DeCoFlow: Structural Decomposition of Normalizing Flows for Continual Anomaly Detection

Hun Im[1], Jungi Lee[1], Subeen Cha[1], Pilsung Kang[1*]
[1]Korea University, Seoul, Republic of Korea
{corresponding_email}@korea.ac.kr

Abstract

Continual anomaly detection (CAD) requires learning to identify defects across sequentially arriving product categories while preserving detection accuracy on previously learned categories. Existing approaches face an *isolation-efficiency dilemma*: hard parameter isolation prevents forgetting but incurs linear model growth, while efficient adaptation methods suffer from density manifold interference.

We observe that normalizing flows (NF) possess a unique *Arbitrary Function Property* in their coupling layers—the mathematical guarantee that invertibility is preserved regardless of subnet implementation—which uniquely enables *zero forgetting* through safe parameter decomposition. Based on this insight, we propose DeCoFlow, a framework that decomposes coupling layer subnets into frozen shared bases and task-specific low-rank adapters (LoRA), achieving complete parameter isolation with only 8% additional parameters per task.

To compensate for frozen base rigidity, we introduce three integral components: Task-Specific Alignment (TSA) for distribution alignment, Tail-Aware Loss (TAL) for decision boundary focus, and Affine Coupling Layer (ACL) for nonlinear manifold correction.

On MVTec-AD with 15 sequential tasks, DeCoFlow achieves 98.05% Image-AUC and 55.80% Pixel-AP with *zero forgetting* (FM=0.0), establishing new state-of-the-art in replay-free continual anomaly detection.

Keywords: Continual Learning, Anomaly Detection, Normalizing Flows, Parameter-Efficient Fine-Tuning, Low-Rank Adaptation

1 Introduction

Background: The Need for Continual AD. Deep learning-based anomaly detection (AD) has evolved from single-class classification (One-Class Classification) learning only normal data distributions to multi-class approaches. However, this assumes a static environment where all data is collected at training time—a significant limitation.

Real manufacturing environments are inherently dynamic: new product classes arrive sequentially. Accumulating all historical data for retraining is impractical due to exponentially growing storage costs and privacy constraints. Consequently, models must learn new tasks without access to previous data, facing the **catastrophic forgetting** problem—the rapid loss of previously learned knowledge that undermines system reliability.

Limitations of Existing Methods. Unlike classification models that only need to adjust decision boundaries, anomaly detection—particularly Normalizing Flow-based methods—must precisely estimate the probability density of normal data. Even minor parameter interference can collapse the likelihood manifold, causing fatal performance degradation. Therefore, merely *mitigating* forgetting is insufficient for AD.

Many SOTA methods (CADIC, ReplayCAD, etc.) adopt replay-based approaches, but data memory costs scale with task count, compromising scalability. Limited buffer sizes cannot cover the complex tail distributions of high-dimensional data, resulting in biased density estimation.

Parameter isolation methods (SurpriseNet, Continual-MEGA) attempt to physically separate parameters to avoid forgetting. However, allocating independent networks per task causes linear model growth, or dividing fixed capacity leads to capacity saturation.

Attempts to apply prompts or adapters (UCAD, DER, TAPD) in feature space still require separate memory banks or additional routing processes, failing to achieve complete efficiency.

1.1 The Isolation-Efficiency Trade-off

Existing Continual AD research faces a trade-off between Isolation (forgetting prevention) and Efficiency (cost effectiveness):

- Complete isolation for performance → model costs explode
- Efficient isolation → precision degradation and aux-

iliary costs

This research proposes a novel solution leveraging NF's structural properties: achieving **No Replay** and **Parameter Efficiency** simultaneously.

1.2 Key Insight: Theoretical and Empirical Foundations

Insight 1: Structural Suitability of Normalizing Flows. Based on the idea that "only the parts requiring change need to be separated, not the entire model," we adopt a strategy decomposing parameters into shared components (w_{shared}) and task-specific deltas (ΔW_{task}):

$$w_{task} = w_{shared} + \Delta w_{task} \tag{1}$$

For this decomposition to work, it must not compromise the model's mathematical integrity. Memory Bank methods directly lead to replay problems. Reconstruction-based methods (AE, VAE) suffer from encoder-decoder coupling that breaks latent consistency under decomposition. Teacher-Student methods require precise feature alignment that becomes unstable with partial freezing.

NF's Structural Suitability (Arbitrary Function Property). In contrast, NF's Affine Coupling Layer:

$$y_2 = x_2 \odot \exp(s(x_1)) + t(x_1) \tag{2}$$

structurally guarantees the invertibility and efficient Jacobian computation of the entire transformation, regardless of how complex the scale (s) and translation (t) functions are.

We reinterpret this **Arbitrary Function Property** as a structural safeguard for efficient isolation. Decomposing subnets into [Frozen Base $+ \Delta W$] form maintains theoretical validity completely. NF's coupling layer is the most suitable model for decomposition as it structurally guarantees invertibility of the entire transformation regardless of subnet implementation.

