Journal of Scientific & Industrial Research
Vol. 82, February 2023, pp. 171-177
DOI: 10.56042/jsir.v82i2.70250

Hyper Parameter Optimization for Transfer Learning of ShuffleNetV2 with Edge Computing for Casting Defect Detection

Narasimha Prasad L V[1]*, Durga Bhavani Dokku[2], Sri Lakshmi Talasila[3] & Praveen Tumuluru[4]

[1]Computer Science and Engineering, Institute of Aeronautical Engineering, Hyderabad, 500 043, Telangana, India
[2]Computer Science and Engineering, CVR College of Engineering, Hyderabad, 501 510, Telangana, India
[3]Computer Science and Engineering, Prasad V. Potluri Siddhartha Institute of Technology, Vijayawada, 520 007, Andhra Pradesh, India
[4]Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, 522 302, Andhra Pradesh, India

Received 23 June 2022; revised 22 September 2022; accepted 07 October 2022

A casting defect is an expendable abnormality and the most undesirable thing in the metal casting process. In Casting Defect Detection, deep learning based on Convolution Neural Network (CNN) models has been widely used, but most of these models require a lot of processing power. This work proposes a low-power ShuffleNet V2-based Transfer Learning model for defect identification with low latency, easy upgrading, increased efficiency, and an automatic visual inspection system with edge computing. Initially, various image transformation techniques were used for data augmentation on casting datasets to test the model flexibility in diverse casting. Subsequently, a pre-trained lightweight ShuffleNetV2 model is adapted, and hyperparameters are fine-tuned to optimize the model. The work results in a lightweight, adaptive, and scalable model ideal for resource-constrained edge devices. Finally, the trained model can be used as an edge device on the NVIDIA Jetson Nano-kit to speed up detection. The measures of precision, recall, accuracy, and F1 score were utilized for model evaluation. According to the statistical measures, the model accuracy is 99.58%, precision is 100%, recall is 99%, and the F1-Score is 100 %.

Keywords: Edge Computing, Industrial Internet of Things, NVIDIA Jetson Nano-kit, ShuffleNetV2

Introduction

Advanced industrial systems demand better product performance and a higher requirement for quality control throughout the manufacturing process.[1–3] Every industry has its quality inspection section responsible for removing defective products. However, the main difficulty is that this inspection is done manually. The process takes a long time and is not always accurate due to human error, which could cost the company a significant amount of money. Casting is a process of manufacturing that entails pouring a liquid substance into a mould with a hollow hole in the chosen shape and then allowing it to solidify. Casting is commonly used to make vital components such as the cylinder head, cylinder block, bridge shell and crankshaft. Pinholes, shrinkage defects, burrs, blow holes, pouring metal defects, mould material defects, metallurgical defects, and so on are all common casting problems. These defects have a detrimental impact on the product's aesthetics, convenience of use, and performance.[4–6] Defect detection considerably reduces the negative effect of product problems.[7]

In the automotive industry, inspecting all safety-critical parts is standard procedure. Traditional computer vision algorithms require predefined features and statistics-based machine learning models.[8,9] Visual inspection of casting faults, on the other hand, is done manually based on human experience or intuition. In a range of industries, vision-based inspection technologies are frequently used to improve recognition accuracy and reduce the cost of manual inspection.[10] Image processing algorithms are employed to construct feature vectors, and subsequent machine learning-based methods are used to create inspection systems.[11] Through robust vision sensors, image processing algorithms and smartly engineered optical transmission systems, machine vision can perform many more jobs than artificial vision.

Industries are striving to decrease time-consuming and unsafe old methods in various fields. With the advent of Industry 4.0, many companies began to concentrate on automating their standard procedures. Automated technologies improved the manufacturing floor and assisted employees in completing previously

*Author for Correspondence
E-mail: lvnprasad@iare.ac.in

difficult or tedious tasks. Similarly, industries might use visual inspection systems based on images, sensors, and infrared to perform faster inspection rates, deliver higher quality solutions, and substantially reduce product inefficiencies through quantitative evaluation.

