Sensors and Materials, Vol. 37, No. 8 (2025) 3785–3802
MYU Tokyo

S & M 4152

Lightweight Vision-transformer-based Coffee Bean Quality Inspection with Class-aware Unsupervised Domain Adaptation

Hsien-Chun Cho,[1] Zih-Ching Chen,[2] and Ching-Yi Chen[1]*

[1]Department of Electrical Engineering, Ming Chuan University, Taoyuan, Taiwan
[2]NVIDIA AI Technology Center, NVIDIA Corporation, Santa Clara, CA, USA

(Received April 21, 2025; accepted August 15, 2025)

Keywords: automated optical inspection, unsupervised domain adaptation, vision transformer, model compression

With the rapid development of automated optical inspection (AOI) technology and imaging sensor systems, image-based coffee bean quality inspection has emerged as an effective alternative to manual sorting. High-resolution imaging sensors are capable of capturing critical surface features of coffee beans, including texture, color, and structural patterns, thereby providing rich input for downstream intelligent classification algorithms. However, most existing research has concentrated on green beans, while labeled datasets for roasted beans remain scarce. This imbalance severely restricts the generalization capability of trained models in real-world applications. To address this issue, a class-aware unsupervised domain adaptation (UDA) framework based on the vision transformer (ViT) architecture is proposed. The framework simultaneously aligns the feature distributions between the source domain (green beans) and the target domain (roasted beans), while enhancing class-level consistency during training. This design effectively mitigates domain shifts induced by variations in coffee processing stages, thereby improving model robustness in cross-domain scenarios. In addition, to enhance deployment efficiency and operational practicality in intelligent sensing environments, a model compression strategy is further introduced. By leveraging the modular dependency structure inherent in transformer-based architectures, we developed an approach that integrates structured pruning with knowledge distillation (KD) to significantly reduce model complexity while preserving classification performance. Experimental results confirm that the proposed method delivers high classification accuracy and generalization capability, demonstrating its potential for deployment in image-based coffee bean quality inspection systems.

1. Introduction

Coffee is one of the most economically valuable commodities in the global beverage market, offering health benefits such as enhanced mental concentration and promoting metabolism when

*Corresponding author: e-mail: chingyi@mail.mcu.edu.tw
https://doi.org/10.18494/SAM5701

consumed in moderation. However, the presence of defective beans that are not properly removed during the processing stage can negatively affect the flavor profile and even pose potential health risks to consumers. Traditional inspection methods heavily rely on manual operations, which are not only labor-intensive and inefficient but also susceptible to subjective judgments, rendering them unsuitable for large-scale production environments. With the advancement of automation and computer vision (CV) technologies, automated optical inspection (AOI) systems that integrate high-resolution imaging sensors with CV algorithms have been widely adopted for quality assessment in the food and coffee industries,(1,2) providing an effective alternative to manual sorting. Imaging sensors are capable of capturing critical surface characteristics of coffee beans, including texture, color, and shape, which serve as the input for intelligent decision-making algorithms. When combined with lightweight models deployed on edge devices, this overall system enables real-time classification and defect detection, significantly enhancing both production line efficiency and inspection accuracy.

In the context of integrating AI and sensor technology, data-driven learning models have become the dominant tools for quality inspection tasks. Particularly when labeled data are abundant, such models can effectively extract latent patterns and structures from sensory data. Deep neural networks (DNNs), as the dominant learning architecture in recent years, have also been effectively applied to coffee bean quality inspection tasks.(3–7) Nevertheless, DNN-based models typically require large-scale labeled datasets for supervised learning, for which it is often assumed that the training and testing data are drawn from the same underlying distribution. When deployed in real-world scenarios where the target domain exhibits substantial distribution shifts from the source domain, model performance often deteriorates significantly.(8) In the context of coffee bean quality inspection, most existing studies have focused primarily on green coffee beans, while research targeting roasted coffee beans remains limited. Moreover, publicly available labeled datasets of roasted beans are extremely scarce. Given the substantial differences in appearance, color, and surface characteristics between green and roasted beans, the distribution discrepancy between the source and target domains poses significant challenges for conventional supervised learning models, severely limiting their generalization capability in cross-domain tasks.

