Knowledge-Based Systems 262 (2023) 110251

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

# Anomaly detection framework for unmanned vending machines[☆]

Zongyang Da[a], Yujie Dun[a,*], Chengxu Liu[a], Yuanzhi Liang[b], Yao Xue[a], Xueming Qian[a,c,d]

[a] School of Information and Communication Engineering, Xi'an Jiaotong University, Xi'an 710049, China
[b] School of Software Engineering, Xi'an Jiaotong University, Xi'an 710049, China
[c] Key Laboratory for Intelligent Networks and Network Security, Ministry of Education, Xi'an Jiaotong University, Xi'an 710049, China
[d] SMILES Laboratory, School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China

ARTICLE INFO

*Article history:*
Received 11 April 2022
Received in revised form 27 December 2022
Accepted 2 January 2023
Available online 5 January 2023

*Keywords:*
Unmanned retail
Anomaly detection
Classification
Loss function
Dataset

ABSTRACT

Unmanned vending machines (UVMs) based on computer vision have shown huge commercial potential in unmanned retail. Anomaly detection is the problem of recognizing the falling-over and occlusion anomalies in UVMs. However, considering various anomaly features with different feature complexities, the existing methods are not sufficiently effective. In this paper, we propose an unmanned retail anomaly detection method based on deep convolutional neural networks (CNNs) called the complexity-classification anomaly detection (ClassAD) framework. The ClassAD method consists of two modules: a complexity rating module and a classification module. These two modules jointly input images of different feature complexity into different-capacity networks, which makes full use of the feature extraction ability of CNNs. ClassAD includes a decomposition feature enhancement approach to enhance inference speed. Furthermore, we introduce complexity-class loss and multi-instance loss in ClassAD for accuracy and learning stability. Experiments show that ClassAD yields promising results compared with state-of-the-art methods in both effectiveness and efficiency. In addition, we propose a UVM-Anomaly dataset for anomaly detection in UVMs.

## 1. Introduction

Vision-based unmanned vending machines (UVMs) [1–3] have emerged and drawn attention in recent years due to their conspicuous commercial value. UVMs enable direct interactions between customers and products. In addition, UVMs help sellers to maintain inventory in a flexible manner. Moreover, a simple camera is enough for the machine, which can achieve large-scale cost savings. Considering the advantages of UVMs, many enterprises such as Amazon, Walmart, Alibaba, etc., have pushed out their own products and tried to occupy the market. However, there still exist some issues that need to be improved. Currently, most researchers focus on the recognition of stock keeping units (SKUs) [4–7] in UVMs.

Anomaly detection [8,9] is also an essential part of UVMs. Anomalies always occur in UVMs because of flexible interactions between containers and customers. As shown in Fig. 1, falling-over anomalies make SKUs hard to recognize, while occlusion anomalies obstruct the view of SKUs. SKU recognition algorithms cannot tackle unpredictable anomalies. Currently, anomaly detection for UVMs lacks sufficient study. The economic loss caused by the internal anomaly occupies a large proportion of the total loss [10]. Thus, there is a strong and practical necessity for anomaly detection in UVMs.

Currently, anomaly detection in UVMs has two main challenges: (1) **feature unpredictability**, which leads to large intraclass variance, and (2) the **imbalance problem**, including feature complexity imbalance and anomaly class imbalance.

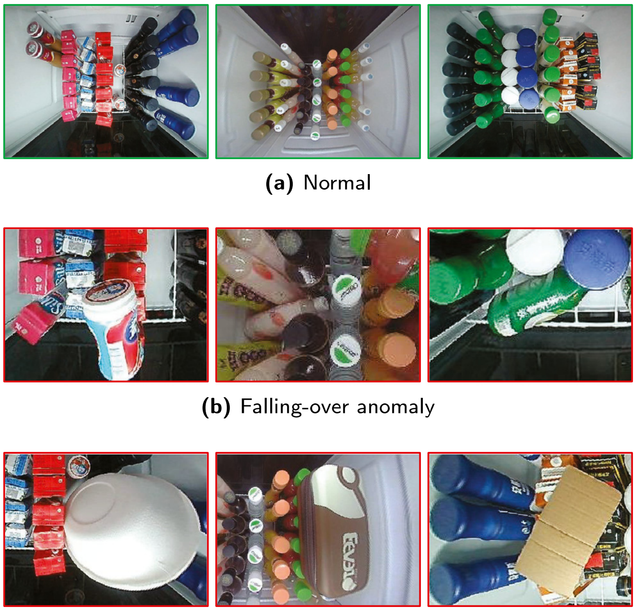
1. **Feature unpredictability:** Feature unpredictability mainly consists of three aspects: (1) a wide variety of SKUs; (2) different angles and positions; and (3) unknown foreign matter. Good performance relies on high similarity within classes and large differences between classes. However, we need to overcome large intraclass variance due to the feature unpredictability of anomaly features.

[☆] This work was supported in part by the NSFC, under Grants 62103317, 62272380 and 61872286, in Part by the Xi'an Science and Technology Planning Project, China under Grant 20YXYJ0005-3, in part by Humanities and Social Sciences Foundation of Ministry of Education, China under Grant 16XJAZH003, in part by Natural Science Foundation of Shaanxi Province, China under Grant 2021JQ-058, in part by the ShaanXi Province, China under Grant 2018JM6092, and in part by the the Science and Technology Program of Xi'an, China under Grant 21RGZN0017.

[*] Corresponding author.

*E-mail addresses:* dzy1134483011@stu.xjtu.edu.cn (Z. Da), dunyj@mail.xjtu.edu.cn (Y. Dun), chengxuliu@stu.xjtu.edu.cn (C. Liu), liangyzh13@stu.xjtu.edu.cn (Y. Liang), xueyao@xjtu.edu.cn (Y. Xue), qianxm@mail.xjtu.edu.cn (X. Qian).

https://doi.org/10.1016/j.knosys.2023.110251
0950-7051/© 2023 Elsevier B.V. All rights reserved.



**(a)** Normal

**(b)** Falling-over anomaly

**(c)** Occlusion anomaly

**Fig. 1.** Illustration of UVM scenes. (a) is the normal situation. (b) is the falling-over anomaly situation. (c) is the occlusion anomaly situation.

2. **Imbalance problem:** The imbalance problem can be divided into two aspects: feature complexity imbalance and class imbalance. (1) Feature complexity imbalance reflects that some SKUs have complex textured surfaces, while the background has simple features. A rough network that uniformly processes images of different feature complexities can hardly determine suitable model capacity. This results in overfitting for low-complexity images and insufficient learning for high-complexity images; (2) Class imbalance refers to detectors evaluating a large number of images, but only a few images contain anomalies. Anomalies occupy only a small area on the images compared with normal samples. Class imbalance causes inefficient training, and normal samples overwhelm and degenerate the network.

To intuitively show the feature complexity imbalance problem, we calculate the peak signal-to-noise ratio (PSNR) values of all the images in our proposed dataset UVM-Anomaly, which is shown in Fig. 2. All the PSNR values are calculated through a well-trained superresolution model RCAN [11]. The PSNR is a widely used metric for quantitatively evaluating image restoration quality, which is related to feature complexity [12]. We divide all the images into three ranges: low-complexity, medium-complexity and high-complexity, based on the PSNR values from high to low. As shown in Fig. 2, images with high PSNR values generally show smooth features, while images with low PSNR values show complex features. We will discuss the feature complexity in Section 6.1.

Fortunately, computer vision offers deep convolutional neural networks (DCNNs) for anomaly detection [13,14]. Li et al. [15] proposed a cuboid-patch-based method for anomaly detection and localization in video surveillance. Massoli et al. [16] proposed a novel framework, named multilayer one-class classification (MOCCA), using autoencoders to train and test DL models on the AD task. Roth et al. [17] proposed PatchCore, which uses a maximally representative memory bank of nominal patch features. In the existing works, the following issues are worthy of attention: (1) The feature distribution directly determines the feasibility of the work in practice. When anomaly features are unpredictable or similar to normal features, the performance will suffer greatly, especially in the case of the one-class classification method. (2) Due to the feature complexity imbalance problem, a single DCNN can hardly strike a balance between effectiveness and efficiency when working on different complexity features. (3) In actual use, the normal situation is far more common than the anomaly situation, and the anomalies will be hidden. Therefore, the class imbalance problem affects the sufficient learning of the detector, especially for the UVM scenarios.

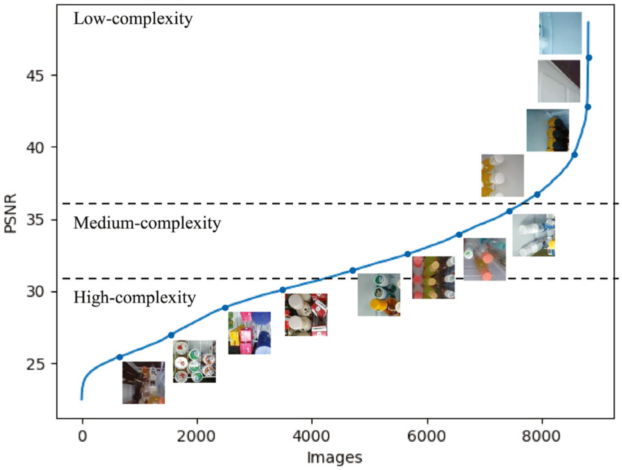


**Fig. 2.** The sorted image-PSNR curve of the UVM-Anomaly dataset. The image patches around the curve are cropped from original images in the dataset.

