Appendix

A. Limitations and Future Work

In this paper, we propose a novel and more discriminative anomaly detection model termed as BGAD to tackle the *insufficient discriminability* issue and the *bias* issue simultaneously. But there are still some limitations to our method. Here, we discuss two main limitations as follows:

One limitation of our model is that we employ normalizing flow to obtain the explicit separating boundary due to its exact log-likelihood estimation ability. However, not all anomaly detection models can generate log-likelihoods, thus our boundary guiding mechanism can't be used in these models directly. A possible solution is that we can use pairwise distances (vector dot product of two features) to substitute log-likelihoods, and then obtain the explicit boundary based on the distribution of normal pairwise distances, and then use BG-SPP loss to optimize the model to learn more discriminative features.

Another limitation is that our method requires anomalous samples to achieve better results, but it is difficult to collect all kinds of anomalies. Thus, generalization performance to unseen anomalies is a critical problem that we should consider, the experimental results in Table 4, 5 and 10 validate our model's generalization capability. However, further improving our model's generalizability and theoretical analysis of model's generalizability are still important and valuable future works. Future work also includes tackling the imbalance problem between normal and abnormal more effectively and attempting to only use pseudo anomalies in our method, such as the generated pseudo anomalies in these works [12, 21, 52].

B. Dataset Details

MVTecAD. The MVTec Anomaly Detection dataset [4] contains 5354 high-resolution images (3629 images for training and 1725 images for testing) of 15 different categories. 5 classes consist of textures and the other 10 classes contain objects. A total of 73 different defect types are presented and almost 1900 defective regions are manually annotated in this dataset.

BTAD. The BeanTech Anomaly Detection dataset [25] contains 2830 real-world images of 3 industrial products. Product 1, 2, and 3 of this dataset contain 400, 1000, and 399 training images respectively.

AITEX. The AITEX [43] is a fabric defect inspection dataset that has 12 defect categories. The original images in this dataset are 4096×256 resolution, we convert this dataset to MVTecAD format following the converting method used in the work [12].

ELPV. The ELPV [11] dataset contains 2642 samples of 300×300 resolution. The dataset is used for solar cell defect inspection and contains two defect categories: mono- and poly-crystalline.

BrainMRI. The BrainMRI is a brain tumor detection dataset obtained by magnetic resonance imaging (MRI) of the brain. The dataset can be downloaded from the Kaggle competition https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection.

HeadCT. The HeadCT is a head hemorrhage detection dataset obtained by a CT scan of the head. The dataset can be downloaded from the Kaggle competition https://www.kaggle.com/datasets/felipekitamura/head-ct-hemorrhage.

C. Implementation Details

As illustrated in Figure 2, we use Efficient-b6 [45] pretrained on ImageNet [54] dataset as the feature extractor to extract three levels of feature maps with $\{4\times, 8\times, 16\times\}$ downsampling ratios. The parameters of the feature extractor are frozen in the training process, only the parameters of the normalizing flow are learnable. The extracted multi-scale features are then transformed to latent space by the normalizing flow, the normalizing flow is constructed by 8 coupling layers similar to [15]. We train BGAD and BGAD$^{w/o}$ using Adam optimizer with $2e-4$ learning rate, 200 train epochs, 32 mini-batch size, and cosine learning rate annealing strategy with 2 warm-up epochs. The normalizer described in sec 3.3 is set to 10 by default, the hyperparameter λ in E.q.(9) is set to 1.0 by default, the default number of transformations in augmentation subset is set as 3. The hyperparameter β is set to 1 by default, and the τ is set to 0.1 by default. With the default hyperparameters, our BGAD can achieve effective performance improvement over NFAD on the six datasets. All the training and test images are resized and cropped to 256×256 resolution from the original resolution. We also utilize a balanced batch sampler to ensure that the ratio of normal and abnormal samples in each mini-batch is 2:1, which can mitigate the rarity problem at the batch level. Our main code is based on the CFLOW implementation made public by the authors of [15] under the MIT license. The pre-trained feature extractor Efficient-b6 is from the timm [54] library under the Apache 2.0 license.

