Abstract—The problem of defects in glass bottles is a significant issue in glass bottle manufacturing. There are various types of defects that can occur, including cracks, scratches, and blisters. Detecting these defects is crucial for ensuring the quality of glass bottle production. The inspection system must be able to accurately detect and automatically determine that the defects in a bottle affect its appearance and functionality. Defective bottles must be identified and removed from the production line to maintain product quality. This paper proposes a glass bottle defect classification using a Convolutional Neural Network with Long Short-Term Memory (CNNLSTM) and an instant base classification. CNNLSTM is used for feature extraction to create a representation of the class data. The instant base classification predicts anomalies based on the similarity of representations of class data. The convolutional layer of the CNNLSTM method incorporates a transfer learning algorithm, using pre-trained models such as ResNet50, AlexNet, MobileNetV3, and VGG16. In this experiment, the results were compared with ResNet50, AlexNet, MobileNetV3, VGG16, ADA, Image threshold, and Edge detection methods. The experimental results demonstrate the effectiveness of the proposed method, achieving high classification accuracies of 77% on the body dataset, 95% on the neck dataset, and an impressive 98% on the rotating dataset.

Keywords—Convolution neural network; glass bottle; defect detection; long short-term memory; inspection machine

I. INTRODUCTION

Defect detection plays a vital role in glass bottle production, safeguarding product quality, consumer safety, brand reputation, and value. By investing in robust flaw detection systems and processes, manufacturers can maintain impeccable standards, safe, and visually appealing glass bottles to their customers [1,2]. Common types of defects that can occur in glass bottles include: 1) Stones are foreign stone grains embedded in the glass, degrading the quality of the bottle. 2) Tears are deformed breaks or fractures on the surface of the bottle. 3) Blisters are raised or swollen areas caused by uneven cooling during manufacturing, affecting both strength and appearance. 4) Cracks are breaks in the surface that compromise the structural integrity of bottle.

The inspection machine uses various technologies to detect defects, including cameras, lasers, and sensors. The inspection machine uses cameras to capture images of the products, and the software analyzes these images to detect defects. When defects are detected, the software sends a signal to the control system to remove the faulty product from the production line.

As for the problem with traditional inspection machines, they cannot inspect defects in complex areas and require the use of imaging techniques to visualize the defects. Because these machines utilize image processing techniques and require various parameter settings to be adjusted by users for defect detection. It results in the inability to completely remove defective bottles from the production line, undermining confidence in the manufacturing process. The detection method on inspection machine for detecting the defects must be accurate, precise, capable of recognizing defect well, and fast in the learning process of defect patterns.

This research proposes a new method for detecting defects in glass bottles using a deep neural network for extracting deep features and instance-based classification. The method uses CNN combined with LSTM to recognize distinctive features of defects and extract those features. The training process is designed to create a set of CNNLSTM models with fewer training iterations, helping to reduce the training time. The instance-based classification is used to classify the defect in the images. It reduces parameter settings by users for defect detection, resulting in a reduction in user workload.

The rest of this paper is organized as follows: Section II presents a background study of glass bottle defect detection techniques. Section III explains the glass bottle defect dataset and describes the proposed glass bottle defect classification method. Section IV explains the evaluation methods, the experimental setup, and the experimental outcomes; and finally, Section V suggests directions for future works and concludes the paper.

II. RELATED WORKS

The existing defect detection techniques mainly focused on the defect in glass bottles. The related methods are as follows. Latina et al. [3] presented defect detection method for detecting defects in glass bottles for the purpose of reusing them and highlights the limitations of the manual inspection in micro, small, and medium enterprises (MSMEs). The research introduced a cost-effective deep learning-based method using the SSD MobileNetV2 model to detect various defects in glass bottles. The method used transfer learning and data augmentation techniques to achieve high accuracy, with up to
98.07% overall system accuracy. Gong et al. [4] presented a machine vision system designed for the automatic online inspection of defects in transparent labels on curved glass bottles. The system used an area-array camera and a custom-made blue dome illumination device to capture high-quality still images by minimizing reflection. To address the challenge of distorted curved geometry, a deformable template matching method was employed for precise defect location. The method also included an adaptive threshold selection strategy that effectively detected small scratches by using global and local threshold values along with a Gaussian fitting algorithm. Additionally, techniques such as skeleton extraction and distance transformation were applied to detect the complete edge contour of Chinese characters with a special font, considering the golden edge printing error. Field tests demonstrated an impressive detection accuracy of 99.5% at a speed of 60 bottles per minute, covering over 60,000 bottles.