To address the challenges above, we propose a new novel framework for anomaly detection in UVMs, namely, complexity-classification anomaly detection (ClassAD). To the best of our knowledge, ClassAD is the first anomaly detection framework in UVMs. (1) ClassAD consists of two modules: **complexity rating module** and **classification module**. The complexity rating module is a simple classification network that classifies the input images into different feature complexity classes. The classification module contains several CNN branches, each of which can classify images into anomaly and normal classes. We design this new complexity rating mechanism which uses two modules jointly, different branches deal with corresponding images with different feature complexity. ClassAD can address the feature complexity imbalance and achieve a balance between effectiveness and efficiency. (2) We propose an image enhancement approach named **decomposition feature enhancement** in front of ClassAD, which utilizes a self-adaptive method to determine the separation standard to separate images into small subimages. This approach can reduce the adverse effect brought by high intraclass variance as well as accelerate inference speed and reduce the training difficulty. (3) We propose **complexity-class loss** and **multi-instance loss**. Complexity-class loss can accelerate the convergence of feature complexity classification; multi-instance loss can reduce class imbalance between the normal and anomaly. These two losses cooperatively make the feature complexity class well-distributed and the classification effective. The whole pipeline is complete and novel, which is effective on anomaly detection on UVMs. (4) Finally, we propose a dataset named **UVM-Anomaly** for anomaly detection in the UVMs. UVM-Anomaly includes 85 types of different SKUs and different kinds of falling-over and occlusion anomalies.

The main contributions of this paper can be summarized as follows:

1. We propose the complexity classing anomaly detection (ClassAD) framework for anomaly detection, especially for

UVM scenes. It achieves a suitable trade-off between fidelity and efficiency for anomaly classification.

2. We propose the decomposition-feature-enhancement approach. It self-adaptively generates subimages to accelerate the training process.
3. We propose two novel loss functions for classification networks. Complexity-class loss speeds up the convergence, and multi-instance loss balances the anomaly and normal instances.
4. We propose a large-scale UVM-Anomaly dataset for SKU anomaly detection.

The remainder of this paper is organized as follows. Related work is reviewed in Section 2. The proposed method is elaborated in Section 3. The decomposition feature enhancement and two loss functions are also explained in this section. The UVM-Anomaly dataset is described in Section 4. Experimental results are shown in Section 5. Discussions are presented in Section 6. We conclude our work in Section 7.

## 2. Related works

In this section, we briefly review and describe some related work on UVMs. Moreover, we introduce some well-performing deep learning methods for anomaly detection in the second part.

### 2.1. Unmanned techniques

At present, the unmanned retail field has three kinds of mainstream features: quick response (QR) code recognition [18], radio frequency identification (RFID) [19], and vision-based techniques. The QR code technique uses QR code labels attached to items; RFID recognizes items with electronic tags and radio frequency transmitting and reception circuits; vision-based techniques utilize cameras and computer vision technology for recognition and detection. We will mainly focus on vision-based unmanned retail works.

Zhang et al. [6] proposed several recently developed deep learning models with their self-compiled large-scale dataset, the images of which were captured in a refrigerator equipped with cameras. Liu et al. [7] proposed a smart unstaffed retail shop scheme based on artificial intelligence and the internet of things. In addition, an end-to-end classification model trained by the Mask-RCNN method was developed for SKU counting and recognition. Zhao et al. [20] proposed a unified object detection framework used for dense scenarios in retail images, which consisted of a hierarchical labelling pattern, a similarity recognition network and an optimized NMS algorithm. Nogueira et al. [21] proposed a low-cost deep learning approach to estimate the number of people in retail stores in real time. Wang et al. [22] proposed an integrated model to coordinate the benefits of all decision subjects and designed a multiobjective optimization method. The existing methods are mainly focused on object detection on SKUs, and most papers emphasize the design of networks. However, few researchers have concentrated on anomaly detection. Our unmanned retail anomaly detection method uses few hardware conditions, only a fish-eye camera, while achieving high accuracy on anomaly classification.

Some researchers have also proposed a variety of datasets. Zhang et al. [2] collected more than 30,000 images of unmanned retail containers using a refrigerator affixed with different cameras under both static and dynamic recognition environments. These images were categorized into ten kinds of beverages. Through manual labelling, the dataset above contained 155,153 instances. Hao et al. [1] proposed a hierarchical large-scale object detection dataset containing 38,000 images of 24 fine-grained and three coarse classes. Cai et al. [23] collected a large-scale object localization and counting dataset including 50,394 images with more than 1.9 million object instances in 140 categories. Wei et al. [24] proposed a dataset containing 200 SKUs and 83,739 images, which included single-product and multiproduct images collected in different scenes. These datasets have a tremendous number of SKUs but no anomaly samples. In contrast, our UVM-Anomaly dataset consists of normal and anomalous samples, and as far as we know, the UVM-Anomaly dataset is the first dataset for anomaly detection in UVMs.

### 2.2. Anomaly detection

Anomaly detection is a technique for classifying whether patterns exist in data that do not conform to normal behaviour. It has been used in a wide range of applications, such as cancer cell detection [8,25], fraud detection [26] and industrial detection systems [27]. The principle of anomaly detection with deep learning is to extract useful feature information from sample images to judge whether or where abnormal conditions exist. Based on the difference in the availability of labels [28], we can divide the present approaches into three main classes: supervised deep anomaly detection [29], semisupervised deep anomaly detection [26] and unsupervised deep anomaly detection [30].

Supervised deep anomaly detection is a commonly used approach in the anomaly detection domain which attaches each instance in the dataset with a normal or anomalous label. Napoletano et al. [31] proposed a region-based method for the detection of anomalies in scanning electron microscope images. Li et al. [15] proposed a cuboid-patch-based method characterized by a cascade of classifiers. Some methods also use one-class classification for anomaly detection. Massoli et al. [16] proposed a novel framework, named multilayer one-class classification (MOCCA), using autoencoders to train and test DL models on the AD task. Roth et al. [17] proposed PatchCore, which uses a maximally representative memory bank of nominal patch features. The performance of the deep supervised classifier is suboptimal due to class imbalance, where normal class instances are far more frequent than anomalous class instances.

Semisupervised deep learning is more widely used than supervised learning in the domain of anomaly detection. An et al. [32] proposed an autoencoder to extract feature information from all normal instances. Akcay et al. [33] introduced an anomaly detection model using a conditional generative adversarial network that jointly learns the generation of high-dimensional image space and the inference of latent space.

Unsupervised deep learning for anomaly detection detects outliers based on intrinsic properties of the data instances. Yao et al. [34] proposed an unsupervised algorithm called KfreqGAN, which is based on an adversarially trained sequence predictor.

These methods can work well in certain scenes. However, in the product scenery, large intraclass variance on feature and complexity imbalance problem hinter the performance. Our proposed classAD is effective and stable in UVM scenery, which can address these problems well.

## 3. Method

In this section, we introduce ClassAD, a novel framework for UVMs. As illustrated in Fig. 3, the network of ClassAD consists of two modules, namely, a complexity rating module and a classification module. To start with, the image $x$ is divided into subimages $x_i$ $(i = 1, \dots, N)$ through the decomposition feature enhancement approach, where $N$ represents the number of separated regions. Then, the calculation procedure for each subimage is independently conducted. We select the subimage $x_i$ as an

