Deep Convolutional Neural Network for Coffee Bean Inspection

Ping Wang,[1*] Hsien-Wei Tseng,[1**] Tzu-Ching Chen,[2] and Chih-Hsien Hsia[3]

[1]College of Artificial Intelligence, Yango University, Fujian 350015, China
[2]Department of Economics, National Taiwan University, Taipei 10617, Taiwan
[3]Department of Computer Science and Information Engineering, National llan University, Yilan 260, Taiwan

(Received December 31, 2020; accepted May 17, 2021)

Keywords: lightweight explainable coffee bean quality testing system, convolutional neural network, knowledge distillation

Coffee is one of the most popular drinks in the world. It contains antioxidants and health-promoting nutrients that can boost one's energy and focus. However, defective beans mixed in with raw beans can easily affect the flavor and even be harmful to human health. The traditional human visual inspection of defective beans is extremely laborious and time-consuming and may result in low-quality coffee due to worker stress and fatigue. We propose a lightweight and explainable intelligent coffee bean quality inspection system that uses deep learning (DL) and computer vision (CV) technologies to assist operators in detecting defects, including mold, fermentation, insect bites, and crushed beans. We use knowledge distillation (KD) to achieve model compression. The basic explainable convolutional neural network (CNN) model is established using the explainable AI (XAI) method. The implemented system has a high identification rate, low complexity, and low power consumption, and can explain the judgment criteria of the complex classification model.

1. Introduction

Coffee is one of the most popular beverages in human society. Drinking a cup of coffee before work is a daily routine for many people. 95% of Taiwan's coffee beans are imported. If raw coffee beans are stored improperly in a hot and humid environment during transport, they are easily contaminated by ochratoxin A, which causes palpitations, headaches, and other symptoms. Moreover, crushed and insect-eaten beans and small stones, if not removed completely, will prevent even roasting, which reduces the flavor and even affects human health.(1)

The traditional method of detecting defective beans and crushed stones has been by human visual inspection, which requires training and long-term experience. The process is not only time-consuming but also prone to misjudgments due to worker stress and fatigue. Thus, the quality and efficiency of the process decrease with increasing working hours. To solve the problem, automated optical inspection (AOI) has been introduced as it uses optical instruments together with computer vision (CV) technology to capture surface images and detect foreign

*Corresponding author: e-mail: pwang@ygu.edu.cn
**Corresponding author: e-mail: hsienwei.tseng@gmail.com

https://doi.org/10.18494/SAM.2021.3277

objects or defects such as abnormal patterns. Introducing AOI into the production line improves inspection accuracy and speed, and production efficiency. It is especially appropriate for mass production.(2–18)

Deep learning (DL) has been adopted in AOI, which is a data-driven algorithm that learns how to classify objects from training data. DL learns to detect defective beans and foreign objects from a database of numerous images of defective and good-quality beans. Therefore, DL-based CV technology has helped the coffee industry enhance the performance of quality inspection and reduce the use of human inspection, which saves cost and time and solves the bottleneck of product testing.

Many researchers have studied methods of automatically classifying the quality of raw coffee beans. Faridah *et al.*(2) used texture analysis and RGB color information to extract the characteristics of coffee beans, and they trained a neural network for classification using the data. Turi *et al.*(3) combined color, morphology, and texture data with an artificial neural network (ANN) to identify the species and region of origin of Ethiopian coffee beans. This method successfully classified four coffee beans from different regions. L*a*b* color measurement values based on Commission Internationale d'Eclairage (CIE) have been used as features, and ANN and Bayesian classifiers (Bayes classifiers) have been applied for the quality classification of raw beans.(4) As convolutional neural networks (CNNs) are efficient in image feature extraction,(5) classification, object inspection, and so forth, deep CNNs are applied in the field of image processing. Thus, several researchers have tried to use a CNN to assess the quality of coffee beans. Huang *et al.*(6) used a CNN to process images of coffee beans and analyze image information. Pinto *et al.*(7) applied a CNN to classify six different types of defective beans. In Ref. 8, hyperspectral signals were processed by using a support vector machine and a CNN to find and classify insect-eaten and rotten beans. A CNN shows good performance in image classification or recognition but has high computational complexity. As a CNN requires large storage space and computing resources,(9) it cannot be used on mobile devices or embedded platforms. This problem must be solved to implement a smart coffee bean inspection system. Although deep neural networks (DNNs) have been successfully applied in many applications, the complex operation process does not ensure reliability. As neural layers become deeper and more complex, the process becomes more difficult for a human to understand. Therefore, it is not easy to effectively apply a DNN. Despite their high degree of regularity, DNNs sometimes produce unpredictable results, potentially causing errors.(10) This requires an understanding of the internal mechanism by which DNNs make decisions.