To address this challenge, transferring knowledge from a source domain with abundant labeled data to a target domain lacking annotations has become an important and challenging research problem. Domain adaptation (DA) techniques aim to enhance the transferability of models across different data distributions, particularly in scenarios where the target domain contains limited or no labeled data. Among various DA approaches, unsupervised domain adaptation (UDA) has emerged as a key solution with the primary objective of learning domain-invariant representations through feature adaptation strategies that align the feature distributions between the source and target domains. Existing UDA methods can generally be categorized into two main approaches. The first category adopts explicit feature alignment strategies, which minimize the statistical distance between the source and target domains to achieve cross-domain feature alignment.(9) The second category is inspired by generative adversarial networks (GANs), leveraging adversarial learning mechanisms combined with a domain discriminator to implicitly align feature distributions across domains.(10,11)

However, despite the widespread application of UDA techniques across various cross-domain tasks, their performance remains subject to several challenges and limitations. Most UDA methods assume that the source and target domains share an identical label space. Nevertheless, in real-world scenarios, label information in the target domain is often unavailable or incomplete. Moreover, in situations where label shifts(12) or shifts in the support(13) exist between domains, learning domain-invariant representations becomes theoretically challenging, and such strategies may fail to guarantee effective cross-domain transfer performance.(14) In addition, many existing UDA methods primarily focus on aligning the global feature distributions between domains, while overlooking the class-specific characteristics of individual samples. This class-agnostic learning approach often results in the failure to learn discriminative feature representations, thereby limiting the performance of the model in downstream tasks such as classification.(15,16)

To address the aforementioned challenges, a class-aware UDA framework based on the vision transformer (ViT)-tiny architecture is proposed for coffee bean quality inspection. The framework begins by utilizing a label predictor and a domain classifier within an adversarial learning setup to align the feature distributions between the source and target domains. During each training iteration, after achieving feature alignment, the system further employs a self-training (ST) mechanism to enhance class-level consistency. High-confidence samples are selectged from the target domain to generate pseudo-labels, which are then incorporated into the training dataset. This iterative pseudo-labeling process effectively improves the model's generalization capability in the roasted bean domain. Furthermore, to reduce model complexity, a lightweight strategy is introduced that integrates structured pruning and knowledge distillation (KD), tailored to the strong interdependences across transformer layers. Structured pruning is first applied to the pretrained ViT-tiny model to remove redundant parameters and compress the model. Subsequently, KD is employed to transfer knowledge from the original teacher model to the compressed student model, achieving a balance between classification accuracy and inference efficiency.

2. Related Approaches

2.1 ViT

Convolutional neural networks (CNNs) are characterized by local connectivity and weight sharing, which allow them to achieve excellent performance in image feature extraction tasks. Along with the continuous evolution of neural network architectures, various classic CNN structures have been proposed, among which the residual neural network (ResNet) is a prominent representative. ResNet effectively addresses the degradation problem commonly encountered in very deep networks by introducing identity mapping mechanisms, allowing for the construction of deeper and more easily trainable models. Consequently, ResNet has become one of the mainstream architectures widely adopted in CV applications. On the other hand, in the field of natural language processing (NLP), the transformer architecture, which is based on the self-attention mechanism, has emerged as the dominant model paradigm.(17) Owing to its superior computational efficiency and scalability, transformer-based models have been successfully scaled to unprecedented sizes, and there is still no evidence of performance saturation with increasing model size or dataset scale. This trend highlights the tremendous potential of transformer architectures for future development across various AI domains.

Inspired by the remarkable success of transformer architectures in the field of NLP, ViT(18) was the first architecture in which the standard transformer framework is directly applied to image recognition tasks. ViT divides an input image into fixed-size patches and flattens each patch into a one-dimensional vector, which is then linearly projected into an embedding vector as the input to the transformer encoder. This design is analogous to the token processing mechanism in NLP tasks, where the image is transformed into a sequence of patch embeddings to facilitate global information modeling and learning, as illustrated in Fig. 1. To preserve the spatial information of the input image, ViT introduces position embeddings to encode the original location of each patch. The overall architecture consists of multiple layers of multihead self-attention and feed-forward networks, with each layer incorporating layer normalization (LN) and residual connections. During the training phase, ViT employs a multilayer perceptron (MLP) head for image classification, while in the fine-tuning stage, it can optionally adopt a single-layer linear classifier. Compared with conventional CNNs, ViT is more capable of capturing long-range dependences and modeling global relationships within images. Consequently, it has demonstrated competitive performance on large-scale image datasets and has emerged as one of the most prominent architectures in recent visual representation learning research.

