CADIC: Continual Anomaly Detection Based on Incremental Coreset

Gen Yang, Zhipeng Deng, Junfeng Man*

***Abstract*—The primary objective of Continual Anomaly Detection (CAD) is to learn the normal patterns of new tasks under dynamic data distribution assumptions while mitigating catastrophic forgetting. Existing embedding-based CAD approaches continuously update a memory bank with new embeddings to adapt to sequential tasks. However, these methods require constructing class-specific sub-memory banks for each task, which restricts their flexibility and scalability. To address this limitation, we propose a novel CAD framework where all tasks share a unified memory bank. During training, the method incrementally updates embeddings within a fixed-size coreset, enabling continuous knowledge acquisition from sequential tasks without task-specific memory fragmentation. In the inference phase, anomaly scores are computed via a nearest-neighbor matching mechanism, achieving state-of-the-art detection accuracy. We validate the method through comprehensive experiments on MVTec AD and Visa datasets. Results show that our approach outperforms existing baselines, achieving average image-level AUROC scores of 0.972 (MVTec AD) and 0.891 (Visa). Notably, on a real-world electronic paper dataset, it demonstrates 100% accuracy in anomaly sample detection, confirming its robustness in practical scenarios. The implementation will be open-sourced on GitHub.**

***Index Terms*—Anomaly detection, continual learning, memory replay, coreset.**

I. Introduction

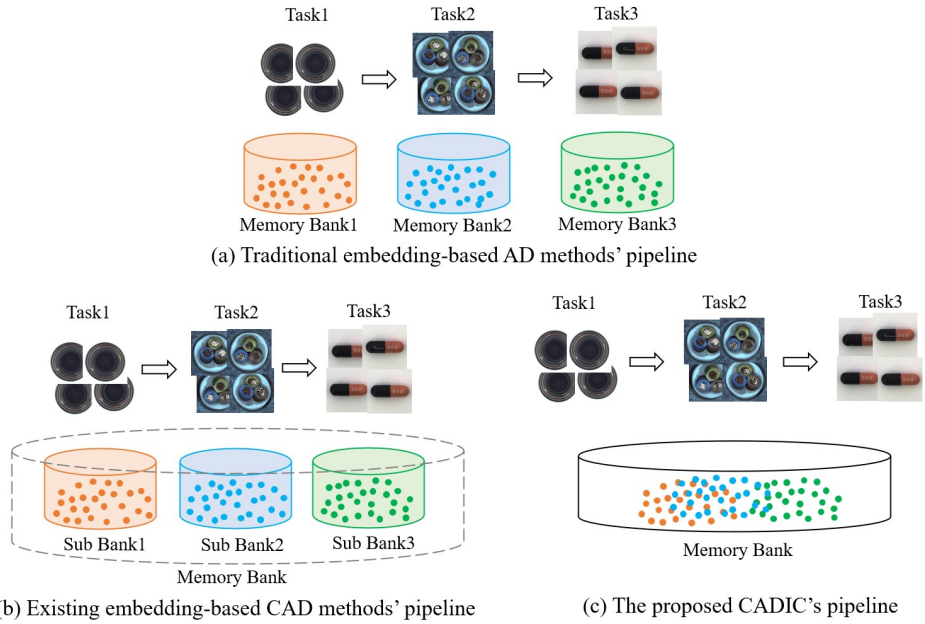
ANOMALY Detection (AD) learns only anomaly-free samples and identify abnormal samples as instances that significantly differ from the learned behavior, playing a crucial role in industrial manufacturing. Most of the existing anomaly detection methods assume that the task is fixed. However, in real-world applications, new products could be added to the production line and there may exist concept drift for the same product during the producing process. These factors decrease the performance of anomaly detection models significantly. Continual Learning (CL) is a learning paradigm proposed to avoid Catastrophic Forgetting. It is also called lifelong learning, incremental learning or continuous learning. In the continual learning setting, the model can learn different tasks continuously. After learning new tasks, the model can still retain knowledge of past tasks.

Continual Anomaly Detection (CAD) is a combination of AD and CL, which aims to learn new AD tasks while retaining performance on previously learned AD tasks. Most of current CAD methods are developed from reconstruction-based AD methods. By distilling previous AD models [1] or retraining on newly updated dataset [2] [3], the model can detect the anomalies of all the tasks based on the reconstruction error. Notably, embedding-based approaches [4] [5] represent another prominent category of AD methods. These approaches utilize frozen feature extractors to store features in a memory bank and detect anomalies based on the similarity between query embeddings and their nearest neighbors in the bank. Compared to reconstruction-based AD methods, embedding-based AD approaches are more stable due to the absence of neural network training processes, and they demonstrate superior anomaly detection performance by identifying subtle anomalies through local patch embeddings. Coincidentally, the replay-based continual learning uses a memory bank to store information of previous tasks. By replaying samples from the memory bank, it keeps the information of past tasks while learn new tasks. Thus, there is a gap between embedding based anomaly detection methods and replay-based continual learning methods, which motivate us to carry out this research.

To address the research gap at the intersection of embedding-based anomaly detection (AD) and continual learning (CL), several methodologies have been proposed. Li et al. [6] introduce the DNE framework, which maintains a memory bank during training to store task-specific embeddings. During inference, Gaussian distributions of past tasks are reconstructed from this bank to identify anomalous inputs through statistical deviation scoring. Liu et al. [7] propose UCAD, which similarly stores multi-task embeddings in a memory bank but enhances it with a trainable prompt module to dynamically infer task types from input images. Anomaly detection is then performed via nearest-neighbor matching against task-specific prototypes. Wu et al. [8] present the DFM method, which unifies embedding extraction and matching within a single model architecture. It employs class-specific adapter modules trained per category while using a frozen backbone for embedding extraction, forming class-conditional embeddings. During inference, the method distinguishes between different classes by retrieving the adapter corresponding to the nearest class-specific embeddings, and directly generates an anomaly map via the Feature Matching Network. While these approaches attempt to reconcile embedding-based AD with CL principles, they share a critical limitation: reliance on class-specific memory banks. This design choice aligns them more closely with conventional AD methods rather than achieving the task-agnostic and incremental learning objectives of CL, thereby leaving the core CL challenges of catastrophic forgetting and task-free adaptation unresolved.

Therefore, we introduce Continual Anomaly Detection Based on Incremental Coreset (CADIC) in this paper. As

Gen Yang, Junfeng Man: Hunan First Normal University, Changsha, China.
Zhipeng Deng: Changsha University of Science and Technology, Changsha, China.
*Correspondence to: 751987807@qq.com



(a) Traditional embedding-based AD methods' pipeline

(b) Existing embedding-based CAD methods' pipeline

(c) The proposed CADIC's pipeline

Fig. 1. Comparison of AD, existing CAD and the proposed CADIC pipeline. (a) Traditional embedding-based AD methods use separate memory banks, each task has its own individual memory bank. (b) Existing embedding-based CAD methods imply only one memory bank, but each task has a specific sub bank. (c) Our embedding-based CAD method (CADIC) utilizes a single memory bank, and all tasks share it.

shown in Fig.1, traditional embedding-based AD methods [4] [5] [12] use separate memory banks, each task has its own individual memory bank. Existing embedding-based CAD methods [6] [7] [8] imply only one memory bank, but each task has a specific sub bank. On the contrary, the proposed CADIC utilizes a single memory bank, and all tasks share it. The main contributions are multifold as follows:

- we propose a novel embedding-based continual anomaly detection method. The method learn new tasks by updating the embeddings in the fixed-size memory bank incrementally, saving most typical embeddings of each task automatically.
- We perform continual anomaly detection experiments on the MVTec AD and VisA dataset. The results show that the proposed method achieves state-of-the-art performance.
- We verify the detection performance of the method for a real-scenario E-paper dataset. The results show that the proposed method can meet the requirements for industrial applications.

II. Related Work

A. Anomaly Detection

AD methods learn solely from anomaly-free samples but can detect various unknown anomalous samples. These methods are split into two main families: reconstruction-based methods and embedding-based methods.