To solve such problems, many studies on model compression methods have been carried out to minimize the consumption of computing resources and time. As a result, accelerated model training and inferencing have been proposed. Model compression methods include network quantization, pruning, and knowledge distillation (KD).(11) A model with high computational complexity, low power consumption, and a small number of parameters is compressed as much as possible without affecting the accuracy rate (ACC).(12–14) However, a DNN still cannot explain the reasons behind its decisions and actions to humans. This problem is solved by AI-related technology, which provides transparent information features and produces explainable AI (XAI) models, thus gaining the trust and confidence of users.

Given the above problems, we propose a lightweight and explainable machine learning (ML) technology for a coffee bean quality assurance system that detects defective coffee beans. The proposed architecture adopts the KD method to simplify the redundant operation of the CNN and the local explainable model-agnostic explanation method (LIME) to approximate the regional target black-box model. The new architecture is expected to predict the output, which helps build a basic explainable DNN model. The architecture is expected to have a high recognition rate, low complexity, and low power consumption, and can explain the judgment criteria for predictions.

2. Related Research

2.1 Residual network (ResNet)

Increasing the number of layers in a CNN yields gradient disappearance and degradation problems in training. The degradation problem comes from the fact that the gradient cannot be transmitted backward, which results in error accumulation, a saturation of the network accuracy, and performance degradation. A residual neural network (ResNet) is modified on the basis of VGG19 (Visual Geometry Group network 19).(15) The residual unit is established through a short-circuit connection mechanism, as shown in Fig. 1. The residual learning mechanism established by ResNet enables deep CNN models with a residual unit structure formed by several stacking hidden layers to be trained easily. When the input is x, the learned feature is denoted as $H(x)$ and the learned residual is denoted as $F(x) = H(x) - x$. When the residual is 0, the accumulation layer only performs identity mapping. The network performance does not decrease with increasing depth. In a real application, the residual will not be 0, which means that the stacked layers learn new features, improving the performance of the network.

2.2 Compression of complex models based on KD

KD is an effective model compression method. It uses the trained complex teacher model to train a small student model. Training the student model as much as possible while maintaining the performance of the teacher model allows its deployment on mobile devices or embedded platforms that have limited computing resources.(12) The method uses a softened softmax

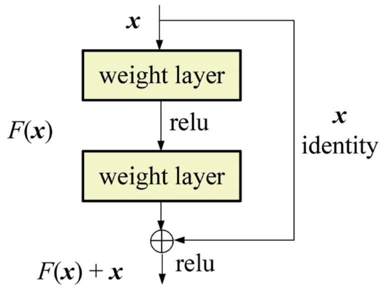


Fig. 1. (Color online) Residual blocks.

probability distribution to achieve model compression, which is appropriate for training a streamlined image classification network. For an image dataset, x_i is the input image, y_i is the standard addition category ($i = 1, 2, \ldots, n$), t is the teacher model, and $P_t = \mathrm{softmax}(z_t/T)$ is the predicted output probability, where z_t is the input of the softmax layer and T is the temperature parameter. Similarly, we define $P_s = \mathrm{softmax}(z_s/T)$ for a student model. The softmax function is defined as

$$P_i = \frac{\exp\left(z_i/T\right)}{\sum_j \exp\left(z_j/T\right)}, \tag{1}$$

where P_i represents the output probability of the ith category and z_i and z_j are the inputs of the softmax layer. $T = 1$ corresponds to a general softmax conversion. The probability distribution of the predicted results among the categories is the probability of extreme values. That is, the probabilities of the correct category and all other categories are close to 1 and 0, respectively. However, the probability distribution generated by the softmax function is flatter and softer for $T > 1$ than for $T = 1$ and provides more implicit information. The degree of similarity between each category is known when the training process of the model adds more judgment conditions. The student model learns in accordance with the loss function

$$L = \alpha \cdot L_{hard} + (1 - \alpha) \cdot L_{soft}, \tag{2}$$

where L_{hard} is a typical cross-entropy loss function (i.e., $T = 1$) in classification problems, L_{soft} is a soft cross-entropy loss function predicted by the teacher model, and α is a weighting factor that balances these two cross-entropy loss functions. Figure 2 shows a schematic diagram of KD.

