Measurement 148 (2019) 106965

Contents lists available at ScienceDirect

Measurement

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

In-line inspection solution for codes on complex backgrounds for the plastic container industry

Qiaokang Liang [a,b,c,*], Wei Zhu [a,b,c,*], Wei Sun [a,b,c,*], Zhun Yu [d], Yaonan Wang [a,c], Dan Zhang [e]

[a] College of Electrical and Information Engineering, Hunan University, Changsha 410082, China
[b] Hunan Key Laboratory of Intelligent Robot Technology in Electronic Manufacturing, Hunan University, Changsha 410082, China
[c] National Engineering Laboratory for Robot Vision Perception and Control, Hunan University, Changsha 410082, China
[d] T-Line Technology CO., LTD, Suqian 223700, China
[e] Department of Mechanical Engineering, York University, Toronto, ON M3J 1P3, Canada

ARTICLE INFO

Article history:
Received 16 June 2019
Received in revised form 31 July 2019
Accepted 16 August 2019
Available online 22 August 2019

Keywords:
Code inspection
Deep learning
ShuffleNet
Transfer learning

ABSTRACT

Machine vision technologies have been widely used for automating the product quality control, but the defect inspection for codes on complex backgrounds is still a challenging task in the plastic container industry. In this work, an efficient and accurate inspection solution based on deep learning was proposed aiming at the detection of codes on complex backgrounds for the plastic container such as beverage packages. Firstly, image processing algorithms such as the region translation method, morphological processing, and image matching technology based on SIFT (Scale Invariant Feature Transform) features were implemented to generate synthetic defective samples, which moderated the class-imbalance problem. Data augmentation strategies were used to increase the amount of training data. Secondly, the ShuffleNet V2 framework was adapted to inspect inkjet codes on complex backgrounds. Additionally, the transfer learning was used to transfer the trained model to other inspection tasks for different kinds of packages. Finally, the proposed approach was built onto an in-line code inspection apparatus for the plastic container industry, and an accuracy of 0.9988 was achieved. The in-line testing results of false detection and omission detection rates demonstrated that the proposed solution can fully meet the production requirements. To the best of our knowledge, this report describes the first time that deep learning has been applied to the industrial defect inspection for the plastic container industry.

1. Introduction

The plastic packaging technologies have been experiencing explosive growth in the beverage industry. In order to ensure the product traceability and quality control, batch number and data codes are marked on the curved plastic surfaces early in the production process. The inkjet printing process, as the most commonly used type of printer, has been widely adopted to spray a digital image of the expiration date and manufacturing date by propelling droplets of ink onto the container. The quality of characters is affected by the performance of code printer and other external factors, and Legible codes on the containers provide consumers with important information and confidence for the inside product. If containers or packages with defective codes such as missing, incorrect, or unreadable codes are not identified in time, the product quality and corporate reputation will be compromised. Therefore, the code inspection that verifies the presence, position, and formation of printed codes is a key checkpoint in the plastic container industry to ensure products meet specifications prior to release and shipment to customers.

With the development of machine learning and image processing technologies [1], the Optical Character Recognition (OCR) technology has been widely used by companies in the field of automatic code inspection on product containers, which greatly improves the accuracy and efficiency of code detection, and reduces their production costs as well as increases their profits.

The general code detection is a composite process that comprises several phases such as the preprocessing, segmentation, feature extraction, and classification. The principal purpose is to compare the recognized characters with the correct codes and judge whether they are qualified. The widely used pixel binarization method for the character segmentation is the threshold-based segmentation method, and characters are segmented by setting a fixed or dynamic threshold. These methods

* Corresponding authors at: College of Electrical and Information Engineering, Hunan University, Changsha 410082, China.
E-mail address: qiaokang@hnu.edu.cn (Q. Liang).

https://doi.org/10.1016/j.measurement.2019.106965
0263-2241/

always require high-quality images and striking contrast between characters and the background. The result of character segmentation has a significant influence on character recognition. An appropriate segmentation algorithm is critical to successfully judge the code quality.

Most of the existing code inspection solutions used in the plastic container industry are based on traditional machine vision methods. Characters of the code are segmented by image segmentation or image matching technologies, and then the code quality is judged according to the hand-craft features. Generally, these methods are capable of handling high contrast images with simple backgrounds, for example, black codes with a white background on the bottom of a can.

However, existing state-of-the-art inspection methods have difficulty solving code inspection problems such as detection tasks for codes of more sophisticated and diverse forms with complex backgrounds. Additionally, with different packaging and printing technologies, the background of codes with various fronts on the packaging is more and more complicated, which makes the inspection more challenging. In practical applications, the captured image quality may be inevitably affected by the illumination condition, movement of the conveyor belt, product temperature, etc. Therefore, traditional machine vision methods are difficult to segment characters and judge the code quality.

Convolutional Neural Networks (CNNs) have been applied to visual tasks such as text recognition, and produced encouraging results [3]. He et al. [4] designed a novel text-attentional CNN to extract deep text features and developed a text detection system by combining the improved MSER method, which achieves a high accuracy of 0.93 with 0.73 recall on the ICDAR 2013 dataset. Ma et al. [5] applied rotated proposals to improve the general object detection algorithm in order to detect arbitrary–oriented scene texts. Not only can the TextBoxes++ algorithm proposed in Ref. [6] detect multi-directional texts efficiently and accurately, but it also form a high-performance text recognition framework by combining with other text recognition modules. Universal text detection and recognition methods are widely applied to automatic number plate recognition tasks [7]. Xie et al. [7] used the angle prediction and YOLO [9] algorithm to realize the detection of license plates in any directions. Li et al. [8] designed an end-to-end method based on the Faster R-CNN [10] that was combined with the RNN (Recurrent Neural Network) and CTC (Connectionist temporal classification) methods for recognition of car license plates. Zhu et al. [11] adopted the FCN [12] to segment candidate regions, and then used the TextBoxes detector [13] to detect text-based traffic signals. During the last few years, CNNs have become more effective in obtaining excellent results for defect detection applications. Chen et al. [14] cascaded three CNNs to detect defects of fasteners on the catenary support device, in which the modified SSD [15] and YOLO were used to locate joints and fasteners respectively, and a lightweight CNN was built to recognize defects. Qiu et al. [16] proposed a segmentation algorithm based on the FCN [12] to achieve the pixel-wise defect detection. In addition, they adopted several methods to improve the segmentation efficiency.

