MoEAD: A Parameter-efficient Model for Multi-class Anomaly Detection

Shiyuan Meng[1], Wenchao Meng[1*], Qihang Zhou[1], Shizhong Li[1], Weiye Hou[1], and Shibo He[1]

Zhejiang University

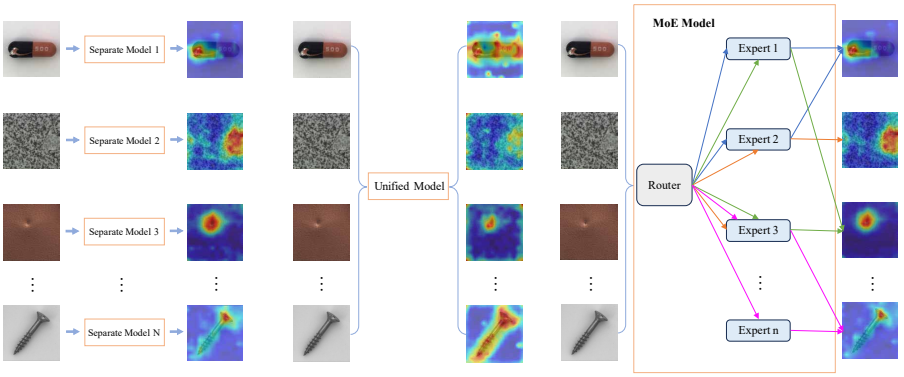
Abstract. Utilizing a unified model to detect multi-class anomalies is a promising solution to real-world anomaly detection. Despite their appeal, such models typically suffer from large model parameters and thus pose a challenge to their deployment on memory-constrained embedding devices. To address this challenge, this paper proposes a novel ViT-style multi-class detection approach named MoEAD, which can reduce the model size while simultaneously maintaining its detection performance. Our key insight is that the FFN layers within each stacked block (i.e., transformer blocks in ViT) mainly characterize the unique representations in these blocks, while the remaining components exhibit similar behaviors across different blocks. The finding motivates us to squeeze traditional stacked transformed blocks from N to a single block, and then incorporate Mixture of Experts (MoE) technology to adaptively select the FFN layer from an expert pool in every recursive round. This allows MoEAD to capture anomaly semantics step-by-step like ViT and choose the optimal representations for distinct class anomaly semantics, even though it shares parameters in all blocks with only one. Experiments show that, compared to the state-of-the-art (SOTA) anomaly detection methods, MoEAD achieves a desirable trade-off between performance and memory consumption. It not only employs the smallest model parameters, has the fastest inference speed, but also obtains competitive detection performance. Code will be available at `https://github.com/TheStarOfMSY/MoEAD`.

Keywords: Multi-class anomaly detection · Unsupervised anomaly detection · Industrial scene

1 Introduction

Anomaly detection (AD) has emerged as a task that receives much attention within the field of computer vision [17], with its objective being to ascertain the presence of anomalous regions within an image and to localize these regions accurately. An anomaly detection network with high accuracy and swift inference speeds aids manufacturing enterprises in improving work efficiency and guaranteeing product quality. Concurrently, due to the limited computational power

* Corresponding Author.



(a) Separate Models for Different Classes **(b)** Unified Model for Different Classes **(c)** MoE Model for Unified Multi-class Anomaly Detection

Fig. 1: The idea of the MoE model for unsupervised unified multi-class anomaly detection. (a) Conventional methods use separate models for different classes, which may lead to high model storage and training time costs. As shown in (c), we use MoE architecture in a unified model like (b). This method sparsely activates experts in an input-dependent way. Due to its adaptability to vast disparities among different types of inputs, this method enables the network to perform better in unified anomaly detection.

of edge devices in industrial production scenarios, the parameter scale of the anomaly detection network should be restricted.

Different from training separate models for diverse object classes, unified anomaly detection aims to use the same framework for multiple categories of anomalies, which conserves a large amount of storage space and training time. Therefore, there is a significant need for a lightweight, fast, unsupervised unified anomaly detection network that also considers accuracy and robustness.

Transformer [33] has achieved commendable performance results in computer vision [6, 9], and models based on the transformer architecture have been active in the field of anomaly detection as well [27, 36]. In this work, we present a novel parameter-efficient ViT-style model, MoEAD, for unsupervised unified anomaly detection and localization. We combine a common Convolutional Neural Network (CNN) backbone with a decoder-only transformer. Compared with the vanilla transformer, we reuse the trainable parameters across transformer blocks, which can compress the model along depth.

However, the direct application of the sharing strategy may inhibit the model from learning different representations in our blocks. To solve this issue, following WideNet [34], we replace the feed-forward network (FFN) layer with the Mixture of Experts (MoE) layer. This layer activates parameters sparsely in response to input tokens via a gating network, enabling the network to have more robust performance while maintaining close FLOPs compared with non-MoE architectures.

The introduction of the MoE layer enhances the capabilities of the model in two respects: First, even while using identical parameters, different blocks of operations within MoE still have the potential to activate distinct experts. This enables the model to learn new representations of input samples with each iteration, just like the version that does not share its parameters. Second, through the routing of the gating function, different experts can process drastically different input samples, as illustrated in Fig. 1. This illustrates that the sparse selection property intrinsic to the MoE architecture determines its unique advantage in unified anomaly detection tasks. Both of these enhancement effects are related to the number of experts that the gating network can select. We quantify this capability using the term *possible paths*.

