstruction" dilemma [35], where anomalies can also be well reconstructed, resulting in the failure of anomaly detection. UniAD and PMAD attempt to mask the adjacent or suspicious anomalies to avoid identical reconstruction. However, due to the diverse scale and shape of anomalies, the masking mechanism cannot completely avoid the abnormal information leakage during reconstruction, the risk of identical reconstruction is still existing. To this end, we consider designing the unified AD model from the normal data distribution learning perspective. The advantage is that we will no longer face the abnormal information leakage risk in principle, as our method is based on normal data distribution and radically avoids reconstruction. Specifically, we employ normalizing flows (NF) to learn the normal data distribution [12].

However, we find that the NF-based AD methods perform unsatisfactorily when applied to the unified AD task. They usually fall into a "homogeneous mapping" issue (see Sec. 3.2), where the NF-based AD models are biased to generate large log-likelihoods for both normal and abnormal inputs (see Fig. 2b).

[4] Class means the category of the object in the image, for industrial AD, it refers to the industrial product category, e.g., in Fig. 1, cable, carpet, etc.

We explain this issue as: the NF-based AD methods [12,36] employ the uni-modal Gaussian prior to learn the multi-class distribution (multi-modal). This can be seen as learning a mapping from a heterogeneous space to the latent homogeneous space. To learn the mapping well, the network may be prompted to take a bias to concentrate on the coarse-grained common characteristics (*e.g.*, local pixel correlations) and suppress the fine-grained distinguishable characteristics (*e.g.*, semantic content) among different class features [14]. Consequently, the network homogeneously maps different class features to the close latent embeddings. Thus, even anomalies can obtain large log-likelihoods and become less distinguishable.

To address this issue, we first empirically confirm that mapping to multi-modal latent distribution is effective to prevent the model from learning the bias (see Fig. 2c). Accordingly, we propose an inter-class Gaussian mixture modeling approach for NF-based AD networks to more effectively capture the complex multi-class distribution. Second, we argue that the inter-class Gaussian mixture modeling can only ensure the features are drawn to the whole distribution but lacks inter-class repulsion, still resulting in a much weaker discriminative ability for different class features. This may cause different class centers to collapse into the same center. To further increase the inter-class discriminability, we propose a mutual information maximization loss to introduce the class repulsion property to the model for better structuring the latent feature space, where the class centers can be pushed away from each other. Third, we introduce an intra-class mixed class centers learning strategy that can urge the model to learn diverse normal patterns even within one class. Finally, we form a hierarchical Gaussian mixture normalizing flow modeling method for unified anomaly detection, which we call HGAD. In summary, we make the following main contributions:

1. We propose a novel unified AD method: HGAD, where the hierarchical Gaussian mixture modeling approach can bring stronger representation capability to the latent space of normalizing flows for accomplishing the unified AD task.

2. We specifically propose three key designs: inter-class Gaussian mixture modeling, mutual information maximization loss, and intra-class mixed class centers learning strategy.

3. Under the unified AD task, our method can dramatically improve the unified AD performance of the previous one-for-one NF-based AD methods (*e.g*, CFLOW-AD), boosting the AUROC from 89.0%/94.0% to 98.4%/97.9%, and also outperform the SOTA unified AD methods (*e.g.*, UniAD).

2 Related Work

Anomaly Detection. 1) *Reconstruction-based approaches* are the most popular AD methods. These methods rely on the assumption that models trained by normal samples would fail in abnormal image regions. Many previous works attempt to train AutoEncoders [18, 37], Variational AutoEncoders [16] and GANs [2, 25] to reconstruct the input images. However, these methods face the "identical shortcut" problem [35]. 2) *Embedding-based approaches* recently show better AD performance by using ImageNet pre-trained networks as feature

extractors [3, 8]. PaDiM [9] extract pre-trained features to model Multivariate Gaussian distribution for normal samples, then utilize Mahalanobis distance to measure the anomaly scores. PatchCore [20] extends on this line by utilizing locally aggregated features and introducing greedy coreset subsampling to form nominal feature banks. 3) *Knowledge distillation* assumes that the student trained to learn the teacher on normal samples could only regress normal features but fail in abnormal features [5]. Recent works mainly focus on feature pyramid [24, 31], reverse distillation [10], and asymmetric distillation [23]. 4) *Unified AD approaches* attempt to train a unified AD model to accomplish anomaly detection for multiple classes. UniAD [35], PMAD [34] and OmniAL [41] are three existing methods in this new direction. UniAD is a transformer-based reconstruction model with three improvements, it can perform well under the unified case by addressing the "identical shortcut" issue. PMAD is a MAE-based patch-level reconstruction model, which can learn a contextual inference relationship within one image rather than the class-dependent reconstruction mode. OmniAL is a unified CNN framework with anomaly synthesis, reconstruction and localization improvements.

GMM in Anomaly Detection. Previous methods, DAGMM [38] and PEDENet [40], also utilize GMM for anomaly detection. However, our method is significantly different from these methods as follows: 1) Different Task. DAGMM is used for non-image data, and PEDENet is used for one-for-one image anomaly detection. Although they both use GMM, it's nontrivial to use them for unified anomaly detection. However, our method can achieve superiority in the unified AD task. 2) Different Approach. DAGMM and PEDENet both predict membership and then use the predicted membership and GMM modeling formulas to directly calculate the mixture component weights and the mean and variance of each mixture component. Our method is based on the learnable class centers and class weights. The way of ours is more beneficial for the hierarchical Gaussian mixture modeling while the way in DAGMM and PEDENet is difficult to achieve this.

Normalizing Flows in Anomaly Detection. In anomaly detection, normalizing flows are employed to learn the normal data distribution [12,15,21,33,36], which maximize the log-likelihoods of normal samples during training. Rudolph *et al.* [21] first employ NFs for anomaly detection by estimating the distribution of pre-trained features. In CFLOW-AD [12], the authors further construct NFs on multi-scale feature maps to achieve anomaly localization. Recently, fully convolutional normalizing flows [22, 36] have been proposed to improve the accuracy and efficiency of anomaly detection. In BGAD [33], the authors propose a NF-based AD model to tackle the supervised AD task. In this paper, we mainly propose a novel NF-based AD model (HGAD) with three improvements to achieve much better unified AD performance.