The development of deep learning technologies has accelerated recently. It has improved object detection, intelligent automation, error detection, autonomous driving, and other industry-related issues.[12] Deep learning allows industries to use non-destructive inspection procedures. Deep learning-based machine vision systems are critical in the Industrial Internet of Things (IIoT) and fully automated manufacturing processes. These methods can be applied to even the tiniest of products to ensure that the inspection process is not obstructed. Another way, it permits a product's quality to be monitored throughout the manufacturing process without risking it. Intelligent machine vision systems were developed by researchers for defective product examination based on data generated by several integrated technologies in modern manufacturing lines. In addition, Industry 4.0 has automated manufacturing procedures by providing personalized and adaptive mass production technologies.[13]

Although Deep Learning algorithms have shown efficiency for detection and classification problems, their use in specific industries is still limited.[14] An automatic casting defect detection system is necessary to address manufacturing industry needs such as low latency response, storage, and high accuracy. The casting defects could be minor in the manufacturing process. If distortion is not correctly observed, a large withdrawal of items could occur, causing severe damage to the brand's reputation and financial losses. Edge Computing (EC) is advantageous to Industry 4.0 since it provides various benefits. The idea behind edge computing is that computation should take place close to data sources. EC is a type of decentralized processing that lets data be handled directly by the device that produces it or a local server. Therefore, we believe that edge computing could have a similar significant impact on society as cloud computing has. Based on its benefits of improving service, and latency, reducing data transmission and easing cloud computing pressure, edge computing is a crucial option to overcome the bottleneck of new technologies like cloud computing, IoT, etc.

This work proposes a casting defect detection method based on EC to meet industry needs. The key contributions of the study are: First, different image transformation methods were employed as data augmentation to improve the model's adaptability; second, by fine-tuning the ShuffleNetV2's hyper-parameters with transfer learning, a lightweight, adaptive, and scalable model that can ensemble the resource-constrained edge device is created, and finally the trained model is deployed to the NVIDIA Jetson Nano-kit as an edge device to speed up detection.

Related Research

Increased automation, manufacturing flexibility, and improved quality are effective strategies for increasing profitability in any company. Traditional and visual inspection approaches, widely used for several years, have shown positive results in many industries detecting cast defects.

Image processing is widely used in defect detection techniques. The Ng *et al.*[15] used valley-emphasis method to construct a defect detection system to maximize between group variance based on the thresholding of original images. In Mery *et al*[16], a method for automatic defect detection in aluminium castings based on a two-step analysis: radioscopic image recognition and tracking was proposed. The Phase-Only Fourier Transform (POFT) is employed in work[17] to detect saliency, effectively improving weak regions. The method enables for more precise detection of the casting defect's location. On the other hand, the defect detection techniques discussed above could only find information such as the estimated location and size of the defect, but not its classification.

Many industries have recently adopted deep learning-based detection, which may be used to identify the location of casting faults and classify them. In their work, the authors developed an upgraded You Only Look Once (YOLOv3) algorithm using the anchor box initialization clustering technique.[18] They included a double-density convolutional layer structure and a model prediction scale to improve the network model for detecting casting flaws. An in-depth convolution neural network was developed[19,20] to perceive the suspicious defect area of casting through a centre-peripheral difference calculation approach based on a selected attention mechanism. Numerous state-of-the-art object analyzers are employed to localize casting defects after the feature extraction layer is detached from the object detection architecture.[21] Different feature extractors can be used to analyze each architecture, but R-CNN is the fastest and produces the best results.

Even if deep learning methods are more efficient than standard approaches, deep learning systems typically require a lot of processing resources to implement. When computational resources are limited, the system's detection performance suffers significantly. Most recent deep learning techniques are integrated with cloud computing to solve this problem. However, data queues caused by the transfer of huge volumes of data (such as videos and images) will significantly impact production effectiveness with this cloud-centric method.[22] A mix of deep learning and edge computing is employed to solve the problem. Edge Computing is an open platform that brings computational and storage abilities closer to customers or data sources by integrating data processing, storage, systems, and applications.[23] A pavement defect detection system based on YOLOv3 (you only look once, version 3) was implemented with a field-programmable gate-array-based edge computing platform.[24]

An edge computing solution has been presented to identify casting defects in the product manufacturing business[25], with low response time, power, and upgradeability. The latency and power consumption issues were better handled with the scalable and lightweight ShuffeNetV2 algorithm with NVIDIA Jetson TX2. Data improvement approaches and cross-entropy loss functions are presented to increase the model's predicting capacity. In this work, a modified CNN model was integrated with edge computing to detect defects in a real-time context, resulting in high energy efficiency, quick reaction time, and scalability.