The normalizing flow in our model is mainly based on Real-NVP [14] architecture, but the convolutional subnetwork in Real-NVP is replaced with a linear subnetwork. Our CNFlow also combines many design efforts of various works on normalizing flows from recent works [2, 18, 19]. As in previous works, the normalizing flow in our model is composed of the so-called *coupling layers*. In our CNFlow, each coupling layer is designed to achieve the forward or inverse affine coupling computation [14] as illustrated in Figure 7. And the native coupling layer is followed by ran-

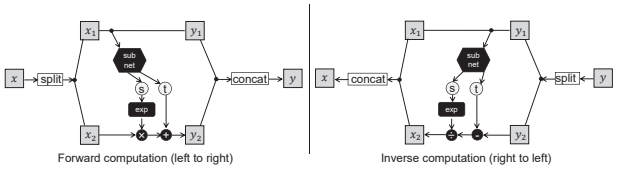

Figure 7. Illustration of a coupling layer. The transformation coefficients are predicted by a subnetwork (*subnet*), which is composed of fully-connected layers, nonlinear activations, batch normalization layers, *etc.*. The forward affine coupling can be calculated as $y_1 = x_1; y_2 = x_2 \odot \exp(s(x_1)) + t(x_1)$, and the inverse affine coupling is then calculated as $x_1 = y_1; x_2 = (y_2 - t(y_1))/\exp(s(y_1))$.

dom and fixed soft permutation of channels [2], and a fixed scaling by a constant, similar to ActNorm layers introduced by [19]. For the coupling coefficients, each subnetwork predicts multiplicative and additive components jointly, as done by [19]. Furthermore, we adopt the soft clamping of multiplication coefficients used by [14]. The implementation of the normalizing flow in our model is based on the FrEIA library https://github.com/VLL-HD/FrEIA, thanks to the authors' contributions, we can implement non-trivial normalizing flow conveniently.

The total parameters of our model are 43M, but the learnable parameters are only 3.6M. Our model can be trained by one GPU card, and the memory usage of our model is only about 2600MB, which means that training our model generally doesn't appear out-of-memory issue. In experiments, we train the model on the MVTecAD dataset with one Titan-XP GPU card, the total training time is about 30 hours, and the inference speed of our model is about 16fps achieved by Titan-XP. Code will be publicly available online.

D. Error Bound Analysis

Proposition 1. *Assume that* $\varphi_{\theta^*} \in \operatorname{argmin}_{\varphi_\theta, \theta \in \Theta}\{\mathcal{L}_{ml} + \lambda \mathcal{L}_{bg-spp}\}$, *and* $y = 0$, $y = 1$ *means normal and abnormal features. Then we have that*

$$\begin{aligned}
&\mathbb{E}_{y_i=0}[\max((b'_n - \log p_i), 0)] + \mathbb{E}_{y_j=1}[\max((\log p_j - b'_a), 0)] \\
&\leq (b_n - b_a)\mathcal{L}_{bg-spp}(\varphi_{\theta^*}) + N/(N+M)[\max(1 + b'_n, -b'_a)] \\
&\leq \frac{(\frac{d}{2}\log(2\pi) - \frac{1}{2})(b_n - b_a)}{\lambda} + N/(N+M) \qquad (10)
\end{aligned}$$

where the $b'_n = b_n - \epsilon, b'_a = b_a + \epsilon, \epsilon \in (0, b_n - b_a)$, N *and* M *are the number of normal and abnormal features.*

proof. The derivation procedure mainly follows the Theorem 3 in [7]. We denote that $\log p_1 \geq \log p_2 \geq \cdots \geq \log p_{N+M}$ is a ranking of the log likelihoods, where N and M are the number of normal and abnormal log likelihoods respectively. Then for $b_n = \log p_N$, we have that

$$\begin{aligned}
&\mathbb{E}_{y_i=0}[\max((b'_n - \log p_i), 0)] + \mathbb{E}_{y_j=1}[\max((\log p_j - b'_a), 0)] \\
&\leq (b'_n - b'_a)\mathcal{L}^0_{bg-spp}(\varphi_{\theta^*}) + N/(N+M)[\max(1 + b'_n, -b'_a)] \\
&\leq (b_n - b_a)\mathcal{L}_{bg-spp}(\varphi_{\theta^*}) + N/(N+M) \qquad (11)
\end{aligned}$$

where the $\mathcal{L}^0_{bg-spp}$ means the ℓ_0 norm based formulation of the BG-SPP loss. The first inequality is obtained by assuming the worst case where $\log p_1, \ldots, \log p_N$ are all misclassified and the others are fallen in (b_a, b_n). The second inequality is obtained as $1 + b'_n \leq 1$ and $-b'_a \leq 1$ when $-1 \leq b'_a < b'_n \leq 0$ satisfies.