Vitis et al. [5] proposes an algorithm that achieves high detection accuracy while significantly reducing processing time. By using adaptive thresholding and analyzing luminous intensity variations, the algorithm effectively detects blob and airline defects while mitigating the impact of tube curvature, rotation, and vibration. Comparative evaluations demonstrate an 86% reduction in processing time, a 268% increase in throughput, and improved detection accuracy compared to existing methods. The algorithm also incorporates Region of Interest reduction techniques and a tuning procedure for parameter adjustment during production batch changes. The performance of the algorithm was assessed in a real environment, and it successfully identified misclassified tubes, suggesting its practical applicability. Zhang et al. [6] suggested a machine learning-based acoustic defect detection (LearningADD) system to replace manual inspection. The system used an improved Hilbert-Huang transform (HHT) to extract features from acoustic signals and a Shuffled Frog Leaping Algorithm (SFLA) to select features. Five deployment strategies were compared and optimized to improve real-time performance. The LearningADD system was validated using data from a real-life beverage factory. The F-measure of the system reached 98.48%. The proposed deployment strategies were verified using experiments on private cloud platforms. The Distributed Heavy Edge deployment strategy outperformed other strategies, with a defect detection time of less than 2.061 seconds for 99% of bottles. Zhou et al. [7] proposed a surface defect detection framework, with a proposal-tuned salient region detection algorithm for detecting and locating surface defects in glass bottles. The algorithm effectively detects small scratches by analyzing luminous intensity variations, while mitigating the impact of tube curvature, rotation, and vibration. Comparative evaluations demonstrate an 86% reduction in processing time, a 268% increase in throughput, and improved detection accuracy compared to existing methods. The algorithm also incorporates Region of Interest reduction techniques and a tuning procedure for parameter adjustment during production batch changes. The performance of the algorithm was assessed in a real environment, and it successfully identified misclassified tubes, suggesting its practical applicability.

III. METHODOLOGY

A. Glass Bottle Dataset

There are two main categories of glass bottle inspection machines: straight glass bottle inspection machines and rotary glass bottle inspection machines. Fig. 1 shows an example of an inspection machine for glass bottles. The straight glass bottle inspection machine is an automated system that utilizes advanced technologies, such as computer vision and image processing, to detect and identify defects in glass bottles. It captures images of the bottles from different angles and analyzes them in real-time to ensure product quality and prevent issues during production and transportation. The straight glass bottle inspection machine is depicted in Fig. 1(a). On the other hand, the rotating glass bottle inspection machine is a specialized automated system designed specifically for inspecting glass bottles using a rotating mechanism. By rotating the bottles, it enables a comprehensive examination of the entire surface, ensuring a high level of accuracy in defect detection. The rotating glass bottle inspection machine is shown in Fig. 1(b).

The bottle inspection machine utilizes a vision camera-based inspection method for fully automated visual inspection of glass bottles. The process is as follows. Set the bottles in the designated area on the conveyor belt of the bottle inspection machine. The bottles are properly aligned and spaced to ensure accurate and consistent imaging. Provide proper lighting conditions for capturing clear and well-illuminated bottle images. Set up the camera parameters such as focus, exposure, and white balance to optimize image quality and clarity. Initiate the image capture process using the bottle inspection machine to trigger the camera to capture images of the bottles as they pass through the inspection area. The machine utilizes two cameras to capture images of the bottle: one camera is focused on the neck, while the other is focused on the body of
the bottle. Fig. 2 illustrates the composition of the side wall of glass bottles; (a) displays images from the neck dataset, (b) provides an example of the body dataset, and (c) showcases sample bottle images from the rotating dataset. The body dataset for glass bottles comprises 571 images, with 368 images showing defects and 203 images without defects. Similarly, the neck dataset consists of 570 images, including 250 images with defects and 320 images without defects. As for the rotating dataset, it comprises 120 images, with 70 images displaying defects and 50 images representing undamaged bottles. Each image in the body and neck datasets is standardized to a size of 900x800 pixels (width x height), ensuring consistency across the dataset. On the other hand, the images in the rotating dataset are resized to a dimension of 1024x800 pixels (width x height). The captured images are transmitted to the detection system for analysis. If any defects, such as stones, tears, or blisters, are identified on the side wall of a bottle, the detection system generates a signal to reject the faulty bottle. This process ensures that only bottles without defects continue downstream in the production line. Fig. 3(a) is shown tear defect, (b) is a blister, and (c) is stone defect on slit wall of bottle. These defects can appear throughout the bottle.

**Fig. 1.** Inspection machine (a) Straight glass bottle inspection machine (b) Rotating glass bottle inspection machine.

**Fig. 2.** Glass bottle dataset (a) Neck dataset (b) Body dataset (c) Rotating bottle dataset.

**B. The Proposed Method**

This section describes the proposed defect detection approach, which is divided into three parts: 1) The network architecture of CNNLSTM for extracting deep features in images, 2) the training process for creating a set of CNNLSTM models and computing the representation of class data, 3) the classification approach that suggests using the extracted deep features from unseen bottle image and applying the distance weight method to classify defect or normal glass bottle images.