2.2 UDA techniques

UDA is one of the key techniques for enabling models to generalize across different domains. Most UDA algorithms aim to reduce the domain gap between the source and target domains by

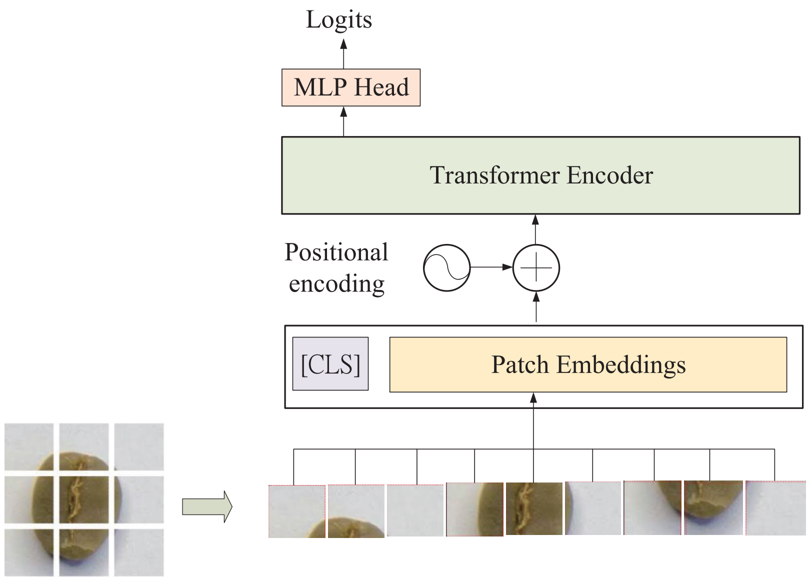
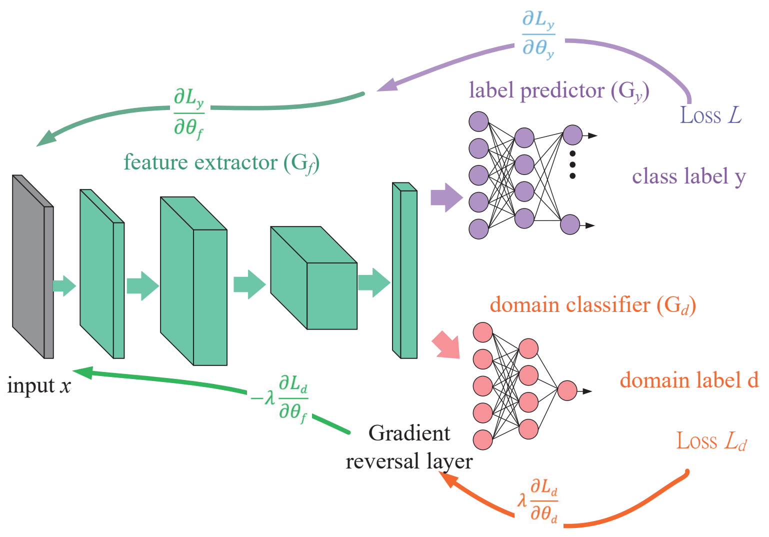


Fig. 1. (Color online) The architecture of ViT.

aligning their feature representations, thereby improving the model's prediction performance on the target domain. Domain-adversarial training of neural networks (DANNs)(19) is one of the representative approaches in UDA. As illustrated in Fig. 2,(19) the architecture of DANN consists of three main components: a feature extractor (G_f), a label predictor (G_y), and a domain classifier (G_d). Through adversarial learning, DANN maps data from both the source and target domains into a shared feature space, thereby learning domain-invariant representations. During the training process, the label predictor is optimized to minimize the classification cross-entropy loss on the labeled source domain samples. The domain classifier is trained to minimize the domain confusion loss, which corresponds to correctly classifying whether the input features come from the source or target domain. In contrast, the feature extractor is optimized to maximize the classification error of the domain classifier, thereby promoting domain confusion. A gradient reversal layer (GRL) is introduced between the feature extractor and the domain classifier to enable adversarial training. This mechanism facilitates the simultaneous optimization of the label classification task and the domain alignment task, allowing the network to learn discriminative yet domain-invariant features during training.