2.3 Explainable neural network model

LIME uses a linear model to locally approximate the prediction results of the target black-box model.[10] It detects changes in the output of the black-box model by slightly perturbing the local

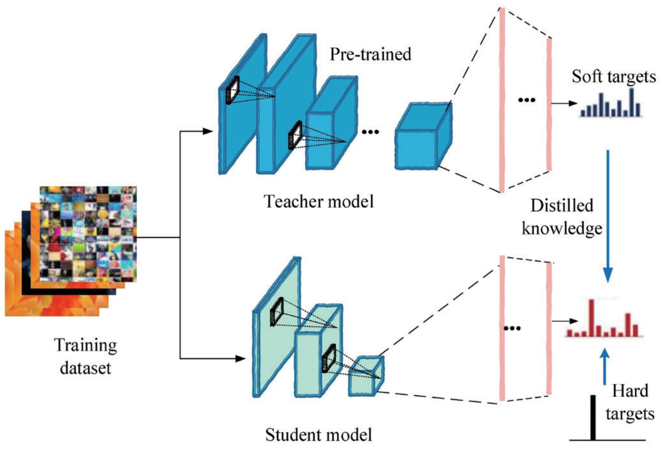


Fig. 2. (Color online) Operating mechanism of KD.

features of the input to determine which features in the training data play the most critical role in decision-making and training the explainable model. However, the model is a local approximation of the black-box model rather than a global approximation. For a complex classification model *f*, LIME finds an explainable model *g* to explain why an input sample is classified into the category predicted by *f*. Assuming that the loss function of a simply explainable model ξ used to approximate *f* is *L*, then π_{x_i} is the kernel function for calculating the similarity. $\Omega(g)$ is the complexity of the explainable model *g*. *G* represents all possible interpretation models, then the problem of interpreting the model ξ to explain $f(x_i)$ can be transformed into an optimization problem. We compare the approximation of the simple model *g* and the original model *f* by minimizing the loss function as shown in Eq. (3).

$$\xi(x_i) = \arg\min_{g \in G}\left[L(f, g, \pi_{x_i}) + \Omega(g) \right] \tag{3}$$

By image recognition, the disturbance to the input image can be understood. For example, the effect of the model on the image recognition is observed after covering several image blocks. On this basis, a simple linear regression model is established, which is used to explain the prediction results to clarify which segmentation images determine the output of the classification model.

3. Architecture of DNN

There are three models: a complex deep model for training, a lightweight model, and a model for explaining the classification decision of coffee beans.

3.1 Complex model for training based on ResNet-18

We first use ResNet-18 to train the coffee bean quality inspection model. The main purpose of training is to achieve high accuracy, and factors such as complexity and power consumption are not considered. ResNet-18 consists of a root block and stacks 1–4. Each stack is formed by superimposing two residual blocks, and all residual blocks use a 3×3 filter with two convolutional layers. Each layer in stacks 1, 2, 3, and 4 has 64, 128, 256, and 512 convolution kernels, respectively. ResNet-18 is connected to a global average pooling layer instead of a fully connected layer for comparison with a traditional classification network (Fig. 3).

3.2 Lightweight student model with KD

To construct a lightweight model, we propose a model based on a depthwise separable convolution algorithm(15) as the student model of KD. The trained ResNet-18 architecture is the teacher model, so the student model learns KD, as shown in Fig. 4. The student model avoids the computational complexity of the CNN algorithm and has fewer parameters. The model architecture in this work is based on a depthwise separable convolution algorithm, as shown in Fig. 5.

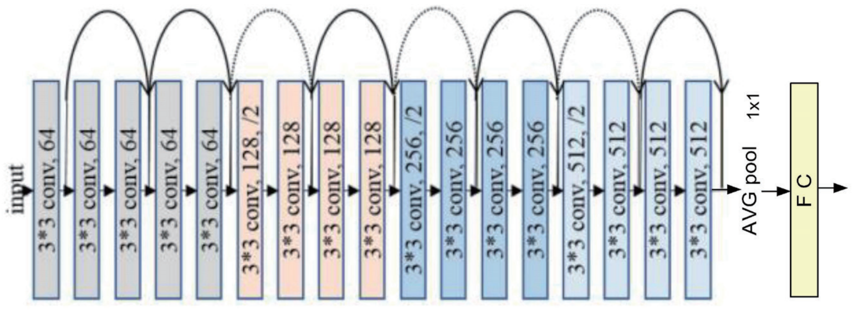


Fig. 3. (Color online) ResNet-18 architecture.