Clearly, maintaining the capacity of the model requires a certain number of experts. Furthermore, we have observed that the reduction of *possible paths* undermines the model's capacity. By stacking two small layers to mimic the power of a large one, we introduce the Stacked MoE (SMoE) layer, which attains optimal performance with a lesser degree of trainable parameters. The methodology used in our model is designed to model diversified industrial anomalies with as few parameters as possible. The experiment part proves that MoEAD is a highly parameter-efficient and effective framework and achieves comparable performance on the task of anomaly detection and localization. We summarize the main contributions of our paper as follows:

- We propose a novel framework, MoEAD, for multi-class anomaly detection that uses MoE architecture in industrial vision anomaly detection.
- To compress the parameter quantity of the model, we share the trainable parameters of the multi-head attention layer and MoE layer. Since the modeling capacity of networks has been enhanced by MoE architecture, sharing parameters across layers can lead to a more lightweight model without significant loss in performance.
- By introducing the SMoE layer, we can use fewer experts and maintain the routing capability of the Gate Network with a reduced number of experts. This design further diminishes the parameter quantity of the model, making it more parameter-efficient.

2 Related Works

2.1 Anomaly detection

Industrial anomaly detection based on images has been a subject of extensive study. Given the unpredictable nature of anomalies in real-world scenarios, there is a growing aspiration to leverage unsupervised learning or zero-shot [12, 19, 41] learning techniques for detecting anomalies. Typical unsupervised anomaly detection methods comprise synthesizing-based [22, 37], embedding-based [5, 30, 40], and reconstruction-based [14, 23, 38] strategies. The reconstruction-based approach is a highly intuitive concept, as it aims to learn the characteristics of normal samples, reconstruct the input data, and identify anomalies through

comparative analysis. Nonetheless, this technique is challenged when tasked with the reconstruction of a wide variety of input samples.

Traditional single-class unsupervised anomaly detection [8, 24, 37] exhibits drawbacks such as wasted training time and storage space, as well as inconvenient usage. Recent works primarily focus on how to conduct unsupervised multi-class anomaly detection using a unified model, which is valuable but challenging. UniAD [35] first successfully accomplishes unified anomaly detection using a vision transformer, and OmniAL [39] uses a unified CNN framework for this task. Subsequent works have employed more diverse techniques; for example, DiAD [16] utilizes diffusion models, while HVQ-Trans [26] adopts a probabilistic framework.

In terms of model lightening, some studies [24] have begun to focus on the importance of computational efficiency in anomaly detection applications. Unfortunately, many works aiming for high precision tend to overlook the constraints on model parameter quantity. In this research, we strictly limit the scale of model parameters and attempt to perform unified multi-class anomaly detection with high parameter efficiency.

2.2 Mixture of Experts

The birth of MoE architecture [18] is inspired by the concept of conditional computation [2]. Within the field of deep learning, Shazeer *et al.* [31] are the pioneers in sparsely-gated MoE, which only activates some relevant *experts* dependent on input to construct large neural networks. GShard [21] firstly applies the MoE concept to the transformer and proposes GShardGate, which restricts the number of tokens processed by one expert. Subsequently, Switch Transformers [13] and Glam [10] follow this idea to further scale the large language models along width to more trainable parameters. In computer vision, MoE also inspires some significant works [1, 15, 28]. Specially, V-MoE [29], a sparse version of the ViT [9], matches the performance of ViT-H while requiring only half of the runtime, showing the potential of MoE architecture in decreasing the model's inference time.

3 Preliminary

Before formally introducing our model framework, we first present some essential information about the MoE architecture in this section. We use MoE architecture to construct an MoE layer and insert it into our model. Fig. 2 is an example of a MoE layer with n experts using Top2 routing. Follow Shazeer *et al.* [31], the MoE layer consists of a set of n trainable expert networks $E_1, E_2, \cdots, E_n$, a gate function G, and a routing network R. Given the input representation x, the output of the MoE layer can be written as:

$$MoE(x) = \sum_{i=1}^{n} G(x)_i E_i(x) + x, \tag{1}$$

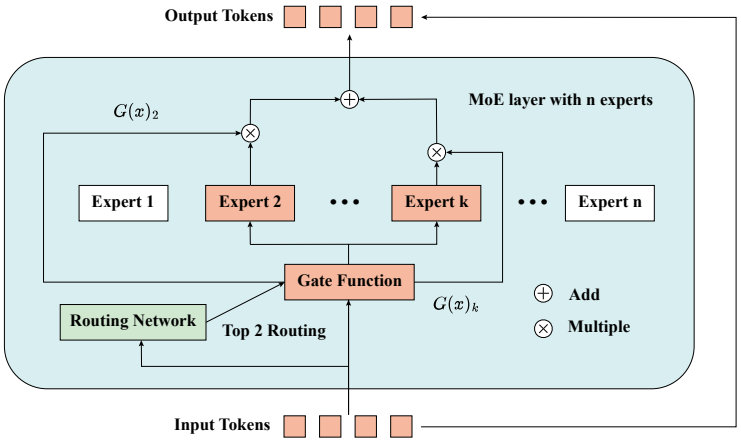


Fig. 2: The architecture of MoE layer used in MoEAD.

where $E_i(x)$ is the output of the i^{th} expert network, which is an FFN in this paper. $G(x)$ is the output of the sparse gate function. If $G(x)_i = 0$, the calculation for the corresponding $E_i(x)$ is unnecessary. We use TopK selection and the softmax function to implement sparsity, our gate function G can be defined as:

$$G(x) = Softmax(TopK(R(x), k)), \tag{2}$$

$$TopK(x, k)_i = \begin{cases} x_i & \text{if } x_i \text{ is the largest } k \text{ elements in } x, \\ -\infty & \text{otherwise,} \end{cases} \tag{3}$$

$$R(x)_i = (x \cdot W_r)_i, \tag{4}$$

where $R(x)$ is the output of the routing network. It is a linear transformation, obtained by multiplying the input by a trainable weight W_r. TopK is an operation that sets all elements of vector $R(x)$ to $-\infty$ (which causes the corresponding gate values to equal 0) except the largest k elements. In this paper, we use $k = 2$ the same as Fig. 2.