**1) Network architecture of CNNLSTM:** CNNLSTM is a convolutional neural network (CNN) combined with LSTM (Long Short-Term Memory) to extract deep features in images. CNNs are effective at extracting spatial features from images, while LSTMs are capable of capturing temporal dependencies in sequential data. The fusion of these two architectures enables CNNLSTM to extract deep features from images. The convolutional layers choose pre-trained deep-learning models available in the Keras library, including VGG16, ResNet50, MobileNetV3, and AlexNet. Using a pre-trained model can reduce the amount of time and resources needed to train a model. It already learned to recognize a variety of features and improve the accuracy of the model. Added LSTM to its convolutional layer. The detail of LSTM unit is shown in Fig. 4.

**Fig. 3.** Defect images. (a) Tear; (b) Blister; (c) Stone.

**Fig. 4.** LSTM.

LSTM comprises multiple memory cells, each consisting of three essential elements: write, read, and forget (delete). During each time step \((t)\), the forget gate unit is updated according to the following process,

\[
\begin{align*}
&f_t = \sigma_d(W_{f}\mathbf{x}_t + U_f h_{t-1} + b_f), \quad (1) \\
&i_t = \sigma_d(W_{i}\mathbf{x}_t + U_i h_{t-1} + b_i), \quad (2) \\
&o_t = \sigma_d(W_{o}\mathbf{x}_t + U_o h_{t-1} + b_o), \quad (3)
\end{align*}
\]

\[
\begin{align*}
&c_t = f_t \odot c_{t-1} + i_t \odot \sigma_e(W_{c}\mathbf{x}_t + b_c), \quad (4) \\
&h_t = o_t \odot \sigma_h(c_t), \quad (5)
\end{align*}
\]

where \(\mathbf{x}_t\) is the input data, \(f_t\) is the forget gate, \(i_t\) is the input/update gate, \(o_t\) is the output gate, \(c_t\) is the cell state
vector, $\sigma$ is the sigmoid activation function, $\odot$ is the Hadamard product (element-wise product), and $h_t$ is the output of the LSTM unit. In the final representation layer, an LSTM, which is a Recurrent Neural Network (RNN) suitable for sequential data, receives inputs from previous convolutional layers. The LSTM produces a set of 512 outputs, which are then passed to the next layer. The outputs of the convolutional layers are subjected to Batch-Normalization (BN) before they are fed into the LSTM layer. The fully connected layers contain 1,000 hidden neurons. The activation function of all the convolutional layers (with a max pooling size of $2 \times 2$) is a Rectified Linear Unit (ReLU) [11]. Deep features are extracted from these layers. The output layer contains one output neuron with a sigmoidal activation function. The size of the input images is set as $224 \times 224 \times 3$ pixels. The CNNLSTM is trained using a Stochastic Gradient Descent (SGD) method [12]. The learning rate is set to 0.01, and the batch size is 8 images. The CNNLSTM neural network architecture is shown in Fig. 5.

Fig. 5. CNNLSTM.

2) The proposed learning process: Our training process aims to minimize the similarity distance between the extracted deep feature vectors and the centroid of the corresponding class with the same label. Simultaneously, it maximizes the similarity distance between the centroid of good bottles and the centroid of defective bottles.

The overall learning process of creating a set of CNNLSTM and representing class data is proposed in this section. The process of creating a set of CNNLSTM models and generating representations of class data is illustrated in Algorithm 1. The inputs of the learning process include a training dataset $S$, a specified number of sub-training sets $K$, a threshold value $\theta$, and the pre-trained weight $m_{k-1}$. The parameter $K$ is a parameter used to determine the number of sub-training sets and the number of CNNLSTM models, which needs to be appropriately adjusted. The value of $K$ affects the number of samples in each sub-training set used for training CNNLSTM and the diversity of CNNLSTM models. $\theta$ is a parameter used to define the initial performance of CNNLSTM in each training iteration. The pre-trained weight $m_{k-1}$ helps to reduce training time and resource requirements, improve performance, and enable the model to perform well in limited data conditions.

