Supplementaries of 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]

Appendix

In the appendix, please note that some equations, tables and figures are referred to the corresponding contents in the main text.

A More Discussions

A.1 Necessities and Motivations

We further explain the necessities and motivations of the individual components in our method. Through the explanations in this subsection, readers can have a clearer understanding of how the overall training objective is designed and how the individual losses are motivated and connected together.

Inter-class Gaussian mixture modeling. The initial motivation of our method is that NF-based AD methods usually fall into the "homogeneous mapping" issue when applied to the unified AD task. 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. The most natural method is to model multiple Gaussian distributions in the latent space. However, the fixed multiple Gaussian distribution centers still result in a relatively fixed whole distribution in the latent space, lacking adaptability. Thus, the inter-class Gaussian mixture modeling is proposed to increase the adaptability of latent distribution for better fitting the complex multi-class normal distribution.

Mutual information maximization. The loss function for the inter-class Gaussian mixture modeling is in Eq. (6), where the logsumexp operator will sum the exp values of all classes, this means that the Eq. (6) only has the drawing characteristic to ensure the latent features are drawn together to the

* Corresponding Author.

whole distribution. As the class centers are randomly initialized, this may cause different class centers to collapse into the same center. Therefore, We need to further introduce class repulsion property. Then, from the mutual information perspective, we propose the mutual information maximization loss for increasing class separating ability. Furthermore, we find that using Entropy as anomaly measurement is beneficial to achieve better results (see Tab. 3c). And minimizing the inter-class entropy can also introduce the class repulsion property. Moreover, directly using entropy as an optimization item is also beneficial for the effect of entropy-based measurement. Thus, we introduce entropy loss in Eq. (9) as a regularizer item.

Learning intra-class mixed class centers. Finally, we consider that in real-world scenarios, even one object class may contain diverse normal patterns. We think modeling intra-class distribution by mixture Gaussian prior should also be beneficial for the results (see Tab. 3a). Moreover, as we further explain in Sec. A.4, another consideration is to guarantee the effectiveness of anomaly determination. The inter-class Gaussian mixture modeling can't effectively guarantee the anomalies that fall into the inter-class Gaussian mixture distribution to be correctly recognized. To this end, we further model the intra-class Gaussian mixture distribution for each class to ensure that the normal distribution of each class still remains compact. Therefore, 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.

Therefore, although our method has four parts, each part is not arbitrarily introduced, but rather well motivated to achieve better unified AD performance. These individual losses are logically well-connected together. Our method mainly introduces a new learning objective for the NF-based AD methods, which usually doesn't increase implementation and model complexity. Thus, We think our method should be general and can be applied to various NF-based AD methods to assist them in improving the unified anomaly detection capability.

A.2 Supervision Information

We summarize the training samples and the supervision information required by our method and other methods in Tab. 1.

Table 1: Training samples and supervision information summarization.

PaDiM	MKD	DRAEM	PMAD	UniAD	OmniAL	FastFlow	CFLOW	HGAD (Ours)
N	N	N+P	N	N	N+P	N	N	N
S	S	S	S	w/o S	w/o S	S	S	S

where N means only using normal samples during training, P means also using pseudo (or synthetic) anomalies during training, S means requiring separating different classes and w/o S means not separating different classes.

If we think that using synthetic anomalies introduces anomalous information during training, DRAEM and OmniAL can also be called as supervised or self-supervised, while others are unsupervised. In addition, both UniAD and OmniAL use additional information to simulate anomalies. UniAD adds noise while OmniAL uses synthetic anomalies to learn how to reconstruct anomalies into normal during training. But our method is entirely based on learning normal feature distribution without any additional information (If synthetic anomalies can be used, our method can easily be combined with BGAD [14] to achieve better unified anomaly detection results). However, the methods based on synthetic anomalies may perform much worse when synthetic anomalies cannot simulate real anomalies well. This will result in limited application scenarios for such methods. For example, on the more challenging VisA dataset, our method significantly outperforms OmniAL (97.1/98.9 vs. 87.8/96.6). Compared to UniAD, results on multiple datasets, such as MVTec3D-RGB, VisA, and Union datasets, also show significant improvements (87.1 vs. 77.5, 97.1 vs. 92.8, 93.5 vs. 86.9).

Our method can be easily extended to completely unsupervised, as industrial images often have significant differences between different classes. For instance, after extracting global features, we can use a simple unsupervised clustering algorithm to divide each image into a specific class. Or we can only require few-shot samples for each class as a reference, and then compute the feature distances between each input sample to these reference samples. In this way, we can also conveniently divide each sample into the most relevant class.

A.3 More Discussions with "identical shortcut"

The "identical shortcut" is essentially caused by the leakage of abnormal information. The process of reconstruction is to remove abnormal information in the input, resulting in the failure of reconstruction in abnormal regions. But if the abnormal features are leaked into the output, this will result in the reconstruction network directly returning a copy of the input as output. This issue usually can be addressed by masking, such as the neighbor masking mechanism in UniAD [16]. However, the "homogeneous mapping" is a specific issue in normalizing flow (NF) based AD methods. In previous NF-based AD methods, the latent feature space is uni-modal. When used for unified anomaly detection, we need to map different class features to the single latent center, this may cause the model more prone to take a bias to map different input features to similar latent features. Thus, with the bias, the log-likelihoods of abnormal features will become closer to the log-likelihoods of normal features, causing normal misdetection or abnormal missing detection. We call this phenomenon as the "homogeneous mapping" issue, rather than casually introducing it. Moreover, as analyzed in Sec. 3.2, we provide a reasonable explanation from the perspective of the formula in normalizing flow. To address this issue, we propose the hierarchical Gaussian Mixture modeling approach, the key designs in our method are completely different from those in

UniAD. As the causes and solutions of the two issues are significantly different, "homogeneous mapping" is not intrinsically equal to "identical shortcut".