Methodology

The methodology section presents the work flow process requirements that enable the flow process in detecting the casting defects of the manufacturing industry with the features of the DL model and EC platform. Manufacturing industries require intelligent and reliable production systems with more benefits. The proposed automated system for casting defect detection should have improved operational efficiency and quick real-time response with lower production costs. We use the ShuffleNetV2 architecture and an edge computing detection system to respond to the specific requirements above. The operational workflow process is shown in Fig. 1.

ShuffleNet V2

The ShuffleNet uses point wise group convolution and channel shuffle to reduce computation costs while preserving accuracy. ShuffleNet's core design uses a novel channel shuffle operation to facilitate information flow between feature channels. The network can be freely customized to the appropriate level of complexity. FLOPs are the most popular metric for measuring a network's efficiency in terms of computation. ShuffleNet v2 assesses the network's computational complexity directly using metrics like speed or memory access cost (FLOPs as well, which act as an indirect metric). The direct metrics are also evaluated on the target platform. However, a few studies have shown that FLOPs do not fully expose the underlying facts; networks with comparable FLOPs differ in their speeds due to several factors, including the level of parallelism and the cost of memory access, the target platform, etc. Since none of them meets the FLOP definition, they are all ignored. ShuffleNet v2 eliminates these problems by providing four rules to represent a network.

Proposed Architecture

The concepts around which the network has been formed shall provide insight into how numerous other direct measurements have been taken into consideration before analyzing the network architecture.

1. Memory access costs are low when the ratio of input to output channels is 1:1, achieved using equal channel widths.
2. Excessive group convolution raises memory access costs: Memory access costs are likely to increase if the group number is too high.
3. Reduced parallelism due to network fragmentation: Parallel computations become more challenging to execute due to network fragmentation.
4. The operations elementwise are non-negligible: Element-wise operations can lengthen memory

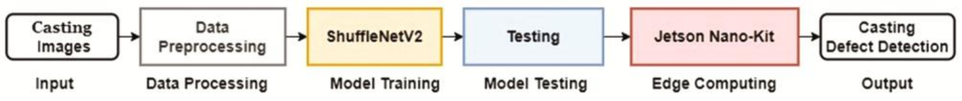


Fig. 1 — Operational workflow process of the proposed model

access times, even though they involve few FLOPs.

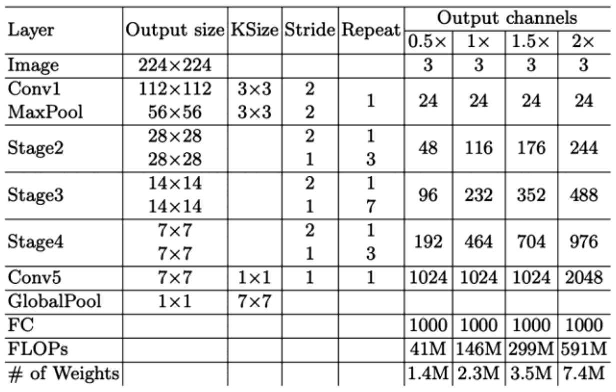
The above measures are incorporated into the architecture of ShuffleNet-V2 to increase network efficiency, as represented in Fig. 2. Two groups, one of which is left as an identity, are formed when the channel splitter splits the channels. The other branch along the three convolutions has the same number of input and output channels. Group-wise, 1 × 1 convolutions don't exist. Examples of element-wise operations constrained to a single branch are ReLU, Concat, and depth-wise convolutions.

Transfer Learning

Transfer learning allows deep learning models trained on massive datasets to perform similar tasks on new datasets. By applying transfer learning, we can use all or a portion of a model already trained for another task to complete a specific task. Moreover, the computation resources and time required for training a model can also be significantly reduced since pre-learned knowledge from other domains and tasks can be reused. A pre-trained model is the term used to describe such a deep learning model. The most well-known examples of pre-trained models are the deep learning models for computer vision based on the ImageNet dataset. Therefore, for building a model from scratch, a pre-trained model as an initial point is preferable. Due to the advantages of transfer learning for improving performance and efficiency, supervised learning is rarely used in the industrial sector.

