This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Hunting Normality from Query Sample via Residual Learning for Generalist Anomaly Detection

Xiaolei Wang[1,2] Yuexin Wang[1,2] Tianhong Dai[4] Huihui Bai[3] Yao Zhao[3] Jimin Xiao[1*]
[1]Xi'an Jiaotong-Liverpool University [2]University of Liverpool
[3]Beijing Jiaotong University [4]Applied Computing Technologies

Abstract

Generalist Anomaly Detection (GAD) seeks to overcome the domain-specific limitations of traditional anomaly detection by training a unified model that can generalize to unseen classes. A promising GAD strategy involves using residual features to create a class-invariant space. However, existing methods that directly model the distribution of residuals face unpredictable risks: there is inconsistency between residual and instance features, i.e., subtle defects may yield small residuals (false negatives), or normal feature residuals could be large due to the diversity of normality (false positives). To address these limitations, we propose a novel residual-based learning framework that re-purposes residuals as a guide to learn instance-level normality, rather than modeling their distribution directly. Our framework features two new attention-based modules: Residual Feature Learning (RFL), which uses learnable proxies to capture diverse patterns from the residual features, and Normality Learning from Support (NLS), which leverages these residual proxies to aggregate query-related normality proxies from the support instance features. These dynamically generated normality proxies are then used to hunt for normality within the query patch features, enabling accurate anomaly localization. Extensive experiments on GAD benchmarks demonstrate the effectiveness of our method. Code is available at HNQ-GAD.

1. Introduction

Visual anomaly detection (AD), a critical task with applications ranging from industrial inspection [11, 18, 38, 44] to medical diagnosis [13, 22], aims to identify rare samples or unexpected patterns that deviate from an established definition of normality. Traditional AD approaches [12, 51, 52] generally train and test a single model within a single, fixed domain. This approach lacks transferability, as these models must be retrained or fine-tuned when deployed on new, unseen classes. To address existing limitations, Generalist Anomaly Detection (GAD) [47, 56] has emerged, which seeks to train one unified model on the source domain that can generalize to detect anomalies in diverse, unseen classes from the target domain.

A promising solution for tackling this class generalization challenge is 'residual feature'. A residual feature is computed by subtracting the nearest normal support feature from a query feature [47, 56]. This operation effectively removes class-specific information, creating a more class-invariant feature space (see Fig.1a). Recently, ResAD [47] proposes to learn the distribution of normal residual features, assuming that anomalies will produce residuals that fall out of distribution. However, this method simply assumes that the residual feature distribution is consistent with the original feature distribution, i.e., normal residuals tend to cluster together while abnormal ones are located outside of the normal residual cluster. Actually, the abnormal residual is only an indirect signal of abnormality, and the significant difference between query and support may not necessarily indicate abnormality. As shown in Fig.1b, a subtle defect may produce a small residual that is incorrectly classified as normal (false negatives). Conversely, the residuals calculated with two normal features may also be large due to the diversity of normality (false positives). The inconsistency between residuals and instance features indicates that modeling residual distributions directly carries unpredictable risks.

To address these limitations, we propose a novel framework that fully exploits the cross-domain capabilities of residuals while mitigating their inherent risks. Recognizing that residuals possess inherent transferability across domains, we first leverage initialized proxies to learn diverse and rich residual patterns. Subsequently, to ensure consistency between residual and instance features, these residual-based proxies serve as guidance signals and are transferred into the normal instance space to learn normal instance patterns. This process is facilitated by a second attention-based module, which utilizes residual-based proxies to adaptively aggregate the most representative normal instance proto-

*Corresponding Author

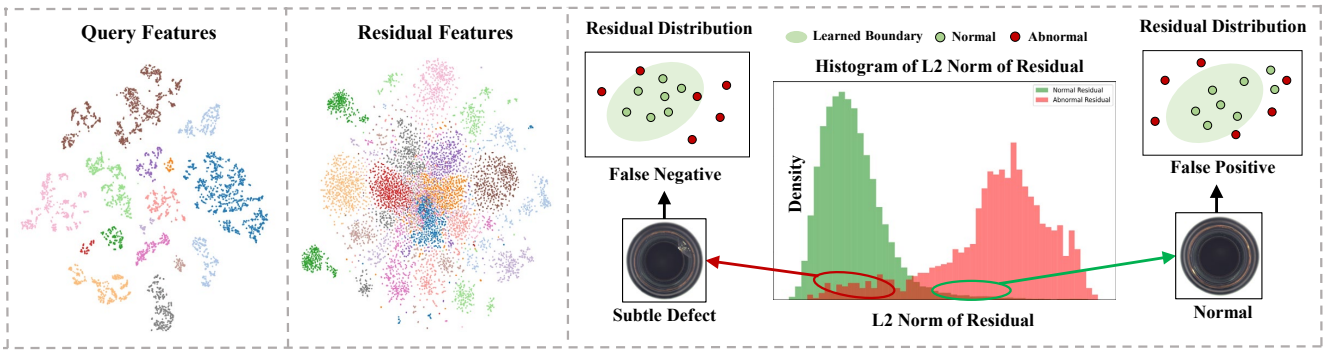


Figure 1. (a) The residual feature effectively removes class-specific information compared to instance features. (b) The inconsistency between residuals and instance features. A subtle defect may produce a small residual that is incorrectly classified as normal (false negatives). Conversely, the residuals calculated with two normal features may also be large (false positives).

types from the support set. By 'hunting normality' using these dynamically generated prototypes based on the specific support and query pairs rather than relying solely on static residual distributions, our approach robustly generalizes to new domains while maintaining accurate anomaly localization. It should be noted that we consider that the patterns of abnormal residuals and instance features are complex, and abnormal residuals are more likely to overlap with normal residuals (Fig.1b). Therefore, we prefer to focus on capturing the normality within the query.

To implement this, we propose a novel residual-based learning framework centered on two new attention-based modules: Residual Feature Learning (RFL) and Normality Learning from Support (NLS). The RFL module first utilizes a set of learnable proxies to process the residual features, capturing their diverse patterns and outputting a set of residual proxies. Next, the NLS module leverages these residual proxies to learn representative normality patterns from the support set features, generating an adaptive set of normality proxies. Finally, these dynamically generated normality proxies are used to hunt for normality within the query patch features to achieve anomaly localization. The insight of this framework is to learn a transferable relationship between residual patterns and instance-level normality, thereby using residuals as a guide to learn instance-level normality rather than modeling their distribution directly. Our contributions can be summarized as follows:

(1) We identify that modeling residual feature distribution directly in GAD tasks carries significant risks, as the inconsistency between residuals and instance features degrades distribution modeling.

(2) We propose a novel framework that re-purposes residual features as a guide to learn instance-level normality patterns from the support set, mitigating the inherent risks in GAD.

(3) The proposed method achieves competitive performance on GAD benchmarks, demonstrating its superiority in generalizing and detecting anomalies in target domains.