Insight 2: Intrinsic Low-Rank Nature of Adaptation (Empirical). Even if structural decomposition is possible, efficiency cannot be achieved if ΔW must be large. We sought to answer "How large must ΔW be?"

Noting that ImageNet-pretrained backbones already extract meaningful task-agnostic representations, NF's role is not learning new features but mapping given space to Gaussian distribution—a structural transformation.

The basic skeleton of this transformation can be shared across tasks, and task-specific differences should be sufficient with Low-Rank. SVD analysis of weight changes during actual training confirmed that over 74% of total energy concentrates in the top 64 ranks. Rank ablation results show negligible performance differences (0.02%) for ranks 16-128, confirming task adaptation is intrinsically low-dimensional.

1.3 Proposed Method: DeCoFlow

Coupling-level Decomposition. DeCoFlow takes an approach modifying only the transformation functions within subnets at NF's Coupling-level. The scale (s) and translation (t) of affine coupling layers are decomposed as:

$$s(x) = s_{base}(x) + \Delta s_{task}(x), \quad \Delta s = B \cdot A \tag{3}$$

Since $s_{base}(x) + \Delta s_{task}(x)$ is still "one function," NF's Arbitrary Function Property ensures mathematical preservation of invertibility and efficient Jacobian computation.

By structuring coupling subnets with Frozen Base and Task-specific LoRA, freezing base weights after Task 0 as a structural anchor shared by all tasks, subsequent tasks train only low-rank LoRA modules. This achieves complete isolation through physical preservation of s_{base} with only **8% parameter addition per task**.

Integral Components for Structural Compensation. While Base Freeze and Low-rank structure secure stability and efficiency, structural gaps remain in capturing distribution differences and fine tail distributions. We introduce TSA (distribution alignment), TAL (tail learning), and ACL (local expressiveness enhancement) to compensate.

1.4 Contributions

1. Demonstrating parameter decomposition is possible through NF's Arbitrary Function Property
2. **DeCoFlow Framework**: An NF-based Continual AD framework achieving practical parameter isolation without data storage through Coupling-level LoRA adaptation. Enables forgetting-free scaling with only 8% parameter addition per task
3. Integrating distribution alignment (TSA), tail learning (TAL), and local expressiveness (ACL) to improve Low-rank adaptation's performance ceiling by 3%p
4. Achieving 98.05% I-AUC on MVTec AD 15-class sequential learning (+3.35%p over existing SOTA)

2 Related Works

2.1 Unified Multi-class Anomaly Detection

Paradigm Shift to Unified Models. Early AD research focused on One-Class Settings training individual models per product class, but this dramatically increases management costs in multi-product environments. UniAD (You et al., NeurIPS 2022) and OmniAL (Zhao et al., CVPR 2023) marked the transition to Unified (Multi-class) AD paradigm—learning multiple classes with a single model. Recent works include MambaAD (He et al., NeurIPS 2024) utilizing State Space Models, and DiAD (He et al., AAAI 2024) and LafitE (Yao et al., 2023) attempting Diffusion-based approaches.

Challenges: Interference & Identity Mapping. When a single model simultaneously learns disparate data distributions, 'Inter-class Interference' occurs (Lu et al., arXiv 2024). Reconstruction-based models are particularly vulnerable to Identity Mapping—reconstructing anomalies unchanged. DecAD (Wang et al., ICCV 2025) addresses this with complex contrastive learning, while Revitalizing Reconstruction (Fan et al., arXiv 2024) requires Latent Disentanglement.

Normalizing Flow. Unlike reconstruction-based models, NF directly optimizes likelihood without reconstructing inputs, structurally avoiding Identity Mapping. This aligns with Dinomaly's (Guo et al., CVPR 2025) "Less is More" philosophy. FastFlow (Yu et al., CVPR 2022), MSFlow (Zhou et al., TPAMI 2024) effectively model spatial context and multi-scale features, proving NF as a powerful AD framework.

However, early NF models were mostly designed for One-class, suffering from distribution overlap when forcing multiple classes to a single normal distribution $\mathcal{N}(0, 1)$. Recent research addresses this through structural advancement:

- **HGAD** (Yao et al., ECCV 2024): Models latent space as hierarchical GMM for probabilistic class-wise distribution separation
- **VQ-Flow** (Zhou et al., arXiv 2024): Introduces hierarchical vector quantization to precisely map multi-modal characteristics to discrete codebooks

Paradigm to Continual Learning. However, HGMNF's GMM component count or VQ-Flow's codebook size are fixed at training initialization, accepting only classes within predefined capacity. This static structure causes 'Structural Mismatch' with heterogeneous data distributions when new tasks arrive. Forcing shared parameter updates inevitably triggers 'Catastrophic Forgetting.' Consequently, adapting to continuously arriving new product lines requires the ability to flexibly learn new distributions while preserving existing knowledge—naturally demanding a transition to Continual Learning paradigm.