4 Method

4.1 MoEAD Architecture

In this section, we will introduce the details of our MoEAD framework. As shown in Fig. 3, MoEAD consists of a conventional CNN backbone, our parameter sharing decoder, and the comparison module after them.

Firstly, we use pre-trained EfficientNet-b4 [32], with its parameters frozen during training, to learn a 2D representation of image input. The output of

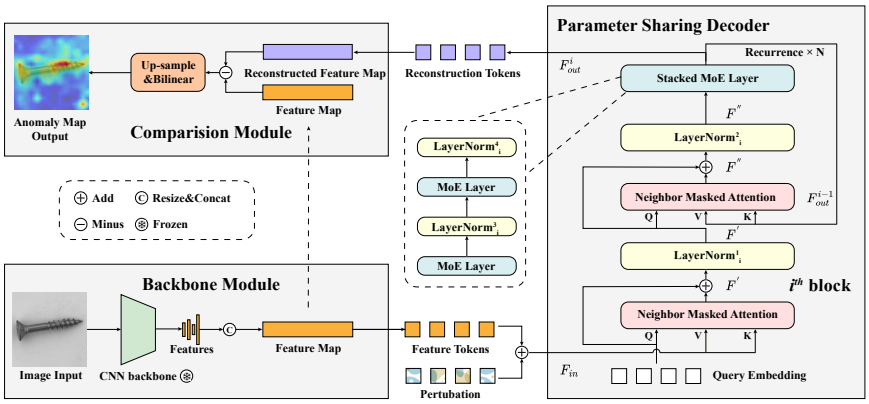


Fig. 3: The overall architecture of our proposed framework MoEAD.

the backbone is a set of extracted image features, and then we resize them and concatenate them into a feature map $F_{\text{in}} \in \mathbb{R}^{C_{org} \times H \times W}$. After tokenization, a linear projection reduces the channel dimension from C_{org} to a smaller dimension C, and we obtain $H \times W$ input feature tokens. To train our model to better reconstruct the input with noise, we use the feature jittering strategy [3]. Perturbations are added to feature tokens before they are processed in our Parameter Sharing Decoder to reconstruct the features.

After the module above, the final output can be viewed as the reconstruction of input token features, and they are then transformed into a reconstructed feature map $F_{\text{out}} \in \mathbb{R}^{C_{org} \times H \times W}$ by another linear projection to recover the decreased channel dimension. In the training procedure, the model is trained with the MSE loss of F_{in} and F_{out} to learn the representation of normal samples. In inference, we define the anomaly score map $S \in \mathbb{R}^{H \times W}$ by calculating the L2 norm of the difference between F_{in} and F_{out}. For anomaly detection, we use the largest value of S after an average pooling to check whether the input image has an abnormal area. For anomaly localization, S is transformed into the origin image size by up-sampling and bi-linear interpolation.

4.2 Parameter Sharing Decoder

Parameter sharing decoder reconstructs the feature tokens in our model, it follows the standard decoder-only transformer architecture. Compared with the vanilla transformer decoder, our network is different in three aspects: (1) We share the parameters of attention and the SMoE layer across our transformer decoder blocks to improve parameter efficiency; (2) We employ a learnable query embedding and use Neighbor Masked Attention (NMA) to avoid recovering the anomalous samples inspired by [35]; (3) The FFN layer is replaced by the SMoE layer to learn more complex representations of the input.

Like other works that share parameters across transformer blocks [7, 20], MoEAD uses the same attention modules and SMoE layer in decoder blocks. However, we use different layer normalization to make sure input tokens learn diversified representations between blocks [34]. Take the i^{th} decoder block for example, it has four individual normalization layers (two of which are concluded in the SMoE layer). The computation in the i^{th} decoder block of MoEAD can be written mathematically as:

$$F^{'} = NMA(q, F_{\text{in}}, F_{\text{in}}), \tag{5}$$

$$F^{'} = LayerNormal_i^1(F^{'} + F_{\text{in}}), \tag{6}$$

$$F^{''} = NMA(F^{'}, F_{\text{out}}^{i-1}, F_{\text{out}}^{i-1}), \tag{7}$$

$$F^{''} = LayerNormal_i^2(F^{''} + F^{'}), \tag{8}$$

$$F_{\text{out}}^{i} = StackedMoE(F^{''}), \tag{9}$$

where F_{in} is the input of the decoder architecture, F_{out}^{i} is the output of the i^{th} block, and q is a learnable query embedding. Learnable query embedding and neighbor masked attention are tricks to prevent the model from reconstructing the anomalies. In each transformer decoder block, we first fuse the learnable query embedding q with F_{in} by the neighbor masked attention mechanism, which masks the neighbor tokens during calculating the attention map. Residual connection and layer normalization are utilized on $F^{'}$, which we obtain before the next neighbor masks attention between $F^{'}$ and the last block's output F_{out}^{i-1}. After two attention modules, we pass all the tokens to the SMoE layer. The details of this procession are in 4.3. Ultimately, this recurrence is executed for N times, and the output of the final block is considered the network's output result, F_{out}.