The image dataset used in this work was collected by a practical production line, in which inkjet codes were sprayed on bottlenecks with colorful backgrounds, as shown in Fig. 1. To fulfill the requirements of the practical plastic container industry, the in-line inspection solution for code inspection should be highly efficient due to the hardware limitations of the equipment, which is controlled by an industrial computer without Graphics Processing Unit (GPU). Therefore, despite the effectivity of both traditional machine vision and CNN-based text detection and recognition methods, the problem associated with in-line inspection for codes on complex backgrounds for the plastic container industry has not yet been satisfactorily resolved. Inspired by the efficient CNNs for mobile devices, we proposed a highly efficient ShuffleNetV2-based approach that can perform defect detection algorithm for inkjet codes with complex backgrounds on individual packages.

The main novelty and contribution of this work come from the following four aspects:

1) A ShuffleNetV2-based approach that can perform in-line defect detection for inkjet codes with complex backgrounds on individual packages for the practical plastic container industry is proposed.
2) This work aims to implement a real-time defect detection algorithm for codes on complex backgrounds for the plastic container industry. To the best of our knowledge, this report for the first time describes that the deep learning has been

Fig. 1. Examples of positive (the first row) and negative (the second row) beverage packages.

applied to the code inspection application for practical industrial instrumentation, which provides great inspiration and guidance to the development of real CNN-based inspection solutions for the plastic container industry.

3) Different from existing CNN based inspection approaches, the proposed algorithm is designed for practical inspection instrumentations. The experimental results prove that the proposed solution is more accurate and significantly efficient compared to previous approaches.
4) The proposed in-line solution has been successfully implemented on the real inspection station in a beverage packaging line. The real-time inspection has been successfully carried out, and achieves outstanding inspection accuracy. The successful application of the proposed solution will be an important reference for the industrialization of deep learning and the code inspection applications for modern industrial firms.

The remainder of this paper is organized as follows. Section 2 briefly introduces the image acquisition system. Section 3 describes the proposed algorithm, including dataset processing and detection algorithm. Sections 4 and 5 deal with the details of our experiments and the obtained results. Section 6 draws some conclusions and outlines further improvements.

2. System overview

The code inspection station was placed after the capper machine and inkjet printer in a beverage packaging line. As shown in Fig. 2, the system consists of an image acquisition system, embedded industrial computer, control system, power supply system, and human-machine interface. The embedded industrial computer is the control unit for the inspection station. The system was equipped with a Gigabit Ethernet vision CCD (Baumer VLG-02C) with an image resolution of 656×490 pixels. As the containers are conveyed through the inspection station, a LED light and the CCD will be triggered. At the same time, the embedded industrial computer will launch the image capture and carried out the real-time inspection.

The schematic diagram of the illumination and optical system is shown in Fig. 3. Two LED light sources are placed on the front and back of the bottle with a certain downward angle. The camera and light sources are set to the trigger mode, which can reduce the entire power consumption of the system. The photoelectric sensor

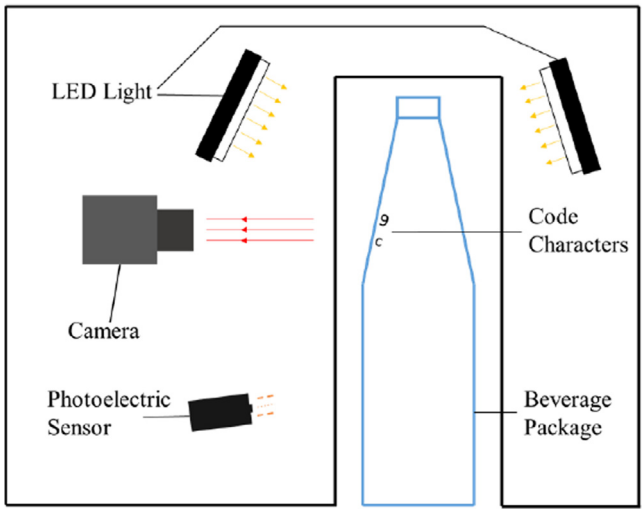

Fig. 3. The light illumination system.

is triggered and sends a signal to the camera and LED light controller when the conveyor transports the product to the particular detection area.

We collected data in a beverage packaging line from Dec. 1, 2018 to Jan. 31, 2019. The inkjet code is composed of the manufacturing date (year, month, day, hour, and minute) and the batch number of the product. Due to the high speed of the packaging line, the image acquisition system captures an image file every 10 s to collect images with different codes as many as possible.

3. Methodology

3.1. Architecture of our solution

In this work, we proposed a novel deep learning-based in-line inspection solution for codes on complex backgrounds. Fig. 4 illustrates the architecture of the solution. In the training phase, we used Python and Tensorflow [38,39] for programming. CNN was used to extract and classify deep features of code images. A variety of methods were used to deal with the class-imbalance problem. After the training is completed, we freeze the model and import it into the detection system that was written using C-Sharp. When

Fig. 2. The code inspection station installed in a beverage packaging line.

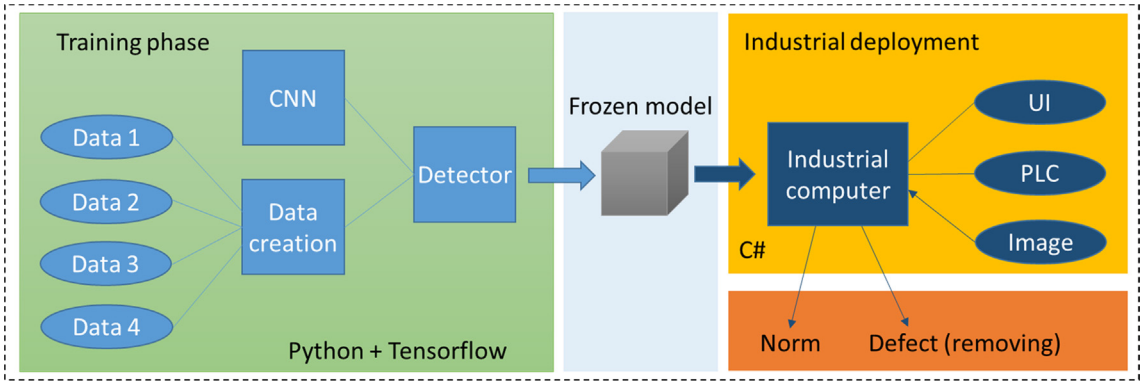

Fig. 4. Architecture of proposed solution.

the industrial camera captures an image, the system calls the frozen model to classify the image and perform the defect inspection.

3.2. Convolutional neural networks

In the past decades, deep convolutional nets have brought about breakthroughs in the field of image processing, whereas the related computer vision competitions and larger datasets have promoted, such as ImageNet dataset [17].