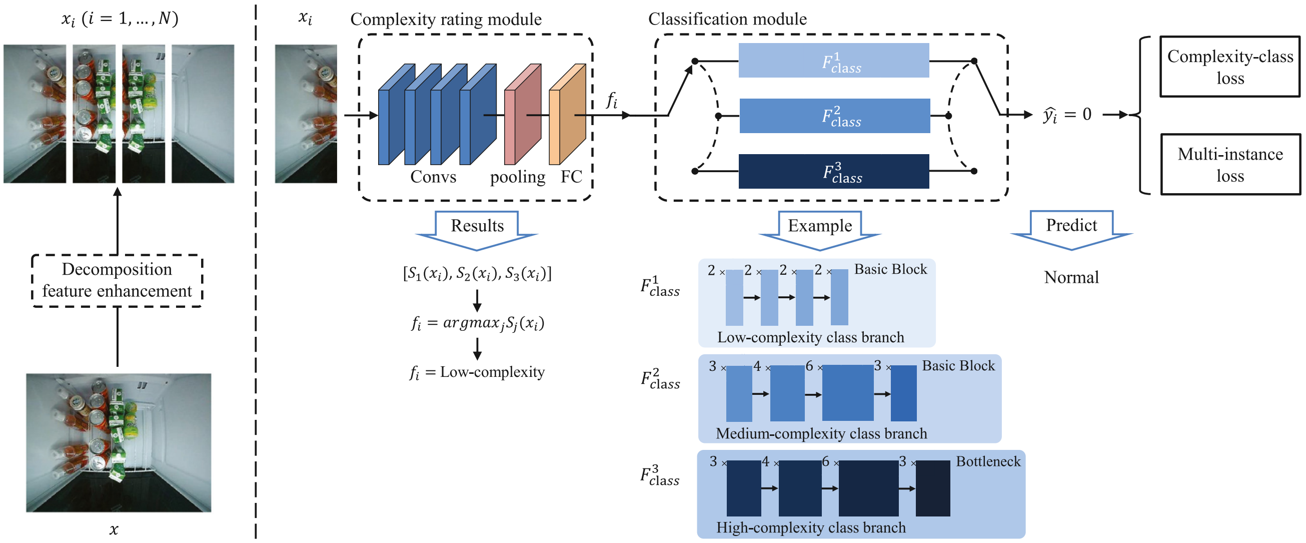

$x_i$ $(i = 1, \dots, N)$
Decomposition feature enhancement
$x$
$x_i$
Complexity rating module
Convs
pooling
FC
$f_i$
Results
$[S_1(x_i), S_2(x_i), S_3(x_i)]$
$f_i = argmax_j S_j(x_i)$
$f_i$ = Low-complexity
Classification module
$F^1_{\text{class}}$
$F^2_{\text{class}}$
$F^3_{\text{class}}$
Example
2 × 2 × 2 × 2 × Basic Block
Low-complexity class branch
3 × 4 × 6 × 3 × Basic Block
Medium-complexity class branch
3 × 4 × 6 × 3 × Bottleneck
High-complexity class branch
$\hat{y}_i = 0$
Complexity-class loss
Multi-instance loss
Predict
Normal

**Fig. 3.** The overview of our proposed ClassAD method. The number of subimages $N = 4$. The number of feature complexity classes $C = 3$. We only demonstrate the procedure for a single subimage $x_i$.

example. (1) The complexity rating module has several convolutional layers to generate the complexity score vector $S_j(x_i)$ $(j = 1, \dots, C)$ for $x_i$. $x_i$ is classified by the index of the maximum complexity score: $f_i = \text{argmax}_j\, S_j\,(x_i)$. $f_i$ represents the complexity class of $x_i$. (2) The classification module contains $C$ branches $F^j_{\text{class}}$ $(j = 1, \dots, C)$. Different branches $F_{\text{class}}$ have similar CNN frameworks but different capacities to deal with subimages with different feature complexities. According to $f_i$, $x_i$ is sent into the corresponding branch, and the anomaly probability value $\widehat{y}_i$ is obtained. The same process is repeated on all the subimages.

Finally, the framework obtains the anomaly probability vector $[\widehat{y_1}, \widehat{y_2}, \dots, \widehat{y_N}]$ for all the subimages and then calculates the anomaly class of the image $x$. In the training process, we utilize complexity-class loss and multi-instance loss to train ClassAD.

### 3.1. Complexity rating module

In this subsection, we elaborate on the complexity rating module. The aim of this module is to discriminate whether the input subimage is easy for CNNs to classify. As shown in Fig. 3, the complexity rating module contains four convolution layers, an average pooling layer and a fully connected layer. The convolution layers are designed to extract shallow feature information, while the average pooling layers fuse spatial per-pixel information. The fully connected layer has several output units, which represent the probabilities of different complexity classes. In our experiment, we set the number of complexity classes to be $C = 3$, and the output complexity score vector for the subimage $x_i$ is represented as $S_j(x_i)$ $(j = 1, \dots, C)$. During the training process, we use the complexity rating module as a classification model and update the parameters. This network has a lightweight structure, and it ensures fast inference speed and low computational cost.

### 3.2. Classification module

In this subsection, we elaborate the classification module with several classification network branches $F^j_{\text{class}}$ $(j = 1, \dots, C)$ for anomaly detection. In our experiments, we set $C = 3$, and the classification module has three branches: low-complexity class branch, medium-complexity class branch, and high-complexity class branch. Each branch is an anomaly detection classification network, and different branches deal with images in different feature complexity ranges. We use ResNet50 [35] as the high-complexity class branch. We control the capacity of the networks and use ResNet18 and ResNet34 as the other two branches.

In the training process, an input image $x$ is divided into subimages $x_i$ $(i = 1, \dots, N)$ by the decomposition feature enhancement approach. After that, the complexity rating module processes each subimage and generates a complexity score vector $S_j\,(x_i)$ $(j = 1, 2, \dots, C)$ for the subimage $x_i$. During the classification module, different from the testing process, each subimage $x_i$ will be sent into all $C$ classification branches. We define the output of the $j$th branch as $F^j_{class}\,(x_i)$. The network multiplies the complexity score vector with all the outputs to generate the final probability vector $\widehat{y}_i$ as

$$\widehat{y}_i = \sum_{j}^{C} S_j\,(x_i) \cdot F^j_{\text{class}}\ \ (x_i)\,, \tag{1}$$

The final probability vector $\widehat{y}_i$ will be involved in the calculation of loss functions and parameter updating. In the testing process, the complexity rating module remains the same. Before the classification module, the network calculates the complexity class $f_i = \text{argmax}_j\, S_j\,(x_i)$ for subimage $x_i$. $x_i$ will be processed with the single $f_i$th branch in the classification module and generate the predicted probability value $\widehat{y}_i$.

### 3.3. Decomposition feature enhancement

In this section, we elaborate on the decomposition feature enhancement approach, which generates subimages $x_i$ $(i = 1, \dots, N)$ from the original input image $x$. As shown in Fig. 4, this approach contains several steps to generate a decomposition standard upon the dataset. When training and testing, the framework uses this standard to generate subimages.

We assume one single image in the dataset as $x$. In the first step, the decomposition feature enhancement approach utilizes canny edge detection and generates a binarized image $b$. In the second step, the approach utilizes Hough vertical line detection. We obtain the pixel-level coordinates of the vertical lines $L$. In the third step, the approach utilizes k-means clustering conducted on the output coordinates of all the dataset images. The clustering parameter $k$ is set as $N - 1$, while $N$ refers to the number of separated regions. After the decomposition feature enhancement, the framework sees the clustering output $P_i$ $(i = 1, \dots, N - 1)$ as the decomposition standard for all the images.

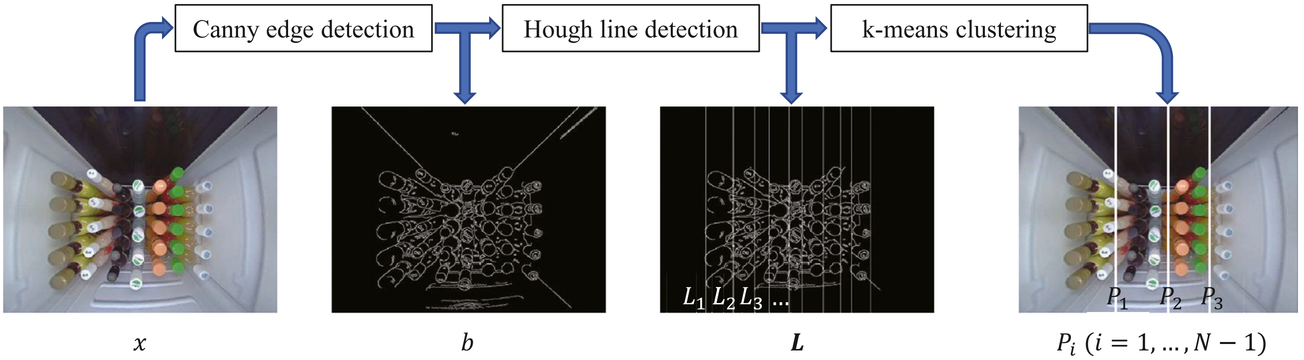


**Fig. 4.** The whole process of the decomposition feature enhancement approach. $x$ is a single image in the dataset. $b$ is the output image of Canny edge detection. $L$ are the coordinates of the detected lines after Hough line detection. $P_i$ $(i = 1, \ldots, N-1)$ are the output of k-means clustering.

### 3.4. Loss functions

In this subsection, we elaborate our two proposed loss functions, namely, complexity-class loss and multi-instance loss. The former is used in the complexity class module, and the latter is used in the classification module.

#### 3.4.1. Complexity-class loss

The complexity-class loss is composed of two parts. The first part is the cross-entropy (CE) loss. CE loss is designed for the classification of different feature complexities. CE loss $L_{CE}(x_i)$ for subimage $x_i$ is defined as

$$L_{CE}(x_i) = -\sum_{j=1}^{C} l_{i,j} \cdot log\left(S_j(x_i)\right), \tag{2}$$

where $C$ is the number of complexity classes; $l_{i,j}$ is the ground-truth complexity class label for $x_i$; and $S_j$ is the $j$th value in the complexity score vector, which indicates the probability that $x_i$ belongs to the $j$th complexity class.