3 Method

3.1 Preliminary of Normalizing Flow Based Anomaly Detection

In subsequent sections, upper case letters denote random variables (RVs) (*e.g.*, X) and lower case letters denote their instances (*e.g.*, x). The probability density

function of a RV is written as $p(X)$, and the probability value for one instance as $p_X(x)$. The normalizing flow models [11, 13] can fit an arbitrary distribution $p(X)$ by a tractable latent base distribution with $p(Z)$ density and a bijective invertible mapping $\varphi : X \in \mathbb{R}^d \rightarrow Z \in \mathbb{R}^d$. Then, according to the change of variable formula [29], the log-likelihood of any $x \in X$ can be estimated as:

$$\log p_\theta(x) = \log p_Z(\varphi_\theta(x)) + \log|\det J| \tag{1}$$

where θ means the learnable model parameters, and we use $p_\theta(x)$ to denote the estimated probability value of feature x by the model φ_θ. The $J = \bigtriangledown_x \varphi_\theta(x)$ is the Jacobian matrix of the bijective transformation ($z = \varphi_\theta(x)$ and $x = \varphi_\theta^{-1}(z)$). The model parameters θ can be optimized by maximizing the log-likelihoods across the training distribution $p(X)$. The loss function is defined as:

$$\mathcal{L}_m = \mathbb{E}_{x \sim p(X)}[-\log p_\theta(x)] \tag{2}$$

In anomaly detection, the latent variables Z for normal features are usually assumed to obey $\mathcal{N}(0, \mathbb{I})$ for simplicity [21]. By replacing $p_Z(z) = (2\pi)^{-\frac{d}{2}} \mathrm{e}^{-\frac{1}{2} z^T z}$, $z = \varphi_\theta(x)$ in Eq. 1, the loss function in Eq. 2 can be written as:

$$\mathcal{L}_m = \mathbb{E}_{x \sim p(X)} \Big[\frac{d}{2} \log(2\pi) + \frac{1}{2} \varphi_\theta(x)^T \varphi_\theta(x) - \log|\det J| \Big] \tag{3}$$

After training, the log-likelihoods of the input features can be exactly estimated by the trained normalizing flow models as $\log p_\theta(x) = -\frac{d}{2}\log(2\pi) - \frac{1}{2}\varphi_\theta(x)^T \varphi_\theta(x) + \log|\det J|$. Next, we can convert log-likelihoods to likelihoods via exponential function: $p_\theta(x) = \mathrm{e}^{\log p_\theta(x)}$. As we maximize log-likelihoods for normal features in Eq. 2, the estimated likelihoods $p_\theta(x)$ can directly represent the normality (a larger value means more normal). Thus, we can convert likelihoods to anomaly scores by $s(x) = 1 - p_\theta(x)$ [33].

3.2 Revisiting Normalizing Flow Based Anomaly Detection Methods

Under the unified AD task, we follow the NF-based anomaly detection paradigm [12, 21] and reproduce the FastFlow [36] to estimate log-likelihoods of the features extracted by a pre-trained backbone. We then convert the estimated log-likelihoods to anomaly scores and evaluate the AUROC metric every 10 epochs. As shown in Fig. 2a, after a period of training, the performance of the model drops severely while the losses continue going extremely small. Accordingly, the overall log-likelihoods become much large. We attribute this phenomenon to the "homogeneous mapping" issue, where the normalizing flows may map all inputs to much close latent variables and then present large log-likelihoods for both normal and abnormal features, thus failing to detect anomalies. This speculation is empirically verified by the visualization results in Fig. 2b (more results in App. Fig. 1), where the normal and abnormal log-likelihoods are highly overlapped. As we explained in Sec. 1, the phenomenon may come from that the model excessively suppresses the fine-grained distinguishable characteristics between

normal and abnormal features. However, as shown in Fig. 2c and 2d, our unified NF-based AD method can more effectively avoid highly overlapped normal and abnormal log-likelihoods, indicating a slighter log-likelihoods bias problem. This encourages us to analyze as follows.

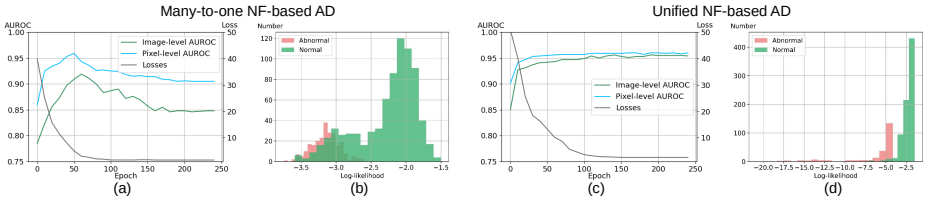

Fig. 2: Comparison between many-to-one and our unified NF-based AD methods on MVTecAD. (a) and (c) show the training losses and the testing anomaly detection and localization AUROCs. (b) shows that the many-to-one NF-based AD model (*i.e.*, training the one-for-one AD method, FastFlow, on multiple classes simultaneously) may have an obvious normal-abnormal overlap under the unified case, while ours (d) can bring better normal-abnormal distinguishability.

Below, we denote normal features as $x_n \in \mathbb{R}^d$ and abnormal features as $x_a \in \mathbb{R}^d$, where d is the channel dimension. We provide a rough analysis using a simple one coupling layer normalizing flow model. When training, the forward affine coupling [11] can be calculated as:

$$x_1, x_2 = \text{split}(x_n)\ ;\ z_1 = x_1, z_2 = x_2 \odot \exp(s(x_1)) + t(x_1)\ ;\ z = \text{cat}(z_1, z_2) \quad (4)$$

where split and cat mean split and concatenate the feature maps along the channel dimension, $s(x_1)$ and $t(x_1)$ are transformation coefficients predicted by a learnable neural network [11]. With the maximum likelihood loss in Eq. 3 pushing all z to fit $\mathcal{N}(0, \mathbb{I})$, the model has no need to distinguish different class features. Thus, it is more likely to take a bias to predict all $s(\cdot)$ to be very small negative numbers ($\rightarrow -\infty$) and $t(\cdot)$ close to zero. The impact is that the model could also fit x_a to $\mathcal{N}(0, \mathbb{I})$ well with the bias, failing to detect anomalies. However, if we map different class features to different class centers, the model is harder to simply take a bias solution. Instead, $s(\cdot)$ and $t(\cdot)$ must be highly related to input features. Considering that $s(\cdot)$ and $t(\cdot)$ in the trained model are relevant to normal features, the model thus could not fit x_a well. We think that the above rough analysis can also be applied to multiple layers. Because the output of one coupling layer will tend to 0 when $s(\cdot)$ and $t(\cdot)$ of the layer are biased. From Eq. 4, we can see that when the output of one coupling layer is close to 0, the output of the next layer will also tend to 0. Therefore, the output after multiple layers will tend to 0, the network is still biased.