Since the birth of AlexNet [18] in 2012, various deep convolutional neural networks have emerged in an endless stream, constantly refreshing the accuracy of various task on the ImageNet dataset. From AlexNet to GoogLeNet [19], VGGNet [20], ResNet [21], DenseNet [22], and SE-Net [23], the image classification accuracy has been getting higher and higher. In addition, the SqueezeNet [24], MobileNet [25], and ShuffleNet [26], which have shone light on compressing models and increasing speed, have also been proposed. The ShuffleNet V2 is superior to many advanced networks in accuracy and speed metrics with the same computation condition, especially when it was applied to devices with limited computing resources. Deep learning based methods must have sufficient accuracy and speed before being applied to the industrial field. These advantages make the ShuffleNet suitable for industrial applications. In this paper, the ShuffleNet V2 network [27] was integrated into our inspection system to detect inkjet codes on complex backgrounds. The ShuffleNet V2 network is proposed to improve model acceleration and compression while keeping high accuracy. Fig. 5 shows the architecture of the ShuffleNet V2 network.

The ShuffleNet V2 network adopts a modular design, which consists of three repeated building shuffle blocks, as shown in Fig. 5. Each block is composed of a $2\times$ down sampling unit and several basic units that are residual structures [21]. The use of residual structures facilitates the efficient flow of gradients within the network. The network finally uses the global average pooling operation to reduce the dimension, avoiding a large number of parameters brought directly by the full connectional layer.

Lots of separable Depthwise Convolutions (DWConvs) were used to replace traditional 3×3 convolutions in the ShuffleNet V2 network, which helps to reduce a large amount of parameters and accelerates the forward propagation. The DWConv is composed of the depthwise convolution and 1×1 convolution. Each filter kernel of depthwise convolution deal with one channel of the input feature map, which is different from traditional convolu-

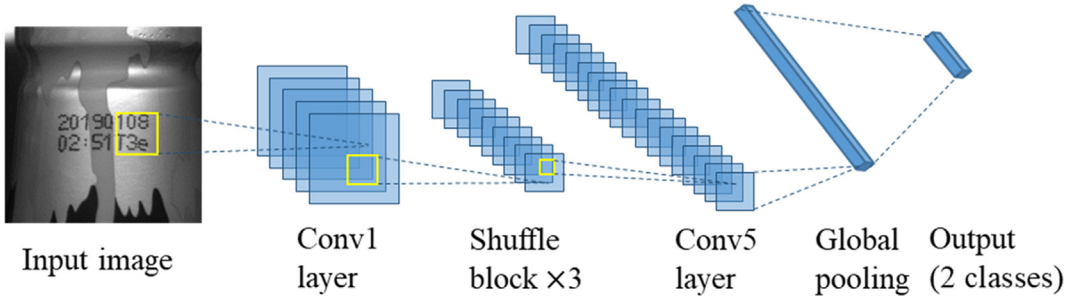

Fig. 5. Architecture of the ShuffleNet V2 network.

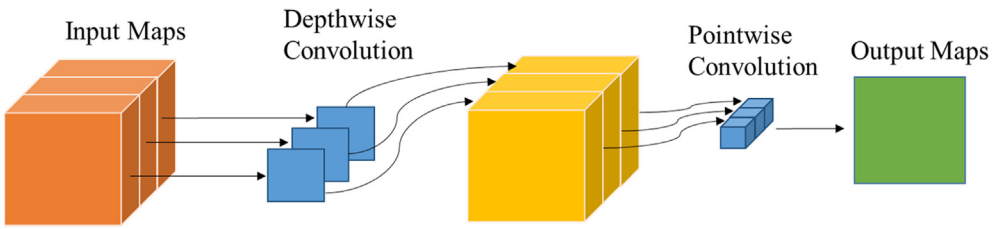

Fig. 6. Separable DWConv.

tions, whose filter kernel deal with all channels of the input feature map. The number of filter kernels of DWConvs must be equal to the number of input and output channels. The 1×1 convolution, also called pointwise convolution, is used to fuse information of all channels and change the number of feature channels. Compared to traditional convolutions, separable DWConvs have fewer parameters and improved efficiency. Fig. 6 illustrates the separable DWConv.

Each convolutional layer is followed by a Batch Normalization (BN) layer, which is used to normalize the output of the current batch [28]. This can accelerate the training process and avoid the overfitting problem. It can be determined as:

$$y = \gamma \frac{x - \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta \tag{1}$$

where x and y represent the current batch and its normalization output, respectively. μ and σ^2 represent the mean and variance of the current batch x, respectively. γ and β are the trainable parameters. ε is a small constant, which is used to avoid zero-division.

The ShuffleNet V2 network uses ReLU (Rectified Linear Unit) as the activation function. The ReLU [29] is a widely used activation function in deep learning, which can enable the network to quickly converge. The ReLU does not saturate and can deal with the gradient disappearance problem. It can be expressed by:

$$f(x) = max(0, x) \tag{2}$$

3.3. Data creation

3.3.1. Data augmentation

The data augmentation is an effective way to virtually enlarge the dataset size, which can reduce the overfitting and increase the generalization ability of the network. In this work, data adjustments are made to the original images in the training and validation datasets before being used in training. Specifically, adjustments that include scaling, rotating, cropping, and changing contrast is applied to the original images, respectively. For each negative original sample, we randomly employ data augmentation several times and creates pairs to synthesize new samples. The following steps are performed for each original image:

(1) Randomly crop a patch from the original image with different scales and aspect ratios from a fixed position. Every cropping adjustment has an equal probability.
(2) Rotate the image by a small angle with a probability of 0.2.
(3) Flip the image horizontally, with a probability of 0.2.
(4) Change the contrast of the image with a random factor that is sampled from [0.7, 1.5], with a probability of 0.25.
(5) Change the brightness of the image with a random factor that is sampled from [0.7, 1.4], with a probability of 0.25.
(6) Resize the image to a scale of 224×224.

The number of negative original samples is much less than that of positive samples. In order to balance the positive and negative samples, we used some image processing technologies that are described in the following sections to automatically synthesize some new negative samples based on the positive samples.