The second part enlarges the gaps among complexity classes. We assume that a larger class gap can accelerate the convergence speed of the network of the complexity class module. The second part $L_S(x_i)$ for $x_i$ is defined as

$$L_S(x_i) = -\sum_{j=1}^{C-1}\sum_{k=j+1}^{C} \left|S_j(x_i) - S_k(x_i)\right|, \text{ s.t. } \sum_{j=1}^{C} S_j(x_i) = 1. \tag{3}$$

$L_S(x_i)$ is the negative number of distances between each pair of complexity scores.

The complexity-class loss $L_{CC}$ is composed of the two parts above:

$$L_{CC} = \omega_1 \cdot L_{CE} + \omega_2 \cdot L_S, \tag{4}$$

where $\omega_1$ and $\omega_2$ are the weights to balance different loss items. $L_{CE}$ is used to ensure the classification quality for feature complexity, and $L_S$ ensures that each branch can be chosen properly.

#### 3.4.2. Multi-instance loss

The multi-instance loss is also composed of two parts. The first part is the binary cross-entropy (BCE) loss, which is defined as

$$L_{BCE}(x_i) = -\sum_{k=1}^{2} y_{i,k} \cdot log\left(\widehat{y_{i,k}}\right), \tag{5}$$

where $y_{i,k}$ is the ground-truth anomaly label for $x_i$. $k = 0$ and $k = 1$ represent the normal and anomaly, respectively. $\widehat{y_{i,k}}$ is the predicted probability value. BCE loss is used for the classification of anomalies and normal conditions.

The second part balances the class imbalance and enhances the overall classification accuracy. The second part deals with the scene in which the number of anomaly instances is smaller than normal. The second part $L_M$ is defined as

$$L_M = \begin{cases} (1-\beta) \cdot \max_{1\le i\le N}\left[-y_{i,k} \cdot \log\left(\widehat{y_{i,k}}\right)\right], \text{s.t.} y_{i,0} = 1 \\ \beta \cdot \sum_{i=1}^{N} -y_{i,k} \cdot \log\left(\widehat{y_{i,k}}\right), \text{s.t.} y_{i,1} = 1 \end{cases}, \tag{6}$$

where $\beta$ is the balanced parameter. The second part differentiates between the weights on the normal and anomalous ground truth. The balance parameter is set to $\beta \ge 0.5$, which can ensure that anomaly instances weigh more when updating network parameters.

The whole multi-instance loss $L_{MI}$ is composed of these two parts:

$$L_{MI} = \omega_3 \cdot L_{BCE} + \omega_4 \cdot L_M, \tag{7}$$

where $\omega_3$ and $\omega_4$ are the weights to balance different loss items. $L_{BCE}$ ensures the classification quality, and $L_M$ makes the network focus more on the anomaly instances.

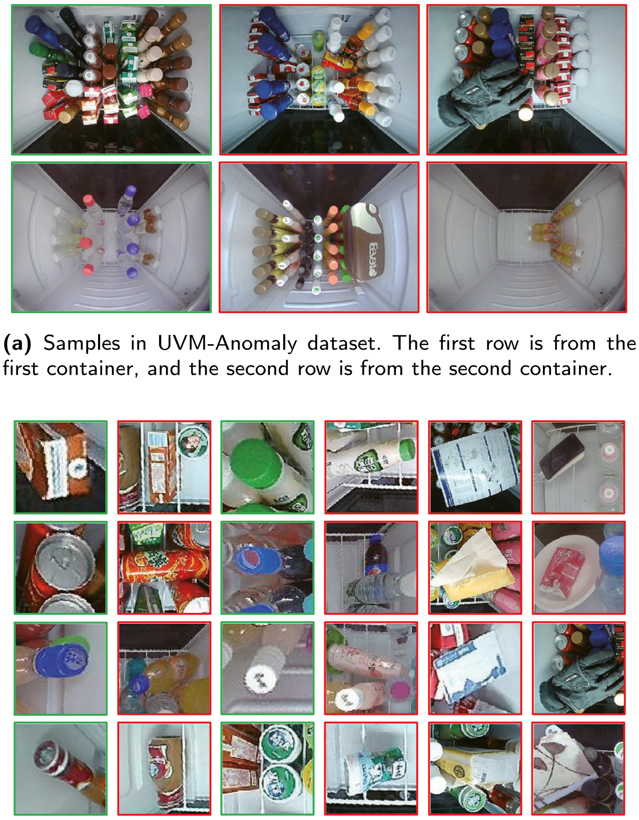
## 4. Dataset

In this section, we introduce our proposed UVM-Anomaly dataset for anomaly detection in UVMs. We provide some basic information on the UVM-Anomaly dataset in Table 1. Some samples in the dataset are shown in Fig. 5. The images in the UVM-Anomaly dataset were collected in two real UVM containers, which were equipped with four 300,000-pixel high-resolution fish-eye cameras. For the container, the length was 600 mm, the width was 500 mm, and the height was 350 mm. There are 85 types of different SKUs in our dataset, including 69 bottled drinks, 9 box-packed beverages and 8 types of cans. All SKUs can be bought in Chinese markets. The images are all of size of 640 × 480.

As we can see from Table 1 and Fig. 5, (1) the variety of SKU features is rich, which shows the feature unpredictability; (2) the number of anomaly samples is smaller than the number of normal samples, and the coverage area of the anomaly instance is also small. These show the class imbalance problem.

We also show some extra information of the dataset related to its SKU number and anomaly type.

1. **SKU number:** In the training set, the SKU number is between 1 and 12: 1208;
   The SKU number between 12 and 30: 1594;
   The SKU number between 30 and 50: 1625.
   In the testing set, the SKU number is between 1 and 12: 41;
   The SKU number between 12 and 30: 103;
   The SKU number between 30 and 50: 117.



**(a)** Samples in UVM-Anomaly dataset. The first row is from the first container, and the second row is from the second container.

**(b)** Intances in UVM-Anomaly dataset

**Fig. 5.** Illustration of some samples and instances in the UVM-Anomaly dataset. Images with green borders are normal, and those with red borders are anomalous.

**Table 1**
The statistics of UVM-Anomaly dataset.

| Set | Label | Image number |
|---|---|---|
| Training set | normal | 2594 |
| | anomaly | 1833 |
| | total | 4427 |
| Testing set | normal | 160 |
| | anomaly | 101 |
| | total | 261 |
| Total | | 4688 |

2. **Anomaly type:** In the training set, the images that contain falling-over: 1684;
   The images that contain falling-over: 670.
   In the testing set, the images that contain occlusion: 88;
   The images that contain falling-over: 35.

## 5. Experiments

In this section, we demonstrate the effectiveness of ClassAD for anomaly detection in the UVMs by contrasting experiments, evaluated by the UVM-Anomaly dataset. We set the number of decomposition regions to $N = 4$ and Section 6.2 will demonstrate its effectiveness. In the complexity-class loss, we set $\omega_1 = 5$ and $\omega_2 = 1$; in the multi-instance loss, we set $\omega_3 = 1$ and $\omega_4 = 0.2$; and we set the balance parameter in the multi-instance loss $\beta = 0.7$. The specific experimental results are shown in the following sections.

### 5.1. Training and testing data

We used the training set of UVM-Anomaly for training. We separated all the images into subimages by the decomposition feature enhancement approach. In the collection of the dataset, anomalies usually occur in partial areas. Thus, there inevitably exist repetitions of normal subimages after decomposition feature enhancement. We removed the repetitive normal subimages, and the total number of training samples was 8603.

These subimages were divided into three complexity classes. We put all the subimages into a well-trained RCAN [11] superresolution model and obtained a peak signal-to-noise ratio (PSNR) value ranking. Then, we divided all the subimages by their PSNR values from high to low, which means feature complexity from low to high. We used a division ratio of $[0.2, 0.3, 0.5]$. Low-complexity images accounted for a small proportion of the whole dataset, and high-complexity images accounted for a large proportion, as shown in Fig. 2. Therefore, the numbers of subimages in the low-complexity range, medium-complexity range and high-complexity range were 1721, 2581, and 4301, respectively. Finally, we obtained the three-complexity-class training data for classification module pretraining.

We used the testing set of UVM-Anomaly for testing. We used the decomposition standard used for the training data to generate subimages. The total number of subimages for testing was 652.

### 5.2. Implementation details

We used the PyTorch framework [42] using NVIDIA RTX 2080 Ti GPUs. ResNet [35] pretrained on ImageNet [43] was used as the backbone of ClassAD. All the training samples were augmented by random rotation, translation, brightness variance and contrast variation. All the samples were resized into 200 × 400. The training batch size was set as 16. We used the SGD optimizer, where momentum was set to 0.9, and weight_decay was set to 0.0005. The initial learning rate was set to 0.001 and was divided by 10 after 20 epochs.