2.2 Continual Learning

Continual Learning addresses the fundamental dilemma between plasticity (learning new knowledge) and stability (preserving old knowledge) in non-stationary environments with sequential data arrival (McCloskey & Cohen, 1989; French, 1999).

Limitations of Existing Research. Early research attempted imperfect trade-offs from a 'balance' perspective. Replay-based methods store past data for stability but face memory constraints and privacy issues. Regularization-based methods like EWC (Kirkpatrick et al., 2017) suppress important parameter changes for stability but sacrifice plasticity.

Structural Isolation & PEFT. Recently, Parameter-Efficient Fine-Tuning (PEFT) attempts structural decoupling rather than compromise. LoRA (Hu et al., ICLR 2022) freezes pre-trained weights while adapting through low-rank matrices, achieving isolation without full replication:

- **Modular Expert Structures**: GainLoRA (Liang et al., arXiv 2025), MINGLE (Qiu et al., NeurIPS 2025) combine task-specific LoRA modules with gating networks in Mixture of Experts structures
- **Geometric and Causal Constraints**: AnaCP (NeurIPS 2025), PAID block interference through geometric structure preservation; CaLoRA (NeurIPS 2025) achieves backward transfer through causal inference
- **Dynamic Subspace Allocation**: CoSO (Cheng et al., NeurIPS 2025) dynamically allocates important subspaces during learning

Gap: Decision Boundary vs Density Manifold. These PEFT techniques are effective for classification but have fundamental limitations for AD. Classification models are robust to minor parameter drift as long as decision boundaries are maintained. However, NF must precisely model probability density with generative characteristics. NF's bijective mapping is highly sensitive to parameter changes—simple adapter insertion can collapse the entire likelihood manifold. Thus, new methodologies preserving NF's invertibility while isolating parameters are required.

2.3 Continual Learning in Anomaly Detection

Replay-based Approaches. CADIC (Yang et al., arXiv 2025) stores representative normal samples as coresets. ReplayCAD (Hu et al., IJCAI 2025) and CRD (Li et al., CVPR 2025) synthesize past data using generative models. However, these have fundamental limitations: generated sample 'fidelity hallucinations' or limited coreset capacity cannot precisely restore tail distributions of normal data.

Regularization & Distillation Approaches. DNE (Li et al., 2022) stores only previous task statistics for memory efficiency; CFRDC uses context-aware constraints to mitigate inter-task interference. However, these tend to assume Gaussian distributions or overly constrain parameter flexibility—unsuitable for NF structures requiring complex nonlinear transformations.

Dynamic Architecture & Prompting. SurpriseNet (arXiv 2023) perfectly prevents inter-task interference through complete isolation but model size scales linearly with tasks. Prompt-based methods like MTRMB (Zhou, You, et al., arXiv 2025), UCAD (Liu et al., AAAI 2024) add task-specific offsets to input features for parameter efficiency but add artificial perturbations to feature space, potentially interfering with NF's precise density estimation requiring topological structure preservation.

Existing Continual AD research fails to simultaneously satisfy: (1) data privacy (no replay), (2) density estimation precision (no regularization limits), (3) parameter efficiency (no isolation explosion). A new methodology is urgently needed that preserves NF's sensitive bijective structure while efficiently isolating and expanding continuously arriving multi-class distributions.

3 Method: DeCoFlow

3.1 Problem Formulation

Continual Anomaly Detection (CAD) assumes T task sequences $\mathcal{D} = \{D_0, D_1, \dots, D_{T-1}\}$ arriving sequentially. Each task $D_t = \{X_i^{(t)}, y_i^{(t)}\}$ contains only normal samples ($y_i = 0$) following unsupervised learning settings. The model learns D_t at time t without access to previous data $D_{0:t-1}$.

The final goal is that model $f_{\theta(t')}$ trained on current task t maintains similar performance on all past tasks as at initial training time—preventing catastrophic forgetting while learning new knowledge.

This research follows Class-Incremental Learning (CIL) scenario where new product classes arrive at each learning stage, and Task ID is not provided at inference. Given only sample x at test time, the model must autonomously select the appropriate expert.

3.2 Architecture Overview

DeCoFlow proposes a framework where functionally specialized modules organically combine to resolve the Isolation-Efficiency Dilemma from Section 1. Instead of training a massive single model, we structurally separate shared knowledge and task-specific adaptation, achieving complete forgetting prevention through parameter interference elimination while maintaining memory efficiency.

