PatchFlow: Leveraging a Flow-Based Model with Patch Features*

Boxiang Zhang [1], Baijian Yang [1], Xiaoming Wang[1], Corey Vian[2]

[1]Purdue University, 401 N. Grant St, West Lafayette, IN 47907, USA
[2]Stellantis - Kokomo Casting Plant 1001 E Blvd, Kokomo, IN 46902, USA
zhan4653@purdue.edu, byang@purdue.edu, wang1747@purdue.edu, corey.vian@stellantis.com

Abstract

Die casting plays a crucial role across various industries due to its ability to craft intricate shapes with high precision and smooth surfaces. However, surface defects remain a major issue that impedes die casting quality control. Recently, computer vision techniques have been explored to automate and improve defect detection. In this work, we combine local neighbor-aware patch features with a normalizing flow model and bridge the gap between the generic pretrained feature extractor and industrial product images by introducing an adapter module to increase the efficiency and accuracy of automated anomaly detection. Compared to state-of-the-art methods (Roth et al. 2021), our approach reduces the error rate by 20% on the MVTec AD dataset (Bergmann et al. 2019), achieving an image-level AUROC of 99.28%. Our approach has also enhanced performance on the VisA dataset (Zou et al. 2022), achieving an image-level AUROC of 96.48%. Compared to the state-of-the-art models, this represents a 28.2% reduction in error. Additionally, experiments on a proprietary die casting dataset yield an accuracy of 95.77% for anomaly detection, without requiring any anomalous samples for training. Defect segmentation on MVTecAD dataset is shown in Figure 4. Our method illustrates the potential of leveraging computer vision and deep learning techniques to advance inspection capabilities for the die casting industry.

Introduction

Die casting is a method where molten metal is injected under high pressure into a mold formed by two hardened steel dies. This process resembles injection molding. It's often used for high-volume production due to the significant investment required for the casting equipment and dies. Despite the high initial costs, the actual manufacturing of parts using die casting is relatively straightforward (Sai, Vinod, and Sowmya 2017). Die casting is crucial in modern manufacturing due to its precision, efficiency, versatility, and cost-effectiveness, especially for large-scale production. It plays a crucial role in various industries, including automotive, aerospace, and electronics. Despite these advantages, one of the primary issues that haunts the die casting industry is surface defects in cast products(Zhang et al. 2023a).

Surface Defects include blisters, peeling, or pits on the surface of the casting. They are often caused by contaminants in the metal, excessive moisture in the mold, or high metal temperature. Surface defects in die casting can lead to a variety of problems, including reduced product quality, increased production costs due to waste, and potentially detrimental impacts on product performance. In the worst-case scenario, these defects can lead to complete product failure. Detecting these flaws, especially in aluminum die casting, presents considerable challenges. Traditional inspection methods, like visual checks and other manual or semi-manual techniques, tend to be labor-intensive, time-consuming, and susceptible to human error.

In light of these challenges, there is a growing need for an automated, efficient, and reliable method to detect surface defects in die casting products. Artificial Intelligence (AI), particularly anomaly detection in computer vision, emerges as a solution to address this issue(Zhang et al. 2023a).

Anomaly detection is the process of identifying data points or patterns that deviate significantly from the expected behavior or norm, proves valuable in tackling quality control challenges in die casting. Figure 4 displays eight examples of products with surface defects, illustrating the pixel-level ground truth of these defects, their corresponding heat maps, and the defects predicted using anomaly detection algorithms.

Anomaly detection models can be viewed as semi-supervised learning models (Bergmann et al. 2019). During the training phase, these models are trained exclusively on normal samples, without any exposure to anomalous instances. During the inference stage, the model is presented with new instances and tasked with determining whether each instance is normal or anomalous.The assumption is that the model will learn the underlying distribution of the normal data, and anything that deviates significantly from this learned distribution can be considered an anomaly. This decision is usually based on some measure of how much each instance deviates from the normal instances that the model was trained on. In computer vision, an instance would typically be an image or a video, and the measure of deviation could be the reconstruction error from a generative model, the feature representations learned by a deep neural network, or the likelihood under a probabilistic model.

Much research has focused on improving the accuracy

*This work is supported by Stellantis

and speed of vision-based anomaly detection tasks (Roth et al. 2021; Zhang et al. 2023b; Lee, Lee, and Song 2022; Tien et al. 2023; Yu et al. 2021; Gudovskiy, Ishizaka, and Kozuka 2022; Ardizzone et al. 2019; Rudolph, Wandt, and Rosenhahn 2021; Rudolph et al. 2022). These studies have traditionally employed Convolutional Neural Network (CNN) architectures, such as ResNet (He et al. 2015) and EfficientNet (Tan and Le 2020), to extract features for anomaly detection.

Embedding-based models require a memory bank, whereas flow-based models necessitate a sophisticated structure with numerous parameters, leading to a high computational load (FLOPs) for mapping the feature distribution to a latent normalized distribution. This mapping is essential for accurately calculating the exact distribution of features in normal samples within the training set. The objective of this work is to streamline the model and enhance the efficiency of flow-based architectures.

The main contribution of this work is list below

- We first combine local neighbor aware patch feature with Normalizing flow model.
- We bridge the gap between the data that the feature extractor was pretrained on and industrial product images by introducing a feature adapter between the feature extractor and the normalizing flow structure.
- We add a bottleneck structure in the flow architecture to reduce computational complexity while preserving the ability to map between distributions.
- We conducted a comprehensive study evaluating our model, which achieved state-of-the-art performance with an image-level AUROC score on the challenging MVTecAD dataset.
- Our results indicate that our model is capable of anomaly detection on both public benchmark datasets as well as practical, real-world applications with high resolution image.