1) Reconstruction-based Methods: The reconstruction-based methods learn to reconstruct normal images during training, and detect anomalies by evaluating the reconstruction errors in the inference stage. Typical methods like AnoGAN [9] and f-AnoGAN [10] reconstruct normal samples with generative adversarial networks. By comparing the difference between the generated samples the input samples, they detect anomalies at the pixel level. Unlike the methods above, which reconstruct normal samples, DRAEM [11] reconstructs simulated anomaly samples. By training the model in an end-to-end manner, it directly outputs the anomaly maps of the input samples. Some other works employ autoencoders, vision transformers, distillation models, and diffusion models as reconstruction frameworks, and achieve competitive detection results.

2) Embedding-based Methods: The embedding-based methods use pre-trained neural networks to extract meaningful embeddings from images, and detect anomalies by analyzing the similarities between embeddings. Typical methods like SPADE [4] and PatchCore [5] save features in a memory bank, and calculate anomaly scores according to the distance between the image embeddings and their nearest neighbors in the memory bank. Works like PaDiM [12] and GAD [13] establish multivariate Gaussian distributions for embeddings stored in the memory bank, and derive anomaly scores for image embeddings by evaluating their Mahalanobis Distances in the distribution. Other works like FastFlow [14] and CFLOW-AD [15] map embeddings into a standard normal distribution with the Normalizing Flow technique. Then, the anomaly scores for image embeddings are computed according to their probabilities in the distribution. In summary, the embedding-based anomaly detection methods use a memory bank to store the normal embeddings, and detect anomalies according to the similarities between embeddings and its nearest neighbors in the memory bank.

B. Continual Learning

In the continual learning setting, models can learn new tasks sequentially without forgetting knowledge of previous tasks. The implementations of continual learning can be divided into three families [16] [17]: parameter isolation methods, regularization methods, and memory replay methods.

1) Parameter isolation methods: In this kind of method, specific network parameters are assigned to specific tasks. The parameters of networks are isolated in two main ways: mask out network parameters or add new networks for new tasks. For the former, PackNet [18] uses weight-based pruning techniques to generate a series of task-specific parameter masks. In the training stage, only mask-free parameters are trained for new tasks. While in the inference stage, only masked parameters are used for prediction. For the latter, Expert Gate [19] uses multiple models to learn multiple tasks. In the training stage, a specialized model and expert gate are trained to learn each task. In the inference stage, the expert gate selects an appropriate model to deal with the task at hand.

2) Regularization methods: For regularization methods, the knowledge of previous tasks is preserved by introduce additional regularization terms. One implementation is to distill knowledge from a past model into the current model. LwF [20] serves as a typical method of this implementation. When training on new tasks, LwF utilizes the output of the previous model as soft label for previous model, translating the knowledge of the former model to the next model, continuously. Another implementation is to penalize the changes of key neural network parameters. EWC [21] is a representative of this method. It assesses the importance of parameters based on their posterior probabilities. When learning new tasks, only the less important parameters are encouraged to adapt to new tasks, while the important ones change little, preserving information from previous tasks effectively.

3) Memory replay methods: This line of works store samples from previous tasks into memory. When learning new tasks, samples in the memory are replayed to avoid forgetting. Samples are replayed in two main ways. First, samples in the memory are reused to train the model. Typically, iCaRL [22] stores a subset of exemplars per class in the training stage, and classifies the testing data according to their distance to the means of all exemplars. Second, samples in the memory are reused to constrain the gradient direction of parameters. Typically, GEM [23] calculates the loss gradient at the memory samples, and constrains the loss gradient at new tasks to the direction that does not decrease the former tasks' performance.

C. Continual Anomaly Detection

Traditional AD methods train separate models for each task, not fit for continuous tasks. CAD applies continual learning techniques into traditional AD methods, aiming to learn new anomaly detection tasks without forgetting old tasks. To lean anomaly detection tasks continuously, Yang et al. [1] propose a reverse distillation-based method for this purpose. When learning a new task, it uses the distillation loss to learn new knowledge from the teacher model in this task. At the same time, it employs pooling distillation loss to learn old knowledge from the student model of the previous task. Pezze et al. [3] introduce a reconstruction-based CAD approach for this purpose. They use a memory module to store old-task's images, and a detection module to reconstruct error for anomaly identification. ReplayCAD [24] leverages compressed semantic embeddings and masks derived from previous tasks to guide a diffusion model in synthesizing samples from old tasks. IUF [25] utilizes singular value decomposition to regulate gradient update directions by projecting gradients onto a subspace orthogonal to the feature representations of old tasks, though this process is computationally expensive. CDAD [26] uses gradient projection to achieve stable continual learning, and it uses an iterative singular value decomposition method based on the transitive property of linear representation to improve computational inefficiency. CFRDC [27] uses a context-aware feature reconstruction model to capture valuable anomaly identification-related knowledge, and utilize an intermediate feature organization strategy to avoid inter-class context conflict. DER [28] dynamically incorporates task-specific adapters and employs learnable prompts to query the corresponding adapter for each task, resulting in successful continual anomaly detection.

III. METHOD

A. Problem Definition

In the CAD paradigm, a unified model is trained non-repetitively on incrementally added tasks and tested on all tasks. To formulate this problem, we assume there are a total of K sequential anomaly detection tasks $\{T_k|k \in (1, 2, ..., K)\}$. Each task T_k corresponds to a dataset D_k. The train set includes only normal samples, while the test set contains both normal and abnormal samples. In each dataset, $X_k \in \mathbb{N}^{H\times W\times 3}$ stands for an image with height H and width W, $Y_k \subset \{0, 1\}^{H\times W}$ indicates whether an image (or a pixel in the image) is normal (0) or abnormal (1). In the training stage, only the k-th task's training samples are used to train model. In the testing stage, test samples of all the previous tasks $\{T_k|k \in (1, 2, ..., K-1)\}$ are used to evaluate the model.

B. Feature extraction

Relevant studies have demonstrated that analyzing the differences between embeddings extracted by pre-trained models can achieve highly accutate anomaly detction. Compared to pre-trained convolutional neural networks (CNNs) [29] [30], pre-trained Vision Transformers (ViTs) [31] excel at capturing global dependencies between arbitrary locations within an image. When employing pre-trained models for anomaly detection, the following two aspects must be considered. First, deeper feature extraction layers. Shallow layers primarily encode low-level texture information, while deeper layers specialize in high-level semantic information. Since normal and abnormal discrimination heavily relies on semantic differences, deeper features provide more accurate anomaly descriptions. Second, larger spatial dimensions of feature maps. Larger feature maps enable finer-grained anomaly maps, resulting in more precise localization at the pixel level and more accurate detection at the image level.

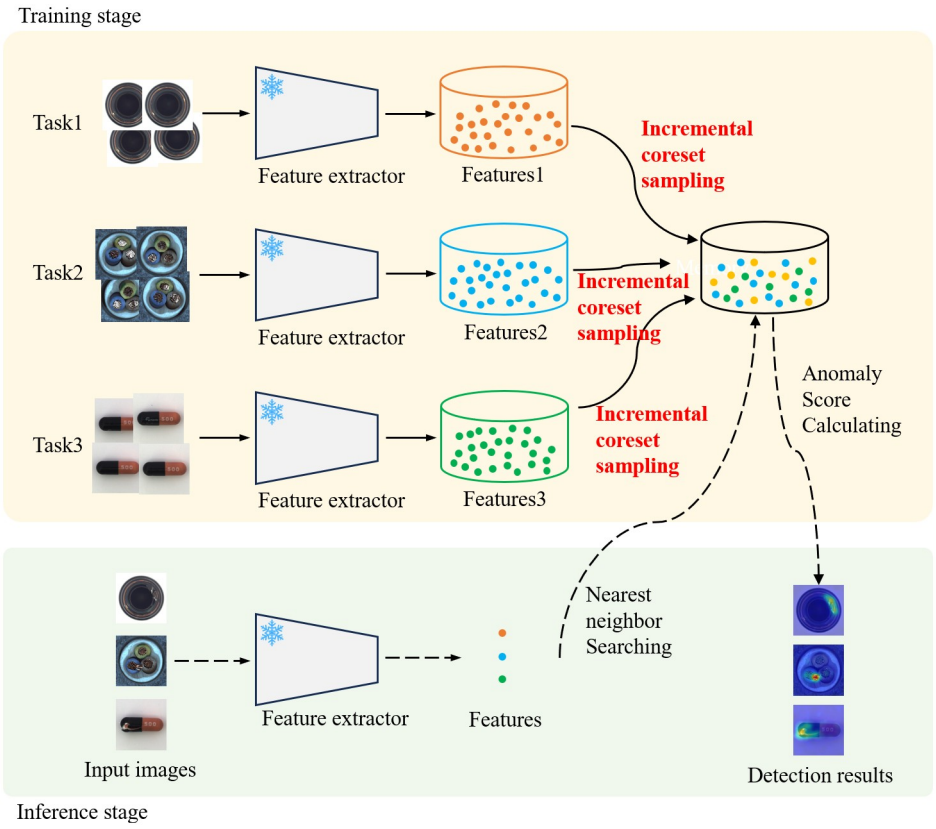


Fig. 2. Framework for the proposed method.