### 5.3. Training details

The training process of ClassAD can be divided into two steps. The first step is to pretrain each branch of the classification module. The second step is to jointly train the complexity rating module and classification module. The reason why we train ClassAD in this fashion is that if we train the whole ClassAD framework from scratch, the training may become trapped in overfitting or unbalance among different branches.

The first step: The pretraining process of the classification module utilized all the training images. The number of total training iterations was set to 40$k$. We only used Binary Cross Entropy loss as in Eq. (5). The second step: The complexity rating module and classification module jointly updated the parameters. The number of total iterations was set to 24$k$. We used complexity-class loss and multi-instance loss to jointly train the complexity rating module.

### 5.4. Comparisons with state-of-the-art methods

We conducted quantitative and qualitative analyses to evaluate the performance of ClassAD for anomaly detection. We quantitatively evaluated the average accuracy, balanced F score (F-1 score) and AUC (area under the receiver operating characteristic curve) score to evaluate the effectiveness; we employed average floating point operations (FLOPs) [44] and frame per second (FPS) to evaluate the efficiency; and we employed the number of parameters to evaluate the model complexity. The

**Table 2**
Comparison of performances of different models for anomaly detection in the UVMs. For the 2nd–4th columns, the best results are highlighted in bold.

| Model | Accuracy | F-1 score | AUC | Average FLOPs | FPS | Parameters |
|---|---|---|---|---|---|---|
| SqueezeNet [36] | 91.60% | 0.927 | 0.898 | 1.2G | 37.86 | 0.8M |
| DenseNet121[37] | 94.85% | 0.949 | 0.940 | 4.54G | 9.05 | 7.1M |
| MobileNetv2[38] | 91.96% | 0.919 | 0.916 | 0.54G | 25.06 | 2.4M |
| MobileNetv3[39] | 93.70% | 0.926 | 0.948 | 0.21G | 20.80 | 1.5M |
| GhostNet [40] | 94.02% | 0.937 | 0.944 | 0.25G | 15.32 | 3.9M |
| MOCCA [16] | 88.79% | 0.866 | 0.894 | – | – | – |
| PatchCore [17] | 90.57% | 0.891 | 0.924 | – | – | – |
| ClassAD-ResNet | **96.28%** | **0.928** | **0.952** | 5.02G | 26.73 | 61.7M |
| ResNet18[35] | 89.94% | 0.890 | 0.901 | 3.03G | 45.81 | 11M |
| ResNet34[35] | 95.04% | 0.911 | 0.948 | 6.10G | 27.97 | 21.3M |
| ResNet50[35] | 96.26% | 0.923 | 0.952 | 6.84G | 20.16 | 23.4M |
| ClassAD-ResNet | 96.28% | 0.928 | 0.952 | 5.02G | 26.73 | 61.7M |
| VGGNet11[41] | 91.90% | 0.894 | 0.904 | 15.70G | 8.17 | 88M |
| VGGNet13[41] | 94.12% | 0.920 | 0.939 | 18.33G | 7.26 | 103.1M |
| VGGNet16[41] | 95.79% | 0.931 | 0.947 | 24.56G | 4.98 | 138M |
| ClassAD-VGGNet | 95.81% | 0.923 | 0.944 | 20.93G | 6.08 | 329.1M |

calculation of FLOPs was based on the 400 × 200 inputs. Our comparisons included two main parts: (1) the first part includes the state-of-the-art DCNN networks (SqueezeNet [36], DenseNet [37], MobileNetv2 [38], MobileNetv3 [39], GhostNet [40]) and the state-of-the-art anomaly detection methods (MOCCA [16], PatchCore [17]); (2) the second part used different backbones to prove the effectiveness of ClassAD, and the backbones included ResNet [35] and VGGNet [41]. All the methods were trained and tested by the same implementation as listed in Section 5.2, and they used the same training and testing dataset.

The effectiveness performances are demonstrated in the first four columns in Table 2. (1) As shown in the first part of Table 2, ClassAD outperforms the state-of-the-art lightweight models on average accuracy, AUC and F-1 score. Compared with state-of-the-art anomaly detection methods, such as MOCCA and PatchCore, ClassAD also performs well. One-class classification methods have a wide application range, but it is difficult for these methods to address the situation in which the anomaly features are similar to normal features. (2) In the second part, ClassAD can achieve similar or even better performance than the original networks with less computation. For example, ClassAD-ResNet performs better than ResNet18, ResNet34 and ResNet50 on accuracy and F-1 score. Similar improvement also occurs on ClassAD-VGGNet. This is because ClassAD properly uses different-capacity branches to process different-complexity images. Further analyses is provided in Section 6.1.

As for efficiency, (1) compared with the lightweight models, such as MobileNetv3 and GhostNet, the ClassAD-ResNet method shows a larger FPS than MobileNetv3 and GhostNet due to the simple framework of residual blocks. (2) Furthermore, compared with the original backbone, ClassAD has fewer average FLOPs because ClassAD utilizes a complexity rating module and some images are sent into lighter branches. As for the inference speed, our method also performs better than the original DCNN, i.e., ClassAD-ResNet outputs 6.57 more images than ResNet50 per second, and ClassAD-VGGNet outputs 1.10 more images than VGGNet16 per second. All these results demonstrate that due to the complexity-rating mechanism, ClassAD can obtain a high speed for classification.

For the model complexity, as shown in the last column of Table 2, the number of parameters of ClassAD is larger than that of the other ResNets. The number of parameters only influences the memory cost on the GPUs. ClassAD has a larger memory cost but obtains a more desirable balance between efficiency and effectiveness.

The qualitative results of ClassAD on the UVM-Anomaly dataset are shown in Fig. 6. (1) In the aspect of anomaly detection, we can see that ClassAD can classify anomalies well. (2) In the aspect of complexity class, we can see that different subimages are classified into diverse complexity classes, which is beneficial to the whole framework for enhancing inference speed and saving computational costs.

## 6. Discussions

In this section, we first discuss the feature complexity problem, which will demonstrate the superiority of ClassAD. Then, we discuss the effectiveness of the decomposition feature enhancement approach. Finally, we discuss the multi-instance loss.

### 6.1. Feature complexity

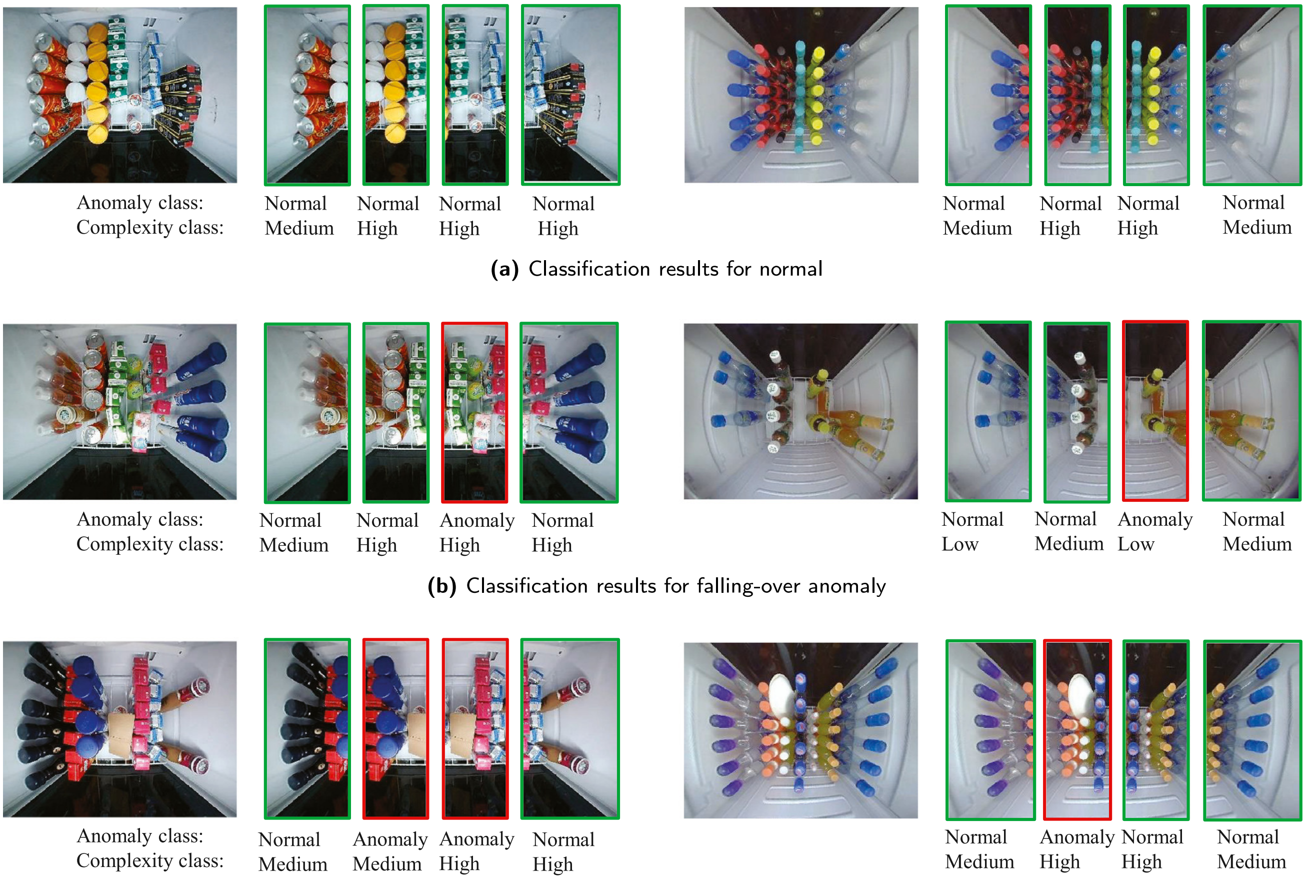
In this subsection, we discuss the performances of different-capacity models (ResNet18, ResNet34, and ResNet50) working on different feature complexity ranges. It will highlight the feature complexity imbalance problem and demonstrate the superiority of ClassAD. Fig. 7 shows the different accuracy change curves in different feature complexity ranges. In Figs. 7(a), 7(b) and 7(c), we show the performances in the low-complexity, medium-complexitylexity and high-complexity ranges.