Furthermore, for any φ_0 satisfying $\mathcal{L}_{bg-spp}(\varphi_0) = 0$, by the optimality of $\varphi_{\theta*}$, we have that

$$\begin{aligned}
\mathcal{L}_{ml}(\varphi_{\theta^*}) + \lambda\mathcal{L}_{bg-spp}(\varphi_{\theta^*}) &\leq \mathcal{L}_{ml}(\varphi_{\theta^0}) + \lambda\mathcal{L}_{bg-spp}(\varphi_{\theta^0}) \\
&= \mathcal{L}_{ml}(\varphi_{\theta^0}) \qquad (12)
\end{aligned}$$

and thus (the similarity function g in BG-SPP loss is specified as the general exponentiated-cosine distance function for simplifying derivation)

$$\begin{aligned}
&\mathcal{L}_{bg-spp}(\varphi_{\theta^*}) \\
&\leq (\mathcal{L}_{ml}(\varphi_{\theta^0}) - \mathcal{L}_{ml}(\varphi_{\theta^*}))/\lambda \\
&\leq \frac{1}{\lambda}\left(\frac{1}{2}\varphi_{\theta^0}(x)^T\varphi_{\theta^0}(x) - \frac{1}{2}\varphi_{\theta^*}(x)^T\varphi_{\theta^*}(x)\right. \\
&\left. + \frac{1}{2}\varphi_{\theta^*}(x)^T\varphi_{\theta^*}(x) + \frac{d}{2}\log(2\pi)\right) \\
&\leq \frac{1}{\lambda}\left(-\frac{1}{2} + \frac{d}{2}\log(2\pi)\right) \\
&= \frac{\frac{d}{2}\log(2\pi) - \frac{1}{2}}{\lambda} \qquad (13)
\end{aligned}$$

The second inequality is obtained by assuming the worst initial states:

$$\varphi_{\theta^0}(x)^T\varphi_{\theta^0}(x) = -1 \qquad (14)$$

By combining the above E.q.(11) and E.q.(13), we have that

$$\begin{aligned}
&\mathbb{E}_{y_i=0}[\max((b'_n - \log p_i), 0)] + \mathbb{E}_{y_j=1}[\max((\log p_j - b'_a), 0)] \\
&\leq \frac{(\frac{d}{2}\log(2\pi) - \frac{1}{2})(b_n - b_a)}{\lambda} + N/(N+M) \qquad (15)
\end{aligned}$$

The above proposition demonstrates that the necessity and usefulness of the BG-SPP loss, because increasing the hyperparameter λ would assist the error bound in converging to zero. And the proposition also implies that increasing anomalies will benefit the reliability of the normal and abnormal discrimination. We also empirically validate the effect of λ on the detection results in Table 7, the results are evaluated on the hard subsets (described in sec L) from the MVTecAD dataset.

Table 7. AUROC and PRO results on the MVTecAD dataset according to the hyperparameter λ.

λ	1	5	10
Image AUROC	0.983	0.984	0.985
Pixel AUROC	0.984	0.985	0.985
PRO	0.945	0.947	0.950

E. BG-SPP Loss Analysis

We can transform the first part of our BG-SPP loss as follows:

$$\begin{aligned}
&\sum_{i=1}^{N} |\min((\log p_i - b_n), 0)| \\
&\quad = \sum_{i=1}^{N} |\max((-\log p_i + b_n), 0)| \\
&\quad = \sum_{i=1}^{N} |-b_n||\max((\log p_i / b_n - 1), 0)| \\
&\quad = \sum_{i=1}^{N} |b_n|\max((\log p_i / b_n - 1), 0)
\end{aligned} \tag{16}$$

Note that as b_n is negative, we can't transform $|\max((-\log p_i + b_n), 0)|$ to $|b_n||\max((-\log p_i / b_n + 1), 0)|$, which is not correct. If we replace $\log p_i / b_n$ by z, the first part of our BG-SPP loss can be rewritten as $\sum_{i=1}^{N} |b_n|\max((z-1), 0)$, which can be seen as a rescaled hinge loss ($\max(1-z, 0)$) but with opposite optimization direction. The second part of our BG-SPP loss can also be transformed to $\sum_{i=1}^{N} |b_n - \tau|\max((-\log p_i / (b_n - \tau) + 1), 0)$, which can be seen as a rescaled version of the hinge loss.