3.3.2. Region translation method

Due to the vibration of the printer and conveyor in the packaging line, part of the printed codes may burn into the background as a ghosting image and leave double interloped codes behind. Therefore, the region translation method was proposed to synthesize this kind of defects since the code color is darker than the background. The region translation method can be defined as the following:

$$P_g(x, y) = \min(P_o(x, y), P_s(x, y)) \tag{3}$$

$$P_s(x, y) = P_o(x + dx, y + dy) \tag{4}$$

where $P_g(x, y)$ is the synthesized image, $P_o(x, y)$ is the original image, and $P_s(x, y)$ is the image translation transform. We obtain the image translation transform by translating the original image along the x and y-axis directions with dx and dy, respectively. The region translation method compares corresponding pixel values from two images one by one, and selects the minimum value as the new pixel value for the synthesized image.

Instead of using the entire image, we select a region of interest to process, which contains the complete code and is as small as possible to reduce the impact of the region translation method on the entire image. We separately transform code images along the x and y-axis to get two kinds of defected images. Examples of synthesized images are shown in Fig. 7.

3.3.3. Morphological processing

Morphological method [37] is one of the most widely used techniques in image processing. It is mainly used to extract image components that are meaningful for expressing and depicting the shape of the region in the image, so that the subsequent recognition work can grasp the most essential shape features of the target object, such as the boundary and connected areas. In addition, techniques such as pixel refinement and trimming burrs are also often used in image pre-processing and post-processing, which are powerful complements to image enhancement technologies. In this paper, three kinds of inkjet code defects, code missing, code blur, and code adhesion are generated by dilation and opening operations of the morphological method.

Erosion and dilation are two most basic operations in morphology. Suppose A and B are collections in Z^2, where B is the structuring element, and A is the object being manipulated (image). The erosion of B to A is defined as the following:

$$A \Theta B = \{z | (B)_z \subseteq A\} \tag{5}$$

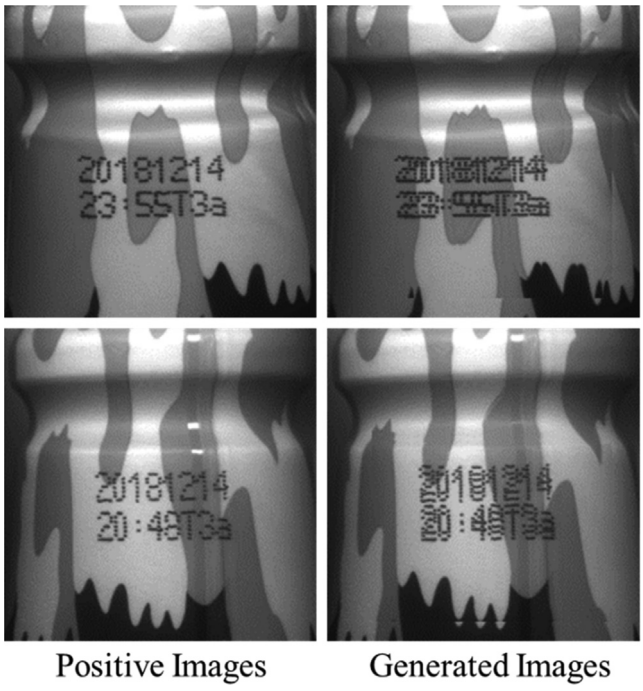

Fig. 7. Examples of generated defective images through the region translation method.

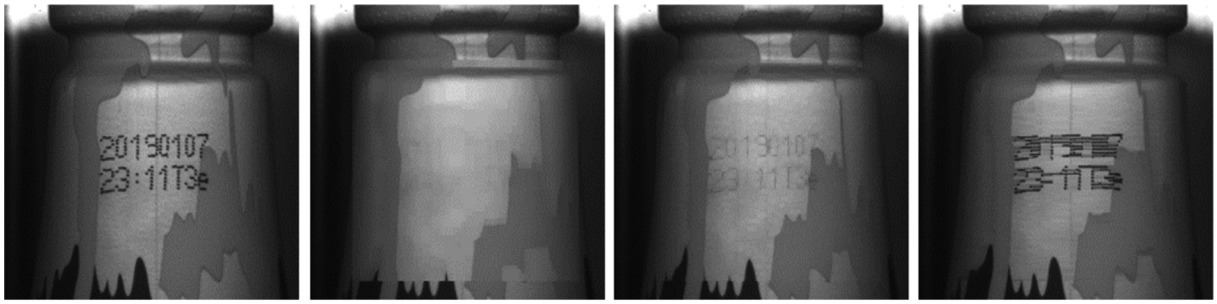

Fig. 8. Examples of generated defective images by morphological processing, where image (a) is original image, image (b), (c), and (d) are defective images with code missing, code blur, and code adhesion, respectively.

This equation indicates that the erosion of B to A is a collection of all points in Z^2, which are contained in A when B is translated by Z. In image processing, the erosion operation can shrink or refine the object. As the structuring element of erosion gradually increases, objects that are smaller than the structuring element disappear one after another. Therefore, it can be used for the filtering process. By selecting the appropriate size and shape of structuring elements, we can filter out some noise points that do not fully contain structural elements. The dilation of B to A is defined as the follows:

$$A \oplus B = \{z | \left(\hat{B}\right)_z \cap A \neq \varnothing\} \tag{6}$$

This formula indicates that the dilation of B to A is to translate B within A, and all the reference points of B and A have at least one common point during the translation. In contrast to erosion operations, dilation can increase or coarsen objects in the image, causing the boundary of the object to expand, and the specific dilation results are related to the image itself and the shape of the structural elements. The dilation operation is often used to bridge the same object that was originally broken in the image. The opening operation of B to A is defined as B eroding A, followed by B dilating A. The opening operation generally smooth the contour of the object, breaks narrow necks and eliminates fine protrusions.

The code detected in this work is black dot matrix characters and the background is relatively shallow. So the dilating operation is used to process the image. We use rectangular structuring elements for dilation. When the size of the structuring element gets large, the code is completely eliminated, and the defective images with missing codes are generated. When the size of the structural elements is small, the color of the coding becomes light, and the defects of the coding blur are generated. By using a rectangular structure element to process the opening operation, a defective image with the code adhesion can be obtained. Resulting defective images are shown in Fig. 8.

3.3.4. Image matching with SIFT

Due to the vibration of the printer and the rotation of the bottle, the printed codes will shift. When use the morphological method to process the captured image, we need to select a larger region of interest. Therefore, it is not easy to finely process the images. For example, the defect with partly missing code cannot be generated if a large region of interest is used for processing. If a small region of interest is used, the code with a large position shift will not be processed.

According to the production requirements, the ratio of the minimum missing length that should be detected in the vertical direction is 1/6. To synthesize these defective images, we need to finely process the images.