The paper is organized as follows: The literature review is presented in the Related Works section, followed by an illustration of the proposed PatchFlow in the Method section. The Experiment section details the dataset, evaluation metrics, and AUROC results. Finally, the paper concludes in the Conclusion section with a brief discussion on future work.

Related Works

The primary objective of anomaly detection in computer vision is to identify deviations from the norm within visual data. This process involves differentiating between standard, expected patterns and those that are irregular or unusual, which are referred to as anomalies. These anomalies can range from minor variations to significant deviations and are crucial for applications such as quality control in manufacturing, surveillance, and medical imaging. Anomaly detection models in computer visions can be broadly classified into three categories: generative-based models, feature-based models, and normalizing flow-based models.

Generative-based Model

Generative-based models are a class of machine learning models that focus on generating or simulating data. The underlying assumption is that these models can learn the data distribution of normal instances. In anomaly detection, generative models are typically trained exclusively on normal data. Once trained, the models attempt to reconstruct new input data based on what they have learned. If the new input data is normal, the models should be able to reconstruct it accurately. However, if the input data contains anomalies, the models will likely struggle to reconstruct it accurately, resulting in a high reconstruction error, which is the basis for anomaly detection. Generative Adversarial Networks (GANs) (Goodfellow et al. 2014),UNet (Ronneberger, Fischer, and Brox 2015), Variational Autoencoders (VAEs) (Kingma and Welling 2022) and diffusion models (Sohl-Dickstein et al. 2015), are capable of generating new data instances that resemble the normal instances in the training set. These models are often complex and require substantial computational resources. Scaling these models for large datasets or high-dimensional data can be challenging. In addition, Generative models can be difficult to train. For example, GANs involve training a generator and a discriminator simultaneously. Models can easily become unstable or fail to converge.

Feature-based Model

In traditional Feature-Based Models, features are explicitly defined and engineered based on domain knowledge. For example, in image processing, features might include edge detection, color histograms, or texture measures. These features are manually crafted and selected to capture relevant information for a specific task. The performance heavily relies on the quality of feature engineering, which can be labor-intensive and may not capture all the nuances in complex data.

Embedding-based models are a modern evolution of feature-based models, primarily used in the context of deep learning. Instead of manually defining the features, the model learns an optimal representation of the data autonomously. These learned representations, known as embeddings, are essentially high-dimensional feature vectors. Embeddings capture complex and abstract patterns in the data, which might not be easily identifiable with traditional feature engineering. Embedding-based models have become increasingly popular in anomaly detection. These models learn a compact representation of normal data and then identify deviations from this norm as potential anomalies. Various techniques can be utilized for feature extraction, spanning simple image statistics to complex deep learning architectures. PADiM (Defard et al. 2020) introduced a feature aggregation technique to improve model performance by combining multi-level descriptors extracted from different layers of a CNN. Specifically, features are computed from both shallow and deep layers, encapsulating low-level visual patterns along with higher-level semantic information. PatchCore (Roth et al. 2021) employs WideResnet50(Zagoruyko and Komodakis 2017) as the backbone feature extractor, combined with a feature aggregation module that incorporates neighboring descriptors to enrich the representations. Anomaly detection is performed by comparing each instance's embedding against a memory bank of features computed from normal samples. Deviations from the distribution of normal embeddings indicate the presence of defects. Thus, embedding-based approaches rely on learning a robust feature space where anomalies separate clearly from defect-free patterns.

Normalizing Flow-based Model

Normalizing Flows (NF) (Rezende and Mohamed 2016) are a class of neural networks that can learn bijectivity transformation N between data distribution $a \in A$ and well-defined densities $b \in B$, such that $B = N(A)$ and $A = N^{-1}(B)$. The well defined distribution could be the standard normal distribution such that $B \sim \mathcal{N}(0, I)$. According to the Change of Variable Theorem, the relation of the distribution of A and B can be defined as:

$$\hat{p_A}(a) = p_B(N(a))|\mathrm{det}(\frac{\partial N(a)}{\partial a^T})| \tag{1}$$

A key advantage of NF models is that the transformation $N : A \rightarrow B$ enables the calculation of the exact distribution of A by equation 1. Real-NVP (Dinh, Sohl-Dickstein, and Bengio 2017) introduces a specific type of normalizing flow that not only uses easily invertible transformations but also has a Jacobian determinant that is straightforward to compute.

In the field of anomaly detection, normalizing flow-based models can learn the exact probability density of the training data and provide a way to transform a simple distribution (like a multivariate Gaussian) into a complex one that matches the distribution of the data. Anomalies can be detected based on their low probability under this learned distribution. In this work we employ normalizing flow-based models to detect anomaly.

DifferNet (Rudolph, Wandt, and Rosenhahn 2021) innovatively integrate AlexNet (Krizhevsky, Sutskever, and Hinton 2012) and a normalizing flow structure with RealNVP (Dinh, Sohl-Dickstein, and Bengio 2017) coupling layer for the task of anomaly detection. It demonstrated impressive capabilities in terms of image-level prediction accuracy, even without the requirement for a large volume of training data. However, despite its achievements, it did not deliver high performance at the pixel level, signifying an area for potential improvement. Subsequent work, Cs-Flow (Rudolph et al. 2022), increased the input resolution of the feature extractor and using a more powerful feature encoder EfficientNet (Tan and Le 2020) as well as introduced the concept of a cross-scale flow structure. This structure facilitated interaction of features from different layers within the normalizing flow . This resulted in a performance enhancement over DifferNet. CFLOW-AD (Gudovskiy, Ishizaka, and Kozuka 2022) enhanced the application of normalizing flow in anomaly detection by introducing conditional normalizing flow (Ardizzone et al. 2019) with positional encoding, which provides position prior information. FastFlow (Yu et al. 2021) integrated transformers with normalizing flow-based anomaly detection models by substituting the feature extractor with Class-Attention in Image Transformers (CaiT) (Touvron et al. 2021a) and Data-efficient Image Transformers (DeiT) (Touvron et al. 2021b).