F. RandAugment-based Pseudo Anomaly Generation

The whole procedure of RandAugment-based Pseudo Anomaly Generation (RPAG) is illustrated in Figure 8. More generated abnormal samples by RPAG are shown in Figure 9.

The advantage of RPAG is that learning from generated samples to recognize irregularities can generalize well to unseen anomalies. The limitation is that RPAG is still not a perfect imitation of real anomalies.

Effect of RandAugment-based Pseudo Anomaly Generation. To show the effectiveness of RandAugment-based Pseudo Anomaly Generation, we show experimental results with or without RPAG in Table 8. It can be found that our method can effectively improve the detection performance even if only five anomalies per category are used. Under the setting of five anomalies, the results can be improved by 0.3%, 0.3%, and 1.0% for image-level AUROC, pixel-level AUROC, and PRO, respectively. Under the setting of ten anomalies, the results can be improved by 0.5%, 0.3%, and 0.7% for image-level AUROC, pixel-level AUROC, and PRO, respectively.

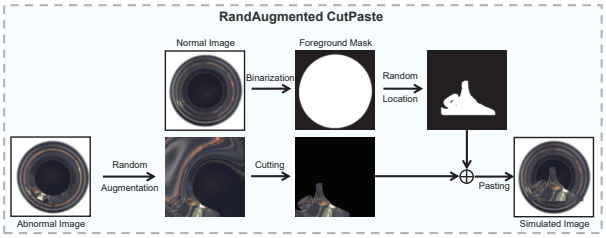

Figure 8. Process to generate abnormal samples.

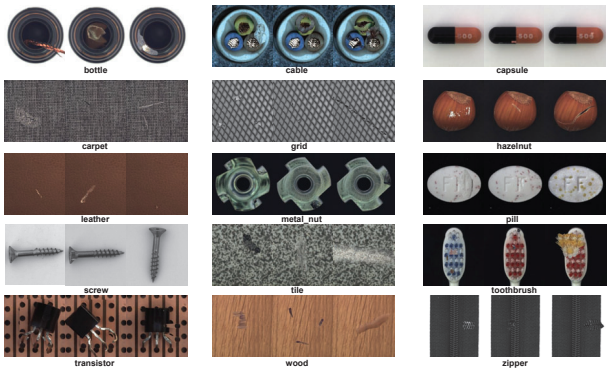

Figure 9. Generated abnormal samples by RPAG. All product categories of the MVTecAD dataset are shown in the Figure.

Comparison with Other Pseudo Anomaly Generation Strategies. In DRAEM [52] and NSA [41], the authors also attempt to utilize synthetic anomalies, we compare our RPAG strategy with their strategies and show results in Table 9. The strategy in [52] is by using texture samples to simulate anomaly regions and by using Perlin noise to capture a variety of anomaly shapes. This strategy may generate invalid anomalies (i.e., anomalies appear in the background) as it will generate anomaly areas in the whole image, while our strategy can ensure to generate more valid anomalies by region limitation. NSA [41] integrates Poisson image editing to seamlessly blend scaled normal patches of various sizes from other normal images. By contrast, the anomalies simulated by our strategy are more realistic than those in [41] as it's based on a few real anomalies. Compared with these anomaly generation strategies, the RPAG is more suitable for our method as it can exploit a few known anomalies more sufficiently.

Table 8. The ablation study on the MVTecAD dataset to verify the effectiveness of RandAugment-based Pseudo Anomaly Generation and Asymmetric Weighting. n: n existing anomalies used in training.

5	10	RPAG	AW	Image AUROC	Pixel AUROC	PRO
				0.968	0.979	0.946
✓				0.982	0.985	0.959
✓		✓		0.985	0.988	0.969
	✓			0.983	0.991	0.969
	✓	✓		0.988	0.992	0.973
	✓	✓	✓	**0.993**	**0.992**	**0.976**

Table 9. Comparison of our RPAG strategy with the pseudo anomaly generation strategies in DRAEM [52] and NSA [41].