2.3 ST

ST(20) is a learning technique that utilizes a pretrained classifier, trained on a small amount of labeled data, to perform inference on unlabeled samples. High-confidence samples are selected and assigned pseudo-labels, thereby increasing the proportion of labeled samples within the training dataset. These pseudo-labeled samples, together with the original labeled data, are then used to retrain the classification model in order to further enhance its classification capability. However, conventional ST methods are typically applicable only when the source and target data samples are drawn from the same underlying distribution, without the presence of domain discrepancy between them.



$\frac{\partial L_y}{\partial \theta_y}$
$\frac{\partial L_y}{\partial \theta_f}$
label predictor (G_y)
Loss L
feature extractor (G_f)
class label y
domain classifier (G_d)
domain label d
input x
$-\lambda \frac{\partial L_d}{\partial \theta_f}$
Gradient reversal layer
Loss L_d
$\lambda \frac{\partial L_d}{\partial \theta_d}$

Fig. 2. (Color online) Architecture of DANN.

Given a labeled dataset $\left\{\left(x^r, \bar{y}^r\right)\right\}_{r=1}^{M}$ and an unlabeled dataset $\left\{\left(x^u\right)\right\}_{u=1}^{N}$, where the number of unlabeled samples is much larger than that of labeled samples, i.e., $N \gg M$, the ST process proceeds as follows. First, an initial classifier model f^* is trained using the labeled dataset. The trained classifier f^* is then applied to the unlabeled samples to generate corresponding predicted labels, resulting in a pseudo-labeled dataset $\left\{\left(x^u, y^u\right)\right\}_{u=1}^{N}$. Subsequently, samples with high prediction confidence are selected in accordance with a predefined confidence threshold. These high-confidence pseudo-labeled samples are removed from the unlabeled dataset and incorporated into the labeled dataset. The classifier is then retrained using the updated labeled dataset to refine the model parameters. This process can be iteratively repeated according to the practical application requirements until the model converges. The overall procedure of the ST algorithm is illustrated in Fig. 3.

2.4 Model compression technique

For the improvement of the usability and computational efficiency of DNNs in resource-constrained environments, model compression techniques have become an important research direction. Techniques such as pruning(21) and KD(22) can effectively reduce the computational complexity, energy consumption, and model size while maintaining accuracy comparable to that of the original model. These techniques significantly optimize the deployment performance of DNN models in practical applications.

2.4.1 Pruning

Pruning is a commonly used model compression technique(21) to remove redundant or less important parameters from a model while maintaining its performance. By eliminating insignificant weights during the training process, one can construct a sparse model structure, allowing zero-valued parameters to be ignored during inference. This enables more efficient computation and reduces inference latency, as illustrated in Fig. 4. Essentially, pruning is

```
ST algorithm
Initialize:
Given a labeled dataset $D_r = \{(x^r, \hat{y}^r)\}_{r=1}^{M}$, and an unlabeled dataset $D_u = \{(x^u)\}_{u=1}^{N}$
While the stopping criteria is not met do
  Train the model $f^*$ using the labeled dataset
  Apply $f^*$ to the unlabeled dataset, and obtain $\{(x^u, y^u)\}_{u=1}^{N}$
  Select high-confidence samples: $\{(x^{conf}, y^{conf})\} \subseteq \{(x^u, y^u)\}_{u=1}^{N}$
  Remove selected samples from unlabeled dataset
Add the selected samples to labeled dataset
End while
```

Fig. 3. Pseudocode of the ST algorithm.