Algorithm 1: Pseudo code of the feature learning network method

<table>
<thead>
<tr>
<th>Input:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S$: A training bottle dataset: ${(x_1, y_1), ..., (x_N, y_N)}$</td>
</tr>
<tr>
<td>$K$: The number of the sub datasets</td>
</tr>
<tr>
<td>$\theta$: A performance threshold value</td>
</tr>
<tr>
<td>$m_{k-1}$: The pre-trained model</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E$: A set of CNNLSTM models</td>
</tr>
<tr>
<td>$C$: A set of representation of class data</td>
</tr>
</tbody>
</table>

1: Randomly split the training dataset into $K$ sub datasets: $S \rightarrow \{s_1, s_2, s_3, ..., s_K\}$.  
2: For $k = 1, 2, ..., K$ do:  
3: $MSE = 0$ and $m_k = m_{k-1}$  
4: While $MSE < \theta$ do:  
5: Train model $m_k$ model with $s_k$  
6: Extract features $F_k$ in images by $m_k$.  
7: Normalize all features in $F_k$ with an L2-norm technique.  
8: For $i \in y \in \{0,1\}$ do:  
9: Compute the representation of each class data $c_{ik}$ from $F_k$.  
10: End For  
11: Evaluate the distances from centroid $c_{ik}$ to every feature of all $F_k$ instances.  
12: Predicted a bottle is normal or abnormal ($h$) is based on the minimum distance between $c_{ik}$ and $F_{k \in E/ik}$.  
13: Compute means squared error MSE from $h$ with true label $y$.  
14: End While  
15: $m_k \in E$ and $c_{ik} \in C$.  
16: End For

To construct a set of CNNLSTM models, our method involves randomly selecting images in the $S$ training dataset and organizing them into several partitions or subsets, i.e., $S \rightarrow \{s_1, s_2, s_3, ..., s_K\}$. The purpose of creating these sub-datasets is to foster diversity in CNNLSTM models and generate representation vectors for both normal bottles and bottles with defects.

The current $s_k$ is the training bottle images for a CNNLSTM model, $m_k$. Let the set models is ensemble $E$. The convolutional layer of the current model uses pre-train weight of previous round. The output of training with five epochs is $m_k$ (Step 5).

In Step 6, deep features vectors $F_k$ are extracted from the images in $s_k$. This process involves removing the output layer from the CNNLSTM and obtaining the output features through the fully connected layers. Let the deep feature vector $f_{jk}$ be an instance extracted from an image in $F_k: f_{jk} = m_k(x_{jk})$. The feature vector consists of 1,000 components, which corresponds to the number of hidden neurons: $f_{jk} = \{v_1, v_2, ..., v_{nk}\}, f_{jk} \in F_k$.  

www.ijacsa.thesai.org
Each feature vector is normalized using an L2-norm technique (Step 7). This normalization technique is commonly used to scale and standardize vectors in machine learning and data analysis. The L2-norm, also known as the Euclidean norm or the L2-norm, is a mathematical measure of the length or magnitude of a vector. In the context of feature normalization, the L2-norm technique calculates the square root of the sum of the squares of each component in the vector, resulting in a normalized vector with a magnitude of 1. Here, we provide an explanation of the concept of L2-norm. Consider a deep feature vector as:

$$ F_k = [f_{1k}, f_{2k}, f_{3k}, \ldots, f_{nk}] $$

where

$$ |f_{jk}| = \sqrt{\sum_{l=1}^{n_k} |v_{lk}|^2} $$

where $v_{lk}$ represents an extracted feature within $f_{jk}$, and $n_k$ denotes the total number of extracted features $v$ in $f_{jk}$. L2 serves as the final output layer of CNNLSTM, producing the extracted features as the output.

The representation of the class data $c_{ik}$ is computed as the average of all feature vectors in $F_k$, which are extracted from instances of the $i$-th class sample data (Step 9). In this step, we compute the centroids for the two classes: the centroid of the good bottles (class 0) and the centroid of the bottles with defects (class 1); $y = \{0, 1\}$. These class centroids enable the recognition of patterns on the side walls of the bottles. The centroid vector $c_{ik}$ is calculated as follows:

$$ c_{ik} = \frac{1}{N_{ik}} \sum_{j \in N_{ik}} f_{jk} $$

where $c_{ik}$ constrain the average values $a : c_{ik} = \{a_1, a_2, \ldots, a_n\}$; and $N_{ik}$ is the number of all data instances of class $i$ in $s_k$. The centroids are representation of class data.

Predicted a bottle is normal class or abnormal class (h) is based on the similarity distance between $c_{ik}$ and $F_{k \in N_{ik}}$ (Step 12). The similarity distances between each class are compared using the Euclidean distance to predict the category of the bottle. Predict normal or abnormal bottles by minimizing the similarity distance value defined as

$$ h_j = \arg\min_i \|f_{jk} - c_{ik}\|_2^2, \text{ for } j = 1 : Nk, $$

where $Nk$ is the number of instances in $s_k$.