All these models utilize features directly extracted from the pretrained backbone. However, the discrepancy of distribution between the images on which the backbone model is pretrained and the industrial images poses a constraint on detection accuracy. Additionally, normalizing flow-based models typically entail a complex structure for distribution mapping, resulting in high computational costs and slow inference speeds. In this study, we mainly focus on addressing the gap of normalizing flow based model.

Method

Model Overview

The PatchFlow model comprises four key components, shown in Figure 1. First, a pretrained feature extractor $\mathcal{E}$ extracts multi-level representations from multi-scale images. Second, a feature aggregation layer $\mathcal{P}$ combines descriptors from different hierarchical levels and scales. Third, a feature adapter $\mathcal{A}$ module reduces the dimensionality of the representations and bridges the gap between the generic pretraining data and specialized industrial product images. The feature adapter addresses the domain shift, while the normalizing flow leverages the power of deep generative models to enable precise anomaly detection. Together, these components yield an accurate and robust anomaly detection architecture suited for quality control across various manufacturing processes. Finally, a normalizing flow $\mathcal{F}$ maps the adapted features to a standardized distribution.

Feature Extractor

Feature extractor $\mathcal{E}$ maps data $x \in \mathbb{R}^{C \times H \times W}$ with C channels, height H and width W from dataset $X = \{x_1, ..., x_D\}$ of size D into a latent space $y \in Y$. For each i^{th} data point, this relationship is denoted as $y_i = \mathcal{E}(x_i)$. To ensure the preservation of both the global abstract information and local special information inherent in the images, instead of solely relying on the features from the final layer, we utilize features from the multiple layers of the feature extractor. $y_i^j = [E(x_i)]^j$ refers to the feature from hierarchy $j \in \{1, 2, ...\}$of the feature extractor for the image $i \in \{1, 2, ..., D\}$. The feature extractor is well trained on large dataset like ImageNet(Deng et al.) and keep fix for both training and testing.

Feature Aggregation

After obtaining features from a feature extractor across different hierarchies, we employed a feature aggregation layer $\mathcal{P}$ to combine the local neighborhood, which consisted of a feature patcher and a featur fuser, both of which do not

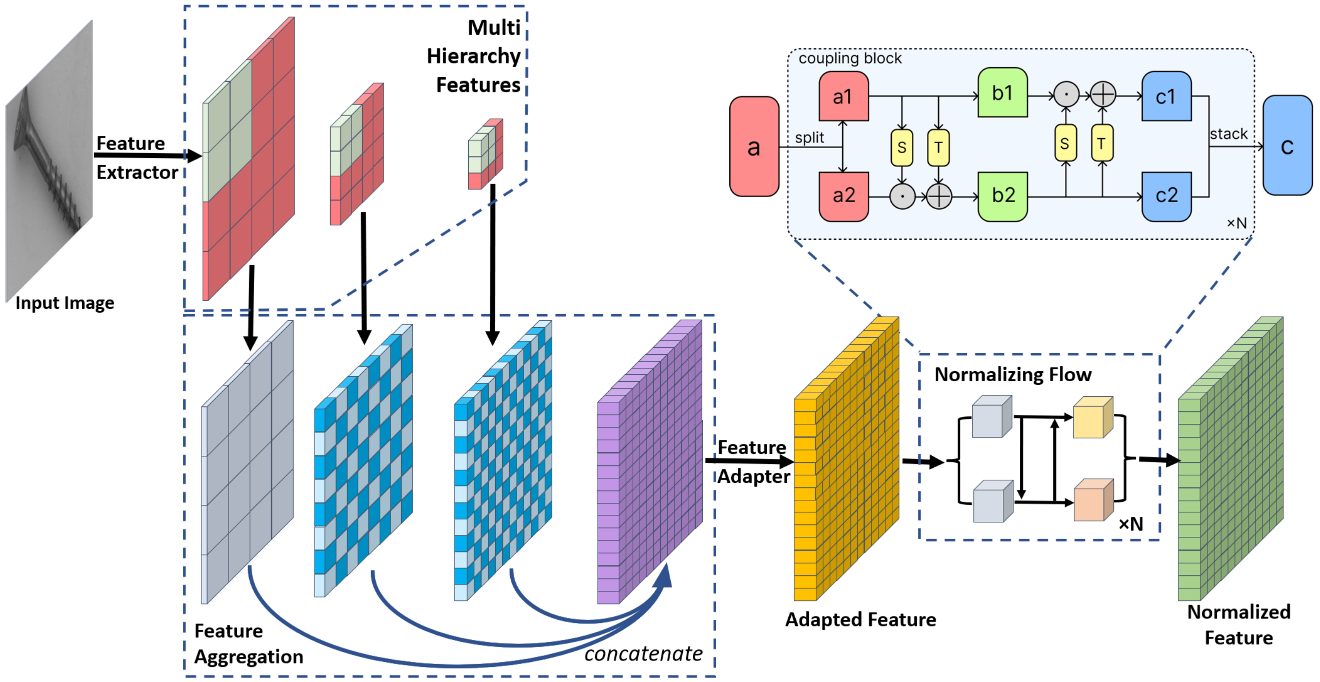

Figure 1: Model Overview. The PatchFlow model comprises four key components. 1, a pretrained feature extractor extracts multi-level representations from multi-scale images. 2, a feature aggregation layer combines descriptors from different hierarchical levels and scales. 3, a feature adapter module reduces the dimensionality of the representations and bridges the gap between the generic pretraining data and specialized industrial product images. 4, a normalizing flow maps the adapted features to a standardized distribution.