Given that the parameters of the model are concentrated in the attention module and SMoE layer, we reduce approximately N times the number of trainable parameters by sharing parameters in each block except for normalization, thereby minimizing the model's complexity. Thanks to the SMoE layer, sharing parameters does not significantly negatively impact the performance of our model.

4.3 SMoE Layer

As shown in Sec. 3, our core idea of the MoE layer is using conditional computation to better learn significantly divergent input samples. However, the potential of the conditional computation is connected with the number of experts. The presence of a large number of experts allows the gating function to more efficiently identify suitable experts to handle specific inputs. We refer to this scenario as the gating function providing more *possible paths* for each token input.

However, an excess of expert networks might lead to problems such as significantly higher memory usage and difficulties in adequately training these experts.

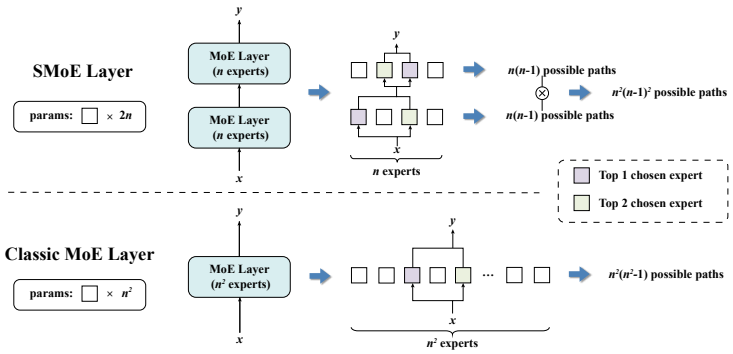


Fig. 4: SMoE layer compared with the classic MoE layer. Two stacked MoE layers enhance the selectable paths through multiplication, thus strengthening the gating selection ability. Compared to a layer with a larger number of experts, this approach achieves a similar effect with fewer trainable parameters.

V-MoE [29] uses 8 experts per layer for datasets with 9 million data points, while it chooses 32 experts for datasets with 300 million. For previous MoE networks [21, 31, 34], expert network counts of 16, 32, 64, and 128 are all often selected.

To ensure the accuracy of the model, the use of a certain number of expert networks is inevitable, but it also results in a larger parameter size in the model. For the purpose of compressing the model size, we stack two smaller layers with n experts to replace the power of a n^2 expert layer like [11]. We call this method the SMoE layer.

As shown in Fig. 4, SMoE layers have a similar number of *possible paths* to a larger one. In the Top2 selection condition, each layer chooses two experts, and these two experts possess distinct weights correlated with the scores given by the routing network. An MoE layer with n experts can provide n potential most relevant expert candidates and $n-1$ sub-relevant expert candidates, yielding a total of $n(n-1)$ possible alternative solutions for certain inputs. By stacking two layers, the $2n$ experts produce $n^2(n-1)^2$ possible choices, which is close to $n^2(n^2-1)$ an MoE layer with n^2 experts can provide.

Experiment results in Sec. 5.5 confirm the SMoE layer is more parameter-efficient than the layer with double experts. It can decrease the trainable parameters needed while scaling the model to achieve better performance.

4.4 Adding Auxiliary Loss

In our model, we dispatch every input token to k experts. However, it is possible to observe an imbalance in load among the experts during training. This is due to the fact that experts who perform well in the early stages of training are more likely to be selected by the gating network. To solve this problem, we define an

auxiliary loss to encourage all experts to be equally important in our framework. The auxiliary loss can be written as:

$$\mathcal{L}_{\text{aux}} = CV(\sum_{x \in X} G_1(x))^2 + CV(\sum_{x \in X} G_2(x))^2, \tag{10}$$

$$\mathcal{L} = \mathcal{L}_{\text{main}} + \lambda * \mathcal{L}_{\text{aux}}, \tag{11}$$

where $G_1(x)$ and $G_2(x)$ are the outputs of the gate function in two MoE layers, which consist of our SMoE layer. In this formula, CV represents *Coefficient of Variation*, which encapsulates the uniformity of the output produced by the gating function. The smaller the auxiliary loss is, the more evenly the experts within the SMoE layer are selected. Finally, the auxiliary loss is added to our main loss. We set hyper-parameter λ 0.01 in our work.

5 Experiments

5.1 Dataset and Metrics

Dataset. We evaluate our model on the MVTec-AD dataset [4] and the VisA dataset [42]. The MVTec-AD dataset consists of over five thousand pictures of 15 classes in the industrial environment. All of the samples in the training set are anomaly-free, which is convenient for unsupervised learning. For abnormal samples in the test set, the ground truth provides image-level and pixel-level annotations for both anomaly detection and localization evaluation. The VisA dataset is similar to the MVTec-AD in use but has more images and different objects. These datasets are widely applied in research for unsupervised multi-class anomaly detection and localization.

Metrics. Following prior work, we use the Area Under the Receiver Operating Curve (AUROC) as our evaluation metric for anomaly detection and anomaly localization. In Sec. 5.2, Sec. 5.4, and Sec. 5.5, I-AUROC represents AUROC of image-level anomaly detection, and P-AUROC represents pixel-level anomaly localization.