3.3 Hierarchical Gaussian Mixture Normalizing Flow Modeling

Overview. As shown in Fig. 3, our HGAD is composed of a feature extractor, a normalizing flow model (details in App. C), and the hierarchical Gaussian

mixture modeling. First, the features extracted by a fixed pre-trained backbone are sent into the normalizing flow model to transform into the latent embeddings. Then, we employ our hierarchical Gaussian mixture modeling method to fit the latent embeddings during training. In Fig. 3, we only show features from one level, but we indeed extract multi-level feature maps and construct the NF model at each level. The pseudo-code of our HGAD is provided in Alg. 1.

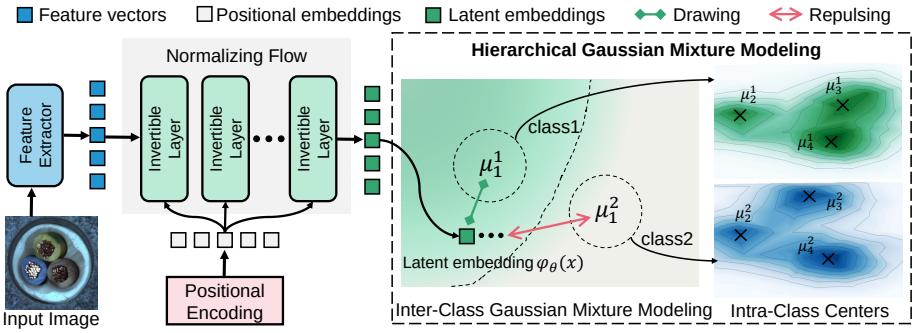

Fig. 3: Model overview. The extracted feature vectors are sent into the normalizing flow model for transforming into latent embeddings. Following [12], we add positional embeddings (*i.e.*, sinusoidal position encoding) to each invertible layer as they are effective for NF-based AD methods. Then, we employ our hierarchical Gaussian mixture modeling approach to fit the latent embeddings, which can assist the model against learning the "homogeneous mapping".

Inter-Class Gaussian Mixture Modeling. As discussed in Sec. 3.2, mapping different class features to different class centers can suffer a slighter log-likelihoods bias problem. To this end, to further better fit the complex multi-class normal distribution in the latent space, we propose the inter-class Gaussian mixture modeling approach. Specifically, a Gaussian mixture model with class-dependent means μ_y and covariance matrices Σ_y (y is used to distinguish different classes), is used as the prior distribution for the latent variables Z:

$$p_{Z|Y}(z|y) = \mathcal{N}(z; \mu_y, \Sigma_y) \quad \text{and} \quad p_Z(z) = \sum\nolimits_y p(y)\mathcal{N}(z; \mu_y, \Sigma_y) \tag{5}$$

For simplicity, we also use unit matrix $\mathbb{I}$ to replace all the class-dependent covariance matrices Σ_y. To urge the network to adaptively learn the class weights, we parameterize the class weights $p(Y)$ through a learnable vector ψ, with $p(y) = \text{softmax}_y(\psi)$, where the subscript y of the softmax operator denotes the class index of the calculated softmax value. The use of the softmax can ensure that $p(y)$ stays positive and sums to one. The ψ can be initialized to 0. With the parameterized $p(Y)$, we can derive the loss function for the inter-class Gaussian mixture modeling as follows (the detailed derivation is in App. E):

$$\mathcal{L}_g = \mathbb{E}_{x\sim p(X)}\left[-\underset{y}{\text{logsumexp}}\left(-\frac{||\varphi_\theta(x)-\mu_y||_2^2}{2}+c_y\right)-\log|\det J|+\frac{d}{2}\log(2\pi)\right] \tag{6}$$

Algorithm 1 HGAD: **H**ierarchical **G**aussian mixture modeling for unified **AD**

Input: Input image $I \in \mathbb{R}^{H\times W\times 3}$, Class $y \in \{1,\dots,Y\}$
Initialization: Class centers $\mu_y^k \leftarrow y + \boldsymbol{r}, \boldsymbol{r} \in \mathbb{R}^{d_k} \sim \mathcal{N}(0,\mathbb{I})$, Learnable vector $\psi^k \leftarrow \mathbf{0} \in \mathbb{R}^Y$, $\psi_y^k \leftarrow \mathbf{0} \in \mathbb{R}^M$; $y \in \{1,\dots,Y\}, k \in \{1,\dots,K\}$
Feature Extraction: K-level feature maps, denoted as $X^k, k \in \{1,\dots,K\}$
for each level $k \in \{1,\dots,K\}$ **do**
  Total loss $\mathcal{L}_{all} \leftarrow 0$
  for each feature $x \in X^k$ **do**
    Obtain the latent representation: $\varphi_\theta(x) \in \mathbb{R}^{d_k}$
    Calculate $c_y = \text{logsoftmax}_y(\psi^k)$ and $c_i^y = \text{logsoftmax}_i(\psi_y^k), i \in \{1,\dots,M\}$
    Calculate **inter-class loss**: $\mathcal{L}_g$ based on Eq. 6
    Calculate **mutual information maximization loss**: $\mathcal{L}_{mi}$ based on Eq. 8
    Calculate **entropy loss**: $\mathcal{L}_e$ based on Eq. 9
    Calculate **intra-class loss**: $\mathcal{L}_{in}$ based on Eq. 10
    Calculate the overall loss: $\mathcal{L} = \lambda_1\mathcal{L}_g + \lambda_2\mathcal{L}_{mi} + \mathcal{L}_e + \mathcal{L}_{in}$
    Update $\mathcal{L}_{all} \leftarrow \mathcal{L}_{all} + \mathcal{L}$
  end for
  Obtain mean loss $\mathcal{L}_{mean}^k = \mathcal{L}_{all}/N$, N is the total number of features
end for
Output: Mean loss $\mathcal{L}_{mean} = \frac{1}{K}\sum_{k=1}^K \mathcal{L}_{mean}^k$

where c_y denotes logarithmic class weights and is defined as $c_y := \log p(y) = \text{logsoftmax}_y(\psi)$, and the subscript y of the logsumexp operator denotes summing the exp values of all classes.