(1) For the low-complexity feature range, we find that the training curves of the three models all level off at approximately 60$k$ iterations. As shown in Fig. 7(a), at 100$k$ iterations, ResNet18 achieves 92.55%, which is lower than the 93.45% accuracy of ResNet50; at 140$k$ iterations, ResNet18 increases to 93.84%, which is higher than the 91.05% accuracy of ResNet50. This shows that in the low-complexity feature range, shallow networks (such as ResNet18) can tie or even outperform deep networks (such as ResNet50).

(2) For the medium-complexity feature range, the performance of ResNet18 is generally worse than that of the other two models. As shown in Fig. 7(b), ResNet50 generally outperforms ResNet34 and ResNet18. However, at 140$k$ iterations, ResNet34 achieves 90.12%, and ResNet50 achieves 90.10%. This shows that in the medium-complexity feature range, ResNet34 can achieve a comparable performance with ResNet50.

(3) For the high-complexity feature range, as shown in Fig. 7(c), ResNet50 generally outperforms ResNet34 and ResNet18 in terms of accuracy. High-complexity features are more difficult for DCNNs to learn and represent. High-complexity features need more computational cost and representative parameters.

In conclusion, the experimental results show that different-capacity networks can perform well in different feature complexity ranges. This shows that a single network has difficulty

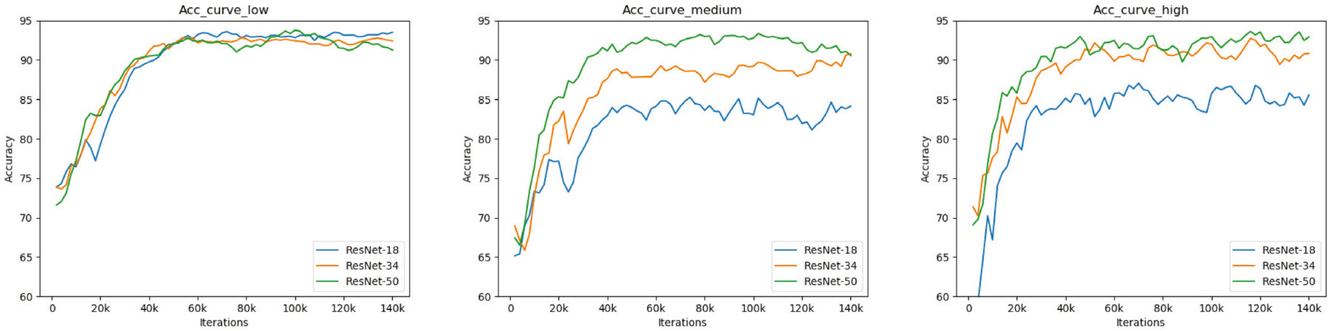


**(a)** Classification results for normal

**(b)** Classification results for falling-over anomaly

**(c)** Classification results for occulusion anomaly

**Fig. 6.** Some qualitative results on the UVM-Anomaly dataset. The anomaly class determines whether the subimage is anomalous; the complexity class is the classified complexity class for each subimage.



**(a)** Accuracy curve when training on the low-complexity range

**(b)** Accuracy curve when training on the meidum-complexity range

**(c)** Accuracy curve when training on the high-complexity range

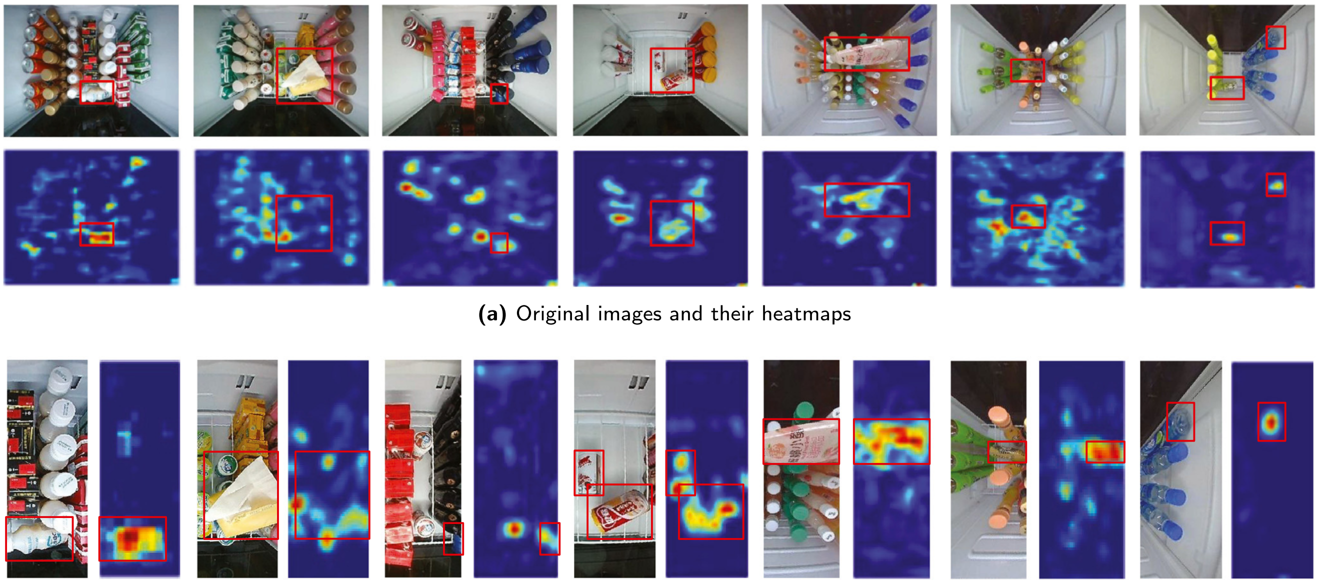
**Fig. 7.** The accuracy curves on the data with different complexity ranges of three classifiers: ResNet18, ResNet34, and ResNet50.

balancing effectiveness and computational resources due to the feature complexity imbalance problem. Therefore, we designed ClassAD to combine shallow and deep models in one framework, which will avoid underfitting and reduce computational costs. Table 2 has proven that ClassAD also obtains an improvement in classification performance.

### 6.2. Discussion on the decomposition feature enhancement approach

In this subsection, we discuss the effectiveness of the decomposition feature enhancement approach. We conducted the comparison experiment on different numbers of decomposition regions $N$ and some other data augmentation approaches, including CutMix [45] and average segmentation. Average segmentation means that we separate the images on average, e.g., a 640 × 480 image into four 160 × 480 subimages. All the experiments were conducted on ResNet50, and we observed the best results in $20k$. The experimental results are shown in Table 3.

As shown in Table 3, the performances of $N > 0$ (the 2nd, 3rd, and 4th) rows are better than the performance of $N = 0$ in all aspects. Compared with other data augmentation approaches, such as cutmix [45] and average segmentation, decomposition feature enhancement also outperforms other approaches. We infer that CutMix is not suitable for situations when anomaly features are



**(a)** Original images and their heatmaps

**(b)** Subimages and their heatmaps

**Fig. 8.** Illustration of several contrasting feature maps of ResNet50 with $N = 0$ and $N = 4$. (a) includes original images and their output heatmaps; (b) includes the corresponding subimages and their output heatmaps. To facilitate comparison, we mark anomaly areas with red boxes.