Model-based Fine-Tuning

Fine-tuning the hyperparameters of the model outperforms the feature extraction method. In deep learning algorithms, hyperparameters are crucial because they describe the training details and directly impact the output of the model.[26,27] Fine-tuning the weights in some of the pre-trained model's layers and training the output classifier can improve the model's performance. The last network layers are typically unfrozen for tuning, while the earlier network layers are usually freezed (particularly for CNN). As a result, we can modify the parameters at the last layers and fully train the current model.



Layer	Output size	KSize	Stride	Repeat	Output channels			
					0.5×	1×	1.5×	2×
Image	224×224				3	3	3	3
Conv1 MaxPool	112×112 56×56	3×3 3×3	2 2	1	24	24	24	24
Stage2	28×28 28×28		2 1	1 3	48	116	176	244
Stage3	14×14 14×14		2 1	1 7	96	232	352	488
Stage4	7×7 7×7		2 1	1 3	192	464	704	976
Conv5	7×7	1×1	1	1	1024	1024	1024	2048
GlobalPool	1×1	7×7						
FC					1000	1000	1000	1000
FLOPs					41M	146M	299M	591M
# of Weights					1.4M	2.3M	3.5M	7.4M

Fig. 2 — ShuffleNet v2 Architecture for different levels of complexities

In this study, the FC layer of the pre-trained ShuffleNetV2 is changed to our casting defect dataset, and knowledge learned from the ImageNet dataset is transferred from one model to another. The pre-trained ShuffleNetV2 model was trained using the ImageNet dataset. Its weights and biases accurately reflect the dataset features. As a result, the pre-trained model has acquired standard features like edges and curves, which it can utilize to solve classification problems. The parameter values must be set correctly to get the best learning results from the backpropagation algorithm. The primary factors determining Deep CNN's learning performance in the backpropagation algorithm are activation function, learning rate and momentum rate. During training, the backpropagation of error computes the volume of error that a node's weights in the network are responsible for. The weight is scaled by the learning rate rather than updated with the full amount. The learning rate controls the speed at which the model learns. When weights are updated for each batch of training, the learning rate controls the amount of distributed error. The momentum rate influences smoothing the optimization process, slowing updates to continue in the previous direction that misleads the gradient descent algorithm. Generally, transfer learning in CNN can be applied in various test cases for fine-tuning the model. However, the best settings for these factors can't be chosen according to any set rules. The best parameters can only be determined through numerous trials and errors. A smaller learning rate value results in minimal error change over time, whereas a more significant learning rate value frequently results in over-fitting.

Test Case-1

The central concept is to use the weighted layers of the pre-trained ShuffleNetV2 model to extract features without changing the model's weights while training. Only the final classification block was altered; it contains two classes and is trainable.

Test Case-2

In contrast to layers of the network, which are more focused on a specific task, the earlier layers assist in general recording features. The higher-order feature representations of the underlying model should be changed to suit the current goal. We can retrain some model layers while keeping others frozen in training.

Test Case-3

The complete ShuffleNetV2 model is trainable in this experiment by considering pre-train weights as initial network weights with the casting defect dataset.

Results and Discussion

The experiment is performed on a local server, and the computer hardware configuration depicts in Table 1. The computer software configuration includes the Ubuntu operating system and the PyTorch deep learning framework. The demonstration of this work is supported by all the tests conducted in this part, and deep extracted features are utilized to assess the findings using a large dataset of defect categories.

Dataset

The dataset[28] provides a top view of the submersible pump impeller corresponding frames, respectively. The dataset contains 7340 images and is all grey-scaled images of size 300 × 300 pixels. The data augmentation is applied to all the frames. There are mainly two categories:

1. Defective
2. OK

The defective and OK images of the given dataset classes are depicted in Fig. 3. The data is divided into a train and a test folder for training a classification model. Both the train and test folder contains deffront and okfront subfolders. The dataset description of total frames with class labels is shown in Table 2.

The Table 3 illustrates the input parameters used to configure the ShuffleNetV2 model and displays the setting for the functional parameters. A single epoch presents the dataset that the neural network accesses in many batches by simultaneously performing a forward and backward pass and iteratively. The batch size measures how much data the network can fit into a batch of 32 with a learning rate of 0.001. The adaptive learning rate optimization algorithm Adam is chosen during the training and verification procedure to readily adapt to the new task without considerably erasing the learned information through the expertise of managing the verification error in the fine-tuning process. Softmax is the most suitable activation function when the output layer of the classifier considers two classes.