Mutual Information Maximization. Next, we further argue that the inter-class Gaussian mixture modeling can only ensure the latent features are drawn together to the whole distribution (parameterized by $\{\mu_y, \psi_y\}_{y=1}^Y$), where the $\{p(y)\}_{y=1}^Y$ can control the contribution of different class centers to the log-likelihood estimation value $\log p_\theta(x)$. This means that the loss function in Eq. 6 only has the drawing characteristic to make the latent features fit the multi-modal distribution, but without the repulsion property for separating among different classes, still resulting in a much weaker discriminative ability for different class features. As the class centers are randomly initialized, this may cause different class centers to collapse into the same center. To address this, we consider that the latent feature z from class y should be drawn close to its corresponding class center μ_y as much as possible while far away from the other class centers. From the information theory perspective, this means that the mutual information $I(Y,Z)$ should be large enough. So, we propose a mutual information maximization loss to introduce the class repulsion property for increasing the class discrimination ability. The loss function is defined as follows (the derivation is in App. E):

$$\mathcal{L}_{mi} = -\mathbb{E}_{y\sim p(Y)}[-\log p(y)] - \mathbb{E}_{(x,y)\sim p(X,Y)}\left[\log \frac{p(y)p(\varphi_\theta(x)|y)}{\sum_{y'} p(y')p(\varphi_\theta(x)|y')}\right] \quad (7)$$

By replacing $p(\varphi_\theta(x)|y)$ with $\mathcal{N}(\varphi_\theta(x); \mu_y, \mathbb{I})$ in Eq. 7, we can derive the following practical loss format (the detailed derivation is in App. E):

$$\mathcal{L}_{mi} = -\mathbb{E}_{(x,y)\sim p(X,Y)}\left[\underset{y}{\text{logsoftmax}}\left(-\frac{||\varphi_\theta(x)-\mu_{y'}||_2^2}{2}+c_{y'}\right)-c_y\right] \tag{8}$$

where $\mu_{y'}$ means all the other class centers except for μ_y, $c_{y'}$ means all the other logarithmic class weights except for c_y, and the subscript y of the logsoftmax denotes calculating logsoftmax value for class y. Note that we also use this representation way for softmax calculation in the following sections (*e.g.*, Eq. 9).

In addition to the mutual information maximization loss, we propose that we can also introduce the class repulsion property by minimizing the inter-class entropy. We use the $-||\varphi_\theta(x)-\mu_y||_2^2/2$ as the class logits for class y, and then define the entropy loss as follows (a standard entropy formula):

$$\mathcal{L}_e = \mathbb{E}_{x\sim p(X)}\left[\sum_y -\underset{y}{\text{softmax}}(-||\varphi_\theta(x)-\mu_{y'}||_2^2/2)\cdot\underset{y}{\text{logsoftmax}}(-||\varphi_\theta(x)-\mu_{y'}||_2^2/2)\right] \tag{9}$$

Learning Intra-Class Mixed Class Centers. In real-world scenarios, even one object class may contain diverse normal patterns. Thus, to better model intra-class distribution, we further extend the Gaussian prior $p(Z|y)=\mathcal{N}(\mu_y, \Sigma_y)$ to mixture Gaussian prior $p(Z|y)=\sum_{i=1}^{M} p_i(y)\mathcal{N}(\mu_i^y, \Sigma_i^y)$, where M is the number of intra-class latent centers. We can directly replace the $p(Z|y)$ in Eq. 5 and derive the corresponding loss function $\mathcal{L}_g$ in Eq. 6 (see App. Eq. (7)). However, the initial latent features Z usually have large distances with the intra-class centers $\{\mu_i^y\}_{i=1}^M$, this will cause the $p(z|y), z\in Z$ close to 0. After calculating the logarithm function, it is easy to cause the loss to be numerically ill-defined (NaN), making it fail to be optimized. To this end, we propose to decouple the inter-class Gaussian mixture modeling and the intra-class latent centers learning. This decoupling strategy is more conducive to learn class centers as we form a coarse-to-fine optimization process. Specifically, for each class y, we learn a main class center μ_1^y and the delta vectors $\{\Delta\mu_i^y\}_{i=1}^M$ ($\Delta\mu_1^y$ is fixed to 0), which mean the offset values from the main center and are used to represent the other intra-class centers: $\mu_i^y=\{\mu_1^y+\Delta\mu_i^y\}_{i=1}^M$. Then, we can directly employ the Eq. 6 to optimize the main center μ_1^y. When learning the other intra-class centers, we detach the main center μ_1^y from the gradient graph and only optimize the delta vectors by the following loss function:

$$\mathcal{L}_{in} = \mathbb{E}_{(x,y)\sim p(X,Y)}\left[-\underset{i}{\text{logsumexp}}\left(-\frac{||\varphi_\theta(x)-(SG[\mu_1^y]+\Delta\mu_i^y)||_2^2}{2}+c_i^y\right)-\log|\det J|\right] \tag{10}$$

where $SG[\cdot]$ means to stop gradient backpropagation, c_i^y denotes logarithmic intra-class center weights and is defined as $c_i^y := \log p_i(y) = \text{logsoftmax}_i(\psi_y)$. Note that $\psi_y\in\mathbb{R}^M$ is specific to class y, $\psi\in\mathbb{R}^Y$ is for the whole distribution.