Thus, we choose the *ViT-Base-Patch8-224* model pre-trained on the ImageNet 21k as the feature extractor. The extractor contains 12 transformer layers. For an input image $I \in \mathbb{R}^{224\times224\times3}$, each layer outputs a feature map $X \in \mathbb{R}^{28\times28\times768}$ that contains 28×28 embeddings $x \in \mathbb{R}^{768}$, and each embedding represents a 8×8 patch on the input image I. In this paper, we choose the 9-th layer. By computing anomaly scores for each x, and upsampling the resulting matrix to the spatial dimensions of I, an anomaly map is generated. This anomaly map enables both image-level anomaly detection and pixel-level anomaly localization. Since the local structure of a high-dimensional manifold is homeomorphic to Euclidean space, we compute the anomaly score of an embedding by measuring the distance between the embedding and its nearest neighbors stored in a memory bank.

C. training stage

The greatest challenge that applying continual learning to unsupervised anomaly detection is how to add new tasks' anomaly-free knowledge into the anomaly detection model. To tackle this problem, we propose an incremental coreset updating mechanism, aiming to store typical image embeddings across all tasks into embdeddiong coreset. Inspired by the mini-batch training paradigm, we update the memory bank with new task embeddings through dynamic batch processing. Specifically, when updating memory buffer C with the current batch embeddings X, we firstly compute the maximum distance from each embedding in X to all embeddings in C as follows:

$$d_{max}(\mathbf{X},\mathbf{C}) = \max_{\mathbf{x}\in\mathbf{X}} \{d(\mathbf{x},\mathbf{C})\}. \tag{1}$$

Then identify the corresponding specific feature element x in X that produces this maximum distance as follows:

$$\mathbf{x} = \arg\max_{\mathbf{x}\in\mathbf{X}} \{d(\mathbf{x},\mathbf{C})\}, \tag{2}$$

where $d(x, C)$ is the distance between x and its nearest neighbors in C expressed as follows:

$$d(\mathbf{x},\mathbf{C}) = \min_{\mathbf{c}\in\mathbf{C}} ||x-c||_2. \tag{3}$$

Furthermore, we compute the minimum nearest neighbor distance between elements in memory bank C:

$$|\mathbf{C}|_{\min} = \min_{\mathbf{c_1},\mathbf{c_2}\in\mathbf{C}} \{d(\mathbf{c_1},\mathbf{c_2})\}, \tag{4}$$

and identify the corresponding element pair (c_1, c_2):

$$\mathbf{c_1},\mathbf{c_2} = \arg\min_{\mathbf{c_1},\mathbf{c_2}\in\mathbf{C}} \{d(\mathbf{c_1},\mathbf{c_2})\}. \tag{5}$$

When the condition

$$d_{max}(\mathbf{X},\mathbf{C}) > |\mathbf{C}|_{\min} \tag{6}$$

is satisfied, we add x from X into C, and delete c_1 from C. Subsequently, we iteratively execute the procedures defined in Equations (1)-(5) until the condition (6) is not satisfied, thereby completing the memory bank updating process. This

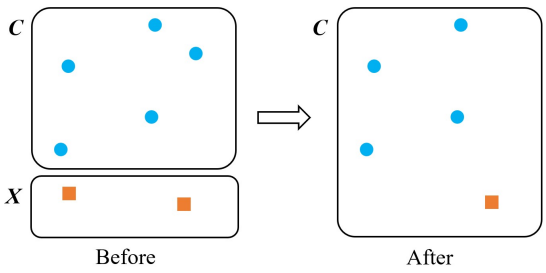


Fig. 3. Illustration of the incremental coreset sampling.

replacement mechanism is universally applied to all tasks' images.

Directly computing the pairwise distances between every vector in X and C is computationally expensive. In this study, the distances between elements in X and C are efficiently calculated using matrix operations, with the formula defined as follows:

$$D = sqrt\{[diag(XX^T)]^T \times \mathbf{1}_{1\times m} + \mathbf{1}_{n\times 1} \times diag(CC^T) - 2XC^T\}, \tag{7}$$

where $diag(\cdot)$ converts the main diagonal elements of a matrix into a row vector while preserving their original order, $sqrt(\cdot)$ computes the arithmetic square root for each element in a matrix independently, $\mathbf{1}_{1\times m}$ is a row vector where all elements are 1, and $\mathbf{1}_{n\times 1}$ is a column vector where all elements are 1.

D. Inference Stage

Since the memory bank only stores normal embeddings, anomaly scores of embeddings can be calculated by measuring their distance to their nearest neighbors in the coreset. A smaller distance indicates greater similarity to normal embeddings, leading to a lower anomaly score; a larger distance suggests less similarity to normal embeddings, resulting in a higher anomaly score. Following PatchCore [5], we calculate the embedding anomaly score as follows:

$$s_{pix} = \min_{\mathbf{c}\in\mathbf{C}} ||x - c||_2. \tag{8}$$

and calculate the image-level anomaly score as follows:

$$s_{img} = \left(1 - \frac{\exp ||x - c^*||_2}{\sum_{c\in\mathcal{N}_b(c^*)} \exp ||x - c||_2}\right) \cdot s^*_{pix}, \tag{9}$$

where s^*_{pix} denotes the largest anomaly score among all embeddings of the test image, c^* is the corresponding embeddings in the memory bank C, and $\mathcal{N}_b(c^*)$ stands for the b embeddings in C that are closest to c^*.

For image-level anomaly detection, the input image is classified as anomalous if its anomaly score s_{img} exceeds a predefined threshold; otherwise, it is deemed normal. For pixel-level anomaly localization, anomaly scores obtained from feature embeddings are spatially upsampled to align with the dimensions of the input image, after which anomalous regions are identified through binarization.

IV. Experiment

The proposed method is evaluated on two widely-used anomaly detection datasets: the MVTech AD [32] and the VisA [33] . The MVTec AD dataset contains 15 subsets, and the VisA dataset contains 12 subsets. Each subset corresponds to a category of object. For each task, the training set contains only anomaly-free samples, while the testing set includes both anomaly-free samples and anomalous samples. We treat each subset as an independent anomaly detection task and order them alphabetically. Thus, there are 15 tasks on the MVTec dataset and 12 tasks on the VisA dataset. The model sequentially learns each task. After completing all tasks, we evaluate the detection metrics across all tasks.

All experiments are conducted on a workstation with an NVIDIA GeForce RTX 4070 GPU, an Intel i5-13600KF CPU, a 32GB of RAM. The Operating System is Windows 10.

A. Evaluation metrics

1) AUROC: The AUROC metric is employed to assess image-level anomaly detection performance in this study. The AUROC value represents the area beneath the ROC curve. The ROC curve is a graphical plot with the FPR (False Positive Rate) on the horizontal axis and the TPR (True Positive Rate) on the vertical axis. Specifically, FPR measures the proportion of false positive samples among all actual negative samples, with lower values indicating better performance. Conversely, TPR quantifies the proportion of true positive samples among all actual positive samples, with higher values being preferable. For a classification model, each classification threshold corresponds to a pair of (FPR, TPR) values. By plotting all such pairs on a coordinate system and sequentially connecting the points, the ROC curve is obtained. AUROC is then calculated as the area under this ROC curve. Thus the AUROC value is threshold-invariant and reflects the intrinsic performance of the model. A higher image-level AUROC (I-AUROC) denotes a better anomaly detection performance.