Table 1 — Hardware Configuration

Hardware	Product Specification
CPU	2 × Intel Xeon Gold 6226R-2.9G
GPU	NVIDIA Quadro RTX 6000, 24GB GDDR6
Motherboard	Power Edge R740/R740 XD
RAM	2 × 64 GB RDIMM
SSD	4 × 960 GB SSD SATA
Edge Device	NVIDIA Jetson Nano-kit

Evaluation Metrics

The classification metrics are used to assess the model performance. The model's overall performance can be improved using various metrics for

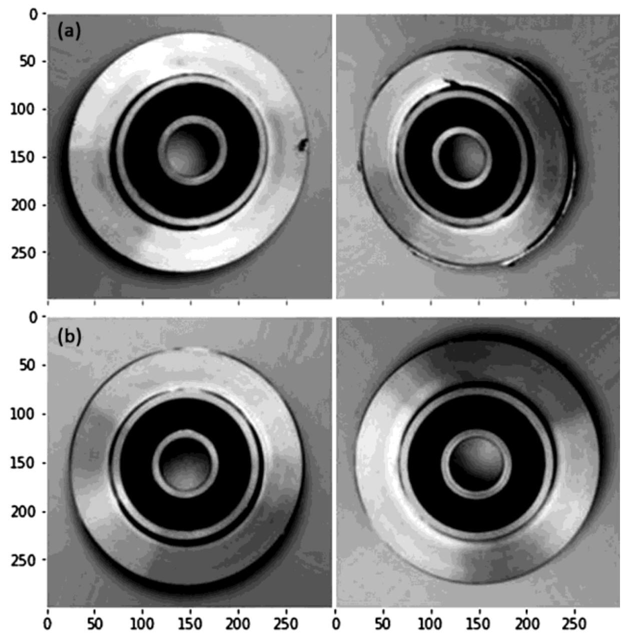


Fig. 3 — Dataset Classes of casting sample: (a) Defective and (b) OK

Table 2 — Dataset description

Classes	Train Dataset	Test Dataset	Total
OK	2875	262	3137
Defective	3758	453	4211
Total	6633	715	7348

Table 3 — Model hyper-parameters

Parameters	Value
Batch Size	32
Learning rate	0.001
Loss	Cross Entropy
Optimizer	Adam
Activation function	ReLU
Epochs	25

Table 4 — Classification Report

Test Cases	Classes	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)
Test case-1	OK	56	66	61	68.81
(Only modified classifier)	Defective	78	70	74	
Test case-2	OK	99	82	90	93.01
(50 % of the model)	Defective	90	100	95	
Test case-3	OK	99	100	99	99.58
(Full model)	Defective	100	99	100	

performance assessment. Before using the model for production on unseen data, these indicators are crucial for assessing its performance. When the model is deployed on unseen data without proper assessment of the model using various evaluation metrics and only based on accuracy, it can lead to poor predictions. True Positive (TP) is a correctly predicted sample of the defective case, and False Positive (FP) is a sample of misclassified defective case. True Negative (TN) is a correctly classified OK case, and False Negative (FN) is a sample of misclassified OK cases.

1. F1 score is a weighted average of precision and recall $\frac{2 \times \text{precision} \times \text{reacall}}{\text{precision} + \text{recall}}$
2. The precision formula is given as $\frac{TP}{FP + TP}$
3. The recall formula can be given as $\frac{TP}{FN + TP}$
4. Accuracy is $\frac{TP + TN}{TN + FP + TP + FN}$

We experimented with the pre-trained model in three test cases and applied fine-tuning process. It can be observed that when we modify only the classifier, i.e test case-1, the precision, recall, F1-score values are 56%, 66%, and 61% for OK class and 78%, 70% and 74% for Defective class. For this case, only 68.81% accuracy is achieved because the model is not trained using the new dataset, and the pre-trained model's weighted layers extracted the features. Still, they did not update the model's weights during training. In test case-2, it can retrain some layers of the model while keeping some frozen in training. In the experiment of test case-2, the first 25 layers are frozen, and the last 25 layers are trainable. The precision, recall, F1-score values are 99%, 82% and 90% for OK class and 90%, 100% and 95% for Defective class. The result shows 93.01% accuracy. The entire model is trainable in the test case-3 experiment, showing 99.58% accuracy. The classification report of the proposed model, along with the metrics, Recall, Precision, F1-Score, and accuracy, is shown in Table 4. The accuracy and loss curves of the test and training data of the ShuffleNet V2 model with an accuracy of 99.58% are shown in Fig. 4(a) and Fig. 4(b).