To assess the extraction and classification performance of a trained CNNLSTM model $m_k$, the Mean Squared Error (denoted as MSE) is measured. If the model error (MSE) exceeds an error threshold value ($\theta$), the deep neural model will undergo an additional training process of 5 epochs (proceed to Step 5). If the condition is met, the deep feature vectors $F_k$ and the centroid $c_k$ are recomputed. Otherwise, the training process is stopped, and the model is obtained. The training iteration $k$ is completed based on the performance condition. The threshold value $\theta$ is used to measure the performance and reduce the number of training iterations in each model generation. It serves as a criterion for selecting a set of models that require fewer training iterations compared to the training process of a normal deep neural network. At this point, the model $m_k$ is added to the ensemble $E (m_k \in E)$, and the centroid $c_k$ is included in the set $c_{ik} \in C$ (Step 15). For the next iteration, we used pre-trained CNNLSTM weights from the previous iteration $k$ to learn the current data $m_{k+1} \rightarrow m_k$. This method eliminates the need to reset the weights each time and only requires fine-tuning on the new set of bottle images. The overview of feature learning method is illustrated in Fig. 6.

3) The classifying process: This section presents a detailed discussion of the proposed similarity voting method. This classification method focuses on utilizing distinctive features of defects for bottle defect detection. The similarity distance is calculated between the representation features of normal bottles and abnormal bottles. The classification steps of our method are outlined in Algorithm 2.

Algorithm2: Pseudocode for classifying bottle defects.

Input:

- $E$: A set of CNNLSTM models
- $C$: A set of centroids of class data
- $x$: A bottle images

Output:

$$ h_i = \arg\min_{l} \sum_{k} s_{dk} $$

1: For $k = 1, 2, \ldots, K$ do:
2: Extract deep features $f_k = m_k (x)$
3: Normalize the deep feature vectors $f_k$ by $L2$ -norm
4: For $i = 1: y$ do:
5: Calculate the distance $d_{ik}$ between centroid $c_{ik}$ to features $f_k$
6: End For
7: Compute the distance weight $s_{dk}$
8: End For

The input for Algorithm 2 consists of a set of CNNLSTM models $E$, the centroid of class data $C$, and the bottle image $x$. The models in set $E$ differ from the decision boundary models of the same architecture. Extract features in bottle
image using each model in $E$ and combine them to predict the final output. Deep feature vectors are extracted from the bottle image: $f_k = m_k(x)$ (Line 2). Then, the deep feature vector $f_k$ is normalized using the L2-norm technique: $f_k = L2(f_k)$ (Step 3). The deep feature vector is normalized to ensure it resides in the same vector space as $c_k$, enabling the calculation of similarity values. The similarity distance $d_{ik}$ between feature and centroid is expressed as follows:

$$
\alpha_{ik} = \|f_k - c_k\|_2, \text{ for } k = 1: K,
$$

where $K$ is the number of ensemble model. Each centroid calculates the similarity distance with the features of the bottle image. The voting weight of each model is computed as follows:

$$
\sum_{i} \sum_{k} a_{ik} = \frac{\alpha_{ik}}{\sum_{i} \alpha_{ik}},
$$

where $s_d_{ik}$ is the voting weight of class $i$ at model $k$. The voting weight is determined by calculating the average distance between the centroid and the features of both normal and abnormal bottles (Step 6).

The final output of this method is prediction class that compute as follows:

$$
h_i = \arg\min_{\sum_{i} \sum_{k} s_d_{ik}},
$$

where $h_i$ is the class or category that the method predicts the input image belongs to based on its defect patterns and relationships from the training bottle images. The proposed classification method is illustrated in Fig. 7.

![Fig. 7. The overall process of glass bottle classification.](image)

IV. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we present a comprehensive set of experiments conducted to assess the performance of the CNNLSTM method.

A. Experimental Setup

These experiments used three glass bottle image datasets, namely the body dataset, the neck dataset, and the rotating dataset. These datasets consist of three primary types of defects: stone, tear, and blister. The body dataset of glass bottles comprises a total of 571 images. For the training set, 292 images, consisting of 161 images with defects and 131 images without defects were used to train models. The remaining 279 images constituted the testing set, with 207 images containing defects and 72 images without defects. The neck dataset encompasses 570 images of glass bottles. The training set include 288 images, with 150 images exhibiting defects and 138 images without defects. The testing set includes 282 images, with 100 images containing defects and 182 images without defects. The rotating dataset encompasses 120 images of glass bottles. The training set include 80 images, with 47 images exhibiting defects and 33 images without defects. The testing set includes 40 images, with 23 images containing defects and 17 images without defects. The distribution of each defect type in both the training and testing datasets for body, neck and rotating bottles is presented in Table I.