Strategy	Image AUROC	Pixel AUROC	PRO
DRAEM [52]	0.970	0.979	0.946
NSA [41]	0.977	0.979	0.946
RPAG	0.988	0.992	0.973

G. Asymmetric Weighting

We further propose Asymmetric Weighting (AW) for the objective function to focus on hard normal features and abnormal features to mitigate the rarity problem.

Weighting for Hard Normal Features. For easy normal features, the weights are assigned as 1. For hard normal features, higher weights should be assigned. Let α_n and γ_n are the normal focusing parameters, we propose Truncated Focal Weighting as follows:

$$Fw_{n_i} = \begin{cases} 1.0, & \text{if} \quad \log p_i > \log p_n; \\ -\alpha_n(1-p_i)^{\gamma_n}\log p_i, & \text{if} \quad \log p_i \leq \log p_n. \end{cases} \tag{17}$$

Weighting for Abnormal Features. Abnormal features with larger log-likelihood can be regarded as hard positives. We propose Reversed Focal Weighting to assign higher weights for abnormal features and much higher weights for hard positives. However, the weighting factors may be less than 1 for abnormal features with smaller log-likelihoods in Reversed Focal Weighting. Therefore, we introduce a truncation term, 1 will be assigned as weights for easy abnormal features. Let α_a and γ_a are the normal focusing parameters, the weighting formula is defined as follows:

$$Fw_{a_j} = \begin{cases} -\alpha_a(1+p_j)^{\gamma_a}\frac{1}{\log p_j}, & \text{if} \quad \log p_j > \log p_a; \\ 1.0, & \text{if} \quad \log p_j \leq \log p_a. \end{cases} \tag{18}$$

Hyperparameter Settings. In E.q.(17), we set $\log p_n = -2$ (features with log-likelihoods larger than -2 can be regarded as easy normal features empirically), and $\alpha_n = 15, \gamma_n = 1$ to make Fw_{n_i} more smooth at $\log p_i = -2$. In E.q.(18), we set $\log p_a = -20$ (features with log-likelihoods less than -20 can be regarded as easy abnormal features empirically), and $\alpha_a = 0.53, \gamma_a = 2$ to make Fw_{a_j} more smooth at $\log p_j = -20$.

Detailed Weighted Learning Objective. The detailed weighted learning objective is formulated as follows:

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{ml} \odot \mathcal{FW}_1 + \lambda \mathcal{L}_{bg-spp} \odot \mathcal{FW}_2 \\ &= \mathbb{E}_{x \in \mathcal{X}^n}\Big[Fw_{n_i} \cdot (\frac{1}{2}\varphi_\theta(x)^T\varphi_\theta(x) \\ &- \sum_{l=1}^{L}\log|\det J_{\varphi_l}(y_{l-1})| + \frac{d}{2}\log(2\pi))\Big] \\ &+ \sum_{i=1}^{|\mathcal{X}^n|} Fw_{n_i} \cdot |\min((\log p_i - b_n), 0)| \\ &+ \sum_{j=1}^{|\mathcal{X}^a|} Fw_{a_j} \cdot |\max((\log p_j - b_n + \tau), 0)| \end{aligned} \tag{19}$$

Effect of Asymmetric Weighting. As shown in Table 8, the detection and localization performance can be further improved by Asymmetric Weighting. The experimental results show that RandAugment-based Pseudo Anomaly Generation and Asymmetric Weighting can mitigate the rarity problem effectively.

H. More Results under the One-Class Setting

In Table 10, we show more results under the one-class setting. All the other results are from [12]. However, in [12], only image-level AUROCs are reported, and the results of some categories shown in Table 10 are missing.

Comparison to Unsupervised Baseline. Our BGAD can outperform the baseline NFAD across all the datasets, especially on the object categories with more complex normal patterns (*e.g.*, Capsule, Screw, Transistor). This shows our model's better generalizability to unseen anomalies.

I. Effect of Semi-Push-Pull Mechanism

In Table 11, we show more comparison results between BGAD† and BGAD. As shown in Table 11, the BGAD† performs less effective than the BGAD, and even worse than the baseline NFAD. The comparison between BGAD and BGAD† shows that the semi-push-pull mechanism in BGAD is critical for mitigating the bias issue.