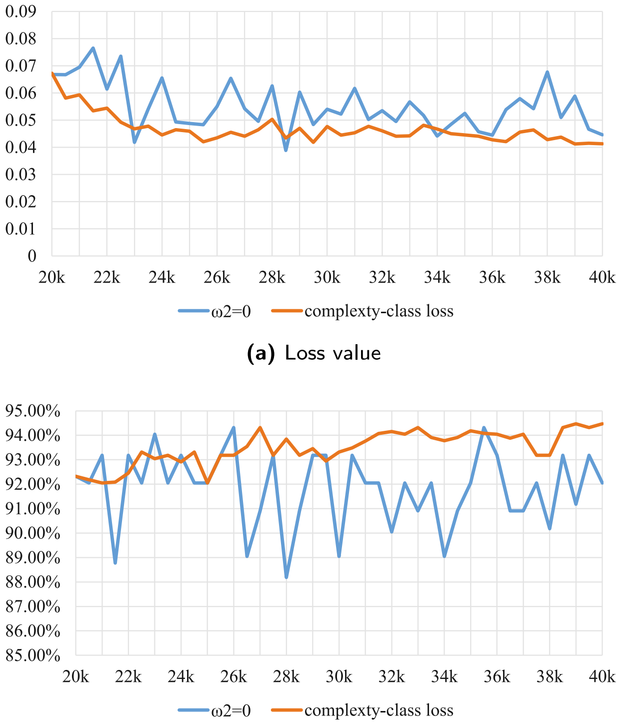
**Table 3**
Comparison of different numbers of decomposition regions $N$ and some other data augmentation approaches. $N = 0$ means that we use no decomposition feature enhancement approach. The best results are highlighted in bold.

| Method | Number | AUC | FLOPs |
|---|---|---|---|
| Decomposition feature enhancement | $N = 0$ | 0.896 | 22.15G |
| | $N = 2$ | 0.928 | 11.58G |
| | $N = 3$ | 0.923 | 8.03G |
| | $N = 4$ | **0.935** | **6.84G** |
| CutMix [45] | – | 0.740 | – |
| Average segmentation | $N = 4$ | 0.904 | – |

similar to normal features, and average segmentation damages the integrity of SKU features. The experimental results demonstrate that decomposition feature enhancement has a desirable effect on anomaly detection in UVMs.

For AUC, $N = 4$ achieves 0.935, which is better than 0.928 of $N = 2$ and 0.923 of $N = 3$; in addition, $N = 4$ also achieves a comparable performance with $N = 2$ for accuracy and F-1 score. AUC is the area under the receiver operating characteristic (ROC) curve, and a high AUC value means that more anomalies are recognized, which is important for practical use. On the other hand, the FLOPs of $N = 4$ is much smaller than those in other situations. This is because small inputs bring only a small computational cost. As a result, we choose $N = 4$.

Additionally, we show some results in Fig. 8 to demonstrate that decomposition feature enhancement is beneficial to feature extraction. In Fig. 8 we can see the unpredictability and irregularity of anomaly features in Fig. 8(a) anomaly features have irregular and scattered heat-map outputs. However, in Fig. 8(b), anomaly features show clear and nearly intact heat-map outputs. This proves that large intraclass variance brought by feature unpredictability increases feature representation difficulty, whereas decomposition feature enhancement can tackle this problem without additional computational cost.



**(a)** Loss value

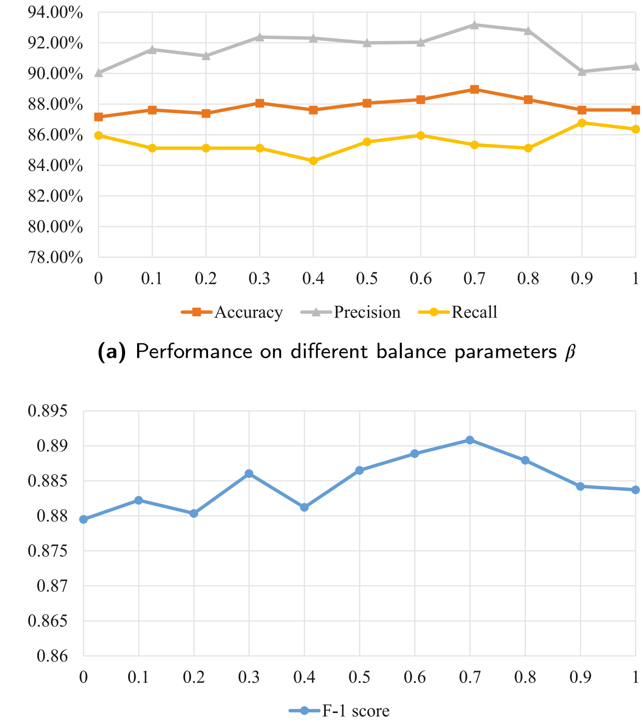
**(b)** Accuracy

**Fig. 9.** Comparison of ClassAD-ResNet with or without complexity-class loss. The x-coordinate is the training iterations.

### 6.3. Discussion on the loss functions

#### 6.3.1. Effect of the complexity-class loss

In this subsection, we discuss the effectiveness of the loss functions and the parameters with ClassAD-ResNet. First, we show the effect of complexity-class loss by simply removing its



**(a)** Performance on different balance parameters $\beta$

**(b)** F-1 score on different balance parameter $\beta$

**Fig. 10.** Comparison of ClassAD-ResNet with different values of balance parameter $\beta$.

**Table 4**
Comparison of different loss functions for anomaly detection on UVM-Anomaly dataset. The best results are highlighted in bold.

| Loss function | Accuracy | F-1 score | AUC |
|---|---|---|---|
| BCE loss | 87.61% | 0.880 | 0.860 |
| Dice loss [46] | 86.71% | 0.865 | 0.854 |
| Focal loss [47] | 88.56% | **0.887** | 0.862 |
| Multi-instance loss | **88.96%** | 0.884 | **0.866** |

second part, which means we set $\omega_2 = 0$. Then, complexity-class loss turns into a simple cross-entropy loss. Fig. 9 shows the changing curve of the loss value and accuracy over iterations. When $\omega_2 = 0$, which means without complexity-class loss, the accuracy curve oscillates and does not converge easily. It shows that without complexity-class loss, the complexity rating module only makes classification without enlarging the gap, but complexity-class loss could make this process smoother.

### 6.3.2. The multi-instance loss

We selected three loss functions for comparison: BCE loss, dice loss [46], and focal loss [47]. Dice loss and focal loss have been proposed to address the class imbalance problem. We conducted all the contrast trials on ResNet50. As shown in Table 4, multi-instance loss outperforms other loss functions in terms of accuracy and AUC. For the F-1 score, multi-instance loss achieves 0.884, which is just 0.003 lower than focal loss. Compared with the fundamental BCE loss, multi-instance loss obtains a huge improvement. This is because multi-instance loss strengthens the weight of anomaly samples to tackle the class imbalance problem.

### 6.3.3. The balance parameter $\beta$

We also discuss the balance parameter $\beta$ in the multi-instance loss. We set the values of $\beta = 0, 0.1, \ldots, 1.0$ to conduct the ablation experiment. The experimental results are shown in Fig. 10. As shown in Fig. 10(a), we can see that the accuracy and precision values of the model on $\beta = 0.7$ are higher than those of the other models. In addition, the recall value of the model on $\beta = 0.7$ also achieves a relatively high level. The reason why the high $\beta$ value brings a high recall value is that the high $\beta$ value strengthens the weight of anomaly samples. As shown in Fig. 10(b), we find that the F-1 score value of the model on $\beta = 0.7$ is higher than the others. In addition, when $\beta = 0$, the multi-instance loss divides no extra attention on the anomaly, and $\beta = 0.7$ performs better than $\beta = 0$. This proves that multi-instance loss is beneficial to imbalanced datasets. Finally, we chose the balance parameter $\beta = 0.7$ in the multi-instance loss considering the overall performances.

## 7. Conclusion and future work

In this paper, we propose a new anomaly detection framework for UVMs named complexity-classification anomaly detection (ClassAD). Our method is composed of a complexity rating module and classification module. It uses a decomposition feature enhancement approach at the beginning. We also propose complexity-class loss and multi-instance loss for ClassAD. In addition, we propose the UVM-Anomaly dataset. Experiments show that ClassAD can enhance the performances of CNNs as well as accelerate inference speed. The discussions show that the decomposition feature enhancement approach is beneficial to model learning and acceleration. In addition, we have also proved that multi-instance loss can address the class imbalance problem.

At present, our method only deals with anomaly detection in a single scene. In our future work, we will investigate the diversity in other scenes.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

Data will be made available on request.

## References

[1] Y. Hao, Y. Fu, Y.-G. Jiang, Take goods from shelves: A dataset for class-incremental object detection, in: Proceedings of the 2019 on International Conference on Multimedia Retrieval, ICMR '19, 2019, pp. 271–278.

[2] H. Zhang, D. Li, Y. Ji, H. Zhou, W. Wu, K. Liu, Towards new retail: A benchmark dataset for smart unmanned vending machines, IEEE Trans. Ind. Inform. (2019) 1.

[3] C. Liu, Z. Da, Y. Liang, Y. Xue, G. Zhao, X. Qian, Product recognition for unmanned vending machines, IEEE Transactions on Neural Networks and Learning Systems (2022) 1–14, http://dx.doi.org/10.1109/TNNLS.2022.3184075.