<table>
<thead>
<tr>
<th>Defect type</th>
<th>Body</th>
<th>Neck</th>
<th>Rotating</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Train</td>
<td>Test</td>
<td>Train</td>
</tr>
<tr>
<td>Stone</td>
<td>81</td>
<td>127</td>
<td>80</td>
</tr>
<tr>
<td>Tear</td>
<td>30</td>
<td>28</td>
<td>20</td>
</tr>
<tr>
<td>Blister</td>
<td>50</td>
<td>52</td>
<td>50</td>
</tr>
<tr>
<td>Normal</td>
<td>131</td>
<td>72</td>
<td>138</td>
</tr>
<tr>
<td>Total</td>
<td>292</td>
<td>279</td>
<td>288</td>
</tr>
</tbody>
</table>

To assess the effectiveness of our proposed method on the defect dataset, various image processing techniques, including image thresholding and edge detection, along with machine learning models such as Anomaly Detection with Autoencoder (ADA), ResNet50, AlexNet, VGG16, and MobileNetV3 were used to construct models for comparison. Image thresholding, a commonly used technique in image processing using OpenCV, involves setting pixel values to either 0 or a maximum value based on a predefined threshold [13]. Edge detection is a
Digital image processing technique used to identify and extract edges in an image, representing abrupt changes in intensity or color between adjacent pixels [14]. ResNet50 is a deep convolutional neural network architecture introduced in 2016 [15]. ResNet-50 consists of 50 layers and employs residual blocks, each containing convolutional layers and shortcut connections. It has achieved state-of-the-art performance in various image recognition tasks. AlexNet is a deep convolutional neural network architecture proposed in 2012 [16]. AlexNet comprises eight layers, including five convolutional layers and three fully connected layers. It introduced several key innovations, such as using ReLU as an activation function and applying dropout regularization. VGG16 developed by the Visual Geometry Group (VGG) at the University of Oxford in 2014, is a widely recognized deep convolutional neural network architecture primarily used for image classification [17]. It consists of 13 convolutional layers followed by 3 fully connected layers, and it has demonstrated outstanding performance on numerous computer vision tasks. MobileNetV3, introduced by Google in 2019, is specifically designed for mobile and embedded devices that have limited computational resources [18]. This architecture aims to strike a balance between model size and accuracy, making it suitable for efficient deployment on devices with constrained hardware capabilities. In addition to these models, we also employed the Anomaly Detection with Autoencoder (ADA) technique. ADA detects anomalies by evaluating the reconstruction loss, comparing it to a predefined threshold. If the reconstruction loss surpasses the threshold, the input is classified as an anomaly.

For training our method and deep learning-based models, we used a training set comprising 292 images for the body dataset and 288 images for the neck dataset. The training epochs are set at 500 for VGG16, RestNet50, MobileNetV3, and ADA. The input error threshold, \( \theta \), of the proposed method is set to 0.2, and the number of ensemble models, \( K \), is set to 3. All the experiments, including the proposed defect detection method and the comparison methods, were conducted on a personal computer equipped with an AMD Ryzen 7 5000 series processor and an NVIDIA GTX 1650 GPU, allowing for efficient computation and analysis of the results.

**B. Experimental Result**

Four evaluation metrics are adopted for the analysis: Accuracy, Recall, Precision, and F1-score. These metrics are widely used in image defect detection and provide comprehensive insights into the performance of the methods. They are defined as follows:

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}
\]

\[
\text{Recall} = \frac{TP}{TP + FN} \tag{14}
\]

\[
\text{Precision} = \frac{TP}{TP + FP} \tag{15}
\]

\[
F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{16}
\]

\( TP \) refers to the number of defect bottles (positive) that are correctly predicted as positive by the model. \( FN \) represents the number of defect bottles (positive) that are incorrectly predicted as bottle without defects (negative) by the model. \( TN \) indicates the number of the bottles without defect (negative) that are correctly predicted as negative by the model. Lastly, \( FP \) signifies the number of the bottles without defects (negative) that are mistakenly predicted as positive by the model.

Fig. 8 presents visualizations of defect classification results of CNNLSTM for the body, neck, and rotation of bottle datasets.

The classification accuracy of various methods on the defect datasets is presented in Table II. The results for the body dataset demonstrate that the standalone VGG16 method outperformed all other methods, achieving an accuracy of 80%. ResNet50 is ranked second in terms of accuracy. Among the CNNLSTM models, CNNLSTM-VGG16 achieved the highest accuracy compared to the other CNNLSTM variants. Conversely, the image threshold method exhibited the lowest accuracy on this dataset.