A.4 The Way to Guarantee Anomalies Out-of-Distribution.

Here, we further explain how to guarantee that anomalies are out-of-distribution. In our method, increasing inter-class distances is to ensure that the latent space has sufficient capacity to accommodate the features of multiple classes. In addition, we also model the intra-class Gaussian mixture distribution for each class to ensure that the normal distribution of each class still remains compact. Therefore, 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 Gaussian mixture distributions. As described in Anomaly Scoring section (sec. 3.4), we can guarantee that anomalies are recognized as out-of-distribution by combining intra-class log-likelihood and inter-class entropy to measure anomalies. Because only if the anomaly is out-of-distribution, the anomaly score based on the association of log-likelihood and entropy will be high, and the detection metrics can be better. The visualization results (decision-level results based on log-likelihood) in Fig. 2 and 1 also intuitively show that our method has fewer normal-abnormal overlaps and the normal boundary is more compact.

A.5 Limitations

In this paper, we propose a novel HGAD to accomplish the unified anomaly detection task. Even if our method manifests good unified AD performance, there are still some limitations of our work. Here, we discuss two main limitations as follows:

One limitation is that our method mainly targets NF-based AD methods to improve their unified AD abilities. To this end, our method cannot be directly utilized to the other types of anomaly detection methods, such as reconstruction-based, OCC-based, embedding-based, and distillation-based approaches (see Related Work, Sec. 2). However, we believe that the other types of anomaly detection methods can also be improved into unified AD methods, but we need to find and solve the corresponding issues in the improvement processes, such as the "identical shortcut" issue [16] in reconstruction-based AD methods. How to upgrade the other types of anomaly detection methods to unified AD methods and how to find a general approach for unified anomaly detection modeling will be the future works.

In this work, our method is mainly aimed at solving unified anomaly detection, it doesn't have the ability to directly generalize to unseen classes. Because, in our method, the new class features usually do not match the learned known multi-class feature distribution, which can lead to normal samples being misrecognized as anomalies. Generalization to unseen classes can be defined as class-agnostic anomaly detection [15], where the model is trained with normal instances from multiple known classes with the objective to detect anomalies from unseen

classes. In the practical industrial scenarios, models with class-agnostic anomaly detection capabilities are very valuable and necessary, because new products will continuously appear and it's cost-ineffective and inconvenient to retrain models for new products. We think our method should achieve better performance on unseen classes than previous NF-based methods due to the ability to learn more complex multi-class distribution, but it's far from solving the problem. How to design a general approach for class-agnostic anomaly detection modeling will be the future works.

A.6 Model Complexity

With the image size fixed as 256×256, we compare the FLOPs and learnable parameters with all competitors. In Tab. 2, we can conclude that the advantage of HGAD does not come from a larger model capacity. Compared to UniAD, our method requires fewer epochs (100 vs. 1000) and has a shorter training time.

Table 2: Complexity comparison between our HGAD and other baseline methods.

	PaDiM	MKD	DRAEM	PMAD	UniAD	FastFlow	CFLOW	HGAD (Ours)
FLOPs	14.9G	24.1G	198.7G	52G	9.7G	36.2G	30.7G	32.8G
Learnable Parameters	/	24.9M	97.4M	163.4M	9.4M	69.8M	24.7M	30.8M
Inference Speed	12.8fps	23fps	22fps	10.8fps	29fps	42.7fps	24.6fps	24.3fps
Training Epochs	/	50	700	300	1000	400	200	100

A.7 Real-world Applications

In industrial inspection scenarios, the class actually means a type of product on the production line. Unified anomaly detection can be applied to train one model to detect defects in all products, without the need to train one model for each type of product. This can greatly reduce the resource costs of training and deploying. In video surveillance scenarios, we can use one model to simultaneously detect anomalies in multiple camera scenes.

B Social Impacts and Ethics

As a unified model for unified anomaly detection, the proposed method does not suffer from particular ethical concerns or negative social impacts. All datasets used are public. All qualitative visualizations are based on industrial product images, which doesn't infringe personal privacy.

C Implementation Details

Optimization Strategy. In the initial a few epochs, we only optimize with $\mathcal{L}_g$ and $\mathcal{L}_{mi}$ to form distinguishable inter-class main class centers. And then we simultaneously optimize the intra-class delta vectors and the main class centers with the overall loss $\mathcal{L}$ in Eq. (11). In this way, we can better decouple the inter-class and intra-class learning processes. This strategy can make the intra-class learning become much easier, as optimizing after forming distinguishable inter-class main centers will not have the problem that many centers initially overlap with each other.