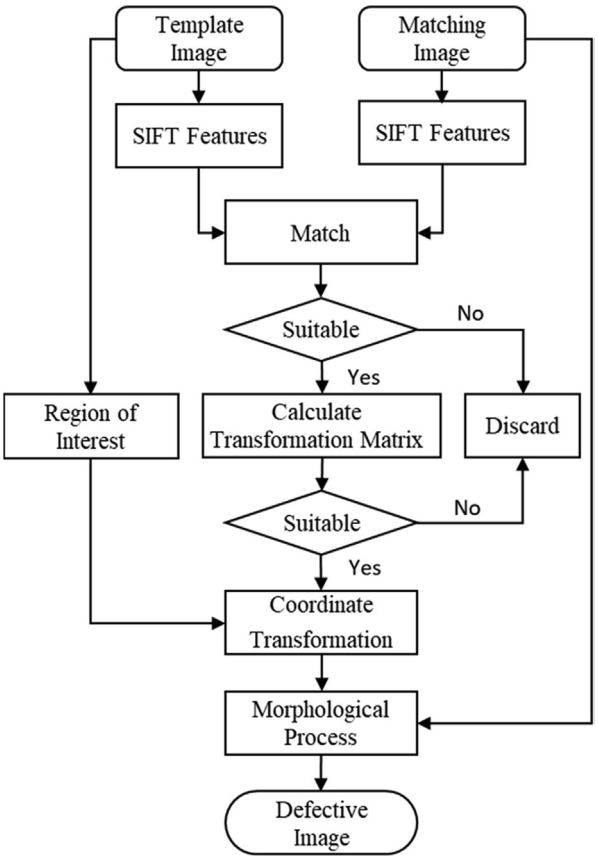

Fig. 9. The flow chart of data processing based on the SIFT feature matching.

To generate a defect at a specific location, the location must be accurately calculated. We used the image matching technique based on the SIFT (Scale-invariant Feature Transform) [30] features to calculate the location, and then use the above morphological methods to process the region of interest. The SIFT is a feature descriptor used in the field of image processing. SIFT features are local features of the image, which maintain invariance to rotation, scale, and brightness variation, and maintain a certain degree of stability against changing the viewing angle, affine transformation, and noise.

Algorithms in this work are written in Python and OpenCV [37]. First, the template image and the image to be matched were read, respectively. Then the template image and the image to be matched were processed separately to obtain respective SIFT feature descriptions. The FLANN-based k-nearest neighbor algorithm was used to perform feature matching to obtain matched feature

Fig. 10. Results of data processing based on the SIFT feature matching. Images (a) and (b) are template image and matching image, respectively. The lines in the image (a) and (b) indicates matched feature points. The white and black rectangle in the image (a) represent the effective region and the region of interest, respectively. Image (c) is the affine transformed image, and image (d) is the generated defective image.

points. The matched feature points were filtered according to its coordinates, and those in the effective region were selected. If the quantity of the feature points in the effective region is insufficient, the image will be discarded and the next image will be processed. The affine transformation matrix was calculated according to the matched feature points in the effective region, and the matched image was affine transformed to observe whether the transformed image meets the requirements. If the matched image does not meet the requirements, it will be discarded. The region of interest is selected on the template image, and its coordinates were transformed using the affine transformation matrix to obtain the region of interest in the matched image. Finally, the region of interest of the matched image was subjected to morphological processing to obtain a corresponding defective image. The processing flow chart is shown in Fig. 9.

Image processing results based on the SIFT feature matching are shown in Fig. 10.

3.4. Industrial deployment

In addition to conducting experiments on existing datasets, we also integrated the inspection algorithms proposed in this work into the in-line detection system. The architecture of the in-line inspection system is shown in Fig. 11. The industrial computer transports the image captured by the industrial camera to the inspection algorithm. Then the image and result are showed on the screen. The touching screen is also used to deal with operator's inputs. The PLC system is used to control the production removing device and inspect the production line. In order to adapt to the device drivers of the industrial camera provided by the manufacturer, the inspection software was programmed by C-Sharp. We implemented the deep learning-based algorithm proposed in this work with the TensorFlowSharp [40].

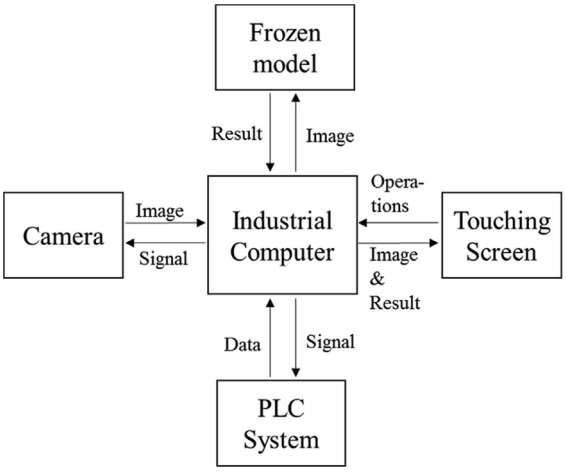

Fig. 11. The architecture of the in-line inspection system.

Fig. 12 illustrates the detection process. The resolution of images, captured by the industrial camera, is 656×490, while the scale of the input of detection algorithm is 224×224. Therefore, a region from the original image is cropped with a scale of 336×336, which is resized to 224×224 in the following steps. In the detection phase, we only implement the forward propagation of deep learning. The ShuffleNet V2 network is used to extract features and classification. The bottle will be removed from the production line if its image is classified as a defective image.

4. Experiments

4.1. Implementation details

When we used the image matching method based on SIFT features, the ratio of the matched image that satisfied the requirement was actually only about 1/6 of all the positive images, since the features that can be used to matching exist only in a small region. In addition, even if the feature matching is successful, the calculated affine transformation matrix is not necessarily perfect. After processing the data with the proposed methods, we got a sufficiently large dataset, as summarized in Table 1.

It is notable that the date of code in training and validation sets are different. The reason why the data set was split in this way is that we try to ensure no intersection between the training and validation sets. This is also consistent with the actual situation since the date of code on beverage packages produced every day is different. Such division method makes our validation results more authentic and credible.

The prepared dataset was converted to TFRecord Format of Tensorflow. We developed the algorithm on ubuntu16.04 with a single GTX 1080TI graphics card.

The stochastic gradient descent (SGD) [31] algorithm was used to optimize the network parameters. The configuration of the training parameters is listed in Table 2.