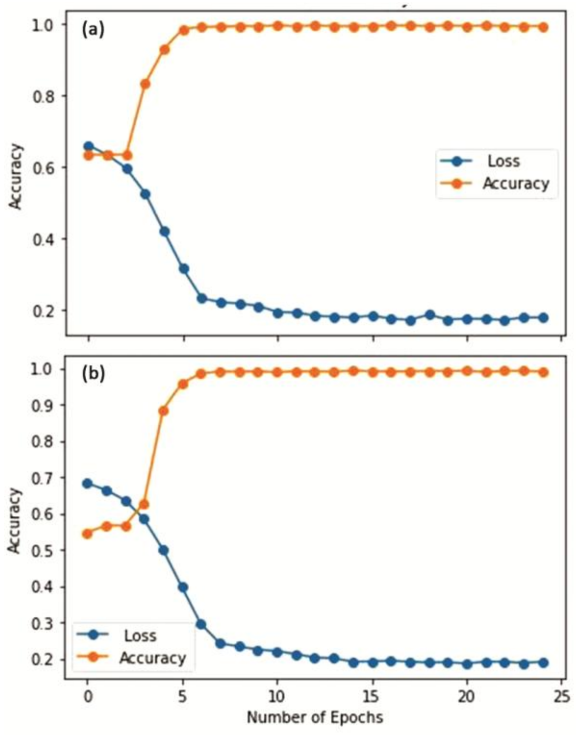


Fig. 4 — ShuffleNet V2 model accuracy and loss curves: (a) Test loss vs accuracy, (b) Train loss vs accuracy

Conclusions

A lightweight ShuffleNetV2-based Transfer Learning model with low power consumption, improved latency, higher efficiency, simple upgradeability, and an automatic visual inspection approach for casting defect detection, is presented in this study. The image transformation techniques were utilized as data augmentation for the casting defect dataset's adaptability. The hyperparameters like learning rate, activation function and batch size are fine-tuned to optimize the model performance by considering three test cases. Test case-3 achieves high accuracy of 99.58%, which is adequate for the industry requirements. The proposed method improves casting production quality and reduces the casting process's scrap rate. To accelerate detection, deploy the learned model to the NVIDIA Jetson Nano-kit edge device. In conclusion, the trained network model improves industrial production efficiency and is quick and precise in defect identification.

References

1 Wang T, Chen Y, Qiao M & Snoussi H, A fast and robust convolutional neural network-based defect detection model in product quality control, *Int J Adv Manuf Technol*, **94(8)** (2018) 3465–3471.

2 Liao Z, Abdelhafeez A, Li H, Yang Y, Diaz O G & Axinte D, State-of-the-art of surface integrity in machining of metal matrix composites, *Int J Mach Tools Manuf*, **143** (2019) 63–91.

3 Kim D H, Kim T J, Wang X L, Kim M, Quan Y J & Oh J W, Smart machining process using machine learning:A review and perspective on machining industry, *Int J Pr Eng Man-Gt*, **5(4)** (2018) 555–568.

4 Zhang H, Shen X, Bo A, Li Y, Zhan H & Gu Y, A multiscale evaluation of the surface integrity in boring trepanning association deep hole drilling, *Int J Mach Tools Manuf*, **123** (2017) 48–56.

5 Rao X, Zhang F, Lu Y, Luo X & Chen F, Surface and subsurface damage of reaction-bonded silicon carbide induced by electrical discharge diamond grinding, *Int J Mach Tools Manuf*, **154** (2020) 103564.

6 Zhang Z & Sun C, Multi-site structural damage identification using a multi-label classification scheme of machine learning, *Measurement*, **154** (2020) 107473.

7 Ravimal D, Kim H, Koh D, Hong J H & Lee S K, Image-based inspection technique of a machined metal surface for an unmanned lapping process, *Int J Pr Eng Man-Gt*, **7(3)** (2020) 547–557.