Model Architecture. The normalizing flow model in our method is mainly based on Real-NVP [6] architecture, but the convolutional subnetwork in Real-NVP is replaced with a two-layer MLP network. As in Real-NVP, the normalizing flow in our model is composed of the so-called coupling layers. All coupling layers have the same architecture, and each coupling layer is designed to tractably achieve the forward or reverse affine coupling transformation [6] (see Eq. (4)). Then each coupling layer is followed by a random and fixed soft permutation of channels [2] and a fixed scaling by a constant, similar to ActNorm layers introduced by [8]. For the coupling coefficients (*i.e.*, $\exp(s(x_1))$ and $t(x_1)$ in Eq. (4)), each subnetwork predicts multiplicative and additive components simultaneously, as done by [6]. Furthermore, we adopt the soft clamping of multiplication coefficients used by [6]. The layer numbers of the normalizing flow models are all 12. We add positional embeddings to each coupling layer, which are concatenated with the first half of the input features (*i.e.*, x_1 in Eq. (4)). Then, the concatenated embeddings are sent into the subnetwork for predicting couping coefficients. The dimension of all positional embeddings is set to 256. The implementation of the normalizing flows in our model is based on the FrEIA library `https://github.com/VLLHD/FrEIA`.

D Datasets

MVTecAD. The MVTecAD [4] dataset is widely used as a standard benchmark for evaluating unsupervised image anomaly detection methods. This dataset contains 5354 high-resolution images (3629 images for training and 1725 images for testing) of 15 different product 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 [9] is an another popular benchmark, which 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.

MVTecAD-3D. The MVTecAD-3D [5] dataset is recently proposed for 3D anomaly detection, which contains 4147 high-resolution 3D point cloud scans paired with 2D RGB images from 10 real-world categories. In this dataset, most anomalies can also be detected only through RGB images. Since we focus on image anomaly detection, we only use RGB images of the MVTecAD-3D dataset. We refer to this subset as MVTec3D-RGB.

VisA. The Visual Anomaly dataset [18] is a recently proposed larger anomaly detection dataset compared to MVTecAD [4]. This dataset contains 10821 images with 9621 normal and 1200 anomalous samples. In addition to images that only contain single instance, the VisA dataset also have images that contain multiple instances. Moreover, some product categories of the VisA dataset, such as Cashew, Chewing gum, Fryum and Pipe fryum, have objects that are roughly aligned. These characteristics make the VisA dataset more challenging than the MVTecAD dataset, whose images only have single instance and are better aligned.

E Detailed Loss Function Derivation

In this section, we provide the detailed derivation of the loss functions proposed in the main text, including $\mathcal{L}_g$ (Eq. (6)), $\mathcal{L}_{mi}$ (Eq. (8)), and $\mathcal{L}_{in}$ (Eq. (10)).

Derivation of $\mathcal{L}_g$. We use a Gaussian mixture model with class-dependent means μ_y and unit covariance $\mathbb{I}$ as the inter-class Gaussian mixture prior, which is defined as follows:

$$p_Z(z) = \sum\nolimits_y p(y)\mathcal{N}(z;\mu_y,\mathbb{I}) \tag{1}$$

Below, we use c_y as a shorthand of $\log p(y)$. Then, we can calculate the log-likelihood as follows:

$$\begin{aligned}
\log p_Z(z) &= \log\big[\sum\nolimits_y p(y)\mathcal{N}(z;\mu_y,\mathbb{I})\big] \\
&= \log\big[\sum\nolimits_y p(y)(2\pi)^{-\frac{d}{2}}\mathrm{e}^{-\frac{1}{2}(z-\mu_y)^T(z-\mu_y)}\big] \\
&= -\frac{d}{2}\log(2\pi) + \log\big(\sum\nolimits_y \mathrm{e}^{c_y}\cdot\mathrm{e}^{-\frac{||z-\mu_y||_2^2}{2}}\big) \\
&= -\frac{d}{2}\log(2\pi) + \log\big(\sum\nolimits_y \mathrm{e}^{-\frac{||z-\mu_y||_2^2}{2}+c_y}\big) \\
&= -\frac{d}{2}\log(2\pi) + \operatorname*{logsumexp}_y\left(-\frac{||z-\mu_y||_2^2}{2}+c_y\right)
\end{aligned} \tag{2}$$

Then, we bring the $\log p_Z(z)$ into Eq. (1) to obtain the log-likelihood $\log p_\theta(x)$ as:

$$\log p_\theta(x) = -\frac{d}{2}\log(2\pi) + \operatorname*{logsumexp}_y\left(-\frac{||\varphi_\theta(x)-\mu_y||_2^2}{2}+c_y\right) + \log|\det J| \tag{3}$$

Further, the maximum likelihood loss in Eq. (2) can be written as:

$$\begin{aligned}
\mathcal{L}_m &= \mathbb{E}_{x\sim p(X)}[-\log p_\theta(x)] \\
&= \mathbb{E}_{x\sim p(X)}\left[-\operatorname*{logsumexp}_y\left(-\frac{||\varphi_\theta(x)-\mu_y||_2^2}{2}+c_y\right) - \log|\det J| + \frac{d}{2}\log(2\pi)\right]
\end{aligned} \tag{4}$$

The loss function $\mathcal{L}_g$ is actually defined as the above maximum likelihood loss $\mathcal{L}_m$ with inter-class Gaussian mixture prior.