2. Related Work

2.1. Anomaly Detection

Visual Anomaly Detection (AD) aims to identify samples or patterns that deviate from our familiar normal one. AD can be divided into multiple sub-tasks based on real-world needs, such as unsupervised AD [3, 38, 42, 43, 45, 48, 50], few-shot AD (FSAD) [14, 23, 27], zero-shot AD (ZSAD) [4, 16, 35, 55], noisy AD [6, 26], open-set AD [13, 57], and 3D AD [8, 29, 54]. Most traditional AD methods follow a one-for-one or one-for-many paradigm, focusing on training and testing a model within a single domain. These methods are typically unsupervised, adopting a one-class classification paradigm (OCC) [39] where the model is trained only on normal samples to identify unseen abnormal patterns. These unsupervised methods generally fall into three main categories: reconstruction-based method [19–21, 34, 40, 44], feature-embedding-based method [28], and augmentation-based method [5, 9, 53]. While effective, these unsupervised methods are designed to detect anomalies on a single data domain and lack transferability. They typically require retraining or fine-tuning when encountering new, unseen classes.

2.2. Few-shot and Generalist Anomaly Detection

Few-Shot Anomaly Detection (FSAD) [14, 15, 23, 31] aims to identify anomalies using only a limited number of normal samples from a target dataset. FSAD methods can be broadly split into two groups: training-free and meta-learning-based methods. Training-free methods, such as FastRecon [14] and UniVAD [17], are often adapted from embedding-based approaches. They use the few-shot normal samples to construct a reference memory bank or calculate distance-based anomaly scores without any network

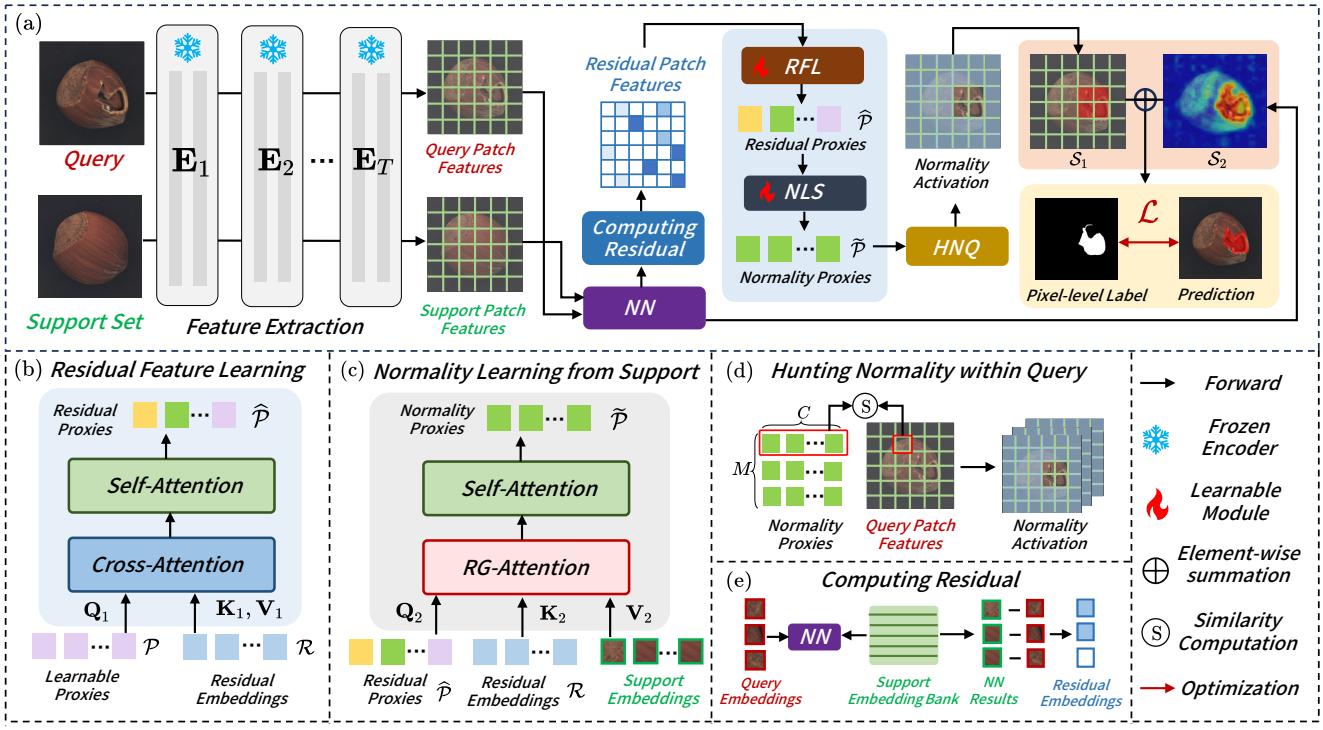


Figure 2. Overview of training phase of our framework. Given a query sample $\mathbf{x}^q$ and its corresponding support samples $\{\mathbf{x}_k^s\}_{k=1}^K$ (normal), we first leverage a frozen encoder $\mathbf{E}$ on query and support inputs, obtaining query patch features $\mathcal{F}^q$ and support patch features $\{\mathcal{F}_k^s\}_{k=1}^K$. Next, residual patch features $\mathcal{R}$ are computed by the NN search and element-wise subtraction. Subsequently, we apply several learnable proxies $\mathcal{P}$ to capture the pattern of residual $\mathcal{R}$, obtaining residual proxies $\widehat{\mathcal{P}}$ in the RFL module. In the next NLS module, $\widehat{\mathcal{P}}$ aims to learn normality from support embeddings by comparing with residual, and the output of NLS is named as normality proxies $\widetilde{\mathcal{P}}$ that can hunt potential normality from query features, achieving anomaly localization.

training. Meta-learning-based methods [23, 29, 49] focus on generalizing the detection ability of model to new target domains. RegAD [23] trains a registration network on seen categories that can then register samples from unseen categories. However, RegAD remains dependent on domain relevance between training and testing data. Based on meta-learning-based FSAD, generalist anomaly detection (GAD) [47, 56] trains one unified model on the source domain that can generalize to detect anomalies in diverse, unseen classes from the target domain. To achieve domain-agnostic anomaly detection, InCTRL [56] firstly proposes a generalist anomaly detection (GAD) task and utilizes residual distance to discriminate anomalies. Subsequently, ResAD [47] directly learning the distribution of residual features to eliminate domain dependencies to achieve GAD.

3. Methodology

3.1. Problem Statement

Generalist anomaly detection (GAD) [47, 56] aims to train a unified model on a source domain, $\mathcal{D}_{\text{source}}$ that can generalize to identify anomalies in a previously unseen target domain, $\mathcal{D}_{\text{target}}$. Following a standard meta-learning framework, the source dataset $\mathcal{D}_{\text{source}} = \{\mathcal{X}, \mathcal{Y}\}$ consists of N samples: $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$, and their corresponding ground-truth (GT) masks: $\mathcal{Y} = \{\mathbf{y}_i\}_{i=1}^N$. During training, the data can be divided into multiple episodes, each consisting of a query sample $\mathbf{x}^q$ with its GT mask $\mathbf{y}^q$ and a support set of K normal samples $\{\mathbf{x}_k^s\}_{k=1}^K$ from the same class as $\mathbf{x}^q$. The model is then trained across these episodes. During inference, the trained model is evaluated on $\mathcal{D}_{\text{target}}$, which contains classes unseen in $\mathcal{D}_{\text{source}}$, using the same episodic structure to locate potential anomalies in the query samples.