2) AUPR: The AUPR metric is used to assess pixel-level anomaly location performance in this study. The AUPR value is the area beneath the PR curve. This curve is constructed by plotting P (Precision) on the vertical axis against R (Recall) on the horizontal axis, visualizing the trade-off between precision and recall across varying thresholds. Specifically, precision represents the proportion of correctly identified positive instances among all predicted positive instances, while recall denotes the proportion of correctly identified positive instances among all actual positive instances. By calculating the pair (R, P) at different thresholds and plotting them on the coordinate, the PR curve is obtained. The AUPR value is then derived by calculating the area under this curve. Since normal regions often occupy significantly larger areas than anomalous regions in images, this leads to a class imbalance problem. AUPR emphasizes the accuracy of positive sample predictions, making it more reliable to assess model performance in this class imbalance scenario. A higher pixel-level AUPR (P-AUPR) indicates a better anomaly location performance.

3) FM: The FM (Forgetting Measure) [2] metric evaluates both image-level and pixel-level performance degradation in this study. The FM value which measures the decline of performance on old tasks when learning new tasks, is a crucial indicator in Continual Learning. Mathematically, for the j-th task after the model has been trained up to task k:

$$f_j^k = \max_{l \in \{1,\ldots,k-1\}} a_{l,j} - a_{k,j}, \forall j < k, \tag{10}$$

where a stands for a specific metric, i.e., the image-level AUROC or the pixel-level AUPR in this paper. Then FM is calculated as follows:

$$FM = \frac{1}{k-1} \sum_{j=1}^{k-1} f_j^k. \tag{11}$$

Thus, the FM value indicates the extent of forgetting on prior tasks. A lower FM demonstrates a better continual learning capability.

B. Results and analysis

We conducted a comparison between our method and other similar approaches. The methods are divided into three paradigms: the joint learning paradigm, the fine-tuning paradigm, and the continual learning paradigm. Under the joint learning paradigm, the model is trained on all tasks simultaneously. The performance of joint learning serves as a soft upper bound for continual learning methods, representing an idealized scenario where catastrophic forgetting is completely mitigated. In contrast, the fine-tuning paradigm trains sequentially on each task without any mechanism to retain previous knowledge, resulting in severe performance degradation on earlier tasks. Consequently, fine-tuning results typically establish a practical lower bound for continual learning evaluation. Continual learning adopts an intermediate training strategy to balance these two extremes. While it shares the same learning objective with joint learning, to master all tasks comprehensively, it operates under the same sequential training constraint as fine-tuning, processing only one task at a time. This fundamental trade-off between retaining past knowledge and acquiring new capabilities defines the core challenge of continual learning. For each paradigm, typical methods are implemented on both the MVTec AD and VisA datasets for anomaly detection. In the Joint experiment, the coreset of each task is constructed with the greedy k-center method, the random sampling method, and the proposed incremental coreset method. The total number of features for all tasks is set to 10000. In the fine-tuning experiment, PatchCore, CFA, SimpleNet, and RD4AD are implemented in a fine-tuning paradigm. In the continual anomaly detection experiment, UCAD and the proposed CADIC are implemented. Replay-CAD, currently a state-of-the-art method in continual anomaly detection, demonstrates high-precision capabilities for both image-level anomaly detection and pixel-level anomaly localization. After sequentially learning all tasks, the final model is evaluated on the test sets across all tasks.

The comparison results are presented in Table I to Table IV. From these tables, we can draw the following conclusions. Firstly, the proposed method can construct a coreset incrementally. After building coresets via random sampling, greedy k-center, and our incremental coreset approach, we conduct both image-level and pixel-level anomaly detection. The results demonstrate that both greedy k-center and incremental coreset significantly outperform random sampling, validating the effectiveness of maximizing inter-point distances for feature selection. Our incremental coreset achieves comparable performance to the greedy k-center method while offering dynamic update capabilities for continual learning. In contrast, the static greedy k-center method lacks adaptability to new data streams. Secondly, the proposed method avoids Catastrophic Forgetting effectively. Direct fine-tuning of conventional anomaly detection models leads to severe performance degradation on historical tasks. After training on all 15 MVTec tasks, PatchCore, CFA, SimpleNet, and RD4AD exhibit poor retention of prior knowledge. Our method maintains near-ideal performance across all tasks, demonstrating its ability to retain discriminative features for historical categories while assimilating new data, thus effectively mitigating catastrophic forgetting. Thirdly, the proposed CADIC method achieves SOTA performance in continual anomaly detection. On the MVTec AD dataset, our CADIC achieves 0.972 image-level AUROC and 0.584 pixel-level AUROC, outperforming UCAD's 0.930 and 0.456, respectively. On VisA, we attain 0.8912 image-level AUROC (+1.72% over UCAD) and 0.438 pixel-level AUROC (+13.8% over UCAD). Notably, CADIC achieves 12.8% and 13.8% higher AUPR scores than UCAD on MVTec and VisA, respectively. These results demonstrate that, the proposed CADIC establishes new state-of-the-art benchmark of continual anomaly detection.

Moreover, to analyze the distribution of different category features within the coreset constructed by the proposed method, we generate a t-SNE visualization of the coreset after completing 15 sequential tasks on the MVTec AD dataset. The results are presented in Fig. 4. It shows that features of each category form distinct clusters with notable spatial separation. This segregated spatial distribution facilitates the mapping of image features to their corresponding categorical clusters, thereby enhancing detection accuracy. Moreover, categories with more diverse image content tend to have a larger number of features stored in the memory bank. The three categories with the most features, i.e., hazelnut (1,839), transistor (734), and screw (538), are all structural images with low local repetition and significant intra-class variations. These categories require extensive feature representation to comprehensively capture all normal variations. In contrast, the three categories with the fewest features, i.e., grid (21), carpet (25), and tile (25), are all texture images characterized by high local repetition and strong intra-class similarity. Consequently, only a limited number of features are required to represent all variations within these categories. Thus, the proposed method can construct a coreset with a variable number of features according to the complexity of images in different tasks, which contributes to improving the overall detection performance.

Furthermore, to analyze the time consumption of the proposed method, we plot the time consumption curve as shown in Fig. 5. The x-axis represents the image sequence number, and the y-axis represents the time consumption. The blue

TABLE I
IMAGE-LEVEL METRICS ON MVTEC AD.

Methods	bottle	cable	capsule	carpet	grid	hazelnut	leather	metal_nut	pill	screw	tile	toothbrush	transistor	wood	zipper	average	FM
Joint_PatchCore	1.000	0.977	0.927	1.000	0.983	0.994	1.000	1.000	0.948	0.920	1.000	0.969	0.958	0.997	0.998	0.978	-
Joint_PatchCore(R)	0.998	0.936	0.868	1.000	0.984	0.979	1.000	0.993	0.921	0.653	0.998	0.975	0.832	0.995	0.972	0.940	-
Joint_CADIC	1.000	0.986	0.921	1.000	0.987	0.990	1.000	1.000	0.945	0.899	1.000	0.972	0.975	0.994	0.996	0.978	-
FT_PatchCore	0.163	0.518	0.350	0.968	0.700	0.839	0.625	0.259	0.459	0.484	0.776	0.586	0.341	0.970	0.991	0.602	0.383
FT_CFA	0.309	0.489	0.275	0.834	0.571	0.903	0.935	0.464	0.528	0.528	0.763	0.519	0.320	0.923	0.984	0.623	0.361
FT_SimpleNet	0.938	0.560	0.519	0.736	0.592	0.859	0.749	0.710	0.701	0.599	0.654	0.422	0.669	0.908	0.996	0.708	0.211
FT_RD4AD	0.401	0.538	0.475	0.583	0.558	0.909	0.596	0.623	0.479	0.596	0.715	0.397	0.385	0.700	0.987	0.596	0.393
CFRDC [27]	0.996	0.900	0.785	0.997	0.980	0.994	1.000	0.995	0.933	0.711	0.991	0.933	**0.997**	0.982	0.984	0.945	-
IUF [25]	0.909	0.541	0.520	0.996	0.695	0.875	0.997	0.643	0.547	0.646	0.940	0.711	0.660	0.953	0.795	0.762	0.067
ReplayCAD [24]	0.990	0.957	0.747	0.980	0.927	0.985	0.974	0.995	0.944	0.795	**0.999**	0.981	0.957	0.984	**0.997**	0.948	0.045
DNE [6]	0.990	0.619	0.609	0.984	**0.998**	0.924	1.000	0.989	0.671	0.588	0.980	0.933	0.877	0.930	0.958	0.870	0.116
UCAD [7]	1.000	0.751	0.866	0.965	0.944	0.994	1.000	0.988	0.894	0.739	0.998	**1.000**	0.874	**0.995**	0.938	0.930	**0.010**
DFM [8]	0.997	0.948	**0.996**	**0.999**	0.990	0.977	1.000	1.000	**0.983**	0.765	0.982	0.997	0.932	0.986	0.987	0.969	0.015
CADIC(Ours)	**1.000**	**0.982**	0.877	0.996	0.983	**0.994**	**1.000**	**1.000**	0.942	**0.906**	0.995	0.954	0.968	0.994	0.990	**0.972**	0.011

TABLE II
PIXEL-LEVEL METRICS ON MVTEC AD.