Key Components & Pipeline Flow. The framework comprises four key stages by data flow:

1. **Feature Extractor**: Backbone network extracting multi-scale features and injecting positional encoding
2. **Preprocessing Adapters**: Task-specific Alignment (TSA) for input distribution alignment and Spatial Context Mixer (SCM) for local context enhancement
3. **Decomposed Normalizing Flow**: Core engine with frozen base and learnable LoRA adapters performing density estimation
4. **Post-processing & Routing**: Affine Coupling Layer (ACL) for nonlinear manifold correction and Prototype Router for appropriate expert selection at inference

3.3 Design Principle: Invertibility-Independence Decomposition

DeCoFlow's core insight is that NF's Coupling Layer provides structural foundation for parameter-efficient continual learning. Particularly, the Scale and Shift networks within Affine Coupling Transformation can be implemented as arbitrary functions without compromising overall model invertibility or Jacobian computation tractability:

$$y_1 = x_1, \quad y_2 = x_2 \odot \exp(s(x_1)) + t(x_1) \tag{4}$$

Critically, this flexibility extends to decomposed subnets: when $s(x) = s_{base}(x) + \Delta s_t(x)$, the sum remains "one function," preserving all NF guarantees. We implement task-specific corrections $\Delta s_t, \Delta t_t$ as low-rank adapters (Hu et al., 2022), achieving complete parameter isolation with sublinear memory growth. We reinterpret NF's Arbitrary Function Property not as 'degree of expressiveness freedom' but as 'structural foundation for efficient isolation.'

The Core Mechanism: Decomposed NF with Frozen Base. Based on this principle, we physically decompose coupling layer subnets into 'Shared Knowledge' and 'Task-specific Adaptation':

$$\text{DecomposedSubnet}(x_{in}) = \text{MLP}_{Base}(x_{in}, \theta_{base}) + \frac{\alpha}{r} B_t(A_t x_{in}) \tag{5}$$

Here θ_{base} is permanently frozen after Task 0 training, serving as anchor shared by all tasks; A_t, B_t are LoRA Adapters learning each task's unique distribution. This structure suppresses memory growth for task count T while perfectly preventing existing knowledge degradation.

Compensating for Rigidity: Integral Components. However, base network freezing inevitably causes Structural Rigidity—feature space fixed to initial task may struggle to accommodate new tasks' heterogeneous distributions. DeCoFlow resolves this through four functions via preprocessing and postprocessing modules:

1. **Task-specific Alignment**: Forces input data statistical distribution alignment to frozen base's optimal operating range
2. **Spatial Context Mixer (SCM)**: Compensates local correlations missed by NF's independent patch

processing

3. **Task-specific Affine Coupling Layer (ACL)**: Performs nonlinear manifold adaptation at output for complex distribution differences beyond linear LoRA
4. **Tail-Aware Loss (TAL)**: Prevents gradient domination by easy samples (Bulk), weighting top high-loss patches to precisely optimize decision boundaries critical for AD

3.4 Feature Extraction

Feature Extractor & Patch Embedding. For meaningful representation extraction from input images, we use ImageNet-pretrained WideResNet-50 as feature extractor. Instead of using single layer outputs, we hierarchically utilize intermediate layer outputs for multi-scale information.

Following PatchCore methodology, extracted feature maps are divided patch-wise to preserve local characteristics. After pooling, we obtain feature tensor $F \in \mathbb{R}^{B\times H\times W\times D}$ where B is batch size, $H \times W$ is patch grid spatial resolution, D is feature dimension.

Positional Encoding. NF structurally has Permutation Invariance, processing each patch embedding independently, potentially losing relative position information—important for 2D images. We introduce 2D sinusoidal positional encoding $P \in \mathbb{R}^{H\times W\times D}$ added to patch embeddings:

$$F' = F + P \tag{6}$$

3.5 Integral Components: Compensating for Frozen Base Rigidity

Frozen Base design blocks catastrophic forgetting but inevitably causes Structural Rigidity—models fixed to Task 0 have limited capacity for heterogeneous distributions or fine local variations of new tasks. We propose Task-Specific Alignment (TSA) and Spatial Context Mixer (SCM) to compensate for specific frozen base paradigm limitations.

3.5.1 Task-specific Alignment (TSA)

After Task 0, Base Flow parameters are frozen, meaning it's fitted to Task 0's distribution statistics. When new task input distributions differ significantly, Covariate Shift occurs. Fixed Base Flow weights cannot trigger optimal activations for new distributions, and limited-capacity (Low-Rank) LoRA modules bear excessive burden correcting global distribution differences, delaying initial convergence and increasing optimization difficulty.

TSA forces input data alignment and recalibration to the form most efficient for fixed Base Flow processing, comprising two stages:

Task-Agnostic Standardization: Apply parameter-free LayerNorm to force input features $F^{(1)}$ to mean 0, variance 1 standard normal:

$$f_{std} = \frac{F^{(1)} - \mathbb{E}[F^{(1)}]}{\sqrt{\mathrm{Var}[F^{(1)}] + \epsilon}} \tag{7}$$

This pre-eliminates scale and distribution gaps between new task data and existing data, reducing NF optimization difficulty and ensuring all data starts from same statistical baseline for training stability.