Besides achieving SOTA performance, we also want to show our framework's parameter efficiency and inference speed. So we use model trainable parameters (Params) and FLoating point OPerations (FLOPs) to represent our framework's advantage in using fewer trainable parameters and less time while detecting anomalies in the industrial scene.

5.2 Performance Comparision

Setup. In experiments, all images in the dataset are revised to 224×224. After backbone, the features are concatenated to form a $272 \times 14 \times 14$ feature map and then tokenized. Without special annotation, we use 4 experts in MoE layer. The

Table 1: Multi-class anomaly detection results with the I-AUROC metric compared with SOTA methods on the MVTec-AD dataset.

Category		DRAEM [37] ICCV'21	RD [8] CVPR'22	UniAD [35] NIPS'22	DiAD [16] AAAI'24	HVQ-Trans [26] NIPS'23	MoEAD
Object	Bottle	97.5	99.6	99.7	99.7	100	100 ± 0.00
	Cable	57.8	84.1	95.2	94.8	99.0	97.6 ± 1.22
	Capsule	65.3	94.1	86.9	89.0	95.4	92.5 ± 0.99
	Hazelnut	93.7	60.8	99.8	99.5	100	100 ± 0.00
	Metal Nut	72.8	100	99.2	99.1	99.9	99.4 ± 0.10
	Pill	82.2	97.5	93.7	95.7	95.8	94.7 ± 0.02
	Screw	92.0	97.7	87.5	90.7	95.6	92.7 ± 0.51
	Toothbrush	90.6	97.2	94.2	99.7	93.6	94.2 ± 0.45
	Transistor	74.8	94.2	99.8	99.8	99.7	99.8 ± 0.01
	Zipper	98.8	99.5	95.8	95.1	97.9	98.3 ± 0.23
Texture	Carpet	98.0	98.5	99.8	99.4	99.9	99.7 ± 0.00
	Grid	99.3	98.0	98.2	98.5	97.0	99.2 ± 0.08
	Leather	98.7	100	100	99.8	100	100 ± 0.00
	Tile	99.8	98.3	99.3	96.8	99.2	99.5 ± 0.05
	Wood	99.8	99.2	98.6	99.7	97.2	98.6 ± 0.02
Mean		88.1	94.6	96.5	97.2	98.0	97.7 ± 0.02

AdamW optimizer [25] is also employed in our approach with a weight decay of 1×10^{-4}. We train our model for 1000 epochs on a single machine using 4 GPUs (NVIDIA GeForce RTX 3090). We run the evaluation 5 times with different random seeds, and the learning rate of all experiments is 2×10^{-4}.

Baselines. We have opted for several classic open-source works as baselines for comparison with our architecture, which include: DRAEM [37], RD [8], UniAD [35], DiAD [16] and HVQ-Trans [26]. To carry out experiments on parameter effectiveness and inference speed, we calculate the model size and FLOPs of these baselines through their available code repositories.

Performance on MVTec-AD dataset. Quantitative results of anomaly detection/localization on MVTec-AD are shown in Tab. 1/Tab. 2, with the best in red and the top 3 in blue. In comparison to other methods, MoEAD performs adeptly across virtually all 15 categories of objectives in anomaly detection, achieving accuracy within the top three of all models in almost every category (13/15). The balanced performance is attributed to the MoE architectural design adopted in our model. By gating network, we ensure that each input object is processed by suitably trained experts. Similar to anomaly detection, MoEAD has achieved uniform localization results, with P-AUROC metrics exceeding 91.0 for every category. Overall, MoEAD has demonstrated excellent performance in anomaly detection and localization, achieving an I-AUROC/P-AUROC of 97.7/97.0, standing next only to HVQ-Trans. Additionally, MoEAD presents remarkable performance in anomaly detection for textural class objects.

Performance on VisA dataset. As shown in Tab. 3, VisA is more difficult than MVTec-AD in anomaly detection since categories in VisA have a more

Table 2: Multi-class anomaly localization results with the P-AUROC metric compared with SOTA methods on the MVTec-AD dataset.

Category		DRAEM [37] ICCV'21	RD [8] CVPR'22	UniAD [35] NIPS'22	DiAD [16] AAAI'24	HVQ-Trans [26] NIPS'23	MoEAD
Object	Bottle	87.6	97.8	98.1	98.4	98.3	98.1 ± 0.00
	Cable	71.3	85.1	97.3	96.8	98.1	97.6 ± 0.02
	Capsule	50.5	98.8	98.5	97.1	98.8	98.6 ± 0.00
	Hazelnut	96.9	97.9	98.1	98.3	98.8	97.9 ± 0.01
	Metal Nut	62.2	93.8	94.8	97.3	96.3	94.7 ± 0.05
	Pill	94.4	97.5	95.0	95.7	97.1	95.8 ± 0.08
	Screw	95.5	99.4	98.3	97.9	98.9	98.9 ± 0.01
	Toothbrush	97.7	99.0	98.4	99.0	98.6	98.4 ± 0.00
	Transistor	64.5	85.9	97.9	95.1	97.9	97.9 ± 0.03
	Zipper	98.3	98.5	96.8	96.2	97.5	97.7 ± 0.03
Texture	Carpet	98.6	99.0	98.5	98.6	98.7	98.3 ± 0.00
	Grid	98.7	99.2	96.5	96.6	97.0	97.4 ± 0.01
	Leather	97.3	99.3	98.8	98.8	98.8	98.8 ± 0.00
	Tile	98.0	95.3	91.8	92.4	92.2	91.8 ± 0.02
	Wood	96.0	95.3	83.2	93.3	92.4	92.9 ± 0.01
Mean		87.2	96.1	96.8	96.8	97.3	97.0 ± 0.01

complex structure. However, despite utilizing a relatively simple network architecture and fewer parameters, we have still managed to achieve an average I-AUROC/P-AUROC metric of 93.1/98.7, only 0.1 lower than SOTA method.