Methods	bottle	cable	capsule	carpet	grid	hazelnut	leather	metal_nut	pill	screw	tile	toothbrush	transistor	wood	zipper	average	FM
Joint_PatchCore	0.820	0.514	0.525	0.770	0.300	0.728	0.224	0.892	0.811	0.336	0.620	0.527	0.637	0.683	0.531	0.594	-
Joint_PatchCore(R)	0.826	0.505	0.510	0.766	0.293	0.712	0.230	0.862	0.785	0.157	0.664	0.561	0.515	0.641	0.565	0.573	-
Joint_CADIC	0.815	0.510	0.519	0.754	0.292	0.744	0.210	0.886	0.815	0.307	0.630	0.530	0.650	0.675	0.528	0.591	-
FT_PatchCore	0.048	0.029	0.035	0.552	0.003	0.338	0.279	0.248	0.051	0.008	0.249	0.034	0.079	0.304	0.595	0.190	0.371
FT_CFA	0.068	0.056	0.050	0.271	0.004	0.341	0.393	0.255	0.080	0.015	0.155	0.053	0.056	0.281	0.573	0.177	0.083
FT_SimpleNet	0.108	0.045	0.029	0.018	0.004	0.029	0.006	0.227	0.077	0.004	0.082	0.046	0.049	0.037	0.139	0.060	0.069
FT_RD4AD	0.055	0.040	0.064	0.212	0.005	0.384	0.116	0.247	0.061	0.015	0.193	0.034	0.059	0.097	0.562	0.143	0.425
CFRDC [27]	0.737	0.518	0.425	0.506	0.243	0.556	0.372	0.666	0.417	0.125	0.454	0.417	0.710	0.380	0.390	0.461	-
IUF [25]	0.289	0.054	0.040	0.440	0.084	0.301	0.330	0.142	0.048	0.012	0.310	0.049	0.065	0.326	0.08	0.171	0.059
ReplayCAD [24]	0.710	0.369	0.337	0.652	**0.338**	0.635	**0.587**	0.656	0.698	**0.329**	0.531	**0.576**	0.605	0.500	**0.539**	0.537	0.055
UCAD [7]	0.752	0.290	0.349	0.622	0.187	0.506	0.333	0.775	0.634	0.214	0.549	0.298	0.398	0.535	0.398	0.456	**0.013**
DFM [8]	0.768	**0.506**	0.241	**0.771**	0.228	0.479	0.432	0.690	0.576	0.242	0.623	0.331	0.501	0.581	0.511	0.511	**0.013**
CADIC(Ours)	**0.790**	0.485	**0.506**	0.753	0.276	**0.749**	0.191	**0.880**	**0.810**	0.328	**0.609**	0.527	**0.650**	**0.686**	0.517	**0.584**	0.015

TABLE III
IMAGE-LEVEL METRICS ON VISA.

Methods	candle	capsules	cashew	chewinggum	fryum	macaroni1	macaroni2	pcb1	pcb2	pcb3	pcb4	pipe_fryum	average	FM
Joint_PatchCore	0.952	0.878	0.940	0.983	0.950	0.902	0.667	0.963	0.892	0.894	0.983	0.995	0.916	-
Joint_PatchCore(R)	0.870	0.750	0.895	0.964	0.883	0.768	0.677	0.949	0.855	0.779	0.921	0.971	0.857	-
Joint_CADIC	0.945	0.825	0.941	0.980	0.954	0.904	0.694	0.967	0.881	0.865	0.972	0.989	0.910	-
FT_PatchCore	0.401	0.605	0.624	0.907	0.334	0.538	0.437	0.527	0.597	0.507	0.588	0.998	0.589	0.361
FT_CFA	0.512	0.672	0.873	0.753	0.304	0.557	0.422	0.698	0.472	0.449	0.407	0.998	0.593	0.327
FT_SimpleNet	0.504	0.474	0.794	0.721	0.684	0.567	0.447	0.598	0.629	0.538	0.493	0.945	0.616	0.283
FT_RD4AD	0.380	0.385	0.737	0.539	0.533	0.607	0.487	0.437	0.672	0.343	0.187	**0.999**	0.525	0.423
IUF [25]	**0.994**	0.692	0.758	0.548	0.677	0.795	0.606	0.563	0.766	0.651	0.512	0.614	0.681	0.085
ReplayCAD [24]	0.924	0.843	0.937	0.961	0.915	**0.889**	**0.805**	0.911	0.849	0.831	**0.978**	0.991	**0.903**	0.055
DNE [6]	0.486	0.413	0.735	0.585	0.691	0.584	0.546	0.633	0.693	0.642	0.562	0.747	0.610	0.179
UCAD [7]	0.778	**0.877**	**0.960**	0.958	**0.945**	0.823	0.667	0.905	0.871	0.813	0.901	0.988	0.874	**0.039**
CADIC(ours)	0.859	0.817	0.926	**0.972**	0.910	0.839	0.733	**0.946**	**0.878**	**0.869**	0.952	**0.993**	0.891	0.043

curve illustrates the time consumption of each single image, and the orange points denote the start time of each category. From the figure, we can conclude the following results. In general, learning individual images from earlier tasks requires more time compared to those from later tasks. As shown in the figure, the earlier tasks like bottle, cable, and capsule take longer than later tasks such as toothbrush, transistor, and zipper. This phenomenon arises because during early tasks, the features in the coreset are too close to each other, leading to frequent updates. While in the later tasks, the distances between adjacent features in the coreset are large enough, leading to fewer element updates. In detail, the processing time of each category decreases rapidly. It shows that, for any specific category, the processing times for the initial few images are significantly longer, but they decrease rapidly afterward. This occurs because at the beginning of learning each category, the coreset contains few representative features of the category. As a result, a substantial amount of features

TABLE IV
PIXEL-LEVEL METRICS ON VISA.

Methods	candle	capsules	cashew	chewinggum	fryum	macaroni1	macaroni2	pcb1	pcb2	pcb3	pcb4	pipe_fryum	average	FM
Joint_PatchCore	0.206	0.617	0.717	0.768	0.485	0.056	0.015	0.790	0.084	0.561	0.359	0.620	0.440	-
Joint_PatchCore(R)	0.213	0.581	0.685	0.770	0.476	0.029	0.010	0.770	0.070	0.496	0.293	0.637	0.419	-
Joint_CADIC	0.196	0.618	0.716	0.771	0.490	0.054	0.015	0.793	0.088	0.573	0.349	0.610	0.439	-
FT_PatchCore	0.012	0.007	0.055	0.315	0.082	0.000	0.000	0.008	0.004	0.007	0.010	0.585	0.090	0.311
FT_CFA	0.017	0.005	0.059	0.243	0.085	0.001	0.001	0.013	0.006	0.008	0.015	0.592	0.087	0.184
FT_SimpleNet	0.001	0.004	0.017	0.007	0.047	0.000	0.000	0.013	0.003	0.004	0.009	0.058	0.014	0.016
FT_RD4AD	0.002	0.005	0.061	0.045	0.098	0.001	0.001	0.013	0.008	0.008	0.013	0.576	0.069	0.201
IUF [25]	0.012	0.017	0.043	0.033	0.107	0.011	0.004	0.019	0.009	0.018	0.021	0.117	0.034	**0.003**
ReplayCAD [24]	**0.241**	0.430	0.555	0.674	0.462	**0.178**	**0.099**	0.793	**0.199**	0.422	0.303	0.625	0.415	0.050
UCAD [7]	0.067	0.437	0.580	0.503	0.334	0.013	0.003	0.702	0.136	0.266	0.106	0.457	0.300	0.015
CADIC(ours)	0.193	**0.631**	**0.718**	**0.763**	**0.476**	0.047	0.019	**0.794**	0.087	**0.559**	**0.337**	**0.631**	**0.438**	0.014