Task-Adaptive Recalibration: Apply task-specific learnable parameters γ_t (Scale) and β_t (Shift) for Affine Transformation on standardized features—not simple restoration but Recalibration to the statistical position most efficient for fixed base model to process that task:

$$\gamma_t = 0.5 + 1.5 \cdot \sigma(\gamma_{raw}) \tag{8}$$

$$\beta_t = 2.0 \cdot \tanh(\beta_{raw}) \tag{9}$$

$$\hat{F}_t = \gamma_t \odot x + \beta_t \tag{10}$$

Here $\sigma(\cdot)$ is sigmoid. γ_t adjusts channel-wise feature importance. γ_t and β_t are constrained to $[0.5, 2.0]$ and $[-2.0, 2.0]$ respectively for stable learning.

Why Affine Transformation? TSA uses affine rather than nonlinear transformation to secure Statistical Consistency by adjusting only 1st and 2nd moments (mean, variance) without damaging the intrinsic geometric structure (shape/pattern) of data.

3.5.2 Spatial Context Mixer (SCM)

NF basically processes inputs as independent (i.i.d.) patches to compute likelihood $p(X) \approx \prod p(x_{u,v})$. However, this has structural limitations overlooking strong Spatial Correlation inherent in image data.

Independent processing has structural limitations in recognizing fine defects like scratches or stains defined by Local Discontinuity with neighboring patches. From probabilistic perspective, normal data distribution depends on surrounding information, requiring conditional probability distribution modeling like $p(x_{u,v}|\text{Neighbors})$.

SCM introduced before NF input aggregates neighboring information through Depthwise Convolution without channel interference, dynamically adjusting mixing ratio between original and context information via learnable gating parameter α:

$$C_{u,v} = \sum_{i,j} W_{i,j} \cdot f^{(2)}_{u+i,v+j}, \quad F^{(3)}_{u,v} = (1-\alpha)\cdot F^{(2)}_{u,v} + \alpha \cdot C_{u,v} \tag{11}$$

Final output $F^{(3)}$ is input to NF as independent entities but informationally already embeds neighbor states. Consequently, NF achieves Pseudo-Dependency Modeling effect—indirectly learning conditional probability $p(X_{u,v}|N_{u,v})$ without architecture changes.

3.6 DeCoFlow: Structural Decomposition of Normalizing Flows

The core engine DeCoFlow receives preprocessed feature tensor $F^{(3)}$, maps it to Gaussian distribution in latent space to estimate normal data probability density. Following RealNVP-based invertible structure, DeCoFlow comprises two core modules for preventing catastrophic forgetting and maximizing AD performance in continual learning: Decomposed Coupling Subnet (DCS) and Affine Coupling Layer.

3.6.1 Decomposed Coupling Subnet (DCS)

Design Philosophy. Decomposed Coupling Subnet (DCS) is the core module generating transformation parameters (scale s, shift t) within each coupling layer. Unlike general coupling layers using simple MLPs, DCS is designed with both Spatial Context Awareness and Task Adaptivity. Particularly, it adopts asymmetric network structure to reflect 'Local Contrast' information—the core of anomaly detection—in scaling transformation.

Structure & Formulation. DCS comprises three stages:

1. **Spatial Context Conditioning**: Unlike existing methods flattening inputs to 1D vectors, we reconstruct to 2D image form restoring spatial structure, then extract local context via $N \times N$ Depthwise Convolution. Reflection ratio is adjusted through learnable gating:

$$\text{ctx} = \alpha \cdot c(x), \quad \text{where} \quad \alpha = \alpha_{max} \cdot \sigma(\theta_{scale}) \tag{12}$$

2. **Context-Aware s-network**: Since anomalies mainly appear as discontinuity (Contrast) with surroundings, scale parameter s is predicted combining original features and context information:

$$s = \text{Linear}_2^{(s)}(\text{ReLU}(\text{Linear}_1^{(s)}([x; \text{ctx}]))) \tag{13}$$

3. **Context-Free t-network**: Distribution position shift t depends on patch-specific characteristics, predicted from original features only without context:

$$t = \text{Linear}_2^{(t)}(\text{ReLU}(\text{Linear}_1^{(t)}([x]))) \tag{14}$$

LoRA-based Decomposition. Each Linear layer above is implemented as LoRALinear for parameter decomposition. After Task 0, base weights $(\mathbf{W}_{base}, \mathbf{b}_{base})$ are frozen as Anchor, and subsequent tasks train only low-rank adapters $(\mathbf{A}_t, \mathbf{B}_t)$.

This structure fundamentally prevents catastrophic forgetting by never modifying existing parameters when learning new tasks. Also, LoRA applies only to linear layers within subnets, not affecting overall NF invertibility and Jacobian computation.

Previous section's SCM and DCS internal Context Conditioning both utilize spatial information but with clearly distinct roles: SCM encodes neighbor information into input itself, while DCS guides scaling considering neighbor relationships during transformation. Experiments show synergistic effects with two modules complementing each other beyond single-module benefits.

3.6.2 Task-Specific Affine Coupling Layer (ACL)

To correct fine nonlinear manifold mismatches between tasks difficult to resolve with frozen base and linear LoRA combination alone, we additionally place Affine Coupling Layer (ACL) at NF output. This layer follows standard coupling structure but trains with completely independent weights per task without parameter sharing.