introduce new trainable parameters. For the patcher, following the approach outlined in (Roth et al. 2021),it combine local neighborhood for each hierarchy respectively. Let $y_i^j \in \mathbb{R}^{c^j,h^j,w^j}$, where c^j, h^j, and w^j represent the dimensions, height, and width of the feature from hierarchy j. With CNN structure, typically, $h^j > h^{j+1}$, $w^j > w^{j+1}$, $c^j < c^{j+1}$. This is extended to $y_i^j(h,w) \in \mathbb{R}^{c^j}$, indicating the feature slice at position (h,w), where $h \in \{1,\ldots,h^j\}$ and $w \in \{1,\ldots,w^j\}$. The neighborhood of position (h,w) with uneven patch size $\int$ is defined as follows:

$$N_{\int}^{(h,w)} = \{(a,b)|a \in [h-[\int/2],..,h+[\int/2]], \\ b \in [w-[\int/2],...,w+[\int/2]]\} \tag{2}$$

The local neighbourhood aware patch feature of image i from hierarchy j at position (h,w) with patch size $\int$ is defined as:

$$p_i^j(N_{\int}^{(h,w)}) = Agg(y_i^j(a,b)|(a,b) \in N_{\int}^{(h,w)}) \tag{3}$$

where Agg is some aggregation function. After obtaining patch features from each hierarchy, $p_{i,\int}^j = Agg(y_i^j)_{\int} \in \mathbb{R}^{c^j,h^j,w^j}$, with $p_{i,\int}^j(h,w) = p_i^j(N_{\int}^{(h,w)}) \in \mathbb{R}^{c^j}$. These features are fused together to create a multi-hierarchy local aware feature. This is achieved by interpolating the deeper features to a size that matches the features from the shallower layers and concatenating them together, as illustrated in the figure. The feature aggregation layer $\mathcal{P}$ takes features from different hierarchies as input and outputs a fused local aware patch feature, $f_{i,\int} = \mathcal{P}_{\int}(y_i^1, y_i^2, \ldots)$, with $f_{i,\int}(h,w) \in \mathbb{R}^{C_{\int}}$, where $C_{\int} = \sum_j c^j$. In addition to incorporating features from multiple hierarchies, our approach involves leveraging features at different scales. Features from various scales undergo the same processing pipeline, yielding several patch features of the same size. These patch features are then straightforwardly concatenated to form a multi-scale hierarchy patch feature. In our work, we specifically use three scales. Consequently, the multi-scale hierarchy patch feature for data i with patch size $\int$ can be expressed as $f_{i,\int}^3(h,w) \in \mathbb{R}^{3\times C_{\int}}$.

Feature Adaptor

The industrial dataset exhibits a distinct distribution compared to the dataset on which the feature extractor was originally trained. To mitigate the impact of these distributional differences and improve the model's performance on the industrial data, we introduce an adaptation layer denoted as $\mathcal{A}$. This layer facilitates the transfer of patch features $f_{i,\int}(h,w)$ to adapted features $g_{i,\int}(h,w) \in \mathbb{R}^{C_g}$, to address these distributional differences and enhance the model's effectiveness in handling the industrial data.

$$g_{i,\int}(h,w) = \mathcal{A}(f_{i,\int}(h,w)) \tag{4}$$

To achieve this adaptation, we employ a single fully-connected layer, a strategy demonstrated to be effective

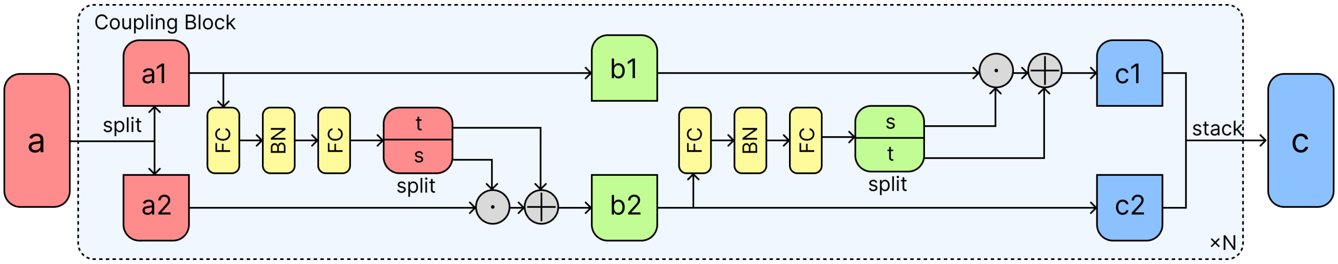

Figure 2: The coupling block features a bottleneck structure, where 'FC' and 'BN' refer to the fully connected layer and bottleneck layer.

in prior work such as (Liu et al. 2023). By incorporating this adaptation mechanism, we aim to enhance the model's adaptability to the specific distributional nuances encountered in the industrial setting.

Flow

To transform the distribution $P_G(g(h,w))$ of the adapted feature $g(h,w)$, features are subsequently passed through a bijective mapping $\mathcal{F}$, where $\mathcal{F} : G \rightarrow Z$. The aim here is to bijectively map the features $g(h,w) \in G$ to a normalized latent space $z(h,w) \in Z$ with normal distribution, such that $Z \sim \mathcal{N}(0,I)$. This bijective transformation is represented as $z(h,w) = \mathcal{F}(g_s(h,w),\, g_s(h,w) = \mathcal{F}^{-1}(z(h,w))$. the estimate distribution $\hat{g}(x,y)$ of $g(x,y)$ is given by equation1 In contrast to prior approaches (Ardizzone et al. 2019; Gudovskiy, Ishizaka, and Kozuka 2022; Rudolph, Wandt, and Rosenhahn 2021; Yu et al. 2021) that map the distribution of image-level features to a normalized latent distribution, our method directly maps each patch-level feature. This strategy offers the advantage of facilitating straightforward defect localization based on the position of normalized latent patch features that deviate from the normal distribution. This approach allows for a more fine-grained and precise identification of defects, as the deviation from the normal distribution is localized at the patch level rather than the entire image.