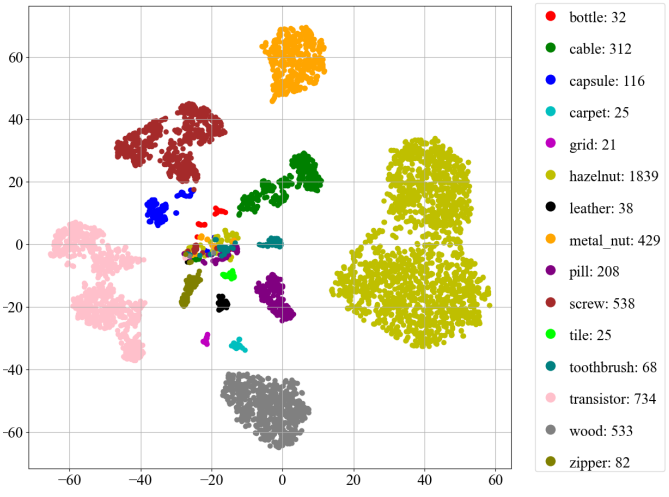


Fig. 4. t-SNE visualization of features sampled from the MVTec AD dataset

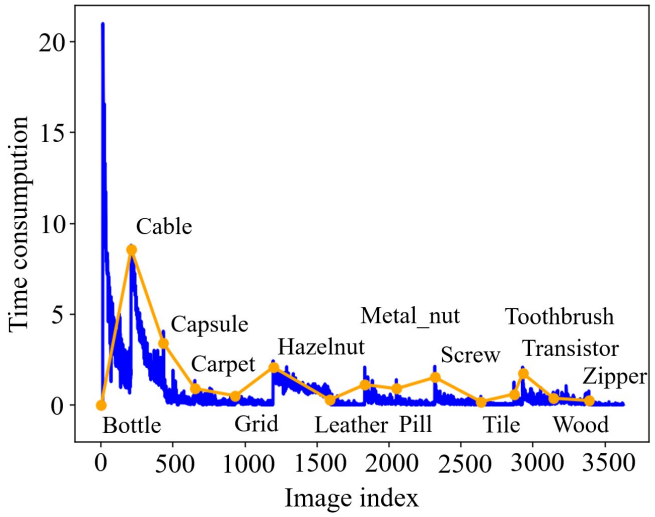


Fig. 5. Time consumption curve for every 100 images sampled. The horizontal coordinate axis is the image batch number (each batch corresponds to 100 images), and the vertical coordinate axis is the time consumed (seconds).

from images of this category are incorporated into the coreset. However, as the features of this category gradually increase, it becomes increasingly difficult for features of this category to be incorporated into the coreset. The reason why the time consumption for the first few images of "Bottle" is close to zero is that the features of these images are directly placed into the coreset for initialization. Moreover, learning complex images takes longer than learning simple ones. As can be seen from the figure, the learning time of complex images such as hazelnut, screw, and transistor is longer than that of simpler images like grid, carpet, and tile. This is because the more complex an image is, the greater the differences among its features, necessitating to preserving more features in the coreset. Conversely, the simpler an image is, the smaller the differences among its features, with only a few highly representative features being likely to enter the coreset.

Fig. 6 illustrates the anomaly localization results of our method on learned tasks after completing the training of a new task on the MVTec AD and VisA datasets. We can observe the following two key points. Firstly, the localization results on old tasks remain largely unchanged even after learning multiple new tasks. These results indicate that the coreset constructed by CADIC effectively preserves the most critical historical knowledge. Secondly, CADIC demonstrates excellent localization capability for subtle defects, reflecting its robustness in perceiving fine-grained features.

C. Ablation Study

In this section, we conduct comprehensive ablation experiments on MVTec AD and VisA benchmarks.

1) Extractor: We perform ablation experiments to explore the influence of different embedding extractors on the continual learning ability of our method. *ResNet50* has been proven by numerous prior works to be an effective feature extractor. By concatenating feature maps of the second and the third block, the embeddings are obtained. *ViT-base-patch16* and *ViT-base-patch8* are both transformer-based neural networks. For *ViT-base-patch16*, each embedding in the feature map corresponds to a 16×16 patch of the original image, resulting in coarser representations; while for *ViT-base-patch8*, each embedding corresponds to an 8×8 patch, yielding finer representations. Table V shows the performance variations across these three extractors. It can be observed that the *ViT-base-patch8* achieves optimal performance in both anomaly detection and localization tasks. Thus, we choose *ViT-base-patch8* as the embedding extractor.

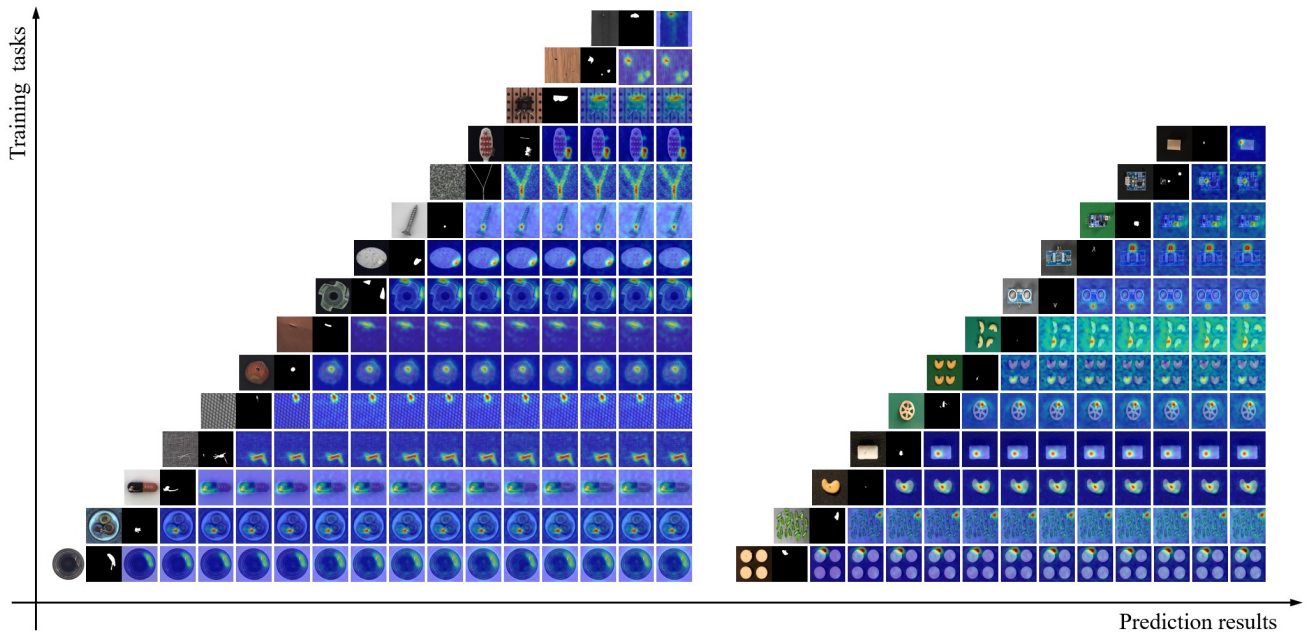


Fig. 6. The results of anomaly localization of CADIC in MVTec AD and VisA dataset, respectively.