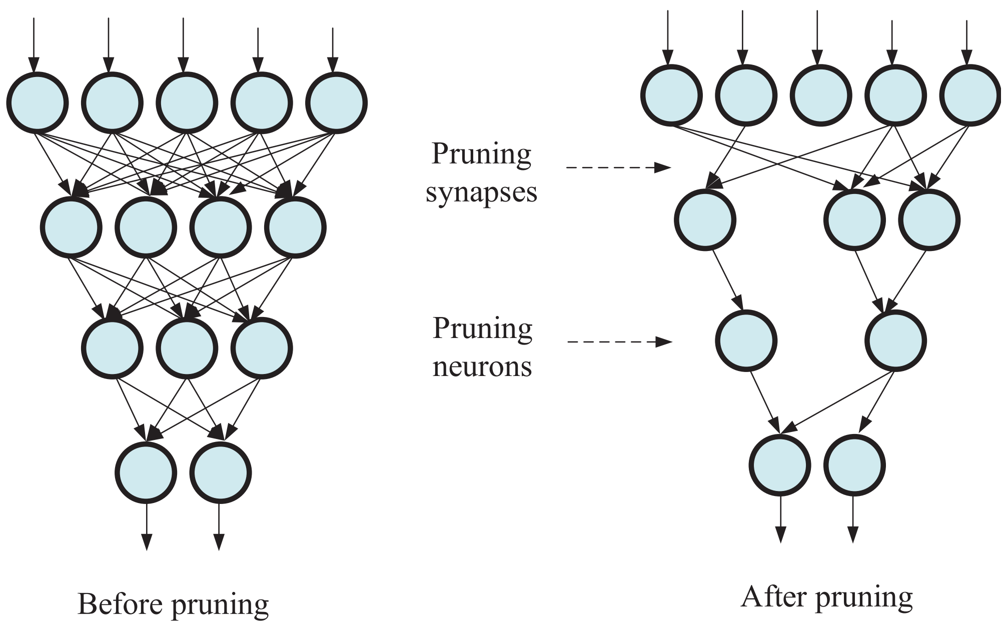


Fig. 4. (Color online) Illustration of the pruning process.

regarded as an optimization strategy that balances the trade-off between model size and prediction performance. The typical pruning process involves three main steps: first, a large-capacity full model is trained to ensure sufficient representation capability; second, parameters or channels are ranked according to their importance, and those with minimal impact on the model output are removed to obtain a compact lightweight model; finally, the pruned model is fine-tuned to recover any performance degradation caused by pruning.

2.4.2 KD

The core concept of KD lies in transferring the knowledge learned by a pretrained, high-capacity teacher model to a more compact student model with fewer parameters, in order to enhance the predictive capability of the latter.(22) During the training process, the student model is trained using a combination of supervision signals: the one-hot encoded ground-truth labels from the original training dataset and the soft probability distributions generated by the teacher model through a softmax function. These two components are jointly optimized within a weighted loss function, which constitutes the overall KD objective, as defined in

$$L_{KD} = \alpha \cdot L_{hard} + \left(1-\alpha\right) \cdot L_{soft}. \tag{1}$$

In this formulation, α is a weighting factor that controls the relative importance between the hard-label loss and the soft-label loss during the overall training process. Specifically, L_{hard} denotes the conventional cross-entropy loss computed with the one-hot encoded ground-truth labels for the classification task, while L_{soft} represents the softened cross-entropy loss based on the probability distributions predicted by the teacher model. The overall architecture and workflow of the KD process are illustrated in Fig. 5.

3. System Design

We propose a cyclic DA framework, referred to as Cycle UDA-ST (CUDA-ST), which decomposes the overall learning task into two sequential subtasks: UDA and ST. These two stages are alternately executed an iteratively to progressively optimize the model performance. In addition, the system is specifically designed considering edge computing environments commonly encountered in intelligent sensing applications. To address the constraints of such deployment scenarios, the architecture incorporates structured pruning and KD strategies, thereby reducing model complexity and enabling high-accuracy quality inspection with low computational cost.

3.1 Cross-domain feature alignment strategy

In the application of coffee bean quality inspection, existing methods often rely on a large amount of labeled green coffee bean data to train classification models. However, labeled datasets for roasted coffee beans are extremely limited. Furthermore, the inherent differences in characteristics and appearance between green and roasted beans make it difficult for classifiers trained solely on green beans to be directly applied to the quality inspection of roasted beans. To enhance the model's capability for cross-domain feature alignment, we develop a UDA-based feature alignment strategy. The overall framework consists of three main modules: a feature