For each Bijective Mapping block, we follow previous works (Ardizzone et al. 2019; Gudovskiy, Ishizaka, and Kozuka 2022; Rudolph, Wandt, and Rosenhahn 2021; Yu et al. 2021), which adopt the coupling layer structure from RealNVP (Dinh, Sohl-Dickstein, and Bengio 2017) as the basic building block in our PatchFlow architecture (see Figure 2).

We adopted a fully connected layer with a bottleneck structure to preserve the capacity for mapping between distributions while reducing the computational complexity (Flops) in the coupling layer. This design choice allows us get a deeper network with less Flops (He et al. 2015), while the bottleneck structure enhances efficiency by constraining the number of parameters involved in the transformation. In our implementation, we employed a three-layer convolutional neural network with a kernel size of 1. Figure 2 shows the detail structure of coupling block.

Objective Function

The objective of the training process is to find optimized parameters θ within the normalizing flow N to minimize the Kullback-Leibler (KL) divergence between the estimated feature distribution $\hat{f}(h,w) \in F$and the actual feature distribution $y \in Y$. We define a transformation $N : Y \rightarrow Z$ where Z follows a standard normal distribution, $Z \sim \mathcal{N}(0,I)$. the KL divergence can be mathematically expressed as follows:

$$\begin{aligned}\mathcal{L}(\theta) &= D^{\theta}_{KL}(p_y|\hat{p_y}) \\ &= \sum_{i=1}^{D} p_y(y_i) log(\frac{p_y(y_i)}{\hat{p_y}(y_i)}) \\ &= \frac{1}{D}\sum_{i=1}^{D}\frac{||N(y_i)||_2^2}{2} - log(|\det(\frac{\partial N(y_i)}{\partial y_i^T})|) + c\end{aligned} \tag{5}$$

In equation 5, the constant term $(const)$ is not directly linked to the model parameters θ. Thus, when applying the backpropagation algorithm to optimize the loss function, this c term does not influence the update of the model parameters. It is effectively treated as a constant component within the loss function, which simplifies the computation during the optimization process.

Mask Generation and Score Function

$z(h,w) \in \mathbb{R}^{C_g}$ such that $Z \sim \mathcal{N}(0,I)$. each element of the latent feature encapsulates spatial information. Therefore, any deviation from the normal distribution indicates not only an anomaly but also suggests its relative position. In order to maintain simplicity in prediction, the generation of anomaly mask and the anomaly score function involve straightforward linear transformations of the normalized latent patch features. Given the normalized features $z \in \mathbb{R}^{C_g \times F_g \times W_g}$, where C_g, H_g, W_g represent the number of channels, height, and width of the feature after feature adaption layer respectively. We upscale features to match the input image size using bilinear interpolation to generate the final anomaly map with size $R^{B \times H \times W}$. This map presents a comprehensive view of the anomalies detected in the input image, effectively leveraging information from various hierarchy and scales to enhance anomaly detection. The anomaly score for each image is then determined by the maximum value in the corresponding anomaly map.

Model	DRÆM	CutPaste	CFlow-AD	CS-Flow	PADIM	PatchCore	PatchFlow
Carpet	97.0	93.9	98.98	**100**	99.1	98.7	**100**
Grid	99.9	**100**	97.64	99.0	97.3	98.2	99.83
Leather	**100**	**100**	98.98	**100**	99.2	**100**	**100**
Tile	99.6	94.6	99.25	**100**	94.1	98.7	99.10
Wood	99.1	99.1	98.99	**100**	94.9	99.2	99.65
Bottle	99.2	98.2	98.89	99.8	98.3	**100**	**100**
Cable	91.8	81.2	**99.66**	99.1	96.7	99.5	99.01
Capsule	98.5	98.2	**98.56**	97.1	98.5	98.1	97.21
Hazelnut	**100**	98.3	98.95	99.6	98.2	**100**	**100**
Metal Nut	98.7	99.9	98.86	99.1	97.2	**100**	**100**
Pill	**98.9**	94.9	98.01	98.6	95.7	96.6	97.35
Screw	93.9	88.7	**98.93**	97.6	98.5	98.1	98.34
Toothbrush	**100**	99.4	97.99	91.9	98.8	**100**	**100**
Transistor	93.1	96.1	96.65	99.3	97.5	**100**	99.54
Zipper	**100**	99.9	99.08	99.7	98.5	99.4	99.21
Average	98.03	96.1	98.62	98.7	97.5	99.1	**99.28**

Table 1: Image level AUROC score of DRÆM (Zavrtanik, Kristan, and Skočaj 2021), CutPaste(Li et al. 2021), PADIM(Defard et al. 2020), PatchCore(Roth et al. 2021), CFlow-AD(Gudovskiy, Ishizaka, and Kozuka 2022), CS-Flow(Rudolph et al. 2022) from the MVTecAD(Bergmann et al. 2019) dataset are presented, with the best scores highlighted in bold.

Experiments

Datasets

Numerous datasets have been specifically designed for anomaly detection tasks. In this study, we evaluate our model using two well-established, large-scale anomaly detection datasets, namely the MVTec AD (Bergmann et al. 2019) dataset and VisA dataset(Zou et al. 2022).

The MVTec AD (Bergmann et al. 2019) dataset is a comprehensive benchmark for anomaly detection methods. It's widely recognized in the field, comprising over 5000 high-resolution images categorized into 15 different object and texture types.