TABLE V
PERFORMANCE COMPARISON OF DIFFERENT EXTRACTORS

Extractor	MVTec AD		VISA	
	I-AUROC	P-AUPR	I-AUROC	P-AUPR
WideResNet50	0.949	0.498	0.831	0.370
ViT-base-patch16	0.940	0.493	0.844	0.350
ViT-base-patch8	**0.972**	**0.584**	**0.891**	**0.438**

TABLE VI
PERFORMANCE COMPARISON OF DIFFERENT LAYERS

Layer	MVTec AD		VISA	
	I-AUROC	P-AUPR	I-AUROC	P-AUPR
1	0.836	0.515	0.722	0.261
3	0.904	0.591	0.799	0.363
5	0.932	0.587	0.831	0.391
7	0.947	**0.612**	0.881	**0.446**
9	**0.972**	0.584	**0.891**	0.438
11	0.946	0.535	0.870	0.404

2) Layer: We also examined the effectiveness of different layers of *ViT-base-patch8*. The network comprises 12 feature extraction layers, each capable of extracting feature maps with dimensions of $28 \times 28 \times 768$. For simplicity, we compare only the performance of odd-numbered layers. The results are shown in the TableVI. It shows that as the layers get deeper, the extracted features become increasingly effective in our method. However, when layers are too close to the output layer, the performance declines. Specifically, the 7th layer achieves the best pixel-level metrics, while the 9th layer excels in image-level metrics. Compared to pixel-level anomaly localization, we prioritize image-level anomaly detection. Therefore, we select the 9th layer as the feature extraction layer in this study.

3) Coreset size : Finally, we study the impact of memory bank size on the model's continual learning capability. Table VII shows there is a significant positive correlation between coreset size and model performance. When the coreset size is small, the model trains faster but exhibits lower anomaly detection metrics. At a coreset size of 20,000, the model achieves the best anomaly detection performance, but its training time becomes excessively long. When the coreset size is 10,000, the model achieves both relatively high detection metrics and acceptable training time. Therefore, this paper sets the coreset size to 10,000.

D. Further study

To investigate CADIC's performance on ultra-long-term anomaly detection tasks, we further conducted continual anomaly detection experiments on a hybrid dataset composed of MVTec and VISA. The results are shown in the table VIII. It shows that, after continual learning across all 27 tasks, CADIC still achieves competitive metrics, with an average image-level AUROC of 0.921 and an average pixel-level AUPR of 0.492. This experiment demonstrates that the proposed CADIC exhibits excellent detection performance in ultra-long-term anomaly detection tasks.

E. Application

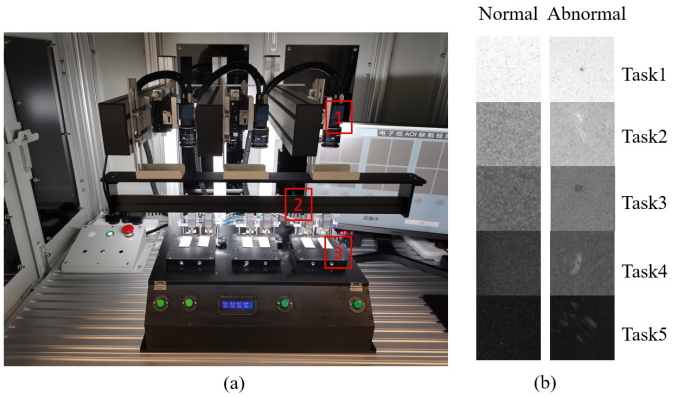
To further demonstrate the practical value of CADIC, we apply CADIC to detect the anomalies of real-world Electronic Paper (E-paper). Fig. 7(a) shows the automatic optical inspection platform. Based on this platform, we collect a large number of electronic paper images and construct an electronic paper anomaly detection dataset. The dataset includes 750 normal images and 250 abnormal images. For E-paper, different defects are only visible under specific grayscale.

TABLE VII
PERFORMANCE COMPARISON OF DIFFERENT CORESET SIZE

Coreset size	MVTec AD				VISA			
	I-AUROC	P-AUPR	Training Time (seconds)	Inference Speed (FPS)	I-AUROC	P-AUPR	Training Time (seconds)	Inference Speed (FPS)
2500	0.955	0.559	**63**	**7.9**	0.828	0.415	**79**	**6.8**
5000	0.969	0.577	379	7.8	0.856	0.427	474	6.4
10000	0.972	0.584	2,503	6.6	0.891	0.438	3,165	6.5
20000	**0.978**	**0.593**	17,038	6.4	**0.916**	**0.439**	22,254	6.5

TABLE VIII
ULTRA-LONG-TERM CONTINUAL ANOMALY DETECTION PERFORMANCE

Category	bottle	cable	capsule	carpet	grid	hazelnut	leather	metal nut	pill	-	-
I-AUROC	0.998	0.984	0.887	0.995	0.967	0.999	1.000	1.000	0.957	-	-
P-AUPR	0.795	0.502	0.476	0.760	0.224	0.710	0.136	0.834	0.803	-	-
Category	screw	tile	toothbrush	transistor	wood	zipper	candle	capsules	cashew	-	-
I-AUROC	0.901	0.995	0.908	0.949	0.990	0.968	0.852	0.846	0.871	-	-
P-AUPR	0.277	0.543	0.508	0.643	0.650	0.403	0.203	0.575	0.766	-	-
Category	chewinggum	fryum	macaroni1	macaroni2	pcb1	pcb2	pcb3	pcb4	pipe_fryum	average	FM
I-AUROC	0.993	0.902	0.764	0.702	0.922	0.809	0.827	0.909	0.989	0.921	0.023
P-AUPR	0.721	0.472	0.026	0.009	0.794	0.081	0.436	0.305	0.644	0.492	0.011



(a) (b)

Fig. 7. (a) Automatic optical inspection platform. 1: Camera, 2: Light source, 3: The inspected E-paper. (b) Five tasks from the E-paper dataset

Therefore, we partition the dataset into five independent tasks according to the grayscale levels of the images. Each task contains 100 normal images for training, 50 normal images and 50 abnormal images for inference. Fig. 7(b) illustrates typical images of each task. We use CADIC to learn the five tasks sequentially, and the detection results are shown in table IX. We observe that CADIC shows outstanding performance across all five tasks. Notably, almost all the tasks' image-level AUROC are 1.0. The results indicate that CADIC achieves perfect classification performance in image-level anomaly discrimination. Moreover, Fig. 8 shows the localization result of the five tasks. we can see that CADIC can localize various types of defects across different tasks effectively. This experiment demonstrates that CADIC exhibits strong practicality for industrial application.

V. CONCLUSION

In this study, we present a novel continual anomaly detection method termed CADIC, which learns sequential tasks through incremental memory bank updates. By employing the pre-trained high-resolution ViT-Base-Patch8-224 backbone as the feature extractor, we achieve highly refined embeddings with enhanced contextual awareness. We introduce a mini-batch updating mechanism that constructs memory banks of multiple tasks incrementally, enabling effective anomaly detection in a continual learning setting. Experiments are conducted on both the MVTec AD dataset and the VisA dataset. The results show that the proposed method surpasses existing SOTA continual anomaly detection methods at both image-level and pixel-level. Our method is intuitive, easy to implement, and highly accurate, demonstrating strong practical applicability.

In future work, we aim to further optimize the memory bank update mechanism. The proposed method involves iteratively computing the Euclidean distances between features in the current update batch and those stored in the memory bank. The overall computation costs is high for large-scale memory banks. Therefore, we will investigate more efficient updating strategies to accelerate the memory bank updating processes.

REFERENCES

[1] A. Yang, X. Xu, Y. Wu, and H. Liu, "Reverse Distillation for Continuous Anomaly Detection," IEEE Trans. Instrum. Meas., vol. 73, pp. 1–13, 2024, doi: 10.1109/TIM.2024.3440374.
[2] Chaudhry, A.; Dokania, P. K.; Ajanthan, T.; and Torr, P. H. 2018. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In Proceedings of the European conference on computer vision, 532–547.
[3] Davide Dalle Pezze, "Continual learning approaches for anomaly detection," arxiv, 2022.
[4] N. Cohen, Y. Hoshen, Sub-image anomaly detection with deep pyramid correspondences (2021). arXiv: 2005.02357.