Extending $\mathcal{L}_g$ for Learning Intra-Class Mixed Class Centers. When we extend the Gaussian prior $p(Z|y) = \mathcal{N}(\mu_y, \mathbb{I})$ to mixture Gaussian prior $p(Z|y) = \sum_{i=1}^{M} p_i(y)\mathcal{N}(\mu_i^y, \mathbb{I})$, where M is the number of intra-class latent centers, the likelihood of latent feature z can be calculated as follows:

$$p_Z(z) = \sum\nolimits_y p(y)\Big(\sum\nolimits_{i=1}^{M} p_i(y)\mathcal{N}(\mu_i^y, \mathbb{I})\Big) \tag{5}$$

Then, following the derivation in Eq. 2, we have:

$$\log p_Z(z) = \log\Big(\sum\nolimits_y p(y) \operatorname*{sumexp}_i \Big[\frac{-||z-\mu_i^y||_2^2}{2} + c_i^y - \frac{d}{2}\log(2\pi)\Big]\Big) \tag{6}$$

where c_i^y is the shorthand of $\log p_i(y)$. The $\mathcal{L}_g$ for learning intra-class mixed class centers can be defined as:

$$\mathcal{L}_g = \mathbb{E}_{x\sim p(X)}\Big[-\log\Big(\sum\nolimits_y p(y) \operatorname*{sumexp}_i \Big[\frac{-||\varphi_\theta(x)-\mu_i^y||_2^2}{2} + c_i^y - \frac{d}{2}\log(2\pi)\Big]\Big) - \log|\det J|\Big] \tag{7}$$

However, as the initial latent features Z usually have large distances with the intra-class centers $\{\mu_i^y\}_{i=1}^M$, this will cause the value after sumexp operation close to 0. After calculating the logarithm function, it's easy to cause the loss to be numerically ill-defined (NaN). Besides, we find that directly employing Eq. 7 for learning intra-class mixed class centers will lead to much worse results, as we need to simultaneously optimize all intra-class centers of all classes to fit the inter-class Gaussian mixture prior. To this end, we propose to decouple the inter-class Gaussian mixture prior fitting and the intra-class latent centers learning. The loss function of learning intra-class mixed class centers is defined in Eq. 11.

Derivation of $\mathcal{L}_{mi}$. We first derive the general format of the mutual information loss in Eq. (7) as follows:

$$\begin{aligned}
\mathcal{L}_{mi} &= -I(Y,Z) = -H(Y) + H(Y|Z) = -H(Y) - H(Z) + H(Y,Z) \\
&= -H(Y) - \mathbb{E}_{x\sim p(X)}\Big[-\log\big(\sum\nolimits_y p(y)p(\varphi_\theta(x)|y)\big)\Big] \\
&+ \mathbb{E}_{(x,y)\sim p(X,Y)}[-\log(p(y)p(\varphi_\theta(x)|y))] \\
&= -H(Y) - \mathbb{E}_{(x,y)\sim p(X,Y)}\left[\log\frac{p(y)p(\varphi_\theta(x)|y)}{\sum_{y'} p(y')p(\varphi_\theta(x)|y')}\right] \\
&= -\mathbb{E}_{y\sim p(Y)}[-\log p(y)] - \mathbb{E}_{(x,y)\sim p(X,Y)}\left[\log\frac{p(y)p(\varphi_\theta(x)|y)}{\sum_{y'} p(y')p(\varphi_\theta(x)|y')}\right]
\end{aligned} \tag{8}$$

Then, by replacing $p(\varphi_\theta(x)|y)$ with $\mathcal{N}(\varphi_\theta(x); \mu_y, \mathbb{I})$ in the mutual information loss, we can derive the following practical loss format for the second part of Eq.

8. We also use c_y as a shorthand of $\log p(y)$.

$$
\begin{aligned}
&- \mathbb{E}_{(x,y)\sim p(X,Y)}\left[\log \frac{p(y)p(\varphi_\theta(x)|y)}{\sum_{y'} p(y')p(\varphi_\theta(x)|y')}\right] \\
&= -\mathbb{E}_{(x,y)\sim p(X,Y)}\left[\log \frac{p(y)\mathcal{N}(\varphi_\theta(x);\mu_y,\mathbb{I})}{\sum_{y'} p(y')\mathcal{N}(\varphi_\theta(x);\mu_{y'},\mathbb{I})}\right] \\
&= -\mathbb{E}_{(x,y)\sim p(X,Y)}\left[\log \frac{(2\pi)^{-\frac{d}{2}}\mathrm{e}^{-\frac{1}{2}(\varphi_\theta(x)-\mu_y)^T(\varphi_\theta(x)-\mu_y)}\cdot \mathrm{e}^{c_y}}{\sum_{y'}(2\pi)^{-\frac{d}{2}}\mathrm{e}^{-\frac{1}{2}(\varphi_\theta(x)-\mu_{y'})^T(\varphi_\theta(x)-\mu_{y'})}\cdot \mathrm{e}^{c_{y'}}}\right] \\
&= -\mathbb{E}_{(x,y)\sim p(X,Y)}\left[\log \frac{\mathrm{e}^{-\frac{||\varphi_\theta(x)-\mu_y||_2^2}{2}+c_y}}{\sum_{y'}\mathrm{e}^{-\frac{||\varphi_\theta(x)-\mu_{y'}||_2^2}{2}+c_{y'}}}\right] \\
&= -\mathbb{E}_{(x,y)\sim p(X,Y)}\left[\underset{y}{\mathrm{logsoftmax}}\left(-\frac{||\varphi_\theta(x)-\mu_{y'}||_2^2}{2}+c_{y'}\right)\right] \qquad (9)
\end{aligned}
$$

By replacing Eq. 9 back to the Eq. 8, we can obtain the following practical loss format of the mutual information loss.