[4] L. Zhao, J. Yao, H. Du, J. Zhao, R. Zhang, A unified object detection framework for intelligent retail container commodities, in: 2019 IEEE International Conference on Image Processing, ICIP, 2019, pp. 3891–3895.

[5] P. Ekanayake, Z. Deng, C. Yang, X. Hong, J. Yang, Naïve approach for bounding box annotation and object detection towards smart retail systems, in: G. Wang, J. Feng, M.Z.A. Bhuiyan, R. Lu (Eds.), Security, Privacy, and Anonymity in Computation, Communication, and Storage, Springer International Publishing, 2019, pp. 218–227.

[6] H. Zhang, D. Li, Y. Ji, H. Zhou, W. Wu, Deep learning-based beverage recognition for unmanned vending machines: An empirical study, in: 2019 IEEE 17th International Conference on Industrial Informatics, Vol. 1, INDIN, 2019, pp. 1464–1467.

[7] L. Liu, B. Zhou, Z. Zou, S. Yeh, L. Zheng, A smart unstaffed retail shop based on artificial intelligence and IoT, in: 2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks, CAMAD, 2018, pp. 1–4.

[8] T. Fernando, H. Gammulle, S. Denman, S. Sridharan, C. Fookes, Deep learning for medical anomaly detection – a survey, ACM Comput. Surv. 54 (7) (2021).

[9] X. Zhang, J. Mu, X. Zhang, H. Liu, L. Zong, Y. Li, Deep anomaly detection with self-supervised learning and adversarial training, Pattern Recognit. 121 (2022) 108234.

[10] Y. Ishii, E. Saneyoshi, M. Sendoda, R. Kondo, Anomaly identification in a liquid-coffee vending machine using electrical current waveforms, in: 2019 IEEE 2nd International Conference on Information and Computer Technologies, ICICT, 2019, pp. 98–101.

[11] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, Y. Fu, Image super-resolution using very deep residual channel attention networks, in: Computer Vision – ECCV 2018, Springer International Publishing, 2018, pp. 294–310.

[12] C. Dong, C.C. Loy, K. He, X. Tang, Image super-resolution using deep convolutional networks, IEEE Trans. Pattern Anal. Mach. Intell. 38 (2) (2016) 295–307.

[13] C. Ye, X. Li, S. Lai, Y. Wang, X. Qian, Scale adaption-guided human face detection, Knowl.-Based Syst. 253 (2022) 109499, http://dx.doi.org/10.1016/j.knosys.2022.109499, URL https://www.sciencedirect.com/science/article/pii/S0950705122007511.

[14] Y. Xue, S. Liu, Y. Li, P. Wang, X. Qian, A new weakly supervised strategy for surgical tool detection, Knowl.-Based Syst. 239 (2022) 107860, http://dx.doi.org/10.1016/j.knosys.2021.107860, URL https://www.sciencedirect.com/science/article/pii/S095070512101039X.

[15] N. Li, F. Chang, C. Liu, Spatial-temporal cascade autoencoder for video anomaly detection in crowded scenes, IEEE Trans. Multimed. 23 (2021) 203–215.

[16] F.V. Massoli, F. Falchi, A. Kantarci, Å. Akti, H.K. Ekenel, G. Amato, MOCCA: Multilayer one-class classification for anomaly detection, IEEE Trans. Neural Netw. Learn. Syst. 33 (6) (2022) 2313–2323, http://dx.doi.org/10.1109/TNNLS.2021.3130074.

[17] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, P. Gehler, Towards total recall in industrial anomaly detection, in: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2022, pp. 14298–14308, http://dx.doi.org/10.1109/CVPR52688.2022.01392.

[18] Y. Liu, J. Yang, M. Liu, Recognition of QR code with mobile phones, in: 2008 Chinese Control and Decision Conference, 2008, pp. 203–206.

[19] R. Want, An introduction to RFID technology, IEEE Pervasive Comput. 5 (1) (2006) 25–33.

[20] L. Zhao, J. Yao, H. Du, J. Zhao, R. Zhang, A unified object detection framework for intelligent retail container commodities, in: 2019 IEEE International Conference on Image Processing, ICIP, 2019, pp. 3891–3895.

[21] V. Nogueira, H. Oliveira, J. Augusto Silva, T. Vieira, K. Oliveira, RetailNet: A deep learning approach for people counting and hot spots detection in retail stores, in: 2019 32nd SIBGRAPI Conference on Graphics, Patterns and Images, SIBGRAPI, 2019, pp. 155–162.

[22] X. Wang, N. Lin, Y. Li, Y. Shi, J. Ruan, An integrated modeling method for collaborative vehicle routing: Facilitating the unmanned micro warehouse pattern in new retail, Expert Syst. Appl. 168 (2021) 114307.

[23] Y. Cai, L. Wen, L. Zhang, D. Du, W. Wang, P. Zhu, Rethinking object detection in retail stores, 2020, ArXiv E-Prints, arXiv:2003.08230.

[24] X.-S. Wei, Q. Cui, L. Yang, P. Wang, L. Liu, RPC: A large-scale retail product checkout dataset, 2019, ArXiv E-Prints, arXiv:1901.07249.

[25] Y. Li, Y. Xue, L. Li, X. Zhang, X. Qian, Domain adaptive box-supervised instance segmentation network for mitosis detection, IEEE Trans. Med. Imaging 41 (9) (2022) 2469–2485, http://dx.doi.org/10.1109/TMI.2022.3165518.

[26] D. Wang, J. Lin, P. Cui, Q. Jia, Z. Wang, Y. Fang, Q. Yu, J. Zhou, S. Yang, Y. Qi, A semi-supervised graph attentive network for financial fraud detection, in: 2019 IEEE International Conference on Data Mining, ICDM, 2019, pp. 598–607.

[27] A. Aldweesh, A. Derhab, A.Z. Emam, Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues, Knowl.-Based Syst. 189 (2020) 105124.

[28] R. Chalapathy, S. Chawla, Deep learning for anomaly detection: A survey, 2019, ArXiv E-Prints arXiv:1901.03407.

[29] M. Hardt, E. Price, E. Price, N. Srebro, Equality of opportunity in supervised learning, in: D.D. Lee, M. Sugiyama, U.V. Luxburg, I. Guyon, R. Garnett (Eds.), Advances in Neural Information Processing Systems,Vol. 29, 2016, pp. 3315–3323.

[30] J. Zhao, Z. Yi, S. Pan, Y. Zhao, Z. Zhao, F. Su, B. Zhuang, Unsupervised traffic anomaly detection using trajectories, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2019.

[31] P. Napoletano, F. Piccoli, R. Schettini, Anomaly detection in nanofibrous materials by CNN-based self-similarity, Sensors 18 (2018) 209.

[32] J. An, S. Cho, Variational autoencoder based anomaly detection using reconstruction probability, in: Special Lecture on IE, 2015, pp. 2:1–18.

[33] S. Akcay, A. Atapour-Abarghouei, T.P. Breckon, GANomaly: Semi-supervised anomaly detection via adversarial training, in: C.V. Jawahar, H. Li, G. Mori, K. Schindler (Eds.), Computer Vision – ACCV 2018, Springer International Publishing, 2019, pp. 622–637.

[34] Y. Yao, J. Ma, Y. Ye, KfreqGAN: Unsupervised detection of sequence anomaly with adversarial learning and frequency domain information, Knowl.-Based Syst. 236 (2022) 107757.

[35] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: The IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2016.

[36] F.N. Iandola, S. Han, M.W. Moskewicz, K. Ashraf, W.J. Dally, K. Keutzer, SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and &lt;0.5MB model size, 2016, ArXiv E-Prints arXiv:1602.07360.

[37] G. Huang, Z. Liu, L. van der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: The IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2017.

[38] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, MobileNetV2: Inverted residuals and linear bottlenecks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2018.

[39] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, Q. Le, Searching for MobileNetV3, in: 2019 IEEE/CVF International Conference on Computer Vision, ICCV, 2019, pp. 1314–1324.

[40] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, C. Xu, GhostNet: More features from cheap operations, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020.

[41] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2014, ArXiv E-Prints arXiv:1409.1556.

[42] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in Pytorch, 2017.

[43] J. Deng, W. Dong, R. Socher, L. Li, K. Li, L. Fei-Fei, ImageNet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.

[44] C. Tian, Y. Xu, Z. Li, W. Zuo, L. Fei, H. Liu, Attention-guided CNN for image denoising, Neural Netw. 124 (2020) 117–129.

[45] S. Yun, D. Han, S. Chun, S.J. Oh, Y. Yoo, J. Choe, CutMix: Regularization strategy to train strong classifiers with localizable features, in: 2019 IEEE/CVF International Conference on Computer Vision, ICCV, 2019, pp. 6022–6031, http://dx.doi.org/10.1109/ICCV.2019.00612.

[46] F. Milletari, N. Navab, S. Ahmadi, V-Net: Fully convolutional neural networks for volumetric medical image segmentation, in: 2016 Fourth International Conference on 3D Vision, 3DV, 2016, pp. 565–571.

[47] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2980–2988.