J. Hyper-parameter Sensitivity

The main tunable hyperparameters of our model are the normal boundary (controlled by β) and the abnormal boundary (controlled by τ). As shown in Table 12, we evaluate different combinations of β (1%, 5%, 10%) and τ (0.1, 0.2, 0.3). From Table 12, we can draw the following main conclusions: 1) β has a more significant effect on performance compared with τ, and pixel-level AUROC is insen-

Table 10. AUC results under the one-class setting, where models are trained with only one anomaly class and tested to detect other anomaly classes. $\cdot/\cdot$ means image-level and pixel-level AUROCs.

Dataset	Known Class	Baseline	Ten Training Anomaly Samples					
		NFAD	DevNet	FLOS	SAOE	MLEP	DRA	BGAD (Ours)
AITEX	Broken_end	0.835/**0.828**	0.658/-	0.585/-	0.712/-	0.732/-	0.693/-	**0.856**/0.824
	Broken_pick	**0.960**/0.982	0.585/-	0.548/-	0.629/-	0.555/-	0.760/-	0.948/**0.983**
	Cut_selvage	0.834/0.830	0.709/-	0.745/-	0.770/-	0.682/-	0.777/-	**0.865/0.844**
	Fuzzyball	0.815/**0.827**	0.734/-	0.550/-	0.842/-	0.677/-	0.701/-	**0.823**/0.825
	Nep	0.834/**0.828**	0.810/-	0.746/-	0.771/-	0.740/-	0.750/-	**0.838**/0.825
	Weft_crack	0.827/**0.692**	0.599/-	0.636/-	0.618/-	0.370/-	0.717/-	**0.841**/0.685
	Mean	0.851/**0.831**	0.683/-	0.635/-	0.724/-	0.626/-	0.733/-	**0.862/0.831**
ELPV	Mono	0.860/-	0.599/-	0.629/-	0.569/-	0.756/-	0.731/-	**0.884**/-
	Poly	0.870/-	0.804/-	0.662/-	0.796/-	0.734/-	0.800/-	**0.880**/-
	Mean	0.865/-	0.702/-	0.646/-	0.683/-	0.745/-	0.766/-	**0.882**/-
Bottle	Broken_large	**1.000**/0.989	-/-	-/-	-/-	-/-	-/-	**1.000/0.991**
	Broken_small	**1.000**/0.987	-/-	-/-	-/-	-/-	-/-	**1.000/0.988**
	Contamination	**1.000**/0.994	-/-	-/-	-/-	-/-	-/-	**1.000/0.996**
	Mean	**1.000**/0.990	-/-	-/-	-/-	-/-	-/-	**1.000/0.992**
Capsule	Crack	0.934/0.990	-/-	-/-	-/-	-/-	-/-	**0.984/0.991**
	Imprint	0.955/0.992	-/-	-/-	-/-	-/-	-/-	**0.990/0.993**
	Poke	0.936/0.989	-/-	-/-	-/-	-/-	-/-	**0.984/0.991**
	Scratch	0.951/0.989	-/-	-/-	-/-	-/-	-/-	**0.994/0.990**
	Squeeze	0.928/0.990	-/-	-/-	-/-	-/-	-/-	**0.988/0.991**
	Mean	0.941/0.990	-/-	-/-	-/-	-/-	-/-	**0.988/0.991**
Grid	Bent	**0.990**/0.994	-/-	-/-	-/-	-/-	-/-	0.989/**0.995**
	Broken	0.982/0.993	-/-	-/-	-/-	-/-	-/-	**0.994/0.994**
	Glue	0.990/0.993	-/-	-/-	-/-	-/-	-/-	**1.000/0.994**
	Metal	0.982/0.993	-/-	-/-	-/-	-/-	-/-	**0.992/0.994**
	Thread	0.982/0.995	-/-	-/-	-/-	-/-	-/-	**0.990/0.996**
	Mean	0.985/0.994	-/-	-/-	-/-	-/-	-/-	**0.993/0.995**
Hazelnut	Crack	**1.000**/0.996	-/-	-/-	-/-	-/-	-/-	**1.000/0.997**
	Cut	**1.000**/0.984	-/-	-/-	-/-	-/-	-/-	**1.000/0.985**
	Hole	0.997/0.981	-/-	-/-	-/-	-/-	-/-	**1.000/0.985**
	Print	0.997/0.981	-/-	-/-	-/-	-/-	-/-	**0.999/0.984**
	Mean	0.998/0.985	-/-	-/-	-/-	-/-	-/-	**1.000/0.988**
Screw	Manipulated	0.887/0.991	-/-	-/-	-/-	-/-	-/-	**0.957/0.993**
	Scratch_head	0.874/0.988	-/-	-/-	-/-	-/-	-/-	**0.927/0.990**
	Scratch_neck	0.861/0.987	-/-	-/-	-/-	-/-	-/-	**0.944/0.989**
	Thread_side	0.927/0.992	-/-	-/-	-/-	-/-	-/-	**0.965/0.994**
	Thread_top	0.878/0.988	-/-	-/-	-/-	-/-	-/-	**0.943/0.989**
	Mean	0.885/0.989	-/-	-/-	-/-	-/-	-/-	**0.947/0.991**
Tile	Crack	0.997/**0.979**	-/-	-/-	-/-	-/-	-/-	**1.000**/0.978
	Glue_strip	0.997/0.967	-/-	-/-	-/-	-/-	-/-	**1.000/0.979**
	Gray_stroke	0.999/0.963	-/-	-/-	-/-	-/-	-/-	**1.000/0.969**
	Oil	0.997/0.959	-/-	-/-	-/-	-/-	-/-	**1.000/0.967**
	Rough	0.998/0.977	-/-	-/-	-/-	-/-	-/-	**1.000/0.987**
	Mean	0.998/0.969	-/-	-/-	-/-	-/-	-/-	**1.000/0.976**
Transistor	Bent_lead	0.979/**0.925**	-/-	-/-	-/-	-/-	-/-	**0.988**/0.921
	Cut_lead	0.982/0.927	-/-	-/-	-/-	-/-	-/-	**1.000/0.935**
	Damaged	0.985/**0.921**	-/-	-/-	-/-	-/-	-/-	**0.989**/0.919
	Misplaced	0.991/0.945	-/-	-/-	-/-	-/-	-/-	**1.000/0.994**
	Mean	0.984/0.929	-/-	-/-	-/-	-/-	-/-	**0.994/0.942**
Wood	Color	**0.995**/0.968	-/-	-/-	-/-	-/-	-/-	0.994/**0.974**
	Combined	**0.994**/0.970	-/-	-/-	-/-	-/-	-/-	**0.994/0.977**
	Hole	0.994/0.968	-/-	-/-	-/-	-/-	-/-	**0.999/0.973**
	Liquid	**0.994**/0.967	-/-	-/-	-/-	-/-	-/-	0.992/**0.970**
	Scratch	**1.000**/0.986	-/-	-/-	-/-	-/-	-/-	0.999/**0.988**
	Mean	**0.995**/0.972	-/-	-/-	-/-	-/-	-/-	**0.995/0.976**