3.2. Overview

The overview of our training framework is illustrated in Fig.2. For each training episode, a query input and its corresponding support samples are first processed by a pre-trained, frozen backbone network to extract patch-level features. The residual features for each query patch are then obtained by performing a Nearest-Neighbor (NN) search against the support set features and calculating their differences. Although the residual features help build a class-invariant feature space, they still exhibit inconsistencies

across instances, which may introduce potential risks. To mitigate this issue, we introduce a learnable module that models and leverages normality information from the support set under the guidance of the residual features. This process is driven by two novel attention-based modules: Residual Feature Learning (RFL) and Normality Learning from Support (NLS). RFL utilizes a set of learnable proxies that interact with residual features through cross-attention to capture diverse residual patterns, generating a set of 'residual proxies'. Next, NLS leverages the residual proxies to learn representative normality patterns from the support set features. It employs a residual-guided comparison to generate a query-related set of 'normality proxies'. Finally, these generated normality proxies are used to *hunt* for normality within the query patch features. Anomaly localization is achieved by computing the similarity between the query features and the normality proxies. By training the framework on the source domain, the model learns a transferable relationship between residual patterns and instance-level normality, enabling generalization to unseen target domains.

3.3. Feature Extraction and Residual Feature

Following prior works [10, 46], Vision Transformer (ViT) is adopted as our feature encoder. As shown in Fig.2a, the frozen encoder $\mathbf{E}$ with T layers first processes a query image x^q and its K support images $\{\mathbf{x}_k^s\}_{k=1}^K$ to extract the intermediate-layer features for both, producing query patch feature $\mathcal{F}^q \in \mathbb{R}^{HW\times C}$ and a set of support patch features $\{\mathcal{F}_k^s\}_{k=1}^K$, respectively, where each $\mathcal{F}_k^s \in \mathbb{R}^{HW\times C}$. Here, H and W are the spatial dimensions of the feature map, and C is the channel dimension. For convenience, we denote the flattened query features as $\mathcal{F}^q = \{f_l^q\}_{l=1}^L$ and the aggregated bank of all support features as $\mathcal{F}^s = \{f_l^s\}_{l=1}^{KL}$, where $L = H * W$.

These features are then used to compute residual features that effectively suppress class-specific information, thus forming a more class-invariant space suitable for GAD. The residual computation is based on the NN search. For each query embedding $f^q \in \mathcal{F}^q$, its nearest neighbor f_*^s is identified from the support bank $\mathcal{F}^s$ using the cosine distance $\mathbf{d}(\cdot,\cdot)$:

$$f_*^s := \underset{f^s\in\mathcal{F}^s}{\mathbf{argmin}} \left(\mathbf{d}(f^q, f^s)\right). \tag{1}$$

The residual embedding r is then defined as the difference between the embedding and its nearest neighbor (Fig.2e):

$$r = f^q - f_*^s, r \in \mathbb{R}^{1\times1\times C}. \tag{2}$$

By performing this operation on all query embeddings, the residual patch features $\mathcal{R} = \{r_l\}_{l=1}^L$ are obtained, which can be expressed as:

$$\mathcal{R} = \mathcal{F}^q - \mathcal{F}_*^s, \tag{3}$$

where $\mathcal{R} \in \mathbb{R}^{HW\times C}$ represents the 'difference' between the query features and the most similar normal features from the support set (see Fig.3A).

3.4. Residual Feature Learning

As shown in Fig.2b, the Residual Feature Learning (RFL) module is designed to capture the varied patterns within the residual features $\mathcal{R}$. We first initialize M learnable proxies, denoted by $\mathcal{P} \in \mathbb{R}^{M\times C}$. Then, RFL employs a cross-attention block, where the proxies $\mathcal{P}$ serve as queries to focus on the residual features $\mathcal{R}$. The *Query*, *Key*, and *Value* for the cross-attention block are defined as:

$$\mathbf{Q}_1 = \mathbf{W}_1^{\mathcal{Q}}\mathcal{P}, \mathbf{K}_1 = \mathbf{W}_1^{\mathcal{K}}\mathcal{R}, \mathbf{V}_1 = \mathbf{W}_1^{\mathcal{V}}\mathcal{R}, \tag{4}$$

where $\mathbf{W}_1^{\mathcal{Q}}$, $\mathbf{W}_1^{\mathcal{K}}$, and $\mathbf{W}_1^{\mathcal{V}}$ are learnable weights. The output of this cross-attention is then passed through a self-attention module $\mathrm{SA}_1(\cdot)$ to produce the final residual proxies $\widehat{\mathcal{P}}$:

$$\widehat{\mathcal{P}} = \mathrm{SA}_1\left(\mathrm{Softmax}\left(\frac{\mathbf{Q}_1\mathbf{K}_1^\top}{\sqrt{d}}\right)\mathbf{V}_1\right), \tag{5}$$

where $\top$ is transpose of matrix, and d is scaling factor [41]. Since both *Key* and *Value* are derived from the residual features $\mathcal{R}$, the proxy $\widehat{\mathcal{P}}$ contains information across the entire residual space and can be considered as prototypes of residual features.

3.5. Normality Learning from Support

The Normality Learning from Support (NLS) module, depicted in Fig.2c, leverages the learned residual proxies $\widehat{\mathcal{P}}$ to actively learn normality patterns from the support set $\mathcal{F}^s$. This is achieved through a residual-guided comparison that produces a set of normality proxies $\widetilde{\mathcal{P}} \in \mathbb{R}^{M\times C}$.

The core of NLS is a Residual-Guided Attention (RG-Attention) mechanism. This is modeled as a cross-attention block where the *Query*, *Key*, and *Value* are defined as:

$$\mathbf{Q}_2 = \mathbf{W}_2^{\mathcal{Q}}\widehat{\mathcal{P}}, \mathbf{K}_2 = \mathbf{W}_2^{\mathcal{K}}\mathcal{R}, \mathbf{V}_2 = \mathbf{W}_2^{\mathcal{V}}\bar{\mathcal{F}}^s, \tag{6}$$

where $\bar{\mathcal{F}}^s = \frac{1}{K}\sum_{k=1}^K \mathcal{F}_k^s$ is the mean vector of the support features. The attention weights (from $\mathbf{Q}_2\mathbf{K}_2^\mathrm{T}$ in Eq. (6)) are computed by measuring the correlation between the residual proxies $\widehat{\mathcal{P}}$ and the residual embeddings $\mathcal{R}$ implicitly. These weights are then used to aggregate the support features ($\mathbf{V}_2$). This process enables the model to learn the correspondence between residual patterns and normality patterns from the support set, and to produce normality proxies $\widetilde{\mathcal{P}}$ that are represented in the instance feature space. Similar to RFL, the NLS module further refines the RG-Attention output through a self-attention block $\mathrm{SA}_2(\cdot)$:

$$\widetilde{\mathcal{P}} = \mathrm{SA}_2\left(\mathrm{Softmax}\left(\frac{\mathbf{Q}_2\mathbf{K}_2^\top}{\sqrt{d}}\right)\mathbf{V}_2\right). \tag{7}$$