5.3 Speed and Model Size

We likewise compare the resources utilized in operation by MoEAD and other SOTA models, the results of which are shown in Tab. 3, with the best in red and the second in blue. Our model, MoEAD, employes the least number of trainable parameters and FLOPs, a feat credited to the simple and parameter-efficient model structure employed by MoEAD. In comparison to UniAD, which also has relatively low resource consumption, we achieve a reduction of 36% in trainable parameters and 49% in computational costs. Furthermore, MoEAD achieve performance metrics surpassed only by HVQ-Trans.

To better understand our lightweight model, we also analyze the proportion of parameters and computational load each layer occupies within a decoder block. Normalization layers occupy little resources in a block. However, the MoE layer occupies 88.8% parameters but only 1.6% FLOPs. The attention layer, on the contrary, occupies 98.0% FLOPs and 11.1% parameters. As discussed in Sec. 4.3, the method of stacking MoE layers essentially substitutes part of the parameters in MoE layers with more computations within the MoE layers. However, in terms of the entire decoder block, the computational load of the MoE layer is much less than that of the two attention layers, yet the parameters that need to be trained significantly exceed those of the two attention layers.

5.4 Ablation Study

Architecture design. To verify the effectiveness of the methodology employed by our model, we conducted ablation experiments on MVTec-AD. All experiments are conducted under the same setup, with the results displayed in Tab. 4a.

Table 3: Efficiency comparison of different methods. Params represents the trainable parameters, excluding the parameters of the backbone portion. UniAD, HVQ-Trans, and MoEAD all utilize the same EfficientNet-b4 [32], making this comparison fair in terms of this aspect. Performances are evaluated by I-AUROC/P-AUROC metrics.

Method	Params	FLOPs	Performance(MVTec-AD)	Performance(VisA)
DRAEM [37]	97.4M	198.0G	88.1/87.2	80.5/87.0
UniAD [35]	7.7M	4.30G	96.5/96.8	91.9/98.6
DiAD [16]	1.3B	$>$ 2.2T	97.2/96.8	86.8/96.0
HVQ-Trans [26]	18.0M	7.38G	98.0/97.3	93.2/98.7
MoEAD	4.9M	2.18G	97.7/97.0	93.1/98.7

We first analyze the impact of introducing the MoE layer to the model. From the experimental results, it can be seen that replacing the FFN layer with the MoE layer improves the model's accuracy but inevitably increases the trainable parameter quantity.

Next, we evaluate the influence of sharing layers in the block or not. Surprisingly, just sharing either the attention or the MoE layer alone will not significantly enhance the model performance compared to sharing both. Therefore, sharing all parameters of these two components across blocks is the most effective choice for compressing model size.

Lastly, we examine the impact of using the SMoE layer. The incorporation of a SMoE layer results in a significant improvement in model accuracy at a relatively small parameter cost. From the results of the ablation experiments, it's evident that, compared to other models, the technical approach we adopted achieves the best parameter efficiency.

Effect of the block numbers in parameter sharing decoder. We also conduct experiments to explore the effect of the block numbers in the parameter sharing decoder, and the result is shown in Tab. 4b. The best performance is achieved with 4 blocks, which leads to two key observations: 1. The input feature tokens are able to learn different representations in every block, therefore, the four-block model is capable of extracting more valuable information compared to the two-block model. 2. Adding more blocks can not result in further performance enhancement. We observe that both the six-block and eight-block models achieve AUROC metrics exceeding 97 in both tasks. However, overfitting happens in later training stages.

Effect of the k value in routing. The k value influences how many experts are chosen in routing. As shown in Tab. 4c, when we set the k value to 2, 3, and 4, the performance of our model almost stays the same but is better than the model with $k = 1$. Our model doesn't perform well in the case of $k = 1$, because only the input sample's top 1 expert can be activated during the training procedure, which makes the model optimization difficult. When $k > 1$, the k value has little influence on our model. As discussed in Sec. 3, we use the softmax function after

Table 4: Experiment results of ablation study.

(a) Ablation studies on architecture design.

MoE	Stacked	Att-Shared	MoE-Shared	Params	FLOPs	I-AUROC	P-AUROC
				4.6M	2.16G	96.1	96.3
		✓		2.8M	2.16G	95.7	96.1
✓				10.9M	2.17G	97.4	96.9
✓		✓		9.1M	2.17G	97.2	96.8
✓			✓	4.6M	2.17G	97.1	96.8
✓		✓	✓	2.8M	2.17G	97.1	96.8
✓	✓			19.3M	2.18G	98.1	97.2
✓	✓	✓		17.5M	2.18G	97.9	97.1
✓	✓		✓	6.7M	2.18G	97.7	97.0
✓	✓	✓	✓	4.9M	2.18G	97.7	97.0

(b) Ablation studies on block numbers.

Layers	FLOPs	I-AUROC	P-AUROC
2	1.68G	97.5	96.8
4	2.18G	97.7	97.0
6	2.68G	97.7	96.8
8	3.18G	97.6	96.8

(c) Ablation studies on the k value.

k value	I-AUROC	P-AUROC
1	96.8	96.7
2	97.7	97.0
3	97.6	97.0
4	97.7	97.0

the TopK selection, which makes the third or fourth expert contribute little to the layer output. That's why we select $k = 2$.