Overall Loss Function. The overall training loss function is the combination of the Eq. 6, Eq. 8, Eq. 9 and Eq. 10, as follows:

$$\mathcal{L} = \lambda_1\mathcal{L}_g + \lambda_2\mathcal{L}_{mi} + \mathcal{L}_e + \mathcal{L}_{in} \tag{11}$$

where the λ_1 and λ_2 are used to trade off the loss items and are set to 1 and 100 by default. The $\mathcal{L}_e$ and $\mathcal{L}_{in}$ are used as auxiliary losses, so we don't ablate weighting factors for them as it will result in too many combinational experiments. In App. A.1, we further explain the necessities and motivations that connect each other of the individual components in our method.

3.4 Anomaly Scoring

As shown in Alg. 1, we actually extract K-level feature maps and construct the NF model at each level. For each test input feature x^k from level-k, $k \in K$, we can calculate its intra-class log-likelihood $\log p_\theta(x^k) = \text{logsumexp}_i(-||\varphi_\theta(x^k) - \mu_i^y||_2^2/2 + c_i^y) + \log|\det J| - d/2\log(2\pi)$ and inter-class negative entropy $nh(x^k) = \sum_y \text{softmax}_y(-||\varphi_\theta(x^k) - \mu_1^{y'}||_2^2/2) \cdot \text{logsoftmax}_y(-||\varphi_\theta(x^k) - \mu_1^{y'}||_2^2/2)$. Note that y in c_i^y denotes which class x^k belongs to, not whether x^k is normal or abnormal. Next, we convert the log-likelihood to likelihood $p_\theta(x^k) = \text{e}^{\log p_\theta(x^k)}$. Then, we upsample all $p_\theta(x^k)$ in the level-k to the input image resolution ($H \times W$) using bilinear interpolation $P_k = b(p_\theta(x^k)) \in \mathbb{R}^{H \times W}$. Finally, we calculate anomaly score map S_l by aggregating all upsampled likelihoods as $S_l = 1 - \sum_{k=1}^{K} P_k$ (see Sec. 3.1). For the inter-class negative entropy, we also follow the above steps to convert to anomaly score map S_e. Then the final anomaly score map is obtained by combining the two maps $S = S_l \odot S_e$, where the $\odot$ is the element-wise multiplication. In this way, even if anomalies fall into the inter-class Gaussian mixture distribution, they are usually in the low-density regions among the inter-class class centers. So, we can still ensure that anomalies are out-of-distribution through intra-class log-likelihoods (please see App. A.4 for more discussions).

3.5 Further Discussions

Explicitly Distinguishing Classes. Our method utilizes the class y of each image, but it doesn't introduce any extra data collection cost compared to one-for-one AD models. The existing AD datasets are collected for one-for-one anomaly detection (*i.e.*, we need to train a model for each class). Thus, the existing AD datasets need to be separated according to classes, with each class as a subdataset. Therefore, one-for-one AD methods also need to distinguish classes, as they require normal samples from the same class to train. Our method actually has the same supervision as these methods. The difference is that we explicitly distinguish classes but they don't explicitly distinguish classes. So, our method still follows the same data organization format as the one-for-one AD models. But the advantage of our unified AD method is that we can train one model for all classes, greatly reducing the resource costs of training and deploying. Moreover, our method only requires separating different classes and does not require pixel-level annotations. For real-world industrial applications, this doesn't incur extra data collection costs, as we usually consciously collect data according to different classes. In App. A.2, we summarize the training samples and the supervision information required by our method and other methods in detail.

PMAD [34] uses class information, and both UniAD [35] and OmniAL [41] use extra information to simulate anomalies. Although we also use class information, our method can achieve superior results than these methods. Thus, we think that whether utilizing class information should not be a strict limitation to the unified AD modeling especially when it already exists.

In App. A.3, we further provide more discussions with the "identical shortcut" issue and point out that the "homogeneous mapping" is not intrinsically equal to the "identical shortcut" issue. Compared to UniAD, our method is an effective exploration for unified anomaly detection in the direction of normalizing flow based anomaly detection. In App. A, we also further discuss the limitations, applications, effective guarantee, and complexity of our method. We also provide an information-theoretic view in App. F.

4 Experiment

4.1 Datasets and Metrics

Datasets. We extensively evaluate our approach on four real-world industrial AD datasets: MVTecAD [4], BTAD [17], MVTec3D-RGB [6], and VisA [42]. The detailed introduction to these datasets is provided in App. D. To more sufficiently evaluate the unified AD performance of different AD models, we combine these datasets to form a 40-class dataset, which we call Union dataset.

Metrics. Following prior works [4, 5, 37], the standard metric in anomaly detection, AUROC, is used to evaluate the performance of AD methods.

4.2 Main Results

Setup. For each dataset, we train one unified model with images from different classes. We use Efficient-b6 [27] as the feature extractor. The parameters of the feature extractor are frozen during training. The layer numbers of the NF models are all 12. The number of inter-class centers is always equal to the number of classes in the dataset. The number of intra-class centers is set as 10 for all datasets (see ablation study in Sec. 4.3). We use the Adam [19] optimizer with weight decay $1e^{-4}$ to train the model. The total training epochs are set as 100 and the batch size is 8 by default. The learning rate is $2e^{-4}$ initially, and dropped by 0.1 after [48, 57, 88] epochs. The evaluation is run with 3 random seeds.

Baselines. We compare our approach with one-for-one AD baselines including: PaDiM [9], MKD [24], and DRAEM [37], and SOTA unified AD methods: PMAD [34], UniAD [35], and OmniAL [41]. We also compare with the SOTA one-for-one NF-based AD methods: CFLOW [12] and FastFlow [36]. Under the unified case, the results of the one-for-one AD baselines and the NF-based AD methods are run with the publicly available implementations.

Quantitative Results. The detailed results on MVTecAD are shown in Tab. 1. We also report the results under the one-for-one setting in App. Tab. 3. By comparison, we can see that the performances of all baselines and SOTA

Table 1: Anomaly detection and localization results on MVTecAD. All methods are evaluated under the unified case. ·/· means the image-level and pixel-level AUROCs.