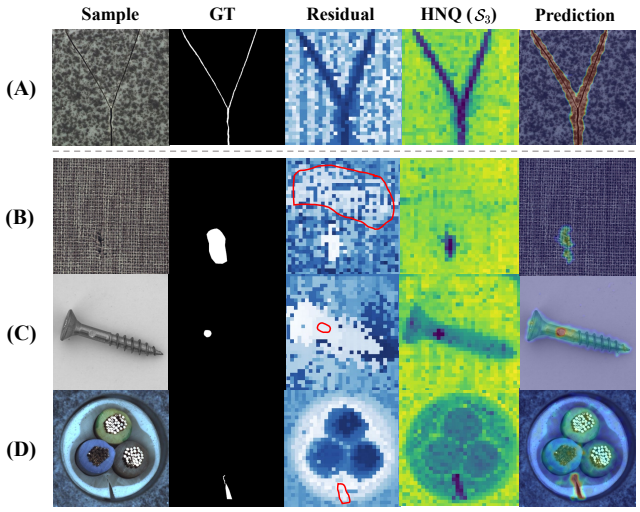


Figure 3. Visualization of residual patch feature and $\mathcal{S}_3$. (A) Residual can be seen as the difference between the query and support. (B-D) However, the significant difference between query and support may not necessarily be abnormal (red regions).

3.6. Hunting Normality within Query

Anomaly Localization Prediction As shown in Fig.2d, we apply similarity computation between normality proxies $\widetilde{\mathcal{P}} \in \mathbb{R}^{M\times C}$ and query patch features $\mathcal{F}^q \in \mathbb{R}^{HW\times C}$ to get the normality activation map. The pixel-level similarity can be written as:

$$\mathcal{S}_0^l = \frac{1}{M}\sum_{m=1}^{M}\frac{f_l^q \cdot \widetilde{\mathcal{P}}_m}{\|f_l^q\|\|\widetilde{\mathcal{P}}_m\|}, \tag{8}$$

where $\|\cdot\|$ is $\mathbf{L}^2$ norm. Based on the pixel-level similarity value, a normality activation map $\mathcal{S}_0 \in \mathbb{R}^{H\times W}$ (see Fig.3) is achieved and an initial anomaly map $\mathcal{S}_1$ is then given by $\mathcal{S}_1 = 1 - \mathcal{S}_0$. Following previous works [10, 38], a second anomaly score map $\mathcal{S}_2$ is further computed based on the NN distance between the query features $\mathcal{F}^q$ and the support features $\mathcal{F}^s$. The final prediction $\mathcal{S}_3$ is then produced by combining these two maps: $\mathcal{S}_3 = \mathcal{S}_1 + \mathcal{S}_2$.

As shown in Fig.3, residual features can sometimes be directly seen as the difference between query and support features (A). However, the abnormal residual serves only as a weak cue of abnormality, and the significant difference between query and support may not necessarily be an anomaly (B-D), which introduces potential risks. Therefore, instead of directly modeling the residuals, we model the relationship between residuals and instance features to mitigate the inconsistency between them. Furthermore, rather than focusing on the complex and diverse patterns of abnormal residuals, we leverage some proxies that are learned from residual and support features to *hunt* the normality within the query ($\mathcal{S}_3$ in Fig.3 shows a superior normal activation).

Optimization Objective Our framework focuses on training a residual-based model to adaptively hunt normality information within the query input. Therefore, the anomaly localization result $\mathcal{S}_3$ of NLQ module should be consistent with the GT mask $\mathbf{y}^q$ of the query image, i.e.,

$$\mathcal{L}_{\text{local}} = \text{Focal}(\mathcal{S}_3, \mathbf{y}^q) + \text{Dice}(\mathcal{S}_3, \mathbf{y}^q), \tag{9}$$

where $\text{Focal}(\cdot)$ and $\text{Dice}(\cdot)$ are the Focal loss [32] and Dice loss [30], respectively. Moreover, the obtained anomaly classification results should also align with image-level query label y^q, i.e.,

$$\mathcal{L}_{\text{global}} = \text{BCE}(\mathcal{S}_3^*, y^q), \tag{10}$$

where $\mathcal{S}_3^*$ is the maximum of prediction $\mathcal{S}_3$ and $\text{BCE}(\cdot)$ denotes BCE loss function. Finally, a unified optimization objective is adopted to train our model, according to Eq. (9) and (10):

$$\mathcal{L} = \mathcal{L}_{\text{global}} + \mathcal{L}_{\text{local}}. \tag{11}$$

3.7. Inference on Target Domain

During inference on the target domain $\mathcal{D}_{\text{target}}$, the trained model is applied using the same episodic structure. For a given test query and support set, the features are extracted, and residual features $\mathcal{R}$ are computed. The trained RFL and NLS modules then dynamically generate a new set of normality proxies $\widetilde{\mathcal{P}}$ specific to the target samples. These proxies are used to generate the normality map $\mathcal{S}_0$ for the target query. The final anomaly score map $\mathcal{S}_3$ is computed by combining $\mathcal{S}_1$ (from $\mathcal{S}_0$) and $\mathcal{S}_2$ (from NN distance), consistent with the training phase.

4. Experiment

4.1. Experimental Setup

Datasets. We conduct experiments on three widely used benchmarks: MVTec AD [1], VisA [58], and BraTS [36]. Our evaluation includes three tasks: (1) training on VisA and testing on MVTec AD;(2) training on MVTec AD and testing on VisA; (3) training on MVTec AD and testing on BraTS. The first two settings assess generalization across different industrial domains, while the third evaluates transferability from industrial to medical applications.

Evaluation Metrics. Following previous works [10, 25, 47], we evaluate both image-level anomaly classification and pixel-level localization performance. For detection, we report the Area Under the Receiver-Operator Curve (AUROC [11]), maximum F1-score at the optimal threshold (F1-max), and Average Precision (AP) based on image-level scores. For segmentation, we utilize pixel-level AUROC, F1-max, and Per-Region Overlap (PRO [1, 2]).

Table 1. Comprehensive GAD results with six image-level and pixel-level metrics on MVTec AD, VisA datasets. When testing with MVTec AD, the training set is Visa, and vice versa. Results marked with $*$ are based on our re-implementation and other results are quoted from works[25, 31, 56]. The best/runner-up results are highlighted in **bold**/underline.