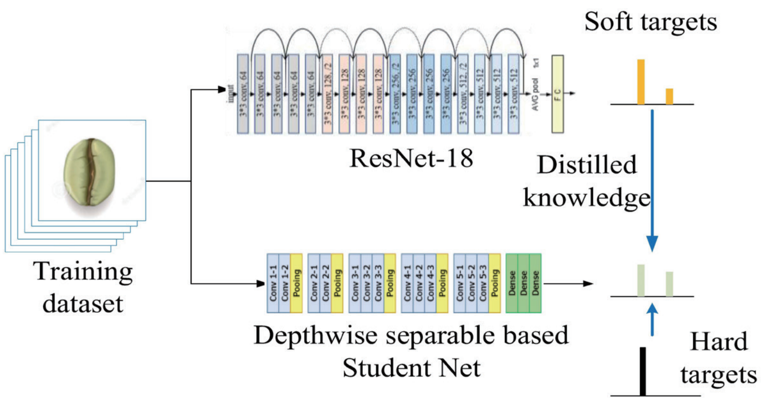


Fig. 4. (Color online) Lightweight CNN.

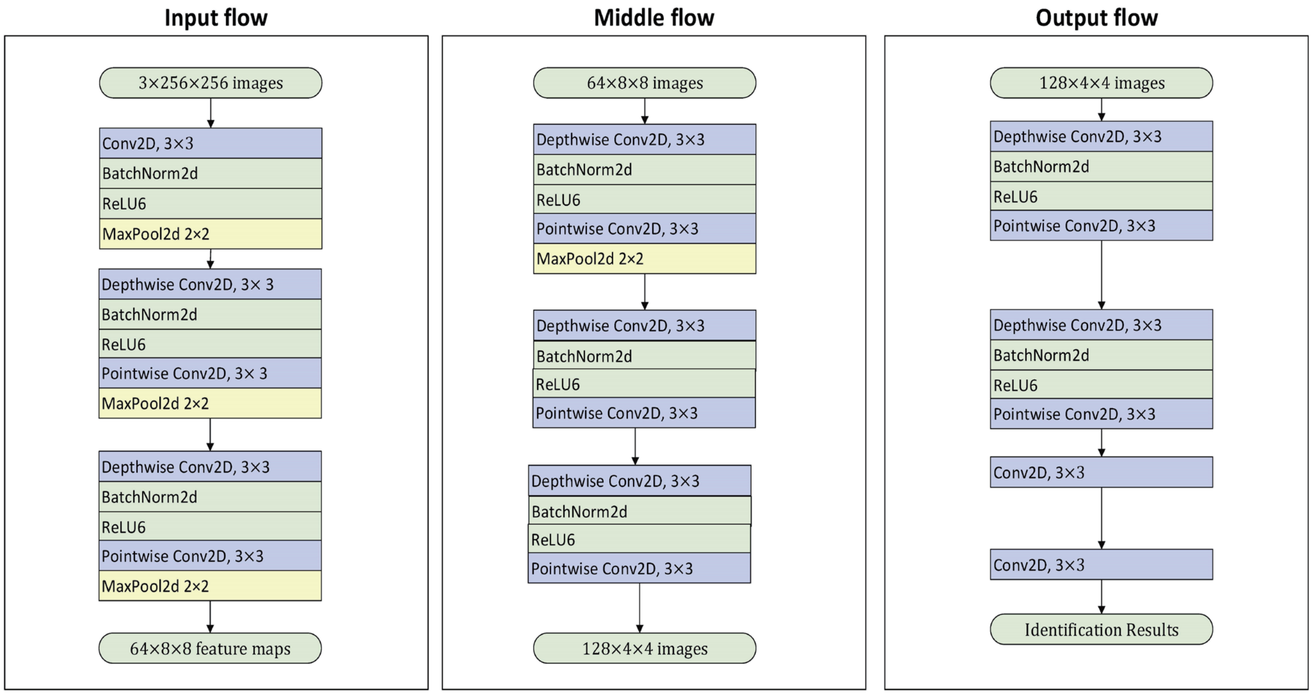


Fig. 5. (Color online) Proposed lightweight model based on depthwise separable convolution algorithm.

3.3 Neural network architecture based on LIME

The coffee bean image classifier decomposes an original input image into explainable local features, as shown in Fig. 6. Then, the input variables are disturbed by covering some parts with gray blocks. Different disturbance settings result in different probabilities of the classifier that predicts the image in the target category. On this basis, a linear regression model based on the locally weighted dataset is obtained. Finally, the feature with the highest positive weight is used to interpret the results of the model to understand why the model makes such a judgment, as shown in Fig. 7.