$$
\begin{aligned}
\mathcal{L}_{mi} &= -\mathbb{E}_{y\sim p(Y)}[-\log p(y)] - \mathbb{E}_{(x,y)\sim p(X,Y)}\left[\underset{y}{\mathrm{logsoftmax}}\left(-\frac{||\varphi_\theta(x)-\mu_{y'}||_2^2}{2}+c_{y'}\right)\right] \\
&= -\mathbb{E}_{y\sim p(Y)}[-c_y] - \mathbb{E}_{(x,y)\sim p(X,Y)}\left[\underset{y}{\mathrm{logsoftmax}}\left(-\frac{||\varphi_\theta(x)-\mu_{y'}||_2^2}{2}+c_{y'}\right)\right] \\
&= -\mathbb{E}_{(x,y)\sim p(X,Y)}\left[\underset{y}{\mathrm{logsoftmax}}\left(-\frac{||\varphi_\theta(x)-\mu_{y'}||_2^2}{2}+c_{y'}\right) - c_y\right] \qquad (10)
\end{aligned}
$$

Intra-Class Mixed Class Centers Learning Loss. The loss function for learning the intra-class class centers is actually the same as the $\mathcal{L}_g$ in Eq. (6). But we note that we need to replace the class centers with the intra-class centers: $\mu_i^y = \{\mu_1^y + \Delta\mu_i^y\}_{i=1}^M$, and the sum operation is performed on all intra-class centers μ_i^y within the corresponding class y. Another difference is that we need to detach the main center μ_1^y from the gradient graph and only optimize the delta vectors. The loss function can be written as:

$$
\mathcal{L}_{in} = \mathbb{E}_{(x,y)\sim p(X,Y)}\left[-\underset{i}{\mathrm{logsumexp}}\left(-\frac{||\varphi_\theta(x)-(SG[\mu_1^y]+\Delta\mu_i^y)||_2^2}{2}+c_i^y\right)-\log|\det J|\right] \qquad (11)
$$

Finally, we note that the use of logsumexp and logsoftmax pytorch operations above is quite important. As the initial $||\varphi_\theta(x)-\mu_y||_2^2/2$ distance values are usually large, if we explicitly perform the exp and then log operations, the values will become too large and the loss will be numerically ill-defined (NaN).

F An Information-Theoretic View

Information theory [11] is an important theoretical foundation for explaining deep learning methods. The well-known *Information Bottleneck principle* [1, 10, 12, 13]

is also rooted from the information theory, which provides an explanation for representation learning as the trade-off between information compression and informativeness retention. Below, we denote the input variable as X, the latent variable as Z, and the class variable as Y. Formally, in this theory, supervised deep learning attempts to minimize the mutual information $I(X, Z)$ between the input X and the latent variable Z while maximizing the mutual information $I(Z, Y)$ between Z and the class Y:

$$\min I(X, Z) - \alpha I(Z, Y) \tag{12}$$

where the hyperparameter $\alpha > 0$ controls the trade-off between compression (*i.e.*, redundant information) and retention (*i.e.*, classification accuracy).

In this section, we will show that our method can be explained by the *Information Bottleneck principle* with the learning objective $\min I(X, Z_{\mathcal{E}}) - \alpha I(Z, Y)$, where $Z_{\mathcal{E}} = \varphi_\theta(X + \mathcal{E})$ and $p(\mathcal{E}) = \mathcal{N}(0, \sigma^2\mathbb{I})$ is Gaussian with mean zero and covariance $\sigma^2\mathbb{I}$. First, we derive $I(X, Z_{\mathcal{E}})$ as follows:

$$\begin{aligned} I(X, Z_{\mathcal{E}}) &= I(Z_{\mathcal{E}}, X) = H(Z_{\mathcal{E}}) - H(Z_{\mathcal{E}}|X) \\ &= \underbrace{\mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}[-\log p(\varphi_\theta(x+\epsilon))]}_{:=A} + \underbrace{\mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}[\log p(\varphi_\theta(x+\epsilon)|x)]}_{:=B} \end{aligned} \tag{13}$$

To approximate the second item (B), we can replace the condition x with $\varphi_\theta(x)$, because φ_θ is bijective and both conditions convey the same information [3].

$$B = \mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}[\log p(\varphi_\theta(x+\epsilon)|x)] = \mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}[\log p(\varphi_\theta(x+\epsilon)|\varphi_\theta(x))] \tag{14}$$

We can linearize $\varphi_\theta(x + \epsilon)$ by its first order Taylor expansion: $\varphi_\theta(x + \epsilon) = \varphi_\theta(x) + J\epsilon + \mathcal{O}(\epsilon^2)$, where the matrix $J = \bigtriangledown_x \varphi_\theta(x)$ is the Jacobian matrix of the bijective transformation ($z = \varphi_\theta(x)$ and $x = \varphi_\theta^{-1}(z)$). Then, we have:

$$\begin{aligned} B &= \mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}[\log p(\varphi_\theta(x) + J\epsilon + \mathcal{O}(\epsilon^2)|\varphi_\theta(x))] \\ &= \mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}[\log p(\varphi_\theta(x) + J\epsilon|\varphi_\theta(x))] + \mathbb{E}_{\epsilon\sim p(\mathcal{E})}[\mathcal{O}(\epsilon^2)] \\ &= \mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}[\log p(\varphi_\theta(x) + J\epsilon|\varphi_\theta(x))] + \mathcal{O}(\sigma^2) \end{aligned} \tag{15}$$

where the $\mathbb{E}_{\epsilon\sim p(\mathcal{E})}[\mathcal{O}(\epsilon^2)]$ is actually the covariance of $p(\mathcal{E}) = \mathcal{N}(0, \sigma^2\mathbb{I})$, thus can be replaced with $\mathcal{O}(\sigma^2)$. Since $p(\mathcal{E})$ is Gaussian with mean zero and covariance $\sigma^2\mathbb{I}$, the conditional distribution is Gaussian with mean $\varphi_\theta(x)$ and covariance