Table 11. AUROC results under the one-class setting. $\cdot/\cdot$ means image-level and pixel-level AUROCs.

Method	Category				
	Bottle	hazelnut	Grid	Tile	Wood
NFAD (baseline)	**1.000**/0.990	0.998/0.986	0.985/0.994	0.998/0.969	**0.995**/0.972
BGAD†	**1.000**/0.983	**1.000**/0.978	0.987/0.988	0.999/0.967	0.994/0.969
BGAD (Ours)	**1.000/0.992**	**1.000/0.988**	**0.993/0.995**	**1.000/0.976**	**0.995/0.976**

sitive to the hyperparameters. 2) Our model is not very sensitive to the margin τ, which means our model can achieve superior results as long as a certain margin is formed between normal and abnormal.

K. Learning Efficiency

In addition to the improvement of detection results, our method can also achieve significant improvement in learning efficiency. To illustrate the learning efficiency, we show AUROC vs epoch curve in Figure 10, specifically, the pixel-level AUROC with a few abnormal samples (FAS) converges rapidly compared to its counterparts. The AUROC can increase a large margin generally only a meta epoch (8 epochs) after adding BG-SPP loss for optimization.

Table 12. AUROC and PRO results on the MVTecAD dataset according to hyperparameter β and τ. $\cdot/\cdot/\cdot$ means mean image-level AUROC, mean pixel-level AUROC and mean PRO.