Method	MVTecAD						VisA					
	Image-level			Pixel-level			Image-level			Pixel-level		
	AUROC	AP	F1-max	AUROC	PRO	F1-max	AUROC	AP	F1-max	AUROC	PRO	F1-max
1-Shot ($K = 1$)												
SPADE [7]	81.0	90.6	90.3	91.2	83.9	42.4	79.5	82.0	80.7	95.6	84.1	35.5
PaDiM [11]	76.6	88.1	88.2	89.3	73.3	40.2	88.2	62.8	75.3	89.9	64.3	17.4
PatchCore[38]	83.4	92.2	90.5	92.0	79.7	50.4	79.9	82.8	81.7	95.4	80.5	38.0
WinCLIP [25]	93.1	96.5	**93.7**	95.2	87.1	**55.9**	83.8	85.1	83.1	96.4	85.1	**41.3**
PromptAD [31]	94.6	**97.1**	-	95.9	87.9	-	86.9	88.4	-	96.7	85.1	-
ResAD* [47]	84.8	92.7	91.2	93.4	83.3	48.2	80.9	83.7	81.3	95.9	79.6	37.8
Ours	**96.0**	96.3	93.1	**96.4**	**92.5**	52.8	**87.3**	**88.5**	**85.7**	**97.6**	**91.4**	39.9
2-Shot ($K = 2$)												
SPADE [7]	82.9	91.7	91.1	92.0	85.7	44.5	80.7	82.3	81.7	96.2	85.7	40.5
PaDiM [11]	78.9	89.3	89.2	91.3	78.2	43.7	67.4	71.6	75.7	92.0	70.1	21.1
PatchCore [38]	86.3	93.8	92.0	93.3	82.3	53.0	81.6	84.8	82.5	96.1	82.6	41.0
WinCLIP [25]	94.4	97.0	**94.4**	96.0	88.4	**58.4**	84.6	85.8	83.0	96.8	86.2	**43.5**
PromptAD [31]	95.7	**97.9**	-	96.2	88.5	-	**88.3**	**90.0**	-	97.1	85.8	-
InCTRL [56]	94.0	96.9	-	-	-	-	85.8	87.7	-	-	-	-
ResAD* [47]	87.2	93.9	92.2	94.8	85.5	50.2	86.6	88.3	84.1	96.5	82.3	40.0
Ours	**97.2**	**97.9**	94.3	**96.8**	**92.6**	53.3	88.0	89.1	**85.9**	**97.6**	**91.8**	41.7
4-Shot ($K = 4$)												
SPADE [7]	84.8	92.5	91.5	92.7	87.0	46.2	81.7	83.4	82.1	96.6	87.3	43.6
PaDiM [11]	80.4	90.5	90.2	92.6	81.3	46.1	72.8	75.6	78.0	93.2	72.6	24.6
PatchCore [38]	88.8	94.5	92.6	94.3	84.3	55.0	85.3	87.5	84.3	96.8	84.9	43.9
WinCLIP [25]	95.2	97.3	94.7	96.2	89.0	**59.5**	87.3	88.8	84.2	97.2	87.6	**47.0**
PromptAD [31]	96.6	**98.5**	-	96.5	90.5	-	89.1	90.8	-	97.4	86.2	-
InCTRL [56]	94.5	97.2	-	-	-	-	87.7	90.2	-	-	-	-
ResAD* [47]	90.7	95.7	93.9	95.8	88.7	53.0	89.3	90.7	86.5	96.8	84.1	41.6
Ours	**97.6**	98.0	**95.1**	**97.1**	**92.4**	56.9	**90.3**	**90.8**	**87.9**	**97.8**	**92.0**	43.4

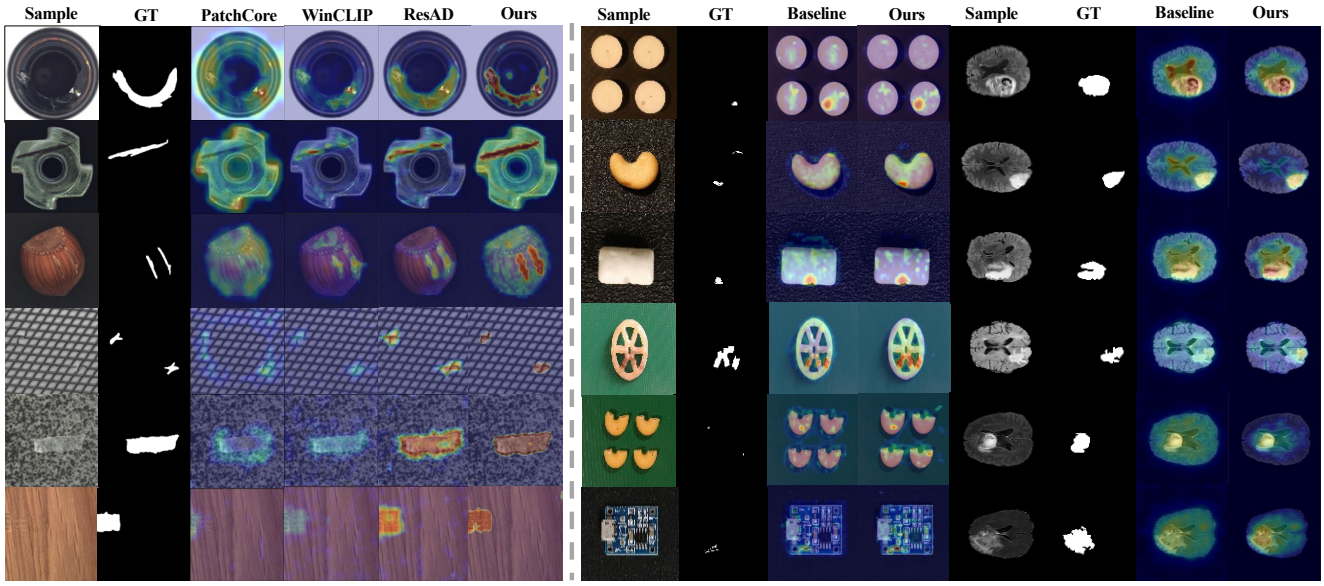
Implementation Details. We employ a lightweight ViT (ViT-Small/14 & DINOv2)[37] as the encoder to ensure low latency. The encoder parameters remain frozen; only the proposed attention modules are updated. Optimization uses AdamW [33] with an initial learning rate of 1×10^{-5}, decaying by a factor of 0.1 at epochs 10 and 15. Training converges within 10 epochs, with 500 episodes per epoch. Input images are resized to 448×448 without augmentation. We set the number of learnable proxies to $M = 25$. Following established protocols [25, 31], the number of normal references is set to $K \in [1, 2, 4, 8]$. The implementation is based on PyTorch, and the experiments are conducted on a single NVIDIA RTX 3090 24GB GPU.

4.2. Comparisons with SOTA Methods

We compare our proposed method with previous state-of-the-art (SOTA) GAD methods on MVTec AD and VisA datasets. Six different image-level and pixel-level metrics are used for quantitative evaluation in Table 1.

MVTec AD. As detailed in Table 1, our method consistently outperforms previous SOTA approaches across various few-shot settings on MVTec AD (training on VisA dataset). In the 4-shot scenario, we achieve superior performance with 97.6 image-level AUROC, 98.0 AP, and 95.1 F1-max, surpassing previous methods such as PromptAD [31] and WinCLIP [25] in key metrics and showing a superior anomaly classification performance. This advantage extends to the 1-shot setting, where our method achieves 96.0 image-level AUROC, significantly outperforming PromptAD [31] and ResAD [47]. Furthermore, our pixel-level metrics, including 92.5 PRO at $K = 1$, demonstrate robust localization capabilities compared to the next best method. Qualitative results in Fig.4a further illustrate our method's ability to effectively reduce false predictions.