$\sigma^2 JJ^T$. Then, we have:

$$\begin{aligned}
B &= \mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}[\log\mathcal{N}(\varphi_\theta(x)+J\epsilon;\varphi_\theta(x),\sigma^2JJ^T)]+\mathcal{O}(\sigma^2)\\
&= \mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}[\log((2\pi)^{-\frac{d}{2}}\cdot(|\sigma^2JJ^T|)^{-\frac{1}{2}}\cdot \mathrm{e}^{-\frac{1}{2}\frac{1}{\sigma^2}\epsilon^T\epsilon})]+\mathcal{O}(\sigma^2)\\
&= \mathbb{E}_{x\sim p(X)}[-\frac{1}{2}\log(|\sigma^2JJ^T|)]-\frac{d}{2}\log(2\pi)-\frac{1}{2\sigma^2}\mathbb{E}_{\epsilon\sim p(\mathcal{E})}[\epsilon^T\epsilon]+\mathcal{O}(\sigma^2)\\
&= \mathbb{E}_{x\sim p(X)}[-\frac{1}{2}\log(|JJ^T|)]-d\log(\sigma)-\frac{d}{2}\log(2\pi)-\frac{1}{2\sigma^2}\mathcal{O}(\sigma^2)+\mathcal{O}(\sigma^2)\\
&= \mathbb{E}_{x\sim p(X)}[-\log|\det J|]-d\log(\sigma)-\frac{d}{2}\log(2\pi)-\frac{1}{2\sigma^2}\mathcal{O}(\sigma^2)+\mathcal{O}(\sigma^2) \qquad (16)
\end{aligned}$$

For the first item (A), we can use the derivation in Eq. 2.

$$\begin{aligned}
A &= \mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}[-\log p(\varphi_\theta(x+\epsilon))]\\
&= \mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}\Big[\frac{d}{2}\log(2\pi)-\operatorname*{logsumexp}_y\Big(-\frac{||\varphi_\theta(x+\epsilon)-\mu_y||_2^2}{2}+c_y\Big)\Big]\\
&= \mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}\Big[-\operatorname*{logsumexp}_y\Big(-\frac{||\varphi_\theta(x+\epsilon)-\mu_y||_2^2}{2}+c_y\Big)\Big]+\frac{d}{2}\log(2\pi)
\end{aligned} \qquad (17)$$

Finally, we put the above derivations together and drop the constant items and the items that vanish with rate $\mathcal{O}(\sigma^2)$ as $\sigma\to 0$. The $I(X,Z_\mathcal{E})$ becomes:

$$I(X,Z_\mathcal{E})=\mathbb{E}_{x\sim p(X),\epsilon\sim p(\mathcal{E})}\Big[-\operatorname*{logsumexp}_y\Big(-\frac{||\varphi_\theta(x+\epsilon)-\mu_y||_2^2}{2}+c_y\Big)-\log|\det J|\Big] \qquad (18)$$

We can find that the $I(X,Z_\mathcal{E})$ has the same formula as the loss $\mathcal{L}_g$ except the constant item $\frac{d}{2}\log(2\pi)$, and $I(Z,Y)=I(Y,Z)=-\mathcal{L}_{mi}$ (see Eq. 8). Thus, the learning objective $\min I(X,Z_\mathcal{E})-\alpha I(Z,Y)$ in *Information Bottleneck principle* can be converted to $\mathcal{L}_g+\alpha\mathcal{L}_{mi}$, which is the first half part of the training loss in Eq. (11).

From the *Information Bottleneck principle* perspective, we can explain our method: it attempts to minimize the mutual information $I(X,Z_\mathcal{E})$ between X and $Z_\mathcal{E}$, forcing the model to ignore the irrelevant aspects of $X+\mathcal{E}$ which do not contribute to fit the latent distribution and only increase the potential for overfitting. Therefore, the $\mathcal{L}_g$ loss function actually endows the normalizing flow model with the compression ability for establishing correct invertible mappings between input X and the latent Gaussian mixture prior Z, which is effective to prevent the model from learning the "homogeneous mapping". Simultaneously, it encourages to maximize the mutual information $I(Y,Z)$ between Y and Z, forcing the model to map different class features to their corresponding class centers which can contribute to class discriminative ability.

G Additional Results

Quantitative Results Under the One-for-one Setting. In Tab. 3, we report the detailed results of anomaly detection and localization on MVTecAD [4]

under the one-for-one setting. We can find that all baselines achieve excellent results under the one-for-one setting, but their performances drop dramatically under the unified case (see Tab. 1 in the main text). For instance, the strong baseline, DRAEM, suffers from a drop of 9.9% and 10.1%. The performance of the previous SOTA NF-based AD method, FastFlow, drops by 7.6% and 2.5%. This demonstrates that the unified anomaly detection is quite more challenging than the conventional one-for-one anomaly detection task, and current SOTA AD methods cannot be directly applied to the unified AD task well. Thus, how to improve the unified AD ability for AD methods should be further studied. On the other hand, compared with reconstruction-based AD methods (*e.g*, DRAEM [17]), NF-based AD methods have less performance degradation when directly applied to the unified case, indicating that NF-based approaches may be a more suitable way for the unified AD modeling than the reconstruction-based approaches.

Table 3: Anomaly detection and localization results on MVTecAD. All methods are evaluated under the one-for-one setting. ·/· means the image-level and pixel-level AUROCs.