![Fig. 8. Examples of classification results of the CNNLSTM.](image)

**TABLE II. THE PERFORMANCE ON THE BODY IMAGES, NECK IMAGES AND ROTATING IMAGES DATASETS IN TERMS OF ACCURACY**

<table>
<thead>
<tr>
<th>Methods</th>
<th>Body dataset Accuracy (%)</th>
<th>Neck dataset Accuracy (%)</th>
<th>Rotating dataset Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNNLSTM-ResNet50</td>
<td>71</td>
<td>95</td>
<td>62</td>
</tr>
<tr>
<td>CNNLSTM-AlexNet</td>
<td>75</td>
<td>89</td>
<td>97</td>
</tr>
<tr>
<td>CNNLSTM-MobileNetV3</td>
<td>64</td>
<td>92</td>
<td>98</td>
</tr>
<tr>
<td>CNNLSTM-VGG16</td>
<td>77</td>
<td>80</td>
<td>90</td>
</tr>
<tr>
<td>ResNet50</td>
<td>79</td>
<td>79</td>
<td>60</td>
</tr>
<tr>
<td>AlexNet</td>
<td>74</td>
<td>92</td>
<td>60</td>
</tr>
<tr>
<td>VGG16</td>
<td>80</td>
<td>85</td>
<td>95</td>
</tr>
<tr>
<td>MobileNetV3</td>
<td>70</td>
<td>72</td>
<td>70</td>
</tr>
<tr>
<td>ADA</td>
<td>74</td>
<td>75</td>
<td>60</td>
</tr>
<tr>
<td>Image threshold</td>
<td>62</td>
<td>65</td>
<td>65</td>
</tr>
<tr>
<td>Edge detection</td>
<td>70</td>
<td>72</td>
<td>63</td>
</tr>
</tbody>
</table>
The neck dataset results reveal that CNNLSTM-ResNet50 attained the highest accuracy of 95%, closely followed by CNNLSTM-AlexNet, CNNLSTM-MobileNetV3, CNNLSTM-VGG16, and AlexNet. Among the standalone methods, AlexNet exhibited the highest accuracy on this dataset. Conversely, the image threshold method yielded the lowest accuracy for the neck dataset.

Upon comparing the results of the rotating dataset images, it is obvious that CNNLSTM-MobileNetV3 achieved the highest accuracy. CNNLSTM-AlexNet and the standalone VGG16 also demonstrated good performance with high accuracies. Conversely, ADA exhibited the lowest accuracy percentages on this dataset.

Overall, the results indicate that different methods perform differently on the three datasets. The standalone VGG16 generally achieved the highest accuracy on both the neck and rotating datasets.

In Table III, the performance measures, including Recall, Precision, and F1-score, are presented for the body dataset. VGG16 demonstrates the highest level of correctness, evident from its superior metrics across Recall, Precision, and F1-score. Following closely in terms of correctness, CNNLSTM-AlexNet achieves relatively high Precision and F1-score. On the other hand, Image Threshold exhibits the lowest level of correctness among the methods listed.

Table IV presents the Recall, Precision, and F1-score metrics of the proposed method and the compared methods for rotating image datasets. Among the CNNLSTM methods, CNNLSTM-MobileNetV3 and CNNLSTM-AlexNet demonstrate outstanding performance, achieving high recall, precision, and F1-score. CNNLSTM-VGG16 also performs well, with consistently high metrics. However, CNNLSTM-ResNet50 exhibits comparatively lower performance across the metrics. When considering individual models, VGG16 stands out with consistently high scores for recall, precision, and F1-score. ResNet50, AlexNet, and MobileNetV3 display varying levels of performance, with some metrics being higher or lower than others. In terms of the additional methods, Image Threshold exhibits relatively high recall but lower precision and F1-score. ADA demonstrates quite low performance across the metrics.

Table VI displays the training time and detection time of all detection methods. Among the CNNLSTM methods, CNNLSTM-VGG16 has the longest training time on all three datasets, followed by CNNLSTM-ResNet50. However, CNNLSTM-ResNet50 has the shortest detection time across all datasets, indicating faster inference speed. CNNLSTM-AlexNet and CNNLSTM-MobileNetV3 also demonstrate relatively shorter training and detection times. For the individual models, ResNet50, AlexNet, VGG16, and MobileNetV3 have consistent training and detection times across all three datasets. Among the additional methods, ADA has relatively short training and detection times, while Image

Table of Values:

<table>
<thead>
<tr>
<th>Methods</th>
<th>Recall (%)</th>
<th>Precision (%)</th>
<th>F1-score (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNNLSTM-ResNet50</td>
<td>80</td>
<td>73</td>
<td>70</td>
</tr>
<tr>
<td>CNNLSTM-AlexNet</td>
<td>82</td>
<td>74</td>
<td>74</td>
</tr>
<tr>
<td>CNNLSTM-MobileNetV3</td>
<td>71</td>
<td>66</td>
<td>63</td>
</tr>
<tr>
<td>CNNLSTM-VGG16</td>
<td>83</td>
<td>75</td>
<td>75</td>
</tr>
<tr>
<td>ResNet50</td>
<td>71</td>
<td>75</td>
<td>73</td>
</tr>
<tr>
<td>AlexNet</td>
<td>73</td>
<td>74</td>
<td>70</td>
</tr>
<tr>
<td>VGG16</td>
<td>86</td>
<td>78</td>
<td>77</td>
</tr>
<tr>
<td>MobileNetV3</td>
<td>57</td>
<td>57</td>
<td>57</td>
</tr>
<tr>
<td>ADA</td>
<td>51</td>
<td>55</td>
<td>53</td>
</tr>
<tr>
<td>Image Threshold</td>
<td>50</td>
<td>53</td>
<td>51</td>
</tr>
<tr>
<td>Edge detection</td>
<td>78</td>
<td>72</td>
<td>67</td>
</tr>
</tbody>
</table>