Category	Baseline Methods			Unified Methods			Normalizing Flow Based Methods		
	PaDiM	MKD	DRAEM	PMAD	UniAD	OmniAL	FastFlow	CFLOW	HGAD (Ours)
Carpet	93.8/97.6	69.8/95.5	98.0/98.6	99.0/97.9	99.8/98.5	98.7/**99.4**	91.6/96.7	98.8/97.5	**100**±0.00/**99.4**±0.05
Grid	73.9/71.0	83.8/82.3	99.3/98.7	96.2/95.6	98.2/96.5	**99.9**/**99.4**	85.7/96.8	95.9/94.1	99.6±0.09/99.1±0.08
Leather	99.9/84.8	93.6/96.7	98.7/97.3	**100**/99.2	**100**/98.8	99.0/99.3	93.7/98.2	**100**/98.1	**100**±0.00/**99.6**±0.00
Tile	93.3/80.5	89.5/85.3	99.8/98.0	99.8/94.5	99.3/91.8	99.6/**99.0**	99.2/95.8	97.9/92.2	**100**±0.00/96.1±0.09
Wood	98.4/89.1	93.4/80.5	**99.8**/96.0	99.6/89.0	98.6/93.2	93.2/**97.4**	98.0/92.0	99.0/92.7	99.5±0.08/95.9±0.09
Bottle	97.9/96.1	98.7/91.8	97.5/87.6	99.8/98.4	99.7/98.1	**100**/**99.2**	**100**/94.0	98.7/96.4	**100**±0.00/98.6±0.08
Cable	70.9/81.0	78.2/89.3	57.8/71.3	93.5/95.4	95.2/**97.3**	**98.2**/**97.3**	90.9/95.2	80.4/92.9	97.3±0.26/95.2±0.49
Capsule	73.4/96.9	68.3/88.3	65.3/50.5	80.5/97.0	86.9/98.5	95.2/96.9	90.5/98.6	75.5/97.7	**99.0**±0.40/**99.2**±0.05
Hazelnut	85.5/96.3	97.1/91.2	93.7/96.9	99.6/97.4	99.8/98.1	95.6/98.4	98.9/96.6	97.1/95.7	**99.9**±0.08/**98.8**±0.05
Metal nut	88.0/84.8	64.9/64.2	72.8/62.2	98.0/91.7	99.2/94.8	99.2/**99.1**	96.5/97.2	87.8/84.4	**100**±0.00/97.8±0.29
Pill	68.8/87.7	79.7/69.7	82.2/94.4	89.4/93.4	93.7/95.0	**97.2**/**98.9**	90.4/96.1	88.0/90.7	96.3±0.73/98.8±0.05
Screw	56.9/94.1	75.6/92.1	92.0/95.5	73.3/96.6	87.5/98.3	88.0/98.0	76.8/95.9	59.5/93.9	**95.5**±0.16/**99.3**±0.12
Toothbrush	95.3/95.6	75.3/88.9	90.6/97.7	95.8/98.2	94.2/98.4	**100**/**99.4**	86.1/97.1	78.0/95.7	91.2±0.37/99.1±0.05
Transistor	86.6/92.3	73.4/71.7	74.8/64.5	97.2/93.3	**99.8**/**97.9**	93.8/93.3	85.7/93.8	86.7/92.3	97.7±0.21/91.9±0.26
Zipper	79.7/94.8	87.4/86.1	98.8/98.3	96.0/96.1	95.8/96.8	**100**/**99.5**	93.8/95.7	92.2/95.7	**100**±0.04/99.0±0.09
Mean	84.2/89.5	81.9/84.9	88.1/87.2	94.5/95.6	96.5/96.8	97.2/**98.3**	91.8/96.0	89.0/94.0	**98.4**±0.08/97.9±0.05

one-for-one NF-based AD methods drop dramatically under the unified case. However, our HGAD outperforms all baselines under the unified case significantly. Compared with the one-for-one NF-based AD counterparts, we improve the unified AD performance from 91.8% to 98.4% and from 96.0% to 97.9%. We further indicate that our model has the same network architecture as CFLOW, and we only introduce multiple inter- and intra-class centers as extra learnable parameters. This means that our novel designs are the keys to improving the unified AD ability of the NF-based AD methods. Moreover, our HGAD also surpasses the SOTA unified AD methods, PMAD (by 3.9% and 2.3%) and UniAD (by 1.9% and 1.1%), demonstrating our superiority. Furthermore, the results on BTAD, MVTec3D-RGB, VisA, and Union dataset (see Tab. 2) also verify the superiority of our method, where we outperform UniAD by 0.9%, 9.6%, 4.3%, and 6.6% in anomaly detection.

Table 2: Anomaly detection and localization results on BTAD, MVTec3D-RGB, VisA, and Union datasets.

Dataset	PaDiM	MKD	DRAEM	PMAD	UniAD	OmniAL	FastFlow	CFLOW	HGAD (Ours)
BTAD	93.8/96.6	89.7/96.2	91.2/91.9	93.8/97.3	94.0/97.2	-/-	92.9/95.3	93.0/96.6	**94.9**±0.08/**98.0**±0.10
MVTec3D-RGB	77.4/96.3	73.5/95.9	73.9/95.5	75.4/95.3	77.5/96.6	-/-	67.9/90.2	71.6/95.7	**87.1**±0.16/**97.7**±0.05
VisA	86.8/97.0	74.2/93.9	85.5/90.5	-/-	92.8/98.1	87.8/96.6	77.2/95.1	88.0/95.9	**97.1**±0.09/**98.9**±0.05
Union	79.0/91.4	72.1/88.9	66.4/82.7	-/-	86.9/95.5	-/-	57.2/78.8	55.7/82.9	**93.5**±0.23/**97.5**±0.14

Qualitative Results. Fig. 4 shows qualitative results. It can be found that our approach can generate much better anomaly score maps than the one-for-one NF-based baseline CFLOW [12] even for different anomaly types. More qualitative results are in the App. Fig. 2.

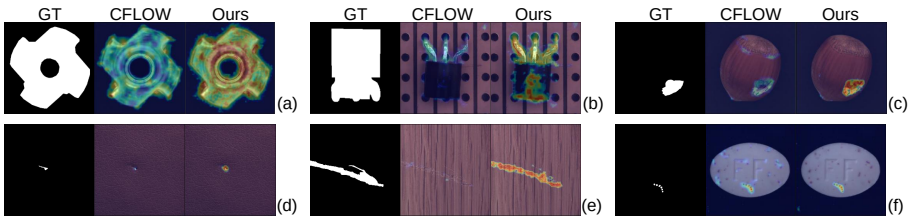

Fig. 4: Qualitative results on MVTecAD. (a) and (b) both represent global anomalies, (c) contains large cracks, (d) shows small dints, (e) contains texture scratches, and (f) shows color anomalies.