Category	**Baseline Methods**			**Unified Methods**		**NF Based Methods**	
	PaDiM	MKD	DRAEM	PMAD	UniAD	FastFlow	CFLOW
Carpet	99.8/99.0	79.3/95.6	97.0/95.5	99.7/98.8	99.9/98.0	100/99.4	100/99.3
Grid	96.7/97.1	78.0/91.8	99.9/99.7	97.7/96.3	98.5/94.6	99.7/98.3	97.6/99.0
Leather	100/99.0	95.1/98.1	100/98.6	100/99.2	100/98.3	100/99.5	97.7/99.7
Tile	98.1/94.1	91.6/82.8	99.6/99.2	100/94.4	99.0/91.8	100/96.3	98.7/98.0
Wood	99.2/94.1	94.3/84.8	99.1/96.4	98.0/93.3	97.9/93.4	100/97.0	99.6/96.7
Bottle	99.9/98.2	99.4/96.3	99.2/99.1	100/98.4	100/98.1	100/97.7	100/99.0
Cable	92.7/96.7	89.2/82.4	91.8/94.7	98.0/97.5	97.6/96.8	100/98.4	100/97.6
Capsule	91.3/98.6	80.5/95.9	98.5/94.3	89.8/98.6	85.3/97.9	100/99.1	99.3/99.0
Hazelnut	92.0/98.1	98.4/94.6	100/99.7	100/98.8	99.9/98.8	100/99.1	96.8/98.9
Metal nut	98.7/97.3	73.6/86.4	98.7/99.5	99.2/97.5	99.0/95.7	100/98.5	91.9/98.6
Pill	93.3/95.7	82.7/89.6	98.9/97.6	94.3/95.5	88.3/95.1	99.4/99.2	99.9/99.0
Screw	85.8/98.4	83.3/96.0	93.9/97.6	73.9/91.4	91.9/97.4	97.8/99.4	99.7/98.9
Toothbrush	96.1/98.8	92.2/96.1	100/98.1	91.4/98.2	95.0/97.8	94.4/98.9	95.2/99.0
Transistor	97.4/97.6	85.6/76.5	93.1/90.9	99.8/97.8	100/98.7	99.8/97.3	99.1/98.0
Zipper	90.3/98.4	93.2/93.9	100/98.8	99.5/96.7	96.7/96.0	99.5/98.7	98.5/99.1
Mean	95.5/97.4	87.8/90.7	98.0/97.3	96.1/96.8	96.6/96.6	99.4/98.5	98.3/98.6

Log-likelihood Histograms. In Fig. 1, we show log-likelihoods generated by the one-for-one NF-based AD method and our method. All categories are from the MVTecAD dataset. The visualization results can empirically verify our speculation that the one-for-one NF-based AD methods may fall into the "homogeneous mapping" issue, where the normal and abnormal log-likelihoods are highly overlapped.

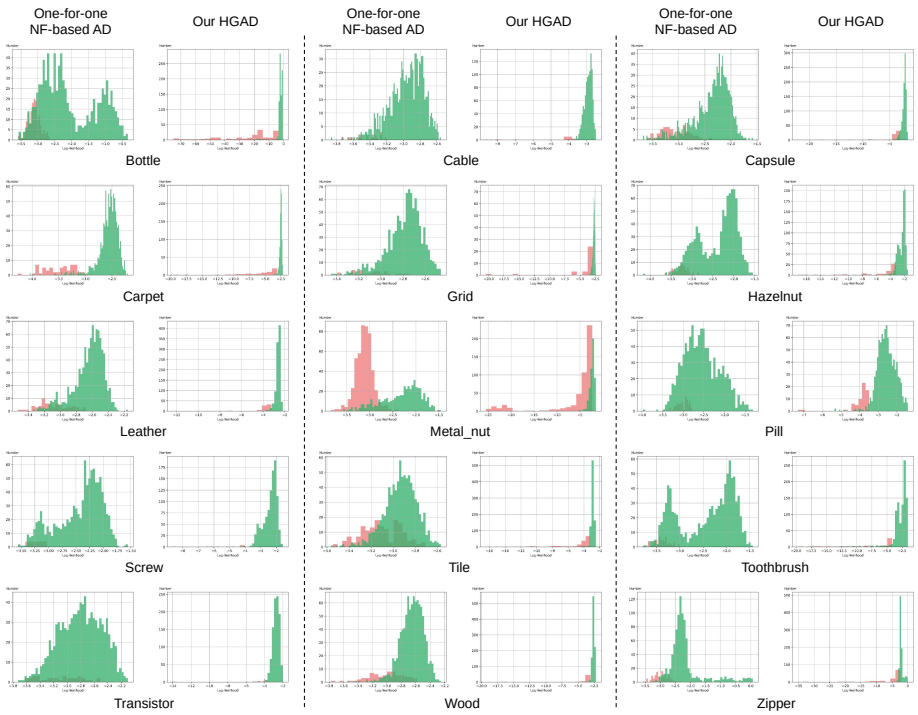

Fig. 1: Log-likelihood histograms on MVTecAD. All categories are from the MVTecAD dataset.

Qualitative Results. We present in Fig. 2 additional anomaly localization results of categories with different anomalies in the MVTecAD dataset. It can be found that our approach can generate much better anomaly score maps that the one-for-one NF-based baseline CFLOW [7] even for different categories from the MVTecAD dataset.