8 Boaretto N & Centeno T M, Automated detection of welding defects in pipelines from radiographic images DWDI, *Ndt & E Int*, **86** (2017) 7–13.

9 Cireşan D C, Meier U, Masci J, Gambardella L M & Schmidhuber J, High-performance neural networks for visual object classification, *arXiv* (2011) preprint arXiv: 1102.0183.

10 Shanmugamani R, Sadique M & Ramamoorthy B, Detection and classification of surface defects of gun barrels using computer vision and machine learning, *Measurement*, **60** (2015) 222–230.

11 Toğaçar M, Ergen B & Cömert Z, Classification of flower species by using features extracted from the intersection of feature selection methods in convolutional neural network models, *Measurement*, **158** (2020) 107703.

12 Janakiramaiah B, Kalyani G & Jayalakshmi A, Automatic alert generation in a surveillance system for smart city environment using deep learning algorithm, *Evol Intell*, **14(2)** (2021) 635–642.

13 Cheng F T, Tieng H, Yang H C, Hung M H, Lin Y C, Wei C F & Shieh Z Y, Industry 4.1 for wheel machining automation, *IEEE Robot Autom Lett*, **1(1)** (2016) 332–339.

14 Zhou X, Li Y & Liang W, CNN-RNN based intelligent recommendation for online medical pre-diagnosis support, *IEEE/ACM Trans Comput Biol Bioinform*, **18(3)** (2020) 912–921.

15 Ng H F, Automatic thresholding for defect detection, *Pattern Recognit Let*, **27(14)** (2006) 1644–1649.

16 Mery D & Filbert D, Automated flaw detection in aluminum castings based on the tracking of potential defects in a radioscopic image sequence, *IEEE Trans Robot Autom*, **18(6)** (2003) 890–901

17 Bai X, Fang Y, Lin W, Wang L & Ju B F, Saliency-based defect detection in industrial images by using phase spectrum, *IEEE Trans Industr Inform*, **10(4)** (2014) 2135–2145.

18 Duan L, Yang K & Ruan L, Research on automatic recognition of casting defects based on deep learning, *IEEE Access*, **9** (2020) 12209–12216.

19 Lin J, Yao Y, Ma L & Wang Y, Detection of a casting defect tracked by deep convolution neural network, *Int J Adv Manuf Technol*, **97(1)** (2018) 573–581.

20 Ferguson M, Ak R, Lee Y T T & Law K H, Automatic localization of casting defects with convolutional neural networks, *IEEE Int Conf Big Data*, (2017) 1726–1735.

21 Ren Z, Fang F, Yan N & Wu Y, State of the art in defect detection based on machine vision, *Int J Pr Eng Man-Gt* **9(2)** (2022) 661–691.

22 Benbarrad T, Salhaoui M, Kenitar S B & Arioua M, Intelligent machine vision model for defective product inspection based on machine learning, *J Sens Actuator Netw*, **10(1)** (2021).

23 Li E, Zeng L, Zhou Z & Chen X, Edge AI: On-demand accelerating deep neural network inference via edge computing, *IEEE Trans Wirel Commun*, **19(1)** (2019) 447–457.

24 Lin Y C, Chen W H & Kuo C H, Implementation of pavement defect detection system on edge computing platform, *Appl Sci*, **11(8)** (2021) 3725.

25 Shi W, Sun H, Cao J, Zhang Q & Liu W, Edge computing—an emerging computing model for the internet of everything era, *J Comput Sci Res Dev*, **54(5)** (2017) 907.

26 Subramanian M, Narasimha Prasad Lv, Janakiramaiah B, Mohan Babu A & Satishkumar Ve, Hyperparameter optimization for transfer learning of VGG16 for disease identification in corn leaves using bayesian optimization, *J Big Data*, **10(3)** (2022) 215–229.

27 Rajalaxmi R R, Narasimha L V, Janakiramaiah B, Pavankumar C S, Neelima N & Sathishkumar V E, Optimizing hyperparameters and performance analysis of LSTM model in detecting fake news on social media, *ACM Trans Asian Low-Resour Lang Inf Process* (2022), https://doi.org/10.1145/3511897

28 Dabhi R, Casting product image data for quality inspection, https://www.kaggle.com/datasets/ravirajsinh45/real-life-industrial-dataset-of-casting-product (27 December 2022)].