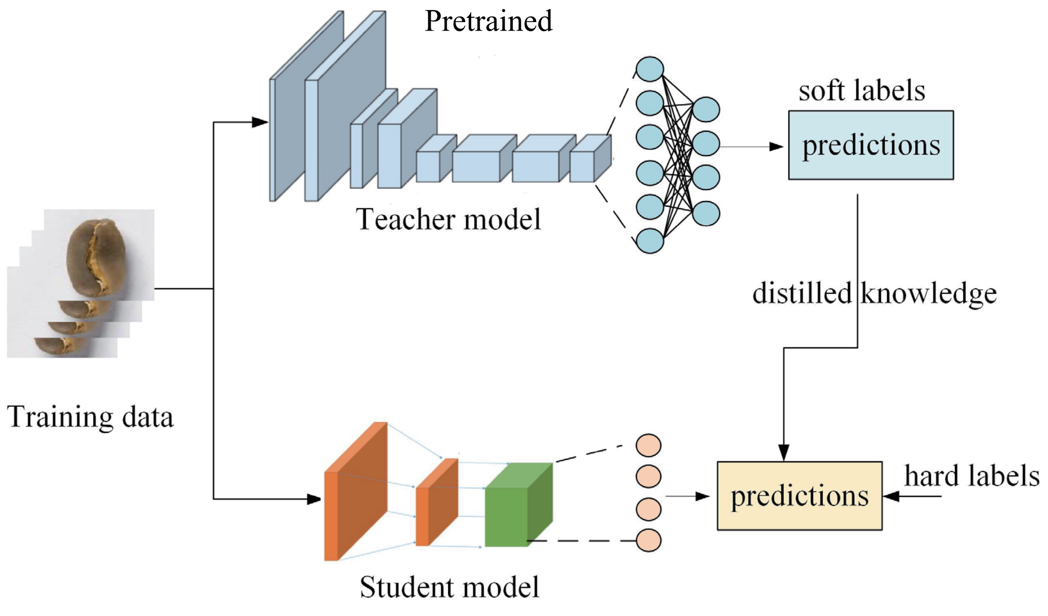


Fig. 5. (Color online) Overall architecture and workflow of the KD process.

extractor, a label predictor, and a domain classifier. The feature extractor is implemented using a lightweight ViT-tiny model, which is responsible for extracting shared feature representations from both source (green beans) and target (roasted beans) domain images. The label predictor performs quality classification on the basis of the extracted features, while the domain classifier determines the domain origin of the feature vectors. An adversarial training strategy is adopted in this architecture. Specifically, a GRL is introduced to encourage the feature extractor to simultaneously enhance the classification capability of the label predictor and generate features that are indistinguishable for the domain classifier. By maximizing the classification error of the domain classifier, the system implicitly aligns the feature distributions between the source and target domains, as illustrated in Fig. 6.

In this architecture, the primary objective of the feature extractor is to learn feature representations that prevent the domain classifier from effectively distinguishing between the source and target domains. The optimal parameter set of the feature extractor, denoted as θ_f^*, is obtained by minimizing the overall loss function.

$$\theta_f^* = \min_{\theta_f} L - \lambda \cdot L_d \tag{2}$$

The label predictor is responsible for predicting the quality category corresponding to the input data, and its training relies solely on the labeled samples from the source domain. In Eq. (3), θ_p^* denotes the optimal parameter set of the label predictor, and L represents the cross-entropy loss computed on the labeled source domain samples.

$$\theta_p^* = \min_{\theta_p} L \tag{3}$$

The domain classifier is a binary classifier designed to determine the domain membership of the input feature vectors. Its training objective is to minimize the domain classification loss. In