References

1. Alemi, A.A., Fischer, I., Dillon, J.V., Murphy, K.: Deep variational information bottleneck. In ICLR (2017)
2. Ardizzone, L., Lüth, C., Kruse, J., Rother, C., Köthe, U.: Guided image generation with conditional invertible neural networks. arXiv preprint arXiv: 1907.02392 (2019)
3. Ardizzone, L., Mackowiak, R., Rother, C., Kothe, U.: Training normalizing flows with the information bottleneck for competitive generative classification. In NeurIPS (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., 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)
6. Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using real nvp. In International Conference on Learning Representations (2017)
7. 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)
8. Kingma, D.P., Dhariwal, P.: Glow: Generative flow with invertible 1x1 convolutions. In Conference and Workshop on Neural Information Processing Systems (2019)
9. 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)
10. Saxe, A.M., Bansal, Y., Dapello, J., Advani, M., Kolchinsky, A., Tracey, B.D., Cox, D.D.: On the information bottleneck theory of deep learning. In ICLR (2018)
11. Shannon, C.E.: A mathematical theory of communication. Bell System Technical Journal (1948)
12. Tishby, N., Pereira, F.C., , Bialek, W.: The information bottleneck method. In The 37th annual Allerton Conference on Communication, Control and Computing (1999)
13. Tishby, N., Zaslavsky, N.: Deep learning and the information bottleneck principle. In IEEE Information Theory Workshop (ITW) (2015)
14. 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)
15. Yao, X., Zhang, C., Li, R., Sun, J., Liu, Z.: One-for-all: Proposal masked cross-class anomaly detection. In AAAI (2023)
16. 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)
17. Zavrtanik, V., Kristan, M., Skocaj, D.: Draem: A discriminatively trained reconstruction embedding for surface anomaly detection. In ICCV (2021)
18. 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)

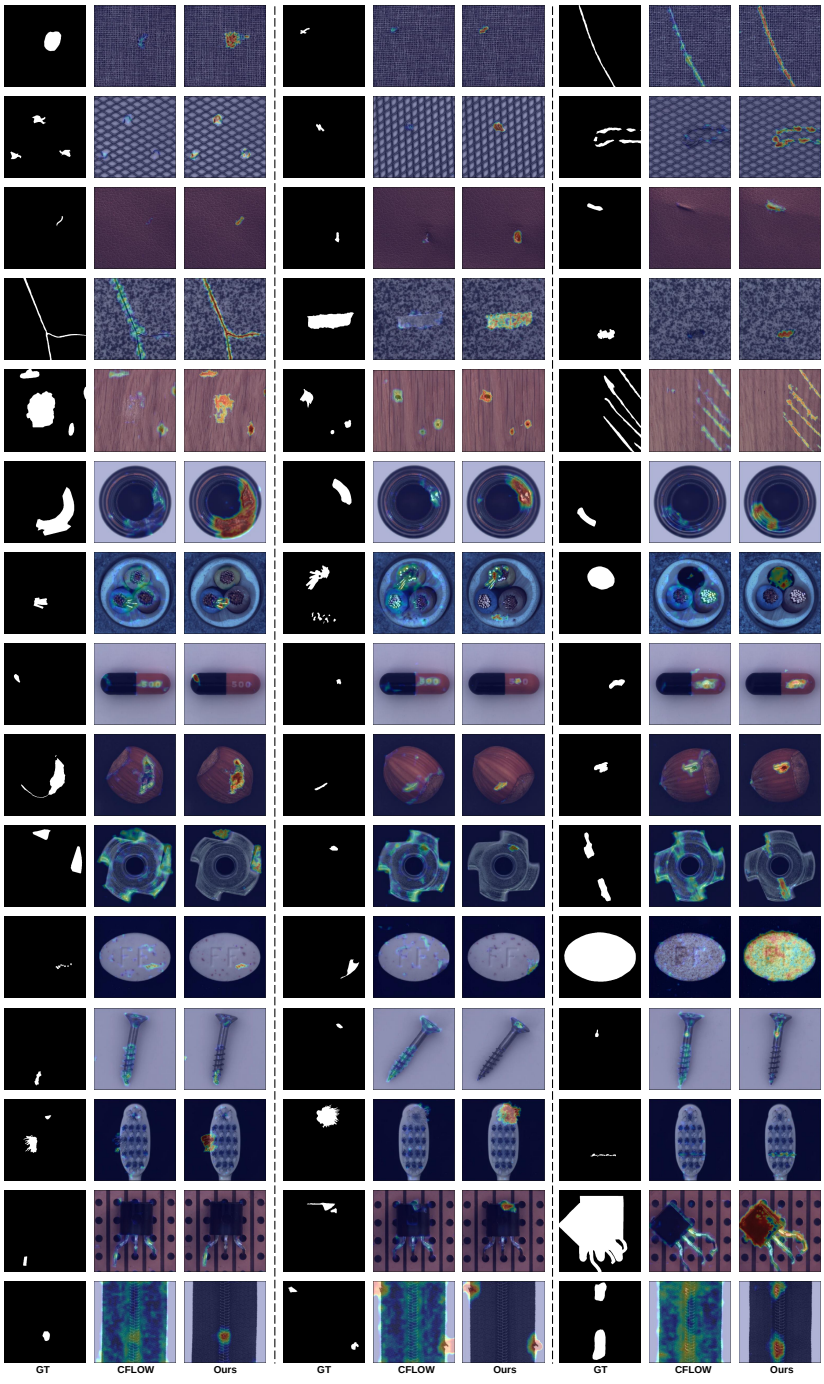

Fig. 2: Qualitative results on MVTecAD. More visualization of anomaly localization maps generated by our method on industrial inspection data. All examples are from the MVTecAD dataset.