VisA. Our method maintains highly competitive performance when training on MVTec AD and testing on the VisA dataset. In the 4-shot setting in Table 1, we achieve 90.3 image-level AUROC, 90.8 AP, and 87.9 F1-max, outperforming ResAD [47] and WinCLIP [25] in image-level



(a) Visualized Examples on MVTec AD **(b) Visualized Examples on VisA and BraTS**

Figure 4. Visualization for anomaly localization results of previous methods and our framework on MVTec AD, VisA, and BraTS datasets.

Table 2. Generalist anomaly detection results with image-level and Pixel-level AUROC on BraTS datasets. Results marked with $*$ are based on our re-implementation. Other results are reported from works [25, 47, 56].

Shot	Method	Image	Pixel	Mean
$K=1$	ResAD* [47]	73.5	91.0	82.3
	Ours	**78.1**	**96.5**	**87.3**
$K=2$	SPADE [7]	58.0	92.8	75.4
	PaDiM [11]	59.4	90.2	74.8
	PatchCore [38]	58.2	93.5	75.9
	RegAD [23]	54.6	81.4	68.0
	WinCLIP [25]	55.9	91.5	73.7
	InCTRL [56]	74.6	-	-
	ResAD* [47]	66.2	91.5	78.9
	Ours	**82.0**	**96.5**	**89.2**
$K=4$	SPADE [7]	66.3	94.8	80.6
	PaDiM [11]	60.6	94.5	77.6
	PatchCore [38]	71.2	95.9	83.6
	RegAD [23]	60.0	87.3	73.7
	WinCLIP [25]	67.3	93.2	80.3
	MVFA* [24]	75.2	92.7	84.0
	InCTRL [56]	76.9	-	-
	ResAD* [47]	74.9	94.3	84.6
	Ours	**85.4**	**97.6**	**91.5**

metrics. These gains are also evident in the 1-shot setting, where we achieve 87.3 image-level AUROC and 88.5 AP, outperforming PromptAD [31] and ResAD [47]. These results confirm the superior performance of our framework to generalize and identify anomalies in diverse target domains. Visualization examples for VisA are provided in Fig.4b.

BraTS. To evaluate robust transferability, we trained on the industrial MVTec AD dataset and tested on the medical BraTS dataset. As shown in Table 2, our method achieves superior performance across all few-shot settings. In the 1-shot scenario, we obtain a mean performance of 87.3, significantly surpassing ResAD [47]. This advantage is amplified in the 4-shot setting, where we achieve an image-level AUROC of 85.4 and a pixel-level AUROC of 97.6, substantially outperforming ResAD [47] and WinCLIP [25]. These results demonstrate robust transferability from industrial to medical imaging domains. Visualization of anomaly localization can be found in Fig.4b.

Efficiency Comparison. As shown in Table 3, compared to previous methods [47, 56], our method outperforms them on three efficiency metrics. Unlike ResAD [47], which spends a lot of training time on fitting the normal residual distribution, the learnable parameters of our framework only involve two attention modules, significantly reducing training time. InCTRL [56] involves modal alignment learning (text-visual), and the entire framework adopts multiple mappings between text and visual spaces, which requires more parameters and also leads to a significant decrease in inference speed.

4.3. Ablation Study

In this section, we investigate the contributions of the main components and the impact of different settings on the proposed RFL and NLS modules.

Table 3. Efficiency comparison between our method and previous works on MVTec AD. Total parameters, FPS, and training time are reported in this table.

Method	Total Parameters (M)	Training Time (H)	Inference Speed (FPS)
InCTRL	117.5	0.7	1.2
ResAD	59.2	20.6	7.8
Ours	**24.4**	**0.5**	**15.8**

Table 4. Ablation study of our method on MVTec AD. RFL: residual feature learning. NLS: normality learning from support.

Variant	RFL	NLS	I-AUROC	P-AUROC
A	–	–	93.2	94.5
B	✓	–	90.4	91.6
C	–	✓	93.8	95.2
D	✓	✓	**96.0**	**96.4**

Table 5. Ablation study of our method on VisA. RFL: residual feature learning. NLS: normality learning from support.

Variant	RFL	NLS	I-AUROC	P-AUROC
A	–	–	82.3	95.5
B	✓	–	80.1	93.2
C	–	✓	83.7	96.2
D	✓	✓	**87.3**	**97.6**

Table 6. The impact of different *Query*, *Key*, *Value* in RFL and NLS modules. All experiments are conducted on MVTec AD, and image-level and pixel-level AUROC are reported.

	$\mathbf{Q}_1$	$\mathbf{K}_1$	$\mathbf{V}_1$	$\mathbf{Q}_2$	$\mathbf{K}_2$	$\mathbf{V}_2$	I	P
i	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{R}$	$\widehat{\mathcal{P}}$	$\mathcal{R}$	$\bar{\mathcal{F}}^s$	**96.0**	**96.4**
ii	$\mathcal{P}$	$\mathcal{R}$	$\bar{\mathcal{F}}^s$	$\widehat{\mathcal{P}}$	$\bar{\mathcal{F}}^s$	$\mathcal{R}$	92.7	93.9
iii	$\mathcal{P}$	$\bar{\mathcal{F}}^s$	$\mathcal{R}$	$\widehat{\mathcal{P}}$	$\mathcal{R}$	$\bar{\mathcal{F}}^s$	95.5	96.1
iv	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{R}$	$\widehat{\mathcal{P}}$	$\bar{\mathcal{F}}^s$	$\bar{\mathcal{F}}^s$	94.4	95.9

Effectiveness of Main Components. Table 4 and 5 demonstrate the performances of four model variants on MVTec AD and VisA datasets, respectively: **A**. Our baseline model only adopts the NN algorithm to generate $\mathcal{S}_2$ as the final anomaly score; **B**. A RFL-only variant leverages $\widehat{\mathcal{P}}$ to hunt normality within the query directly, i.e., $\widetilde{\mathcal{P}}_m$ in Eq.(8) is replaced by $\widehat{\mathcal{P}}_m$. **C**. A NLS-only variant adopts initialized $\mathcal{P}$ as *Query* in Eq.(6). **D**. Our entire framework consists of RFL and NLS modules. The baseline model (**Variant A**) achieves 93.2 and 82.3 image-level AUROC on MVTec AD and VisA, respectively. Interestingly, employing only the RFL module (**Variant B**) to directly hunt for normality results in a performance drop compared to the baseline. This confirms our early hypothesis that directly utilizing residual features or their proxies is risky due to the inconsistency between residuals and instance features. Conversely, the NLS-only model (**Variant C**) brings moderate improvements over the baseline. This is because randomly initialized proxies can also learn some normality under supervised training, even without residual guidance. However, the most significant performance gains are observed when both modules are integrated (**Variant D**). This demonstrates that RFL effectively captures diverse residual patterns, which then serve as necessary guidance for NLS to accurately learn instance-level normality, validating the synergy between these two components.