4. Results and Discussion

4.1 Dataset

An open-source dataset of coffee bean images was used for testing.[17] The dataset contains 4626 images of green coffee beans under the same light source, among which there are 2150 and 2476 images of good and defective beans, respectively. In the experiment, the dataset was divided into a training set of 4000 images and a testing set of 626 images.

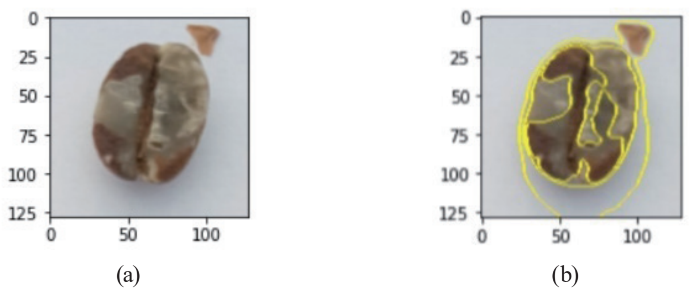


Fig. 6. (Color online) Coffee bean image decomposed into explainable local features: (a) original raw bean image and (b) segmented image.

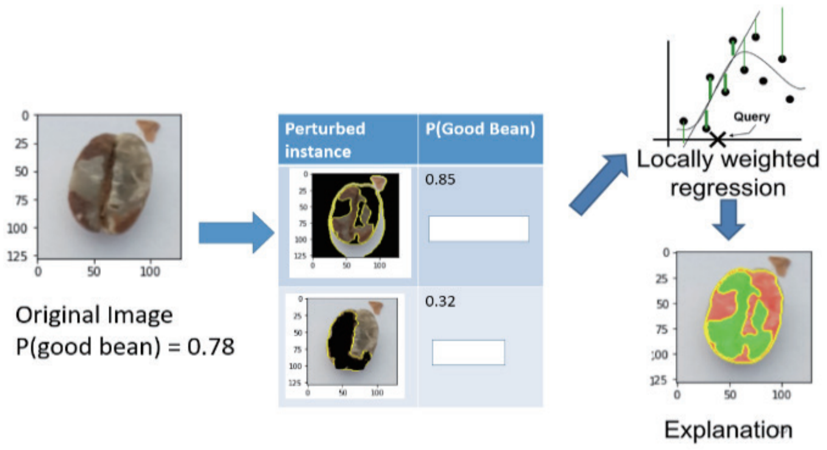


Fig. 7. (Color online) Linear regression model and the feature with the highest positive weight for XAI.

4.2 Training performance and accuracy

ResNet-18 constructed a large complex model, that is, the teacher model. The depthwise separable convolution algorithm was used for a lightweight student model that is similar to the VGG16 architecture. The ResNet-18 model was trained by switching from the adaptive moment estimation (ADAM) optimizer to the stochastic gradient descent (SGD) training method.(18) First, the ADAM optimizer was used as the optimizer with a learning rate (lr) of 0.1 for training. When the accuracy of the training set reached 95%, the ADAM optimizer was replaced with the SGD method. The accuracy of the ResNet-18 model reached about 93% in the testing set for lr of 0.01 and momentum of 0.002.

In the experiment, the student model was trained on the basis of KD where the training method of SWATS was used and the optimizer was changed to SGDM when the accuracy of the training dataset reached 95%. The accuracy of the training set reached 79% for lr of 0.01 and momentum of 0.002, as shown in Fig. 8. The model converged quickly when using Adam, but the fluctuation range of the ACC and the loss of the testing set became large.

Next, the KD design method was used with ResNet-18 for the teacher model, with a distilled model used for training the student model. The SGDM was used as the optimizer to continue training. The accuracy of the lightweight model with KD reached 91%, which was a better performance than the training method with KD, as shown in Fig. 9. The accuracy of the training method in which the student model was trained from scratch for ACC was 79%, whereas that of the training method with the student model using KD for ACC was 91%. The student model