ACL applies additional invertible transformation to latent variable z_{base}, correcting higher-order moments (kurtosis, skewness) that linear layers cannot capture. This precisely aligns final latent distribution to target standard normal $\mathcal{N}(0, 1)$, adding ACL's Jacobian determinant to final likelihood computation for maximizing density estimation accuracy:

$$z_{final} = f_{ACL}^{(t)}(z_{base}) \tag{15}$$

3.7 Training Objective

DeCoFlow's training is based on likelihood maximization—NF's basic principle—combined with tail-aware loss considering AD characteristics for optimization.

3.7.1 Likelihood Calculation

Model training and anomaly detection proceed through log-likelihood computation measuring how well input data maps to latent space's normal distribution.

Input x sequentially passes through TSA, SCM, DCS, and ACL to transform to final latent vector z_{final}. During this process, log values of Jacobian determinants (volume change rates) at each invertible transformation stage are accumulated:

$$\log|\det J_{total}| = \sum \log|\det J_{DCS}| + \sum \log|\det J_{ACL}| \tag{16}$$

Both DCS and ACL use affine coupling structure, computing log determinants identically. Each block splits input to x_1, x_2, generating constrained scale parameter $\tilde{s}$ for numerical stability:

$$\tilde{s} = \alpha_{clamp} \cdot \tanh(s/\alpha_{clamp}) \tag{17}$$

Since Jacobian is block lower-triangular, log determinant is simply sum of scale parameters:

$$\log|\det J_{layer}| = \sum_{i=1}^{D/2} \tilde{s}_i \tag{18}$$

Total transformation log determinant sums values from M DCS blocks and N ACL blocks.

Assuming final latent variable z_{final} follows multivariate standard normal $\mathcal{N}(0,1)$, latent space log probability density is:

$$\log p(z_{final}) = -\frac{1}{2}\|z_{final}\|_2^2 - \frac{D}{2}\log(2\pi) \tag{19}$$

By Change of Variable Formula, final log likelihood in input space $\mathbf{x}$:

$$\log p(x) = \log p(z_{final}) + \log|\det J_{total}| \tag{20}$$

3.7.2 Tail-Aware Loss (TAL)

Standard negative log-likelihood (NLL) loss makes models focus on majority 'normal' patches (Bulk) easy to learn. From $\nabla_\theta L \propto \nabla_\theta \log \sum p(x_i)$, easy sample gradients dominate overall learning direction, under-optimizing 'difficult' patches in tail regions critical for AD decision boundary formation.

This worsens with frozen base. LoRA's limited capacity should focus on precisely adjusting decision boundaries rather than perfectly fitting bulk data.

We introduce Tail-Aware Loss (TAL) weighting top $K\%$ high-loss patches by patch-wise NLL:

$$L_{total} = (1-\lambda_{tail})\cdot\mathbb{E}[L_{NLL}] + \lambda_{tail}\cdot\mathbb{E}_{top-k}[L_{NLL}] \tag{21}$$

Here $L_{NLL} = -\log p(x_{u,v})$, and λ_{tail} adjusts tail region importance. This loss guides limited-capacity adapters to intensively learn regions most critical for AD.

3.8 Task Routing & Adapter Activation

This framework assumes Class-Incremental Learning (CIL), where no prior Task ID information is provided at inference. The model must automatically identify appropriate task from input data alone and activate corresponding adapters (LoRA, TSA, ACL). Wrong task selection leads to inappropriate parameter activation and significant detection performance degradation, making accurate routing a core system challenge.

3.8.1 Prototype-Based Task Routing

We use Mahalanobis distance-based Prototype Router (Lee et al., 2018) for automatic task recognition.

During each task t's training, we collect feature vectors of normal samples to build task-specific prototypes. Prototypes comprise mean vector $\mu_t \in \mathbb{R}^D$ (averaged backbone final layer output) and covariance matrix $\Sigma_t \in \mathbb{R}^{D\times D}$.

Here f^t is backbone's final output vector (Global Average Pooled Feature), N_t is sample count, $\lambda_{reg} = 10^{-5}$ is regularization for numerical stability. Precision matrix Σ_t^{-1} is pre-computed and stored for inference speed.

At inference, we compute Mahalanobis Distance between feature vector from input image x_{img} and all trained task prototypes, selecting task t^* with minimum distance and activating its associated parameters.

Why Mahalanobis over Euclidean Distance:

- **Distribution shape consideration**: Considers each task feature distribution's spread and correlation, enabling more accurate distinction even with distribution overlap
- **Scale invariance**: Precision matrix Σ^{-1} automatically normalizes different scales across feature dimensions

Experimentally, Mahalanobis distance achieves 3-5% higher routing accuracy than Euclidean. Parameters to store for inference are only mean vector and precision matrix per task: for $T = 15$ tasks, $D = 768$ dimensions, total memory is only ∼35MB—negligible compared to total model size.