Impact of Different Settings in RFL and NLS. We further explore the impact of different *Query*, *Key*, and *Value* assignments within our attention modules, as detailed in Table 6. The default configuration (Row **i**), where RFL uses learnable proxies $\mathcal{P}$ to query residuals $\mathcal{R}$, and NLS uses the resulting residual proxies $\widehat{\mathcal{P}}$ to query residuals $\mathcal{R}$ while aggregating support features $\bar{\mathcal{F}}^s$, achieves the best performance. Modifying these settings leads to noticeable performance degradation. **Variant ii** mainly modifies the NLS module by setting its $\mathbf{V}_2$ to $\mathcal{R}$ instead of the mean support features. This results in a significant performance drop to 92.7 and 93.9. This confirms our hypothesis that the NLS module must learn normality patterns by aggregating information from the instance-level support features rather than from the residual space. As shown in **Variant iii**, we replace $\mathcal{R}$ with ($\bar{\mathcal{F}}^s$) for $\mathbf{K}_1$ in RFL, which may interfere with the learning of residual patterns, degrading performance. **Variant iv** modifies the NLS module by changing its $\mathbf{K}_2$ to $\bar{\mathcal{F}}^s$, where normal instance pattern learning is no longer guided by residuals, declining performance.

5. Conclusion

In this paper, we addressed the challenges of Generalist Anomaly Detection (GAD), specifically identifying the significant risks associated with directly modeling residual feature distributions due to their inconsistency with instance features. To mitigate these risks, we proposed a novel framework that re-purposes residuals as guidance to learn instance-level normality patterns from the support set. Our approach mainly consists of two modules: Residual Feature Learning (RFL), which captures diverse patterns from residual features, and Normality Learning from Support (NLS), which leverages these patterns to generate dynamic normality proxies. By employing these proxies to hunt for normality within query samples, our method achieves robust generalization to new domains. Comprehensive experiments on GAD benchmarks demonstrate that our method achieves competitive performance across various few-shot settings.

6. Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62471405, 62331003, 62301451), Basic Research Program of Jiangsu (BK20241814), Suzhou Basic Research Program (SYG202316) and XJTLU REF-22-01-010.

References

[1] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. MVTec AD – a comprehensive real-world dataset for unsupervised anomaly detection. In *CVPR*, 2019. 5

[2] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. In *CVPR*, 2020. 5

[3] Tri Cao, Jiawen Zhu, and Guansong Pang. Anomaly detection under distribution shift. In *ICCV*, 2023. 2

[4] Yunkang Cao, Jiangning Zhang, Luca Frittoli, Yuqi Cheng, Weiming Shen, and Giacomo Boracchi. AdaCLIP: Adapting CLIP with hybrid learnable prompts for zero-shot anomaly detection. In *ECCV*, 2024. 2

[5] Qiyu Chen, Huiyuan Luo, Chengkan Lv, and Zhengtao Zhang. A unified anomaly synthesis strategy with gradient ascent for industrial anomaly detection and localization. *ECCV*, 2024. 2

[6] Yuanhong Chen, Yu Tian, Guansong Pang, and Gustavo Carneiro. Deep one-class classification via interpolated gaussian descriptor. In *AAAI*, 2022. 2

[7] Niv Cohen and Yedid Hoshen. Sub-image anomaly detection with deep pyramid correspondences. *ArXiv*, abs/2005.02357, 2020. 6, 7

[8] Alex Costanzino, Pierluigi Zama Ramirez, Giuseppe Lisanti, and Luigi Di Stefano. Multimodal industrial anomaly detection by crossmodal feature mapping. In *CVPR*, 2024. 2

[9] Zhewei Dai, Shilei Zeng, Haotian Liu, Xurui Li, Feng Xue, and Yu Zhou. Seas: few-shot industrial anomaly image generation with separation and sharing fine-tuning. In *ICCV*, 2025. 2

[10] Simon Damm, Mike Laszkiewicz, Johannes Lederer, and Asja Fischer. AnomalyDINO: Boosting patch-based few-shot anomaly detection with dinov2. *arXiv*, 2024. 4, 5

[11] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. PaDiM: A patch distribution modeling framework for anomaly detection and localization. In *ICPR*, 2021. 1, 5, 6, 7

[12] Hanqiu Deng and Xingyu Li. Anomaly detection via reverse distillation from one-class embedding. In *CVPR*, 2022. 1

[13] Choubo Ding, Guansong Pang, and Chunhua Shen. Catching both gray and black swans: Open-set supervised anomaly detection. In *CVPR*, 2022. 1, 2

[14] Zheng Fang, Xiaoyang Wang, Haocheng Li, Jiejie Liu, Qiugui Hu, and Jimin Xiao. FastRecon: Few-shot industrial anomaly detection via fast feature reconstruction. In *ICCV*, 2023. 2

[15] Bin-Bin Gao. Learning to detect multi-class anomalies with just one normal image prompt. In *ECCV*, 2024. 2

[16] Zhaopeng Gu, Bingke Zhu, Guibo Zhu, Yingying Chen, Ming Tang, and Jinqiao Wang. AnomalyGPT: Detecting industrial anomalies using large vision-language models. In *AAAI*, 2024. 2

[17] Zhaopeng Gu, Bingke Zhu, Guibo Zhu, Yingying Chen, Ming Tang, and Jinqiao Wang. Univad: A training-free unified model for few-shot visual anomaly detection. In *CVPR*, 2025. 2

[18] Denis Gudovskiy, Shun Ishizaka, and Kazuki Kozuka. CFLOW-AD: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. In *WACV*, 2022. 1

[19] Jia Guo, Shuai Lu, Lize Jia, Weihang Zhang, and Huiqi Li. Recontrast: Domain-specific anomaly detection via contrastive reconstruction. In *NeurIPS*, 2023. 2

[20] Jia Guo, Shuai Lu, Weihang Zhang, Fang Chen, Huiqi Li, and Hongen Liao. Dinomaly: The less is more philosophy in multi-class unsupervised anomaly detection. In *CVPR*, 2025.