TABLE IX
PERFORMACE OF THE PROPOSED CADIC IN THE E-PAPER DATASETS

Measurements	task1	task2	task3	task4	task5	average	FM
I-AUROC	1.000	1.000	1.000	1.000	0.999	1.000	0
P-AUPR	0.490	0.555	0.596	0.592	0.371	0.521	0.010

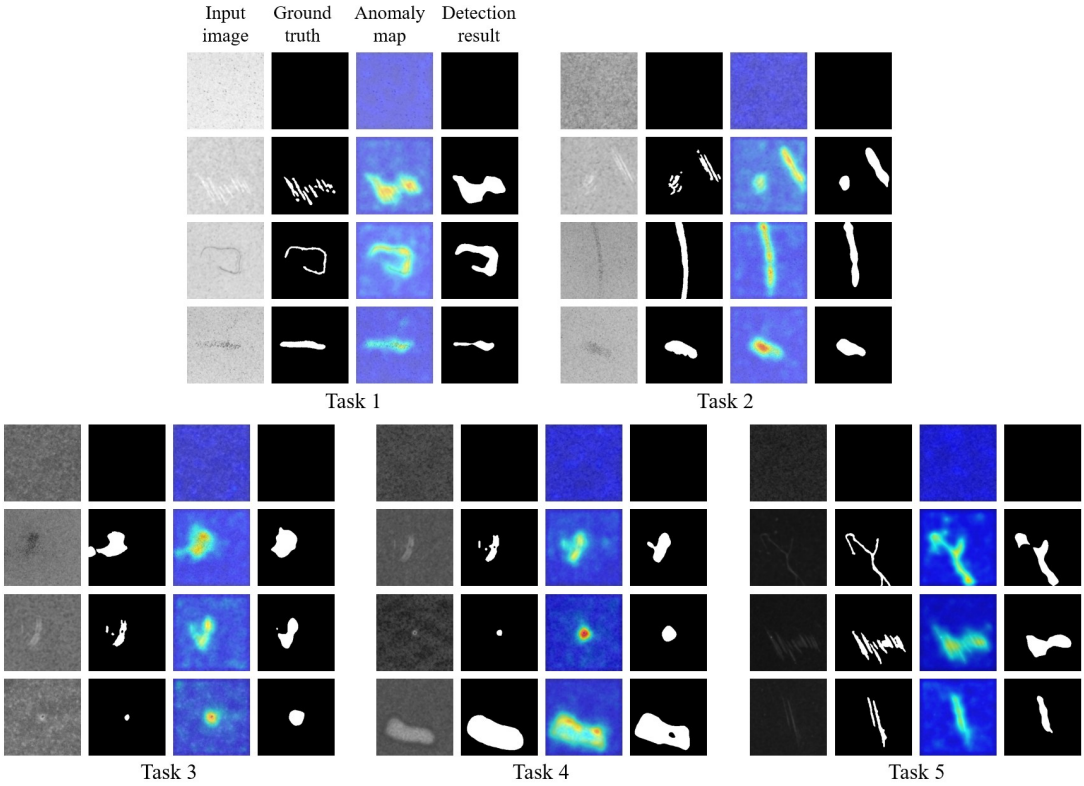


Fig. 8. The results of anomaly localization of CADIC in E-paper dataset.

[5] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, P. Gehler, Towards total recall in industrial anomaly detection, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 14318–14328.

[6] W. Li et al., "Towards Continual Adaptation in Industrial Anomaly Detection," in *Proc. ACM Int. Conf. Multimedia*. 2022, pp. 2871–2880.

[7] J. Liu et al., "Unsupervised Continual Anomaly Detection with Contrastively-Learned Prompt," in *Proc. AAAI Conf. Artif. Intell.*, 2024, pp. 3639–3647.

[8] S. Wu et al., "DFM: Differentiable Feature Matching for Anomaly Detection," 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 2025, pp. 15224-15233, doi: 10.1109/CVPR52734.2025.01418.

[9] Thomas Schlegl et al,"Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery", arXiv preprint arXiv:1703.05291, 2017.

[10] .SCHLEGL T, SEEBCK P, WALDSTEIN S M, et al. F-anogan: fast unsupervised anomaly detection with generative adversarial networks[J]. Medical Image Analysis, 2019, 54: 30-44.

[11] V. Zavrtanik, M. Kristan, and D. Skocaj, "DRÆM – A discriminatively trained reconstruction embedding for surface anomaly detection," in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada: IEEE, Oct. 2021, pp. 8310–8319. doi: 10.1109/ICCV48922.2021.00822.

[12] T. Defard, A. Setkov, A. Loesch, and R. Audigier, "PaDiM: A Patch Distribution Modeling Framework for Anomaly Detection and Localization," in *Proc. Int. Conf. Pattern Recognit.*, 2021, pp. 475–489.

[13] LIN J, CHEN S, LIN E, et al. Deep feature selection for anomaly detection based on pretrained network and gaussian discriminative analysis[J]. IEEE Open Journal of Instrumentation and Measurement, 2022, 1: 1-11.

[14] YU J, ZHENG Y, WANG X, et al. FastFlow: Unsupervised anomaly detection and localization via 2d normalizing flows[DB/OL]. (2021-10-16)[2024-03-21]. https://arxiv.org/abs/2111.07677.

[15] D. Gudovskiy, S. Ishizaka, K. Kozuka, Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 98–107

[16] M. Delange et al., "A continual learning survey: Defying forgetting in classification tasks," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–1, 2021, doi: 10.1109/TPAMI.2021.3057446.

[17] B. Wickramasinghe, G. Saha, and K. Roy, "Continual Learning: A Review of Techniques, Challenges, and Future Directions," *IEEE Trans. Artif. Intell.*, vol. 5, no. 6, pp. 2526–2546, Jun. 2024, doi: 10.1109/TAI.2023.3339091.

[18] A. Mallya and S. Lazebnik, "PackNet: Adding Multiple Tasks to a Single Network by Iterative Pruning," in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT: IEEE, Jun. 2018, pp. 7765–7773. doi: 10.1109/CVPR.2018.00810.

[19] R. Aljundi, P. Chakravarty, and T. Tuytelaars, "Expert Gate: Lifelong Learning with a Network of Experts," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI: IEEE, Jul. 2017, pp. 7120–7129. doi: 10.1109/CVPR.2017.753.

[20] Z. Li and D. Hoiem, "Learning without Forgetting," IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 12, pp. 2935–2947, Dec. 2018, doi: 10.1109/TPAMI.2017.2773081.

[21] J. Kirkpatrick et al., "Overcoming catastrophic forgetting in neural networks," Proc. Natl. Acad. Sci. U.S.A., vol. 114, no. 13, pp. 3521–3526, Mar. 2017, doi: 10.1073/pnas.1611835114.

[22] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental Classifier and Representation Learning," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu,

HI: IEEE, Jul. 2017, pp. 5533–5542. doi: 10.1109/CVPR.2017.587.
[23] Lopez-Paz D, Ranzato M A. Gradient episodic memory for continual learning[J]. Advances in neural information processing systems, 2017, 30.
[24] L. Hu et al., "ReplayCAD: Generative Diffusion Replay for Continual Anomaly Detection," in *Proc. 34th Int. Joint Conf. Artif. Intell.*, 2019.
[25] J. Tang et al., "An Incremental Unified Framework for Small Defect Inspection," in *Proc. Eur. Conf. Comput. Vis.*, 2025, pp. 307–324.
[26] X. Li et al., "One-for-More: Continual Diffusion Model for Anomaly Detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 4766-4775.
[27] J. Pang and C. Li, "Context-aware feature reconstruction for class-incremental anomaly detection and localization," *Neural Networks*, vol. 181, p. 106788, Jan. 2025, doi: 10.1016/j.neunet.2024.106788.
[28] Y. Cai, et al., "Dynamic Expert Routing for Unsupervised Continual Anomaly Detection," *IEEE Trans. Ind. Inf.*, pp. 1–10, 2025.
[29] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.
[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 770–778. doi: 10.1109/CVPR.2016.90.
[31] A. Dosovitskiy et al., "AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE," 2021.
[32] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "MVTec ADA comprehensive real-world dataset for unsupervised anomaly detection," in *Proc. IEEE/CVF Conf. Computer Vis. Pattern Recognit.*, 2019, pp. 9592–9600.
[33] Y. Zou, J. Jeong, L. Pemula, D. Zhang, and O. Dabeer, "Spot-the-difference self-supervised pre-training for anomaly detection and segmentation," in Proc. Eur. Conf. Comput. Vis., 2022, pp. 392–408.