The linear decay method was used to adjust the learning rate in the training process, which is defined as

$$ms = \frac{N}{Bs} \times E \tag{7}$$

$$s = \min(gs, ms) \tag{8}$$

$$lr' = (lr - elr) \times \left(1 - \frac{s}{ms}\right) + elr \tag{9}$$

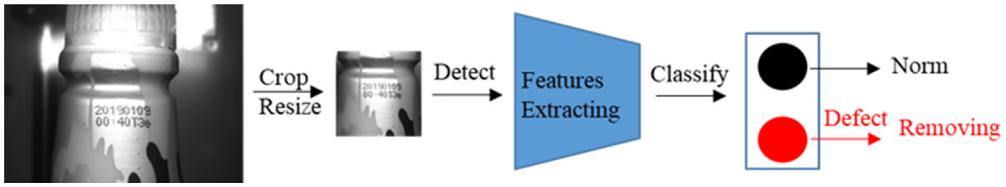

Fig. 12. Inspection process of the in-line inspection system.

where *ms*, *N*, *s* and *gs* represent max step, the number of training images, the step used to calculate learning rate, and the current global step, respectively.

4.2. Experimental results

There is a scale factor in the ShuffleNet V2 network used to control the complexity of the model. For comparison, we made ablation studies on the performance of the proposed algorithm with different complexity of the ShuffleNet V2 network. Most of the negative samples in the dataset were artificially produced. In order to better evaluate the performance of the proposed solution, we calculated the accuracy of the algorithm on the entire validation dataset and the error rate on original images. In addition to the necessary clipping and scaling, we did not perform other data augmentation operations, but instead used the trained model to directly predict the original images. The experimental results are shown in Table 3.

We compared the ShuffleNet V2 network with different complexities of 0.33×, 0.5×, and 1.0×. According to the results shown in Table 3, the more complex the model is, the more parameters and the slower the model becomes. For the accuracy on the whole validation set, the accuracy becomes higher as the complexity of the model increases.

The highest accuracy rate of 0.9988 was obtained with the complexity of 1.0×. The accuracy is defined as

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{10}$$

where *TP*, *TN*, *FP* and *FN* are defined in Table 4.

The Error Rate (ER) on original positive images is called the False Detection rate (FDr), which is defined as

$$FDr = \frac{FN}{TP + TN + FP + FN} \tag{11}$$

The ER on original negative images is called the Omission Detection rate (ODr), which can be defined as

$$ODr = \frac{FP}{TN + FP} \tag{12}$$

The ODr is 0.0000 for the three models, which indicates that our algorithm is able to detect all defective images. Among the three models, the FDr of ShuffleNet V2 network with the complexity of 1.0 is lowest, which is 0.0012. Reliability is of the utmost importance in industrial applications. Considering the lowest FDr and ODr, we chose ShuffleNet V2 network with the complexity of 1.0× as the final inspection model.

We compared our proposed algorithms with some existing code detection methods. For each method, we list the objects that it detects, main methods, accuracy, and possible applications, as shown in Table 5.

As shown in Table 5, though these existing methods have high accuracy, their applications are limited and can only be used for simple background or high contrast images. It's also noted that they are incapable of processing images with complex backgrounds. Traditional machine vision methods can also handle complex background images by designing sophisticated illumination methods, pre-processing technologies, etc. Additionally, they cannot learn many features simultaneously, therefore only target images with fewer content changes can be processed. Sometimes a change in the image brightness will require additional manual adjustment of the parameters, which requires frequent post-maintenance and increases business costs.

The proposed model can learn enough features to handle various types of image changes and implement code defect inspection. Specifically, images with changes of brightness, contrast, or character size can be appropriately inspected by the proposed model. Another benefit of the model is that the required subsequent main-

Table 1
Dataset details.

	Training set	Validation set
Date of code	12.15, 12.17, 12.26, 01.09	12.13, 12.14, 01.07, 01.08, 01.10
Number of original positive samples	11,657	10,212
Number of original negative samples	198	168
Total number of positive samples	23,314	20,430
Total number of negative samples	32,016	25,195

Table 2
The configuration of training parameters.

Method	Parameters	Value
SGD	Batch size (Bs)	128
	Initial learning rate (lr)	0.0625
	End learning rate (elr)	10^{-7}
	Weight decay	0.00005
	Momentum	0.9

Table 3
Comparison of ShuffleNet V2 network with different complexity.

Model	ShuffleNet V2		
Scale factor	0.33×	0.5×	1.0×
Accuracy on the whole dataset	0.9967	0.9983	0.9988
Error Rate on the original positive images	0.0038	0.0021	0.0012
Error Rate on the original negative images	0.0000	0.0000	0.0000
Parameters	0.14 M	0.35 M	1.27 M
Training speed (step/second)	14.7	12.2	8.2

Table 4
The definitions of TP, TN, FP, and FN.

	Actual Class		
Predicted Class		defect-free	defective
	defect-free	True Positive (TP)	False Positive (FP)
	defective	False Negative (FN)	True Negative (TN)

Table 5
The performance comparison of different methods.

Method	Inspection object	description	Accuracy	Application
Pedersen et al. [2] (2016)	Printed text	Local thresh-based binarization & projection histogram & character recognition	0.9857	Printed text & high contrast image
Feng et al. [33] (2018)	Laser code on the cap of beer bottle	Feature points-based template matching & Machine learning	0.99997	Laser code & simple background image
Sun et al. [34] (2018)	Dairy production date code	Gray value-based segmentation & Improved template matching	0.97	High contrast image & simple background image
Feng et al. [35] (2018)	Printed character	SIFT features-based image matching & binarization & pixel value difference algorithm	–	Printed text & high contrast image
Qian et al. [36] (2018)	Bottle cap printed code	Improved template matching algorithm based on Matlab image integral	0.8883	High contrast image
Ours	Inkjet code on beverage package	Morphological processing & image matching & deep CNN	0.9988	Simple or complex background image

Fig. 13. Examples of synthesized defective images.

Fig. 14. The positive (a) and negative (b) image samples. Two images are nearly the same, and their characters are both supposed to be '201901082242T3e'. However, the second line of characters in Fig. 14(b) is incomplete (almost half codes is missing in the vertical direction), which is the difference between the two images.

tenance is simple due to the strong generalization ability and robustness of deep learning.

4.3. Visualization

4.3.1. Synthesized defective images

In the training and validation datasets, most of defective images are synthesized using the proposed algorithms. This is mainly because that products with defective codes in the production line do not occur frequently. Actually, the case of missing parts of the code is the most difficult to be inspected in all kinds of defects. Therefore, the amount of defective images have been extremely augmented to ensure the high performance of the inspection model. Fig. 13 shows some examples of synthesized defective images.