5.5 Experts' Number Analysis

The performance of our framework has relation with the number of experts in SMoE layer. To investigate the influence of experts number, we conduct more experiments on MVTec-AD. As shown in 5, when we use more experts per layer, our model perform better in I-AUROC and P-AUROC. Surprisingly, the metric capability of model with SMoE layer with n experts closely matches the performance of the non-SMoE layer with n^2 experts, indicating that the SMoE layers indeed achieve superior filtering effects. In contrast, using an MoE layer with $2n$ experts, although bearing a similar quantity of params with the stacked layer, does not introduce new gating schemes. Such a model can not perform as well as the stacked arrangement.

By experimenting with the value of n set to 2, 3, 4, and 5 respectively, we obtaine similar results across all instances. In summary, until the number of experts increases to a point where they cannot be adequately trained by the dataset, using a SMoE layer to replace the conventional mixture of expert layers per layer can save parameters for $n^2 - 2n$ experts. MoEAD with more experts performs stronger in evaluation, which shows our framework is scalable and makes a great balance in performance and model size.

Table 5: Experiment results of experts' number analysis.

	Experts	Stacked	Params	FLOPs	I-AUROC	P-AUROC
n=2	n	✓	2.8M	2.17G	97.1	96.8
	n^2		2.8M	2.17G	97.1	96.8
n=3	n	✓	3.9M	2.17G	97.3	96.9
	$2n$		3.9M	2.17G	97.2	96.8
	n^2		5.5M	2.18G	97.4	96.9
n=4	n	✓	4.9M	2.18G	97.7	97.0
	$2n$		4.9M	2.18G	97.3	96.9
	n^2		9.1M	2.19G	97.6	97.0
n=5	n	✓	6.0M	2.18G	97.9	97.1
	$2n$		6.0M	2.18G	97.4	96.9
	n^2		13.9M	2.21G	98.0	97.2

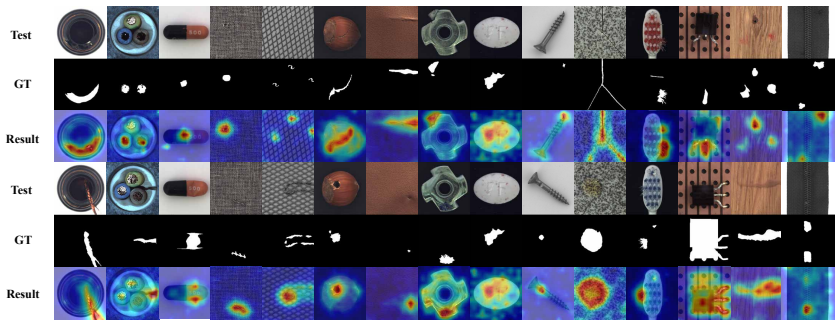


Fig. 5: Qualitative illustration of our anomaly localization results on MVTec-AD. From top to bottom: anomaly sample for test, ground-truth, and final anomaly map.

6 Conclusion

In this paper, we propose a noval architecture, MoEAD, for unsupervised unified multi-class anomaly detection. By introducing the MoE architecture, we are able to cyclically utilize the trainable parameters within the model without overly compromising performance. Further, the SMoE layer in our model reduces the number of experts required to achieve equivalent routing results, compressing the model additionally. Experiments carried out on the MVTec-AD dataset demonstrate that with minimal trainable parameters and FLOPs, our model achieves a performance that is merely 0.3% lower than the best method.

Although our method effectively reduces model parameters and complexity, there is still room for improvement in accuracy. In the future, we will proceed exploring the potential of the MoE architecture for efficient, high-precision, unsupervised multi-class anomaly detection. We aim to draw inspiration from the feature pyramid architecture in order to extract multi-scale features, thus enhancing the model's ability to localize anomalies with greater precision.

Acknowledgements

This work was supported by National Key R&D Program of China 2022ZD0118702, National Natural Science Foundation of China 92367205, U23A20326, and Zhejiang Provincial Natural Science Foundation of China LZ22F030011.

References

1. Ahmed, K., Baig, M.H., Torresani, L.: Network of experts for large-scale image categorization. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14. pp. 516–532. Springer (2016)
2. Bengio, Y.: Deep learning of representations: Looking forward. In: International conference on statistical language and speech processing. pp. 1–37. Springer (2013)
3. Bengio, Y., Yao, L., Alain, G., Vincent, P.: Generalized denoising auto-encoders as generative models. Advances in neural information processing systems **26** (2013)
4. Bergmann, P., Fauser, M., Sattlegger, D., Steger, C.: Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9592–9600 (2019)
5. Bergmann, P., Fauser, M., Sattlegger, D., Steger, C.: Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4183–4192 (2020)
6. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European conference on computer vision. pp. 213–229. Springer (2020)
7. Dehghani, M., Gouws, S., Vinyals, O., Uszkoreit, J., Kaiser, L.: Universal transformers. In: International Conference on Learning Representations (2018)
8. Deng, H., Li, X.: Anomaly detection via reverse distillation from one-class embedding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9737–9746 (2022)
9. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2020)
10. Du, N., Huang, Y., Dai, A.M., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A.W., Firat, O., et al.: Glam: Efficient scaling of language models with mixture-of-experts. In: International Conference on Machine Learning. pp. 5547–5569. PMLR (2022)
11. Eigen, D., Ranzato, M., Sutskever, I.: Learning factored representations in a deep mixture of experts. arXiv preprint arXiv:1312.4314 (2013)
12. Esmaeilpour, S., Liu, B., Robertson, E., Shu, L.: Zero-shot out-of-distribution detection based on the pre-trained model clip. In: Proceedings of the AAAI conference on artificial intelligence. vol. 36, pp. 6568–6576 (2022)
13. Fedus, W., Zoph, B., Shazeer, N.: Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. The Journal of Machine Learning Research **23**(1), 5232–5270 (2022)