3.9 Anomaly Scoring & Inference

The entire inference pipeline transforms input images to latent space learned with normal distribution, measuring probability density to determine anomaly status. The core idea: normal samples have high probability density, anomalies have low density.

Step 1: Feature Extraction. Pass input image through pre-trained CNN backbone (WideResNet-50) to extract multi-layer feature maps. Split patch-wise with adaptive average pooling for 14×14 spatial resolution patch embeddings. Add 2D sinusoidal positional encoding to preserve position information.

Step 2: Normalizing Flow Transformation. With selected task t's adapters activated, input x transforms in order:

1. **TSA**: Align distribution to frozen base's statistics
2. **SCM**: Exchange information between adjacent patches via Depthwise Convolution
3. **DCS**: Transform to latent variable z through 8 LoRA-equipped affine coupling layers
4. **ACL**: Task-specific nonlinear manifold adaptation through 2 blocks

During this process, log determinants accumulate for total transformation volume change rate $\log|\det J_{total}|$.

Step 3: Anomaly Score Computation. Assuming final latent variable $z_{final} \in \mathbb{R}^{H \times W \times D}$ follows $\mathcal{N}(0,1)$, compute log likelihood at each patch position (h, w).

Anomaly Score is defined as negative log-likelihood:

$$s_{h,w} = -\log p(x_{h,w}) \tag{22}$$

Final image-level anomaly score aggregates via mean or top-K patch score average.

4 Experiments

4.1 Experimental Setup

This section details experimental setup designed to support DeCoFlow's core hypotheses and objectively evaluate quantitative performance.

Datasets and Protocols. We use MVTec-AD (Bergmann et al., 2019), standard benchmark for industrial anomaly detection, and ViSA (Zou et al., 2022) with complex defect structures.

Experiments follow 1-Class-Per-Task (1×1) scenario reflecting realistic manufacturing environments with sequential class learning. We strictly adhere to Zero-Replay (no previous task data storage) and Task-Agnostic (no Task ID at inference) settings to evaluate model adaptability and routing performance.

Evaluation Metrics. We compare against normalization (EWC), rehearsal (Replay), architecture isolation (PackNet) methods, as well as latest PEFT-based continual learning techniques (L2P, DualPrompt) and AD-specialized models (CADIC, ReplayCAD). Performance metrics include I-AUC/P-AUC for detection accuracy and **P-AP (Pixel Average Precision)** for localization precision, with **Forgetting Measure (FM)** and routing accuracy as key stability indicators.

Implementation Details. All experiments use ImageNet-pretrained WideResNet-50-2 as frozen backbone. Core architecture comprises 6 DCS (Decomposed Coupling Subnet) blocks and 2 ACL blocks with LoRA rank 64 balancing efficiency and expressiveness. Training proceeds in two phases: first task learns base structure then freezes, subsequent tasks optimize only adapter parameters with AdamP optimizer (LR=3×10^{-4}) for 60 epochs.

4.2 Main Results

MVTec-AD. This experiment verifies that DeCoFlow achieves superior detection performance and stability compared to existing methods in MVTec-AD 15-task sequential learning.

ViSA. (Results and analysis pending)

Verification of Zero Forgetting. Confirms whether proposed structural decomposition strategy—frozen base with independent adapters—works as theoretically intended during actual long-sequence learning. Tracks whether initial task knowledge is preserved as learning progresses, visually verifying parameter isolation effectiveness.

(Heatmap or line Figure format results pending)

Analyzing Figure 3 heatmap visualizing performance changes over 5 tasks and Table 3 BWT metrics: control Fine-tuning model shows sharp Task 0 performance degradation with each new task, ultimately losing ∼70%p performance. Conversely, DeCoFlow maintains Task 0 performance at initial state (100%) even after 15th task training, recording Backward Transfer (BWT) 0.0%.

4.3 Ablation Study

4.3.1 Effectiveness of Components

Quantifies individual contributions of three key compensating elements—TSA, TAL, ACL—to overall performance through Leave-one-out experiments removing each component from Full Model.

4.3.2 Structural Necessity

Determines whether proposed components are simply general performance boosters or specialized elements compensating 'Structural Rigidity' caused by frozen base. 2×2 factorial design experiments with (Base: Frozen vs Trainable) × (Component: Absent vs Present) analyze interaction effects.

4.3.3 LoRA Rank Sensitivity

Validates hypothesis that "Task Adaptation is intrinsically low-rank." Measures model performance sensitivity varying LoRA rank from 16 to 128, confirming efficient adaptation possible without excessive parameters.

4.3.4 Architecture Depth and Stability

Analyzes flow block depth's impact on performance and training stability, verifying ACL's contribution to deep network training stability.

4.3.5 Task 0 Selection Sensitivity

DeCoFlow permanently freezes base parameters after first task (Task 0). Therefore, "Does initial task selection cause bias determining overall model performance?" is essential for verifying framework generality. We set 5 classes with different structural characteristics (simple objects, complex textures, fine structures) as Task 0, train full 15 tasks, and analyze final performance sensitivity.