4.3.2. Deep features of ShuffleNet V2

It is well known that the success of CNN in the field of image processing depends on its powerful feature extraction capabilities. Researchers have been trying to improve its feature extraction capability by deepening and widening the network. Displaying feature maps during training helps us understand the feature extraction process of deep learning and improve the algorithm. Fig. 14 shows positive and negative images, and Fig. 15 illustrates their comparison of feature maps.

From the Conv1 to Conv5 layer, the network is getting deeper and deeper, and the extracted features become more and more abstract. Comparing the feature maps of positive and negative images, one can know that the feature maps of the two are very similar in the lower layers. The more convolutional layers are, the greater the difference in feature mapping, which enables the CNN to classify.

4.4. In-line testing results

The proposed framework is implemented on an industrial computer (eBox-3622 series, NODKA) with a 3.20 GHz Intel Core

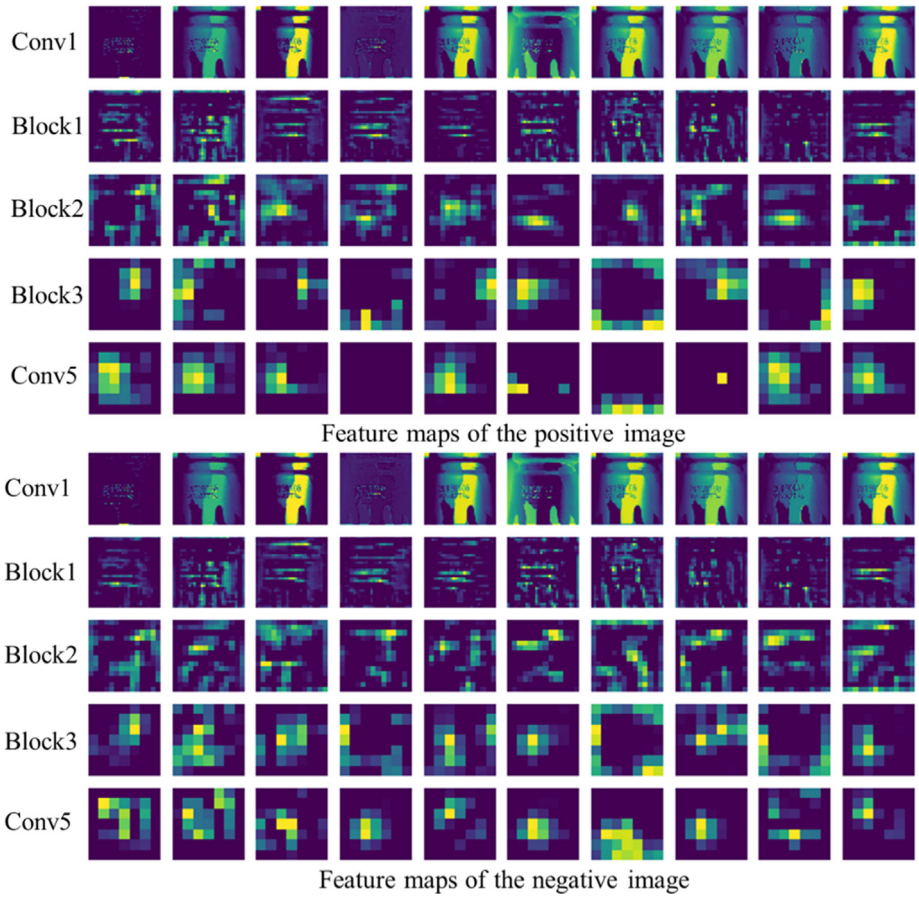

Fig. 15. Output feature maps of the positive and negative images. For the sake of observation, we scale all feature maps to the same size. Conv1: the first convolutional layer. Blocks 1–3: blocks of the ShuffleNet V2. Conv5: the final convolutional layer.

Fig. 16. The user interface of the inspection system.

i5-3470 CPU, 4G RAM, and the operating system is 64-bit Windows 7. Fig. 16 shows the user interface of the detection software, in which the ROI represents the cropped image. The detection results are displayed in real time on the right side of the interface.

A large dataset containing more than 100,000 images was created using all the images we collected. After training, the model is used for on-site inspection. Fig. 2 demonstrates our code inspection system that was working on a practical production line.

According to the results of the inspection system that are obtained in the production line, the detection speed is about 25 FPS (Frame Per Second), which is much faster than that of the conveyor belt, and all the defective types can be detected. The false detection rate of the defective products is around 0.08%. The defects that can be detected on the site include missing code, missing part of the code, code repetition, and blurred code, etc. In the vertical direction, the inspection system can detect a minimum missing length ratio of 1/6, which fully meets the production requirements.

5. Transfer learning

Deep learning practitioners can always obtain satisfactory performance with large-scale datasets. However, sometimes such large-scale datasets, especially defective images, are always inaccessible in practical applications. In this work, a large number of positive samples are easy to collect, but there are few negative samples, which leads to very serious data imbalances. Therefore, it takes a lot of time to manually create plenty of negative samples using the data processing methods that were described in Section 3. Additionally, reusing knowledge from the source domain to the target task is usually an ideal scenario. The transfer learning method is a popular machine learning method, in which a model developed for a task is reused as a starting point for a model on a second task.

In this work, we collected a total of 22,000 original images, and a dataset with 100,000 images was created using data preprocessing methods. The transfer learning [32] techniques were used to transfer the trained models to other code inspection tasks for different kinds of bottle packages. Specifically, the trained model was used to inspect the coding defects of the two other types of bottles, as shown in Fig. 17.

Unlike the previous scenario, we only randomly collected 2084 images in five days, and 175 of them were defective images. We used half of them as a training set and the others as a validation set. In the training phase, we used the model that was trained on the old dataset as the pre-trained model, and then use the new dataset to fine-tune the model. We set the initial learning rate to 0.00625 and trained for 60 epochs. In the end, an inspection accuracy of 0.9959 was obtained on the validation set, with an FDr of 0.0019 and an ODr of 0.0000. More importantly, the entire training process took only 5 min.

Fig. 17. Examples of two other types of bottle with different sizes.

Similarly, we also tested the model in a practical production line. The results showed that various defects can be successfully detected with a false detection rate of 0.1%, and the minimum ratio of the missing segment that can be detected is 1/6. Through the transfer learning, we only need to collect a small amount of data to get a satisfying model, which greatly shortens the development period of new inspection systems.