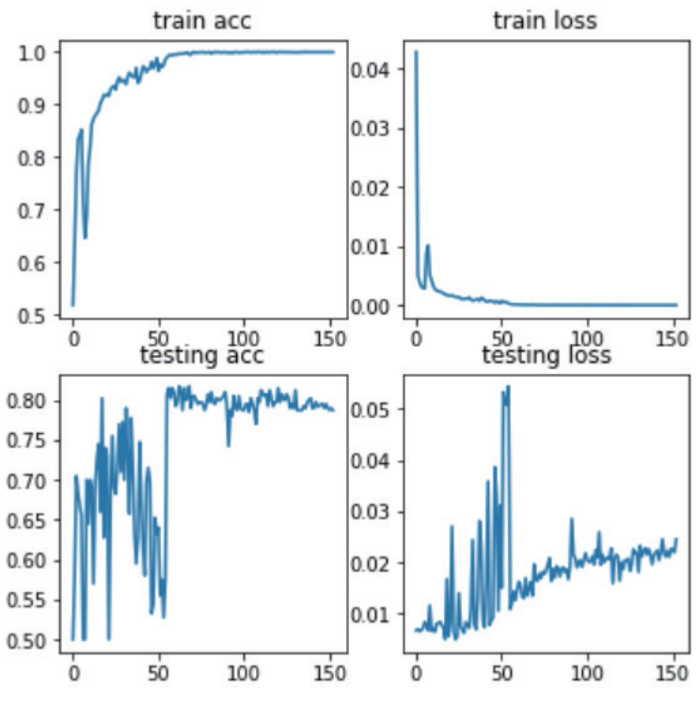


Fig. 8. (Color online) Results of lightweight model training without KD.

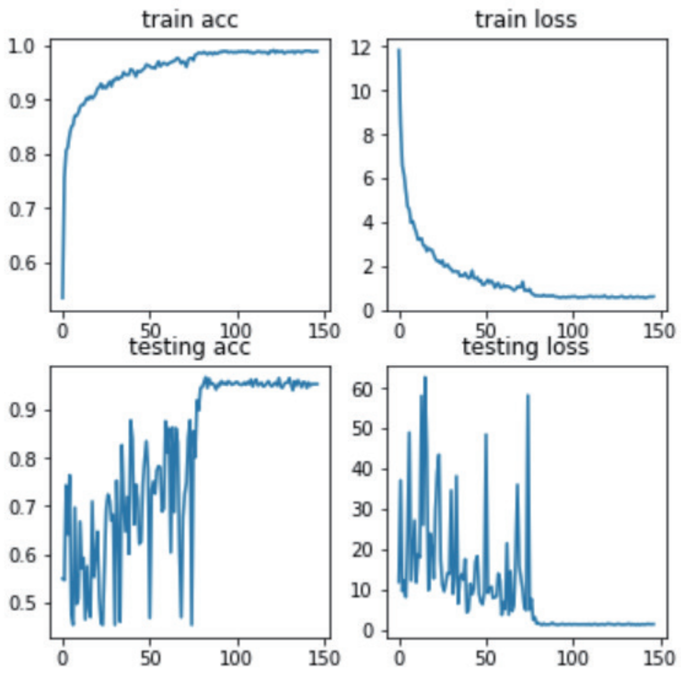


Fig. 9. (Color online) Results of lightweight model training with KD.

Table 1
Model parameters and memory consumption.

Model	Parameters	Memory cost (kB)
ResNet-18	1104000	43000
Student model	256779	1023

architecture used a quarter of the number of parameters as the ResNet-18 model, and its memory load was about 40 times less than that of ResNet-18, as shown in Table 1.

4.3 Interpretability of the model

A. Significance inspection of raw coffee bean images

Saliency maps show the important variables of a model. In the saliency maps in Fig. 10, light green spots show defective parts of coffee beans. Then, the loss function is identified by the proposed model that focuses on the different values of pixels in the images between defective and good coffee beans.

B. Explainable model

LIME establishes an explainable model based on the quality analysis of raw coffee beans. Figure 11(a) shows the result of the analysis of sour and moldy beans by LIME. The result shows that the parts in green focus on the shape and uneven color of the defective beans as these parts have a positive effect on correcting a prediction. The classification results of defective beans are