VGG16 outperforms the CNNLSTM method in terms of accuracy on the body image dataset. The VGG16 is advantaged by being pre-trained on a large and diverse collection of images. Its capacity to identify various features associated with bottle defects, coupled with the requirement for a comprehensive and diverse training set, contributes to its superior performance.

Table IV presents the Recall, Precision, and F1-score metrics of the proposed method and the compared methods for neck image datasets. Among the CNNLSTM methods, CNNLSTM-ResNet50 demonstrates the highest recall, precision, and F1-score, indicating its strong performance. CNNLSTM-MobileNetV3 follows closely behind with good recall, precision, and F1-score. Among the individual models, AlexNet demonstrates good performance, achieving high recall, precision, and F1-score. MobileNetV3 exhibits moderate performance with decent recall and F1-score, but its precision is relatively lower. ADA demonstrates moderate performance across all metrics. On the other hand, Image Threshold and Edge detection methods exhibit lower performance, with lower recall, precision, and F1-score.
Threshold and Edge detection methods have negligible training times but slightly longer detection times compared to the CNNLSTM and individual models.

Overall, the experimental results demonstrate the effectiveness of the proposed CNNLSTM models for defect detection on all three datasets. The proposed models achieved competitive accuracy and performed well in terms of evaluation metrics. The training and detection times varied among the models, with CNNLSTM-AlexNet demonstrating the shortest training time. These findings can guide the selection of appropriate models based on the trade-off between accuracy and computational time requirements in practical applications.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Body dataset</th>
<th>Neck dataset</th>
<th>Rotating dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Training time (second)</td>
<td>Detection time (second)</td>
<td>Training time (second)</td>
</tr>
<tr>
<td>CNNLSTM-ResNet50</td>
<td>154.29</td>
<td>0.11</td>
<td>143.55</td>
</tr>
<tr>
<td>CNNLSTM-AlexNet</td>
<td>103.59</td>
<td>0.12</td>
<td>59.22</td>
</tr>
<tr>
<td>CNNLSTM-MobileNetV3</td>
<td>141.93</td>
<td>0.12</td>
<td>80.69</td>
</tr>
<tr>
<td>CNNLSTM-VGG16</td>
<td>304.41</td>
<td>0.074</td>
<td>304.84</td>
</tr>
<tr>
<td>ResNet50</td>
<td>1,500</td>
<td>0.5</td>
<td>1,500</td>
</tr>
<tr>
<td>AlexNet</td>
<td>690</td>
<td>0.3</td>
<td>690</td>
</tr>
<tr>
<td>VGG16</td>
<td>1,800</td>
<td>0.5</td>
<td>1,800</td>
</tr>
<tr>
<td>MobileNetV3</td>
<td>780</td>
<td>0.3</td>
<td>780</td>
</tr>
<tr>
<td>ADA</td>
<td>180</td>
<td>0.2</td>
<td>180</td>
</tr>
<tr>
<td>Image Threshold</td>
<td>-</td>
<td>0.002</td>
<td>-</td>
</tr>
<tr>
<td>Edge detection</td>
<td>-</td>
<td>0.001</td>
<td>-</td>
</tr>
</tbody>
</table>

V. Conclusion

This research presents an approach for detecting defects in glass bottles using a combination of deep convolutional neural networks with long short-term memory (CNNLSTM) and instance-based classification algorithms. The CNNLSTM combines two types of deep neural networks to extract features and create a representation of class data, which is then used for defect detection. The proposed method is compared with other well-known defect detection methods. Experimental results demonstrate that the proposed method outperforms other defect detection methods in terms of detection performance. Additionally, the proposed method requires less training time, making it suitable for efficient glass bottle defect detection in production lines.

In future work, we intend to extend the application of the proposed algorithm to different types of bottles and explore additional defect types. The flexibility of the CNNLSTM allows for varying the number of models and designing customized detection layers to adapt to specific datasets. Furthermore, there is potential to extend this method to semi-supervised learning, enabling the detection of defects in other product categories. These future directions will further enhance the capabilities and applicability of our approach in defect detection and contribute to the advancement of quality control systems in various industries.

REFERENCES