14. Gong, D., Liu, L., Le, V., Saha, B., Mansour, M.R., Venkatesh, S., Hengel, A.v.d.: Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1705–1714 (2019)
15. Gross, S., Ranzato, M., Szlam, A.: Hard mixtures of experts for large scale weakly supervised vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6865–6873 (2017)
16. He, H., Zhang, J., Chen, H., Chen, X., Li, Z., Chen, X., Wang, Y., Wang, C., Xie, L.: A diffusion-based framework for multi-class anomaly detection. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 8472–8480 (2024)
17. Huang, C., Guan, H., Jiang, A., Zhang, Y., Spratling, M., Wang, Y.F.: Registration based few-shot anomaly detection. In: European Conference on Computer Vision. pp. 303–319. Springer (2022)
18. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural computation **3**(1), 79–87 (1991)
19. Jeong, J., Zou, Y., Kim, T., Zhang, D., Ravichandran, A., Dabeer, O.: Winclip: Zero-/few-shot anomaly classification and segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 19606–19616 (2023)
20. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942 (2019)
21. Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y., Krikun, M., Shazeer, N., Chen, Z.: Gshard: Scaling giant models with conditional computation and automatic sharding. In: International Conference on Learning Representations (2020)
22. Li, C.L., Sohn, K., Yoon, J., Pfister, T.: Cutpaste: Self-supervised learning for anomaly detection and localization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9664–9674 (2021)
23. Liang, Y., Zhang, J., Zhao, S., Wu, R., Liu, Y., Pan, S.: Omni-frequency channel-selection representations for unsupervised anomaly detection. IEEE Transactions on Image Processing (2023)
24. Liu, Z., Zhou, Y., Xu, Y., Wang, Z.: Simplenet: A simple network for image anomaly detection and localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20402–20411 (2023)
25. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2018)
26. Lu, R., Wu, Y., Tian, L., Wang, D., Chen, B., Liu, X., Hu, R.: Hierarchical vector quantized transformer for multi-class unsupervised anomaly detection. Advances in Neural Information Processing Systems **36** (2024)
27. Mishra, P., Verk, R., Fornasier, D., Piciarelli, C., Foresti, G.L.: Vt-adl: A vision transformer network for image anomaly detection and localization. In: 2021 IEEE 30th International Symposium on Industrial Electronics (ISIE). pp. 01–06. IEEE (2021)
28. Pavlitskaya, S., Hubschneider, C., Weber, M., Moritz, R., Huger, F., Schlicht, P., Zollner, M.: Using mixture of expert models to gain insights into semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 342–343 (2020)
29. Riquelme, C., Puigcerver, J., Mustafa, B., Neumann, M., Jenatton, R., Susano Pinto, A., Keysers, D., Houlsby, N.: Scaling vision with sparse mixture of experts. Advances in Neural Information Processing Systems **34**, 8583–8595 (2021)

30. Roth, K., Pemula, L., Zepeda, J., Schölkopf, B., Brox, T., Gehler, P.: Towards total recall in industrial anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14318–14328 (2022)
31. Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., Dean, J.: Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In: International Conference on Learning Representations (2016)
32. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
33. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
34. Xue, F., Shi, Z., Wei, F., Lou, Y., Liu, Y., You, Y.: Go wider instead of deeper. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, pp. 8779–8787 (2022)
35. You, Z., Cui, L., Shen, Y., Yang, K., Lu, X., Zheng, Y., Le, X.: A unified model for multi-class anomaly detection. Advances in Neural Information Processing Systems **35**, 4571–4584 (2022)
36. You, Z., Yang, K., Luo, W., Cui, L., Zheng, Y., Le, X.: Adtr: Anomaly detection transformer with feature reconstruction. In: International Conference on Neural Information Processing. pp. 298–310. Springer (2022)
37. Zavrtanik, V., Kristan, M., Skočaj, D.: Draem-a discriminatively trained reconstruction embedding for surface anomaly detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8330–8339 (2021)
38. Zavrtanik, V., Kristan, M., Skočaj, D.: Dsr–a dual subspace re-projection network for surface anomaly detection. In: European conference on computer vision. pp. 539–554. Springer (2022)
39. Zhao, Y.: Omnial: A unified cnn framework for unsupervised anomaly localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3924–3933 (2023)
40. Zhou, Q., He, S., Liu, H., Chen, T., Chen, J.: Pull & push: Leveraging differential knowledge distillation for efficient unsupervised anomaly detection and localization. IEEE Transactions on Circuits and Systems for Video Technology (2022)
41. Zhou, Q., Pang, G., Tian, Y., He, S., Chen, J.: Anomalyclip: Object-agnostic prompt learning for zero-shot anomaly detection. In: The Twelfth International Conference on Learning Representations (2023)
42. Zou, Y., Jeong, J., Pemula, L., Zhang, D., Dabeer, O.: Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. In: European Conference on Computer Vision. pp. 392–408. Springer (2022)