[21] Haoyang He, Jiangning Zhang, Hongxu Chen, Xuhai Chen, Zhishan Li, Xu Chen, Yabiao Wang, Chengjie Wang, and Lei Xie. A diffusion-based framework for multi-class anomaly detection. In *AAAI*, 2024. 2

[22] Kelei He, Chen Gan, Zhuoyuan Li, Islem Rekik, Zihao Yin, Wen Ji, Yang Gao, Qian Wang, Junfeng Zhang, and Dinggang Shen. Transformers in medical image analysis. *Intelligent Medicine*, 3(1):59–78, 2023. 1

[23] Chaoqin Huang, Haoyan Guan, Aofan Jiang, Ya Zhang, Michael Spratling, and Yan-Feng Wang. Registration based few-shot anomaly detection. In *ECCV*, pages 303–319, 2022. 2, 3, 7

[24] Chaoqin Huang, Aofan Jiang, Jinghao Feng, Ya Zhang, Xinchao Wang, and Yanfeng Wang. Adapting visual-language models for generalizable anomaly detection in medical images. In *CVPR*, 2024. 7

[25] Jongheon Jeong, Yang Zou, Taewan Kim, Dongqing Zhang, Avinash Ravichandran, and Onkar Dabeer. WinCLIP: Zero-/few-shot anomaly classification and segmentation. In *CVPR*, 2023. 5, 6, 7

[26] Xi Jiang, Jianlin Liu, Jinbao Wang, Qiang Nie, Kai Wu, Yong Liu, Chengjie Wang, and Feng Zheng. SoftPatch: Unsupervised anomaly detection with noisy data. In *NeurIPS*, 2022. 2

[27] Mingyu Lee and Jongwon Choi. Text-guided variational image generation for industrial anomaly detection and segmentation. In *CVPR*, 2024. 2

[28] Jiarui Lei, Xiaobo Hu, Yue Wang, and Dong Liu. PyramidFlow: High-resolution defect contrastive localization using pyramid normalizing flow. In *CVPR*, 2023. 2

[29] Wenqiao Li, Xiaohao Xu, Yao Gu, Bozhong Zheng, Shenghua Gao, and Yingna Wu. Towards scalable 3d anomaly detection and localization: A benchmark via 3d anomaly synthesis and a self-supervised learning network. In *CVPR*, 2024. 2, 3

[30] Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. Dice loss for data-imbalanced nlp tasks. In *ACL*, 2020. 5

[31] Xiaofan Li, Zhizhong Zhang, Xin Tan, Chengwei Chen, Yanyun Qu, Yuan Xie, and Lizhuang Ma. PromptAD: Learning prompts with only normal samples for few-shot anomaly detection. In *CVPR*, 2024. 2, 6, 7

[32] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 5

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 6

[34] Wei Luo, Yunkang Cao, Haiming Yao, Xiaotian Zhang, Jianan Lou, Yuqi Cheng, Weiming Shen, and Wenyong Yu. Exploring intrinsic normal prototypes within a single image for universal anomaly detection. In *CVPR*, 2025. 2

[35] Wenxin Ma, Xu Zhang, Qingsong Yao, Fenghe Tang, Chenxu Wu, Yingtai Li, Rui Yan, Zihang Jiang, and S Kevin Zhou. Aa-clip: Enhancing zero-shot anomaly detection via anomaly-aware clip. In *CVPR*, 2025. 2

[36] Bjoern H Menze, Andras Jakab, Stefan Bauer, Jayashree Kalpathy-Cramer, Keyvan Farahani, Justin Kirby, Yuliya Burren, Nicole Porz, Johannes Slotboom, Roland Wiest, et al. The multimodal brain tumor image segmentation benchmark (brats). *IEEE transactions on medical imaging*, 34(10):1993–2024, 2014. 5

[37] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. 6

[38] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *CVPR*, 2022. 1, 2, 5, 6, 7

[39] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *ICML*, 2018. 2

[40] Tran Dinh Tien, Anh Tuan Nguyen, Nguyen Hoang Tran, Ta Duc Huy, Soan Duong, Chanh D Tr Nguyen, and Steven QH Truong. Revisiting reverse distillation for anomaly detection. In *CVPR*, 2023. 2

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 4

[42] Xiaolei Wang, Tianhong Dai, Huihui Bai, Yao Zhao, and Jimin Xiao. Unifying reconstruction and density estimation via invertible contraction mapping in one-class classification. In *NeurIPS*, 2025. 2

[43] Xiaolei Wang, Xiaoyang Wang, Huihui Bai, Eng Gee Lim, and Jimin Xiao. Cnc: Cross-modal normality constraint for unsupervised multi-class anomaly detection. In *AAAI*, 2025. 2

[44] Xiaolei Wang, Xiaoyang Wang, Huihui Bai, Eng Gee Lim, and Jimin Xiao. Decad: Decoupling anomalies in latent space for multi-class unsupervised anomaly detection. In *ICCV*, 2025. 1, 2

[45] Xiaolei Wang, Xiaoyang Wang, Huihui Bai, Eng Gee Lim, and Jimin Xiao. Icc: Intra-cluster contraction for pedestrian anomaly detection under one-class classification setting. *IEEE Transactions on Artificial Intelligence*, 2025. 2

[46] Yuexin Wang, Xiaolei Wang, Yizheng Gong, and Jimin Xiao. Normal-abnormal guided generalist anomaly detection. In *NeurIPS*, 2025. 4

[47] Xincheng Yao, Zixin Chen, Chao Gao, Guangtao Zhai, and Chongyang Zhang. ResAD: A simple framework for class generalizable anomaly detection. In *NeurIPS*, 2024. 1, 3, 5, 6, 7

[48] Xincheng Yao, Yan Luo, Zefeng Qian, and Chongyang Zhang. Adpretrain: Advancing industrial anomaly detection via anomaly representation pretraining. In *NeurIPS*, 2025. 2

[49] Xincheng Yao, Chao Shi, Muming Zhao, Guangtao Zhai, and Chongyang Zhang. Resad++: Towards class agnostic anomaly detection via residual feature learning. *arXiv preprint arXiv:2509.23741*, 2025. 3

[50] Jihun Yi and Sungroh Yoon. Patch SVDD: Patch-level SVDD for anomaly detection and segmentation. In *ACCV*, 2020. 2

[51] Zhiyuan You, Lei Cui, Yujun Shen, Kai Yang, Xin Lu, Yu Zheng, and Xinyi Le. A unified model for multi-class anomaly detection. In *NeurIPS*, 2022. 1

[52] Jiawei Yu, Ye Zheng, Xiang Wang, Wei Li, Yushuang Wu, Rui Zhao, and Liwei Wu. FastFlow: Unsupervised anomaly detection and localization via 2d normalizing flows. *arXiv preprint arXiv:2111.07677*, 2021. 1

[53] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. DRAEM-a discriminatively trained reconstruction embedding for surface anomaly detection. In *ICCV*, 2021. 2

[54] Qiyin Zhong, Xianglin Qiu, Xiaolei Wang, Zhen Zhang, Gang Liu, and Jimin Xiao. Famrd: Frequency-aware multimodal reverse distillation for industrial anomaly detection. In *ACM MM*, 2025. 2

[55] Qihang Zhou, Guansong Pang, Yu Tian, Shibo He, and Jiming Chen. AnomalyCLIP: Object-agnostic prompt learning for zero-shot anomaly detection. In *ICLR*, 2024. 2

[56] Jiawen Zhu and Guansong Pang. Toward generalist anomaly detection via in-context residual learning with few-shot sample prompts. In *CVPR*, 2024. 1, 3, 6, 7

[57] Jiawen Zhu, Choubo Ding, Yu Tian, and Guansong Pang. Anomaly heterogeneity learning for open-set supervised anomaly detection. In *CVPR*, 2024. 2

[58] Yang Zou, Jongheon Jeong, Latha Pemula, Dongqing Zhang, and Onkar Dabeer. Spot-the-difference self-supervised pretraining for anomaly detection and segmentation. In *ECCV*, 2022. 5