The VisA (Zou et al. 2022) dataset consists of 12 subsets, each corresponding to a different object. It includes a total of 10,821 images, with 9,621 normal and 1,200 anomalous samples. The subsets range from complex structures such as different types of printed circuit boards (PCBs), to multiple instances within a single view, and objects that are mostly aligned. Anomalous images in this dataset present various flaws, including surface defects like scratches, dents, and color spots, and structural defects like misplacement or missing parts.

We also collected images of valve bodies, each with a 2592×1944 resolution. In total we collected 112 defect-free samples and 36 synthesized anomaly samples generated by randomly placing nails on the valves as shown in Figure 3. For testing, we used 34 of the 112 defect-free samples along with all 36 synthesized anomaly samples. The remaining 78 defect-free samples were used for training.

Evaluation Metrics

Given that the anomaly detection model is exclusively trained on normal data, it can only produce anomaly scores at either the image or pixel level. To identify defects, a threshold is essential. To avoid manual threshold selection, following established methodologies, we calculate both image-level AUROC (Area Under the Receiver Operating Characteristic curve) and pixel-level AUPRO (Area Under the Precision-Recall curve) (Bergmann et al. 2019) to evaluate the performance of our model in image-level classification and pixel-level defect localization.

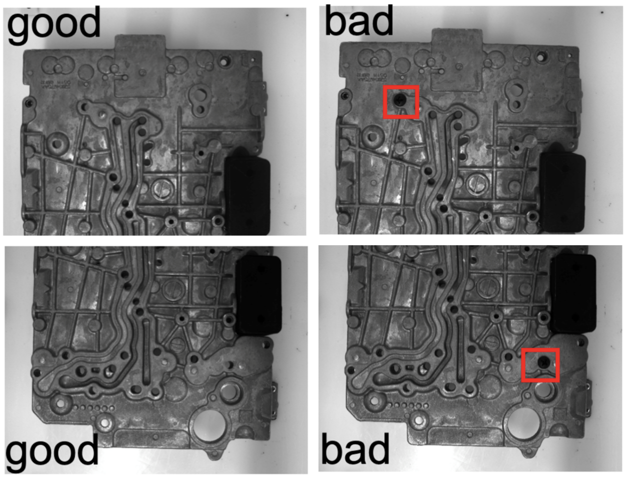

Figure 3: Demonstration of collected valve body data. The valve body data used for demonstration was collected with synthesized defects randomly generated by placing nails on the valve surface.

Models	DRÆM	CutPaste	CFlow-AD	PADIM	PatchCore	PatchFlow
Im-AUROC	88.7	81.9	91.5	89.1	95.1	**96.48**

Table 2: Image level AUROC score of DRÆM (Zavrtanik, Kristan, and Skočaj 2021), CutPaste(Li et al. 2021), PADIM(Defard et al. 2020), PatchCore(Roth et al. 2021), CFlow-AD(Gudovskiy, Ishizaka, and Kozuka 2022), from the VisA(Zou et al. 2022) dataset are presented, with the best scores highlighted in bold.

Implementation Details

We resize the input images to 768×768. Importantly, our approach avoids the use of any data augmentation or normalization techniques on the input images. Features are extracted from the 12th, 19th, and 35th blocks of EfficientNet B5, forming a 3-scale feature pyramid with dimensions $1488 \times 96 \times 96$. Subsequently, a feature adaptor is employed to decrease the feature size to $768 \times 96 \times 96$. The computational complexity and FLOPs of our model scale proportionally with the dimensions of this reduced feature representation. The bottleneck dimension of the flow is fixed at 128.

Results

We compare PatchFlow with several state-of-the-art methods using different learning approaches, including: the generative model-based methods DRÆM (Zavrtanik, Kristan, and Skočaj 2021), CutPaste(Li et al. 2021); the embedding-based model PADIM(Defard et al. 2020), PatchCore(Roth et al. 2021); and the flow-based model CFlow-AD(Gudovskiy, Ishizaka, and Kozuka 2022), CS-Flow(Rudolph et al. 2022) on both MVTecAD dataset(Bergmann et al. 2019) and VisA dataset(Zou et al. 2022). The numerical results for image level anomaly detection on the MVTecAD dataset are listed in 1. As shown, the PatchFlow method outperforms the state-of-the-art PatchCore by reducing the error from 0.9% to 0.72%, which leads to a **20% reduction of error** in terms of image-level anomaly detection, as measured by the image-level AUROC evaluation metric result in 99.28 auroc. PatchFlow consistently demonstrates high performance across various materials, making it a standout model in this evaluation. In table 2 we compare the performance of various state-of-the-art models on VisA(Zou et al. 2022) dataset. PatchFlow reduce the error of 4.9% to 3.52% which leads to **28.2% reduction of error** compared with PatchCore model.

The improvement of PatchFlow over other models on both dataset is notably significant. The high AUROC score of PatchFlow indicates its robustness and reliability in various scenarios, making it a potentially more versatile and effective tool in practical applications.

In the visualization presented in Figure 4, both per-pixel anomaly ground truth maps and corresponding predicted heatmaps and predicted anomalous mask are displayed. This illustration effectively showcases PatchFlow's capability, particularly highlighting its proficiency in accurately locating small defects. The detailed comparison between the ground truth and the model's predictions underlines the precision with which PatchFlow can identify even minute anomalies.

In addition to assessing detection and localization performance, we conduct an evaluation of the computational complexity. The total FLOPs amount to 1.25×10^{10}, resulting in a processing speed of 2.7 frames per second (FPS) when utilizing a single Nvidia GeForce RTX2080Ti GPU.