Table 3: Ablation studies on MVTecAD. *SGC*, *FMC*, *ICG*, *MIM*, and *Intra* mean single center, fixed multiple centers, inter-class Gaussian mixture modeling, mutual information maximization, and intra-class mixed class centers learning, respectively.

(a) Hierarchical Gaussian mixture prior.

SGC	FMC	**ICG**	**MIM**	**Intra**	Det.	Loc.
✓	-	-	-	-	89.0	94.0
-	✓	-	-	-	93.1	95.8
-	-	✓	-	-	94.5	96.7
-	-	✓	✓	-	96.3	96.6
-	✓	-	-	✓	96.3	97.1
-	-	✓	✓	✓	**97.7**	**97.6**

(b) Number of intra-class centers.

# Centers	Det.	Loc.
3	97.5	97.3
5	97.6	97.1
10	**97.7**	**97.6**
15	97.6	97.3
20	97.4	97.2

(c) Anomaly criterion.

Logps	**Entropy**	Det.	Loc.
✓	-	94.4	97.1
-	✓	96.6	97.0
✓	✓	**97.7**	**97.6**

(d) Hyperparameters.

λ_1	λ_2	Det.	Loc.	λ_1	λ_2	Det.	Loc.
1	1	96.5	96.7	0.5	100	98.4	97.8
1	5	97.6	97.3	**1**	**100**	**98.4**	**97.9**
1	10	97.7	97.6	5	100	98.3	97.9
1	50	98.2	97.8	10	100	98.3	97.8
1	**100**	**98.4**	**97.9**	20	100	97.8	97.6

(e) Optimization strategy.

	Det.	Loc.
-	96.5	97.0
✓	**97.7**	**97.6**

4.3 Ablation Studies

Hierarchical Gaussian Mixture Modeling. 1) Tab. 3a verifies our confirmation that mapping to multiple class centers is effective. With the fixed multiple centers (FMC, which is only based on fixed multiple centers but without Gaussian mixture modeling), image-level and pixel-level AUROCs can be improved by 4.1% and 1.8%, respectively. By employing the inter-class Gaussian mixture modeling to learn the latent multi-class distribution, the AUROCs can further be improved by 1.4% and 0.9%. 2) The effectiveness of Mutual information maximization (MIM) is proven in Tab. 3a, where adding MIM brings promotion by 1.8% for detection. This shows that to better learn the complex multi-class distribution, it is necessary to endow the model class discrimination ability to avoid multiple centers collapsing into the same center. 3) Tab. 3a confirms the efficacy of intra-class mixed class centers learning. With the FMC as the baseline,

introducing to learn intra-class mixed class centers could bring an increase of 1.6% for detection and 1.0% for localization, respectively. Finally, combining these, we form the hierarchical Gaussian mixture modeling method to achieve the best results.

Number of Intra-Class Centers. We conduct experiments to investigate the influence of intra-class centers in each class. The results are shown in Tab. 3b. The best performance is achieved with a moderate number: 10 class centers. A larger class center number like 20 does not bring further promotion, which may be because the class centers are saturated and more class centers are harder to train. For other datasets, we also use 10 as the number of intra-class centers.

Anomaly Criterion. Only taking the log-likelihood and the entropy as the anomaly criterion can achieve a good performance, while our associated criterion outperforms each criterion consistently. This illustrates that the associated anomaly scoring strategy is more conducive to guarantee that anomalies are recognized as out-of-distribution.

Hyperparameters. We ablate the hyperparameters λ_1 and λ_2 in Tab. 3d. Note that the experiments in Tab. 3a, b, c, and e are conducted with λ_1 and λ_2 set to 1 and 10. The results in Tab. 3d show that the larger λ_2 can achieve better unified AD performance. The larger λ_2 can urge the network more focused on separating different class features into their corresponding class centers, indicating that the class discrimination ability is of vital significance to accomplish unified anomaly detection.

Optimization Strategy. We found that simultaneously optimizing inter-class and intra-class centers by the beginning would bring instability. Thus, we employ a two-stage optimization strategy (see App. C for details). The strategy can better decouple the inter-class and intra-class learning processes, thereby bringing better results.

5 Conclusion

In this paper, we focus on how to unify anomaly detection regarding multiple classes. For such a challenging task, popular normalizing flow based AD methods may fall into a "homogeneous mapping" issue. To address this, we propose a novel HGAD against learning the bias with three key improvements: inter-class Gaussian mixture modeling, mutual information maximization, and intra-class mixed class centers learning strategy. Under the unified AD setting, our method can improve NF-based AD methods by a large margin, and also surpass the SOTA unified AD methods.

Acknowledgements

This work was supported in part by the National Natural Science Fund of China (62371295), the Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102), and the Science and Technology Commission of Shanghai Municipality (22DZ2229005).