4.4 Analysis

4.4.1 Why Normalizing Flow?

Validates theoretical rationale for performing structural decomposition based on Normalizing Flow rather than other generative (VAE, AE) or knowledge distillation (Teacher-Student) models. Compares performance and training stability when applying LoRA to each architecture's core modules under identical backbone and CL protocol.

4.4.2 Coupling-level vs. Feature-level Adaptation

Analyzes performance differences based on adapter placement location, establishing why placing adapters inside NF (Coupling-level) is superior to input stage (Feature-level). Validates hypothesis that existing prompt tuning (Feature-level) disrupts NF's density manifold.

4.4.3 SVD Analysis of Weight Updates

Analyzes whether weight changes (ΔW) required for task adaptation actually have low-rank characteristics through SVD, validating LoRA Rank 64 appropriateness.

4.4.4 Gradient Redistribution by Tail-Aware Loss

Analyzes gradient distributions during training to elucidate TAL's performance contribution mechanism.

4.4.5 ACL Transformation Analysis

Analyzes whether ACL performs simple linear scaling or qualitatively different nonlinear correction.

4.4.6 Routing Accuracy and Confusion Analysis

Analyzes causes of prototype-based router achieving $\sim$100% accuracy and diagnoses potential Confusion Risk.

4.4.7 Computational Cost

Quantitatively evaluates whether Computational Overhead of DeCoFlow's 'Zero Forgetting' is acceptable for practical deployment environments.

5 Conclusion

We presented DeCoFlow, a continual anomaly detection framework resolving the isolation-efficiency dilemma through normalizing flow's unique structural properties. By identifying the Arbitrary Function Property as theoretical foundation for safe parameter decomposition, we enabled complete task isolation with only 8% parameter overhead per task.

The three integral components—Task-Specific Alignment, Tail-Aware Loss, and Affine Coupling Layer—were systematically validated as providing amplified benefits under frozen base constraints through interaction effect analysis.

DeCoFlow achieves state-of-the-art performance (98.05% I-AUC, 55.80% P-AP) with *zero forgetting* on MVTec-AD, outperforming replay-based methods without storing any data.

Future Work. (1) Adaptive rank selection for overhead reduction. (2) Higher-resolution variants for improved localization. (3) Contrastive routing for high inter-task similarity scenarios. (4) Extension to video anomaly detection with temporal NF.

References

[1] Roth, K., et al.: Towards Total Recall in Industrial Anomaly Detection. CVPR (2022)

[2] Yu, J., et al.: FastFlow: Unsupervised Anomaly Detection and Localization via 2D Normalizing Flows. arXiv (2021)

[3] McCloskey, M., Cohen, N.J.: Catastrophic Interference in Connectionist Networks. Psychology of Learning and Motivation (1989)

[4] French, R.M.: Catastrophic Forgetting in Connectionist Networks. Trends in Cognitive Sciences (1999)

[5] Yang, Y., et al.: CADIC: Continual Anomaly Detection with Incremental Classes. arXiv (2025)

[6] Hu, X., et al.: ReplayCAD: Replay-based Continual Anomaly Detection. IJCAI (2025)

[7] SurpriseNet: Continual Novelty Detection (2023)

[8] Liu, Z., et al.: UCAD: Unsupervised Continual Anomaly Detection. AAAI (2024)

[9] Dinh, L., et al.: Density Estimation Using Real-NVP. ICLR (2017)

[10] Hu, E.J., et al.: LoRA: Low-Rank Adaptation of Large Language Models. ICLR (2022)

[11] Bergmann, P., et al.: MVTec AD: A Comprehensive Real-World Dataset. CVPR (2019)

[12] You, Z., et al.: A Unified Model for Multi-class Anomaly Detection. NeurIPS (2022)

[13] Zhao, Y., et al.: OmniAL: A Unified CNN Framework for Anomaly Localization. CVPR (2023)

[14] He, Y., et al.: MambaAD: Exploring State Space Models. NeurIPS (2024)

[15] He, Y., et al.: DiAD: A Diffusion-based Framework. AAAI (2024)

[16] Zhou, Y., et al.: MSFlow: Multi-Scale Flow-based Framework. TNNLS (2024)

[17] Yao, X., et al.: HGAD: Hierarchical Gaussian Mixture for Anomaly Detection. ECCV (2024)

[18] Kirkpatrick, J., et al.: Overcoming Catastrophic Forgetting in Neural Networks. PNAS (2017)

[19] Liang, Y., et al.: GainLoRA: Gated LoRA for Continual Learning. arXiv (2025)

[20] Qiu, S., et al.: MINGLE: Mixture of LoRA Experts for Continual Learning. NeurIPS (2025)

[21] Cheng, Y., et al.: CoSO: Continual Subspace Optimization. NeurIPS (2025)

[22] Zou, Y., et al.: SPot-the-Difference Self-Supervised Pretraining. ECCV (2022)