	Actual Normal	Actual Abnormal
Predicted Normal	34	3
Predicted Abnormal	0	33

Table 3: confusion matrix of model prediction on collected valve body dataset.

The result on Collected data is shown in Table 3.As shown all 34 normal samples were correctly identified. This is reflected in the True Positives (TP) count of 34 in the confusion matrix. Out of 36 bad samples, 3 were incorrectly identified as normal. This is shown as the False Positives (FP) count of 3. The overall accuracy of the model is 95.71%. This indicates a high level of correctness in the model's predictions across all samples.The F1 Score is 95.77%, which suggests a balanced performance between precision and recall. This is particularly valuable in scenarios where both false positives and false negatives are equally costly. Visualization of per-pixel anomaly heatmaps overlaid on valve body images is shown in Figure 5.

Ablation Study

Flow Steps	1	2	3	4	5
Im-AUROC	**99.28**	98.89	98.73	98.57	98.48

Table 4: image level AUROC score of PatchFlow with different number of Flow steps

We conduct a comprehensive ablation study to evaluate the relationship between the number of flow steps and model performance. The results are shown in Table 4. As the number of flow steps increases, the performance of PatchFlow decreases.

Besides we test the influence of number of scale in scale pyramid.As Shown in Table 5, the model utilizing a 3-scale image pyramid achieves the best performance. As the number of scales increases, the model is able to obtain more abstract and generalized information about the image by viewing a larger portion of the scene or object.

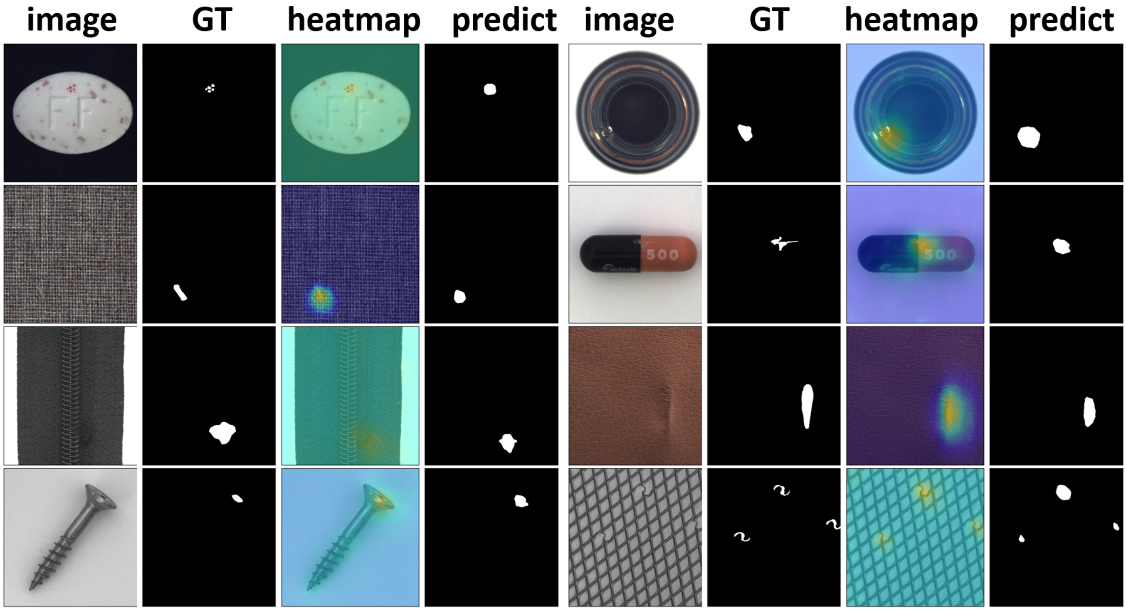

Figure 4: Visualization of per-pixel anomaly Groud Truth; heatmaps and predict anomaly mask

Number of Scale	3	2	1
Image AUROC	**99.28**	98.84	97.78

Table 5: Relation between number of scale and Image level AUROC score

Conclusions

In this work, we have proposed a novel approach for automated anomaly detection in industrial product images. Our method combines local neighbor-aware patch features with a normalizing flow model. A key contribution is the introduction of an adapter module, which bridges the gap between a generic pretrained feature extractor and specialized industrial product images. Additionally, we incorporated a bottleneck structure to reduce computational complexity while preserving the mapping capabilities of the flow architecture.

Through comprehensive empirical evaluation, our model achieves new state-of-the-art performance for image-level anomaly detection on the challenging MVTec AD dataset(Bergmann et al. 2019) , reducing the error rate by 20% compared to prior SOTA(Roth et al. 2021) and attaining an image level AUROC score of 99.28% and VisA Dataset(Zou et al. 2022), reducing the error rate by 28.2% compared with (Roth et al. 2021) and attaining an image level AUROC score of 96.48%. PatchFlow's performance on both datasets, as highlighted by its leading AUROC score, represents a notable advancement in the field of anomaly detection, setting a new benchmark for future models and applications. Further experiments on a proprietary die casting dataset yield a defect detection accuracy of 95.77%, without requiring any anomalous training data.

These results highlight the potential of leveraging computer vision and deep learning to enhance automated visual inspection for critical manufacturing techniques such as die casting. Our method illustrates a viable path to improving quality control and reducing waste in industrial production environments. This work constitutes meaningful progress, and could provide a strong foundation for future research and adoption in real-world inspection systems.