References

1. Acsintoae, A., Florescu1, A., Georgescu, M.I., Mare, T., Sumedrea1, P., Ionescu, R.T., Khan, F.S., Shah, M.: Ubnormal: New benchmark for supervised open-set video anomaly detection. In CVPR (2022)
2. Akcay, S., Atapour-Abarghouei, A., Breckon, T.P.: Ganomaly: Semi-supervised anomaly detection via adversarial training. In ACCV p. 622–637 (2018)
3. Bergman, L., Cohen, N., Hoshen, Y.: Deep nearest neighbor anomaly detection. arXiv preprint arXiv: 2002.10445 (2020)
4. Bergmann, P., Fauser, M., Sattlegger, D., Steger, C.: Mvtec ad - a comprehensive real-world dataset for unsupervised anomaly detection. In CVPR (2019)
5. Bergmann, P., Fauser, M., Sattlegger, D., Steger, C.: Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. In CVPR (2020)
6. Bergmann, P., Jin, X., Sattlegger, D., Steger, C.: The mvtec 3d-ad dataset for unsupervised 3d anomaly detection and localization. In Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (2021)
7. Biase, G.D., Blum, H., Siegwart, R., Cadena, C.: Pixel-wise anomaly detection in complex driving scenes. In CVPR (2021)
8. Cohen, N., Hoshen, Y.: Sub-image anomaly detection with deep pyramid correspondences. arXiv preprint arXiv: 2005.02357v3 (2020)
9. Defard, T., Setkov, A., Loesch, A., Audigier, R.: Padim: a patch distribution modeling framework for anomaly detection and localization. In ICPR International Workshops and Challenges (2021)
10. Deng, H., Li, X.: Anomaly detection via reverse distillation from one-class embedding. In CVPR (2022)
11. Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using real nvp. In International Conference on Learning Representations (2017)
12. Gudovskiy, D., Ishizaka, S., Kozuka, K.: Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. In IEEE Winter Conference on Application of Computer Vision (2022)
13. Kingma, D.P., Dhariwal, P.: Glow: Generative flow with invertible 1x1 convolutions. In Conference and Workshop on Neural Information Processing Systems (2019)
14. Kirichenko, P., Izmailov, P., Wilson, A.G.: Why normalizing flows fail to detect out-of-distribution data. In NeurIPS (2020)
15. Li, R., Zhang, C., Zhou, H., Shi, C., Luo, Y.: Out-of-distribution identification: Let detector tell which i am not sure. In ECCV (2022)
16. Liu, W., Li, R., Zheng, M., Karanam, S., Wu, Z., Bhanu, B., Radke, R.J., Camps, O.: Towards visually explaining variational autoencoders. In CVPR (2020)
17. Mishra, P., Verk, R., Fornasier, D., Piciarelli, C., Foresti, G.L.: Vt-adl: A vision transformer network for image anomaly detection and localization. In 30th IEEE International Symposium on Industrial Electronics (2021)
18. Park, H., Noh, J., Ham, B.: Learning memory-guided normality for anomaly detection. In CVPR (2020)
19. P.Kingma, D., Ba, J.L.: Adam: A method for stochastic optimization. In ICLR (2015)
20. Roth, K., Pemula, L., Zepeda, J., Scholkopf, B., Brox, T., Gehler, P.: Towards total recall in industrial anomaly detection. In CVPR (2022)
21. Rudolph, M., Wandt, B., Rosenhahn, B.: Same same but differnet: Semi-supervised defect detection with normalizing flows. In IEEE Winter Conference on Application of Computer Vision (2021)

22. Rudolph, M., Wehrbein, T., Rosenhahn, B., Wandt, B.: Fully convolutional cross-scale-flows for image-based defect detection. In IEEE Winter Conference on Application of Computer Vision (2022)
23. Rudolph, M., Wehrbein, T., Rosenhahn, B., Wandt, B.: Asymmetric student-teacher networks for industrial anomaly detection. In WACV (2023)
24. Salehi, M., Sadjadi, N., Rohban, S.H., R.Rabiee, H.: Multiresolution knowledge distillation for anomaly detection. In CVPR (2021)
25. Schlegl, T., Seeb¨ock, P., Waldstein, S.M., Schmidt-Erfurth, U., Langs, G.: Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In International Conference on Information Processing in Medical Imaging (2017)
26. Sultani, W., Chen, C., Shah, M.: Real-world anomaly detection in surveillance videos. In CVPR (2018)
27. Tan, M., Le, Q.V.: Efficientnet: Rethinking model scaling for convolutional neural networks. In International Conference on Machine Learning (2019)
28. Tian, Y., Pang, G., Liu, F., Chen, Y., Shin, S.H., Verjans, J.W., Singh, R., Carneiro., G.: Constrained contrastive distribution learning for unsupervised anomaly detection and localisation in medical images. Medical Image Computing and Computer Assisted Intervention pp. 128–140 (2021)
29. Villani, C.: Topics in optimal transportation. American Mathematical (2003)
30. Vojir, T., Sipka, T., Aljundi, R.: Road anomaly detection by partial image reconstruction with segmentation coupling. In ICCV (2021)
31. Wang, G., Han, S., Ding, E., Huang, D.: Student-teacher feature pyramid matching for unsupervised anomaly detection. In British Machine Vision Conference (2021)
32. Yao, X., Li, R., Qian, Z., Luo, Y., Zhang, C.: Focus the discrepancy: Intra- and inter-correlation learning for image anomaly detection. In ICCV (2023)
33. Yao, X., Li, R., Zhang, J., Sun, J., Zhang, C.: Explicit boundary guided semi-push-pull contrastive learning for supervised anomaly detection. In CVPR (2023)
34. Yao, X., Zhang, C., Li, R., Sun, J., Liu, Z.: One-for-all: Proposal masked cross-class anomaly detection. In AAAI (2023)
35. You, Z., Cui, L., Shen, Y., Yang, K., Lu, X., Zheng, Y., Le, X.: A unified model for multi-class anomaly detection. In NeurIPS (2022)
36. Yu, J., Zheng, Y., Wang, X., Li, W., Wu, Y., Zhao, R., Wu, L.: Fastflow: Unsupervised anomaly detection and localization via 2d normalizing flows. arXiv preprint arXiv:2111.07677 (2021)
37. Zavrtanik, V., Kristan, M., Skocaj, D.: Draem: A discriminatively trained reconstruction embedding for surface anomaly detection. In ICCV (2021)
38. Zhang, B., Song, Q., Min, M.R., Cheng, W., Lumezanu, C., Cho, D., Chen, H.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In ICLR (2018)
39. Zhang, J., Xie, Y., Pang, G., Liao, Z., Verjans, J., Li, W., Sun, Z., He, J., Li, Y., Shen, C., Xia, Y.: Viral pneumonia screening on chest x-rays using confidence-aware anomaly detection. Medical Imaging pp. 879–890 (2021)
40. Zhang, K., Wang, B., Kun, C.C.: Pedenet: Image anomaly localization via patch embedding and density estimation. In Pattern Recognition Letters (2021)
41. Zhao, Y.: Ominal: A unified cnn framework for unsupervised anomaly localization. In CVPR (2023)
42. Zou, Y., Jeong, J., Pemula, L., Zhang, D., Dabeer, O.: Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. In ECCV (2022)