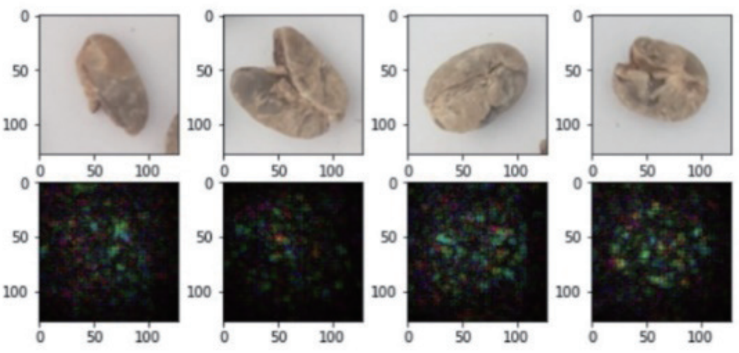
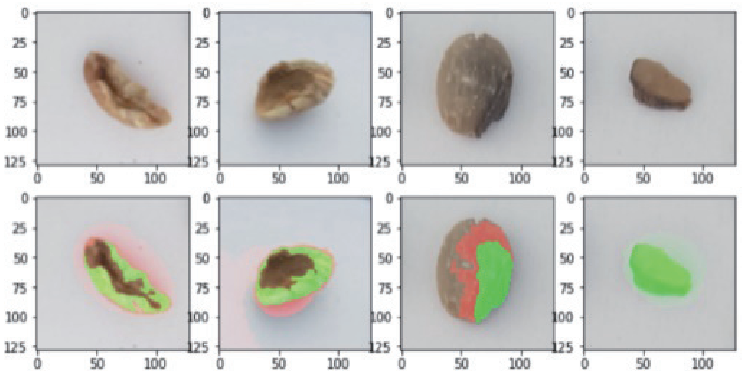
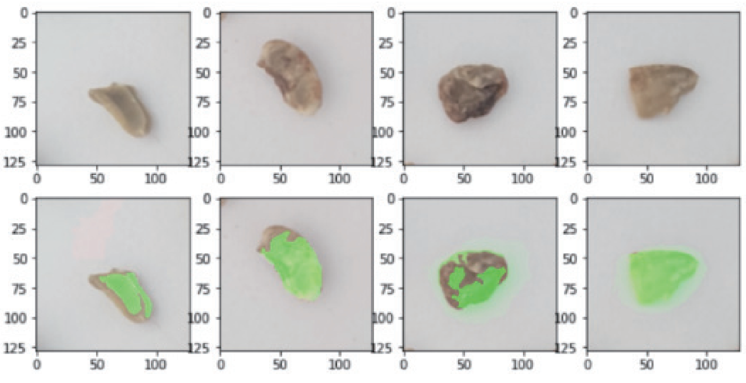


Fig. 10. (Color online) Saliency maps of raw coffee beans produced by the proposed model.



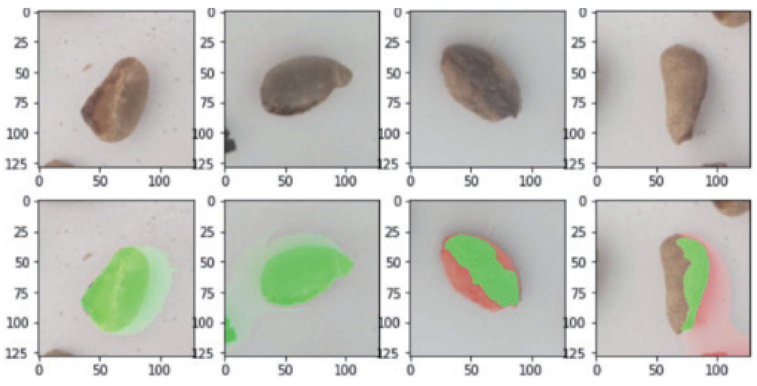
(a)



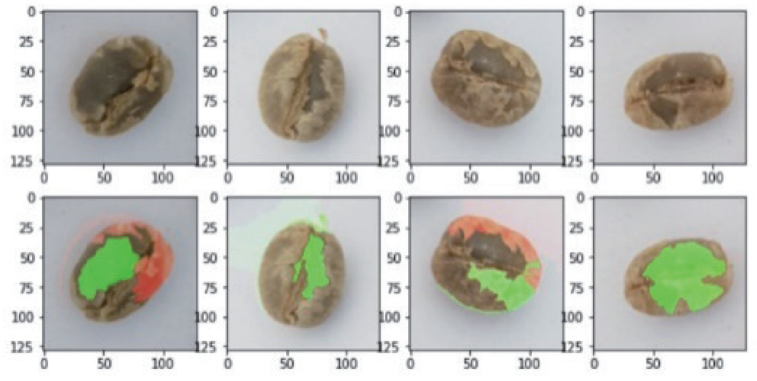
(b)

Fig. 11. (Color online) Analysis results of coffee beans produced by the explainable model. (a) Sour, moldy beans. (b) Crushed beans and beans bitten by insects.

shown in Figs. 11(b) and 11(c). Figure 11(b) shows that the model focuses on the broken shapes to determine crushed beans. Figure 11(c) shows that the model judges beans to be black and moldy from the discolored parts. Good beans are represented as green parts, which have a positive impact on the prediction results and focus on uniform and light green parts of coffee beans. The



(c)



(d)

Fig. 11. (Continued) (Color online) Analysis results of coffee beans produced by the explainable model. (c) Bad beans. (d) Good beans.

white spots of the green beans adversely affect the accuracy of prediction. The model regards white spots as defects, as shown in Fig. 11(d). The colored parts contribute to improving the model. For example, the image enhancement of data with significant characteristics can help improve the model and increase the overall performance.