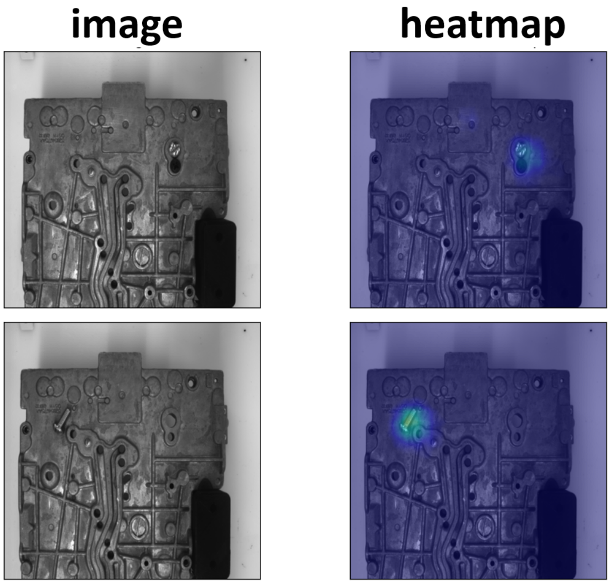

Figure 5: Defect localization heatmaps for valve body images. Brighter green regions indicate areas with higher anomaly scores.

References

Ardizzone, L.; Lüth, C.; Kruse, J.; Rother, C.; and Köthe, U. 2019. Guided Image Generation with Conditional Invertible Neural Networks. arXiv:1907.02392.

Bergmann, P.; Fauser, M.; Sattlegger, D.; and Steger, C. 2019. MVTec AD — A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9584–9592.

Defard, T.; Setkov, A.; Loesch, A.; and Audigier, R. 2020. PaDiM: a Patch Distribution Modeling Framework for Anomaly Detection and Localization. *CoRR*, abs/2011.08785.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. ???? Imagenet: A large-scale hierarchical image database.

Dinh, L.; Sohl-Dickstein, J.; and Bengio, S. 2017. Density estimation using Real NVP. arXiv:1605.08803.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, 2672–2680.

Gudovskiy, D.; Ishizaka, S.; and Kozuka, K. 2022. 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 (WACV)*, 98–107.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition. arXiv:1512.03385.

Kingma, D. P.; and Welling, M. 2022. Auto-Encoding Variational Bayes. arXiv:1312.6114.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. NIPS'12, 1097–1105. Red Hook, NY, USA: Curran Associates Inc.

Lee, S.; Lee, S.; and Song, B. C. 2022. CFA: Coupled-hypersphere-based Feature Adaptation for Target-Oriented Anomaly Localization. arXiv:2206.04325.

Li, C.-L.; Sohn, K.; Yoon, J.; and Pfister, T. 2021. CutPaste: Self-Supervised Learning for Anomaly Detection and Localization. arXiv:2104.04015.

Liu, Z.; Zhou, Y.; Xu, Y.; and Wang, Z. 2023. SimpleNet: A Simple Network for Image Anomaly Detection and Localization. arXiv:2303.15140.

Rezende, D. J.; and Mohamed, S. 2016. Variational Inference with Normalizing Flows. arXiv:1505.05770.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv:1505.04597.

Roth, K.; Pemula, L.; Zepeda, J.; Schölkopf, B.; Brox, T.; and Gehler, P. 2021. Towards Total Recall in Industrial Anomaly Detection. arXiv:2106.08265.

Rudolph, M.; Wandt, B.; and Rosenhahn, B. 2021. Same Same But DifferNet: Semi-Supervised Defect Detection with Normalizing Flows. In *Winter Conference on Applications of Computer Vision (WACV)*.

Rudolph, M.; Wehrbein, T.; Rosenhahn, B.; and Wandt, B. 2022. Fully Convolutional Cross-Scale-Flows for Image-based Defect Detection. In *Winter Conference on Applications of Computer Vision (WACV)*.

Sai, T. V.; Vinod, T.; and Sowmya, G. 2017. A critical review on casting types and defects. *Engineering and Technology*, 3(2): 463–468.

Sohl-Dickstein, J.; Weiss, E. A.; Maheswaranathan, N.; and Ganguli, S. 2015. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. arXiv:1503.03585.

Tan, M.; and Le, Q. V. 2020. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. arXiv:1905.11946.

Tien, T. D.; Nguyen, A. T.; Tran, N. H.; Huy, T. D.; Duong, S. T.; Nguyen, C. D. T.; and Truong, S. Q. H. 2023. Revisiting Reverse Distillation for Anomaly Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 24511–24520.

Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Sablayrolles, A.; and Jégou, H. 2021a. Training data-efficient image transformers & distillation through attention. arXiv:2012.12877.

Touvron, H.; Cord, M.; Sablayrolles, A.; Synnaeve, G.; and Jégou, H. 2021b. Going deeper with Image Transformers. arXiv:2103.17239.

Yu, J.; Zheng, Y.; Wang, X.; Li, W.; Wu, Y.; Zhao, R.; and Wu, L. 2021. FastFlow: Unsupervised Anomaly Detection and Localization via 2D Normalizing Flows. arXiv:2111.07677.

Zagoruyko, S.; and Komodakis, N. 2017. Wide Residual Networks. arXiv:1605.07146.

Zavrtanik, V.; Kristan, M.; and Skočaj, D. 2021. DRAEM – A discriminatively trained reconstruction embedding for surface anomaly detection. arXiv:2108.07610.

Zhang, B.; Yang, B.; Vian, C.; and Wang, X. 2023a. A.I. Vision System for Automated Casting Quality Inspection. *Die Casting Engineer*, 10–16.

Zhang, H.; Wu, Z.; Wang, Z.; Chen, Z.; and Jiang, Y.-G. 2023b. Prototypical Residual Networks for Anomaly Detection and Localization. arXiv:2212.02031.

Zou, Y.; Jeong, J.; Pemula, L.; Zhang, D.; and Dabeer, O. 2022. SPot-the-Difference Self-Supervised Pre-training for Anomaly Detection and Segmentation. *arXiv preprint arXiv:2207.14315*.