6. Conclusion

This work proposed a novel in-line inspection solution for code on complex backgrounds for plastic containers. We introduced some image processing methods to deal with data imbalance, including the data augmentation, region translation method, morphological processing, and SIFT features-based image matching. The ShuffleNet V2 network was employed to perform an accurate defect detection with an accuracy of 0.9988. Compared to existing methods, the proposed algorithm can deal with images on complex backgrounds. In addition, we integrated the proposed algorithm into a practical industrial inspection system. In-line testing results demonstrated that our solution obtained excellent performances, with lower false detection rate and omission detection rate. The proposed solution can automatically identify containers or packages with missing, incorrect, or unreadable codes to ensure only properly coded items reach customers. Furthermore, the use of transfer learning required fewer images to train a new detection model and greatly reduces the development costs of new inspection systems for different types of beverage packages.

We hope the proposed solution could inspire future works of practical inspection for modern industrial firms. Future research will focus on improving light-weight architectures and applying them to inspect defects of other practical industrial instrumentations.

Declaration of Competing Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC 61673163), the Chang-Zhu-Tan National Indigenous Innovation Demonstration Zone Project (2017XK2102), and the Hunan Key Laboratory of Intelligent Robot Technology in Electronic Manufacturing (IRT2018003).

References

[1] Kang Gui, Junfeng Ge, Lin Ye, Lizhen Huang, The piezoelectric road status sensor using the frequency scanning method and machine-learning algorithms, Sens. Actuat., A 287 (2019) 8–20.
[2] J.B. Pedersen, K. Nasrollahi, T.B. Moeslund, Quality inspection of printed texts, in: 2016 International Conference on Systems, Signals and Image Processing (IWSSIP), IEEE, 2016, pp. 1–4.
[3] Q. Ye, D. Doermann, Text detection and recognition in imagery: a survey, IEEE Trans. Pattern Anal. Mach. Intell. 37 (7) (2015) 1480–1500.
[4] T. He, W. Huang, Y. Qiao, J. Yao, Text-attentional convolutional neural network for scene text detection, IEEE Trans. Image Process. 25 (6) (2016) 2529–2541.
[5] J. Ma et al., Arbitrary-oriented scene text detection via rotation proposals, IEEE Trans. Multimedia 20 (11) (2018) 3111–3122.
[6] M. Liao, B. Shi, X. Bai, TextBoxes++: a single-shot oriented scene text detector, IEEE Trans. Image Process. 27 (8) (2018) 3676–3690.
[7] L. Xie, T. Ahmad, L. Jin, Y. Liu, S. Zhang, A new CNN-based method for multi-directional car license plate detection, IEEE Trans. Intell. Transp. Syst. 19 (2) (2018) 507–517.
[8] H. Li, P. Wang, C. Shen, Toward end-to-end car license plate detection and recognition with deep neural networks, IEEE Trans. Intell. Transp. Syst. 20 (3) (2019) 1126–1136.
[9] Joseph Redmon et al., You only look once: Unified, real-time object detection, Proceedings of the IEEE conference on computer vision and pattern recognition, 2016.

[10] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (6) (2017) 1137–1149.
[11] Y. Zhu, M. Liao, M. Yang, W. Liu, Cascaded segmentation-detection networks for text-based traffic sign detection, IEEE Trans. Intell. Transp. Syst. 19 (1) (2018) 209–219.
[12] Jonathan Long, Evan Shelhamer, Trevor Darrell, Fully convolutional networks for semantic segmentation, Proceedings of the IEEE conference on computer vision and pattern recognition, 2015.
[13] M. Liao, B. Shi, X. Bai, X. Wang, W. Liu, TextBoxes: A fast text detector with a single deep neural network, in: Proc. 31st AAAI Conf. Artif. Intell. (AAAI), 2017, pp. 4161–4167.
[14] J. Chen, Z. Liu, H. Wang, A. Núñez, Z. Han, Automatic defect detection of fasteners on the catenary support device using deep convolutional neural network, IEEE Trans. Instrum. Meas. 67 (2) (Feb. 2018) 257–269.
[15] Liu Wei, et al. "Ssd: Single shot multibox detector." European conference on computer vision. Springer, Cham, 2016.
[16] L. Qiu, X. Wu, Z. Yu, A high-efficiency fully convolutional networks for pixel-wise surface defect detection, IEEE Access 7 (2019) 15884–15893.
[17] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, Li Fei-Fei, ImageNet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, 2009, pp. 248–255.
[18] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.
[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9.
[20] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
[21] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[22] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700–4708.
[23] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 7132–7141.
[24] Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. arXiv preprint arXiv:1602.07360.
[25] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T. & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.
[26] X. Zhang, X. Zhou, M. Lin, J. Sun, Shufflenet: An extremely efficient convolutional neural network for mobile devices, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6848–6856.
[27] N. Ma, X. Zhang, H.T. Zheng, J. Sun, Shufflenet v2: Practical guidelines for efficient cnn architecture design, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 116–131.
[28] Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.
[29] Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th international conference on machine learning (ICML-10) (pp. 807-814).
[30] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vision 60 (2) (2004) 91–110.
[31] Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In Proceedings of COMPSTAT'2010 (pp. 177-186). Physica-Verlag HD.
[32] Huh, M., Agrawal, P., & Efros, A. A. (2016). What makes ImageNet good for transfer learning? arXiv preprint arXiv:1608.08614.
[33] W. Feng, C. Fang and F. Sun, "Fast defect detection algorithm based on date code," in Application Research of Computers, vol. 36, no. 7, online publication: http://www.arocmag.com/article/02-2019-07-058.html, 2019.
[34] X. Sun, J. Liu and G. Gao, "Study on visual code-based defect detection technology for production date of dairy packaging," In Food & Machinery, http://kns.cnki.net/kcms/detail/43.1183.TS.20180917.1511.026.html, 9 September 2018.
[35] Q. Feng, L. Wu and X. Wang, "Design of automatic printing character defects detection system based on machine vision," in Journal of Nanchang University (Engineering &Technology), vol. 40, no. 4, pp. 385-389 and 396, December 2018.
[36] X. Qian, J. Zhou, S. Tian, S. Li, Improved template matching algorithm based on machine vision for detecting printed code, J. Mech. Electr. Eng. 35 (4) (2018) 442–446.
[37] Rafael C.Gonzalez, Richard E.Woods, Q. Ruan and Y. Ruan, "Digital Image Processing, Third Edition," China edition, in Publishing House of Electronics Industry, Beijing, pp. 404-411, May 2017.
[38] Website: https://docs.opencv.org/3.3.0/dc/dc3/tutorial_py_matcher.html.
[39] Website: https://github.com/TropComplique/shufflenet-v2-tensorflow.
[40] Website: https://github.com/migueldeicaza/TensorFlowSharp.