5. Conclusion

Even though great progress has been made in DL methods and their applications in the past ten years, DL does not provide sufficient reliability. As the complexity of a neural network increases, its inferential logic process becomes more difficult to understand. We combined DL and CV technology to establish a system for inspecting coffee bean quality and obtained a lightweight and explainable high-performance CNN model through KD and model compression. The proposed architecture accurately classifies the quality of coffee beans and allows understanding of the model's feature marks of the image, which makes the information open and transparent.

References

1 B. Cheng, A. Furtado, H. E. Smyth, and R. J. Henry: Trends Food Sci. Technol. **57** (2016) 20. https://doi.org/10.1016/j.tifs.2016.09.003

2 F. Faridah, G. O. F. Parikesit, and F. Ferdiansjah: TELKOMNIKA **9** (2011) 547. https://doi.org/10.12928/telkomnika.v9i3.747

3 B. Turi, G. Abebe, and G. Goro: East African J. Sci. **7** (2013) 1. https://www.researchgate.net/publication/276264808_Ethiopian_Roasted_Coffee_Classification_Using_Imaging_Techniques

4 E. M. de Oliveira, D. S. Leme, B. H. G. Barbosa, M. P. Rodarte, and R. G. F A. Pereira: J. Food Eng. **171** (2016) 22. https://www.cheric.org/research/tech/periodicals/doi.php?art_seq=1383618

5 A. Krizhevsky, I. Sutskever, and G. E. Hinton: Adv. Neural Inform. Process Syst. **25** (2012) 1. https://doi.org/10.1145/3065386

6 N.-F. Huang, D.-L. Chou, C.-A. Lee, F.-P. Wu, A.-C. Chuang, Y.-H. Chen, and Y.-C. Tsai: IET Smart Cities **2** (2020) 167. https://doi.org/10.1049/iet-smc.2020.0068

7 C. Pinto, J. Furukawa, H. Fukai, and S. Tamura: Porc. 2017 Int. Conf. Advanced Info., Concepts, Theory, and Applications (ICAICTA, 2017) 1. https://doi.org/10.1109/ICAICTA.2017.8090980

8 S.-Y. Chen, C. Y. Chang, C.-S. Ou, and C.-T. Lien: Remote Sens. **12** (2020) 15.

9 M. Arredondo-Velázquez, J. Diaz-Carmona, A. Barranco-Gutiérrez, and C. Torres-Huitzil: IEEE Latin Am. Trans. **18** (2020) 971. https://latamt.ieeer9.org/index.php/transactions/article/view/3104

10 F. K. Došilović, M. Brčić, and N. Hlupić: Proc. 2018 Int. Conv. Information and Communication Technology, Electronics and Microelectronics (2018) 0210.

11 B. L. Deng, G. Li, S. Han, L. Shi, and Y. Xie: Proc. IEEE **108** (2020) 485.

12 G. Hinton, O. Vinyals, and J. Dean: Distilling the Knowledge in a Neural Network (2015). https://arxiv.org/abs/1503.02531

13 J.-S. Chiang, C.-H. Hsia, S.-H. Chang, W.-H. Chang, H.-W. Hsu, Y.-J. Dai, C.-Y. Li, and M.-H. Ho: 2010 SICE Annu. Conf. (2010) 2269.

14 W.-M. Li, C.-H. Hsia, and J.-S. Chiang: 2009 IEEE Int. Symp. Circuits and Systems (2009) 750. https://doi.org/10.1109/ISCAS14548.2009

15 K. He, X. Zhang, S. Ren, and J. Sun: Identity Mappings in Deep Residual Networks (2016). https://arxiv.org/abs/1603.05027v3

16 A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam: MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Application (2017). https://arxiv.org/abs/1704.04861v1

17 Coffee-cobra: https://github.com/edgeryders/coffee-cobra (accessed May 2020).

18 N. S. Keskar and R. Socher: Improving Generalization Performance by Switching from Adam to SGD (2017). https://arxiv.org/abs/1712.07628v1