β \ τ	0.1	0.2	0.3
1%	**0.9936**/0.9920/0.9749	0.9935/0.9920/0.9752	0.9930/0.9918/0.9748
5%	0.9916/0.9922/0.9759	0.9918/**0.9923**/0.9763	0.9922/0.9921/0.9762
10%	0.9915/0.9922/0.9759	0.9920/0.9922/**0.9764**	0.9925/0.9922/0.9762

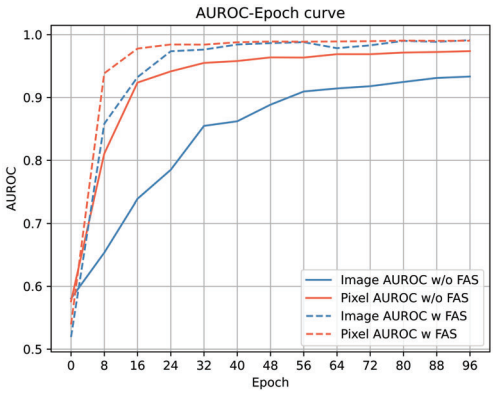

Figure 10. AUROC vs epoch curve of cable category on the MVTecAD dataset.

L. Details of Hard and Unseen Subset Selection

In order to thoroughly verify the effectiveness of our method, we further construct two more difficult subsets from the MVTecAD dataset. The first subset is constructed to evaluate the detection performance, thus we select the subset based on the image-level AUROC. Specifically, we select the first subset based on the misclassification at the image level, *i.e.* anomaly categories are selected if several samples of these categories are detected as normal. The second subset is constructed to evaluate the localization performance, thus we select the subset based on the pixel-level AUROC. Specifically, anomaly categories are selected if their pixel-level AUROCs are the lowest among all anomaly categories. The constructed subsets are shown in Table 13 (Note: As there is only one anomaly class in the toothbrush category, for simplicity, we set the easy and hard subset as the same). In order to verify the generalization capability of our model, we only use the easy subsets as the training set and validate results on the hard subsets. Thus, the hard subsets are utilized as the unseen subsets for generalization capability evaluation.

Table 13. Two hard subsets constructed from the MVTecAD dataset.

		First Subset		Second Subset	
	Category	Easy Anomaly Categories	Hard Anomaly Categories	Easy Anomaly Categories	Hard Anomaly Categories
Textures	Carpet	color, cut, hole, metal_contamination	thread	color, cut, hole, metal_contamination	thread
	Grid	broken, metal_contamination, thread	glue, bent	bent, broken, glue, metal_contamination	thread
	Leather	color, fold, glue, poke	cut	color, cut, glue, poke	fold
	Tile	crack, glue_strip, gray_stroke, oil	rough	crack, glue_strip, gray_stroke, oil	rough
	Wood	color, combined, hole, liquid	scratch	color, combined, hole, liquid	scratch
Objects	Bottle	broken_large, broken_small	contamination	broken_large, broken_small	contamination
	Cable	bent_wire, combined, cut_inner_insulation,cut_outer_insulation,missing_cable	cable_swap, missing_wire, poke_insulation	bent_wire, cable_swap, combined, cut_inner_insulation, missing_cable, missing_wire, poke_insulation	cut_outer_insulation
	Capsule	crack, squeeze	faulty_imprint, poke, scratch	crack, faulty_imprint, poke, scratch	squeeze
	Hazelnut	crack, hole, print	cut	cut, hole, print	crack
	Metal nut	bent, color, flip	scratch	bent, color, scratch	flip
	Pill	color, combined, contamination, faulty_imprint, pill_type	crack, scratch	color, combined, contamination, crack, faulty_imprint, scratch	pill_type
	Screw	scratch_head, scratch_neck, thread_top	manipulated_front, thread_side	manipulated_front, scratch_head, scratch_neck, thread_top	thread_side
	Toothbrush	defective	defective	defective	defective
	Transistor	bent_lead, cut_lead, misplaced	damaged_case	bent_lead, damaged_case, misplaced	cut_lead
	Zipper	broken_teeth, combined, fabric_border, rough, split_teeth	fabric_interior, squeezed_teeth	broken_teeth, combined, fabric_border, fabric_interior, split_teeth, squeezed_teeth	rough