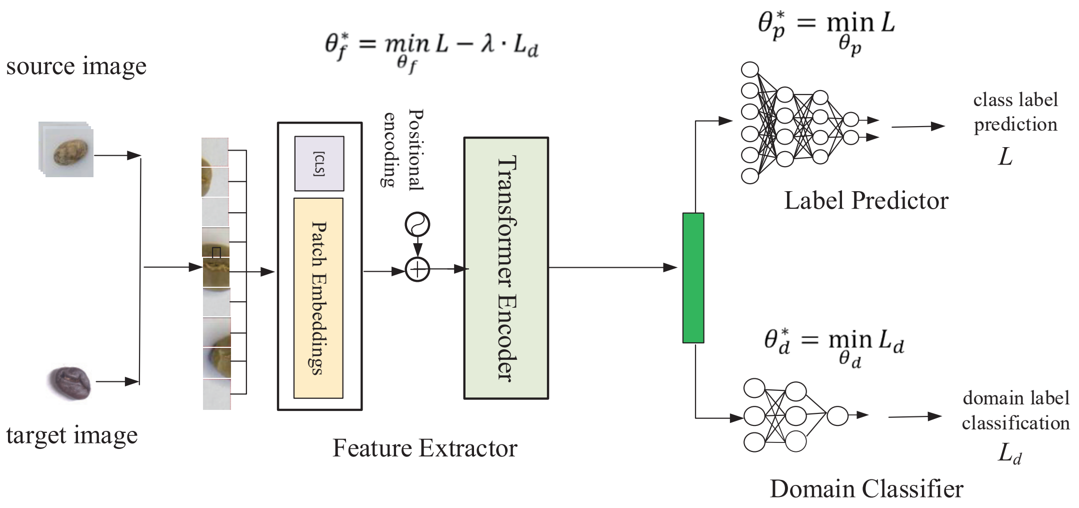


Fig. 6. (Color online) Cross-domain feature alignment strategy based on UDA.

Eq. (4), θ_d^* denotes the optimal parameter set of the domain classifier and is optimized by adjusting the parameter set θ_d to minimize the domain confusion loss L_d.

$$\theta_d^* = \min_{\theta_d} L_d \tag{4}$$

3.2 ST for enhanced class-level alignment

In each training iteration, after completing the feature alignment process, the system further applies the ST mechanism to perform pseudo-labeling on the high-confidence samples from the target domain. The generated pseudo-labeled samples are incorporated into the training dataset to enhance the model's class-level alignment capability. This step improves class consistency and enhances the model's generalization on the target domain. The overall process is illustrated in Fig. 7.

3.3 Model compression strategy based on pruning and KD

In this study, a lightweight ViT-tiny model is adopted as the backbone architecture of the feature extractor. To further enhance the model's deployability in embedded environments or real-time application scenarios, we design a lightweight strategy that integrates pruning and KD to effectively reduce the model complexity while maintaining satisfactory performance. As illustrated in Fig. 8, the proposed approach first applies structured pruning to the pretrained ViT-tiny model to remove redundant parameters, resulting in a compact student model. Subsequently, the KD mechanism is employed to transfer knowledge from the large-capacity teacher model to the student model, guiding the student model to learn both the output

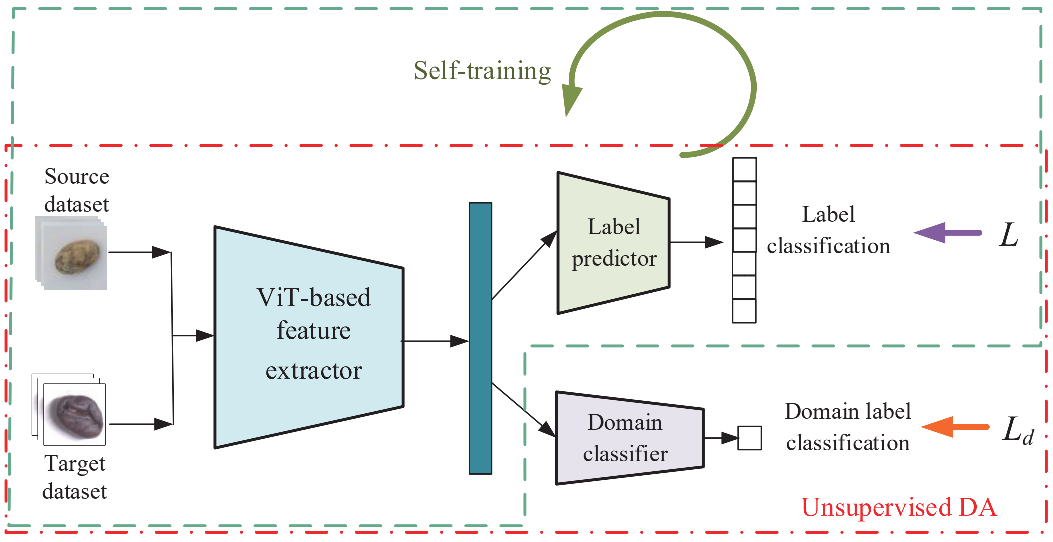


Fig. 7. (Color online) Class-level alignment enhancement using ST.

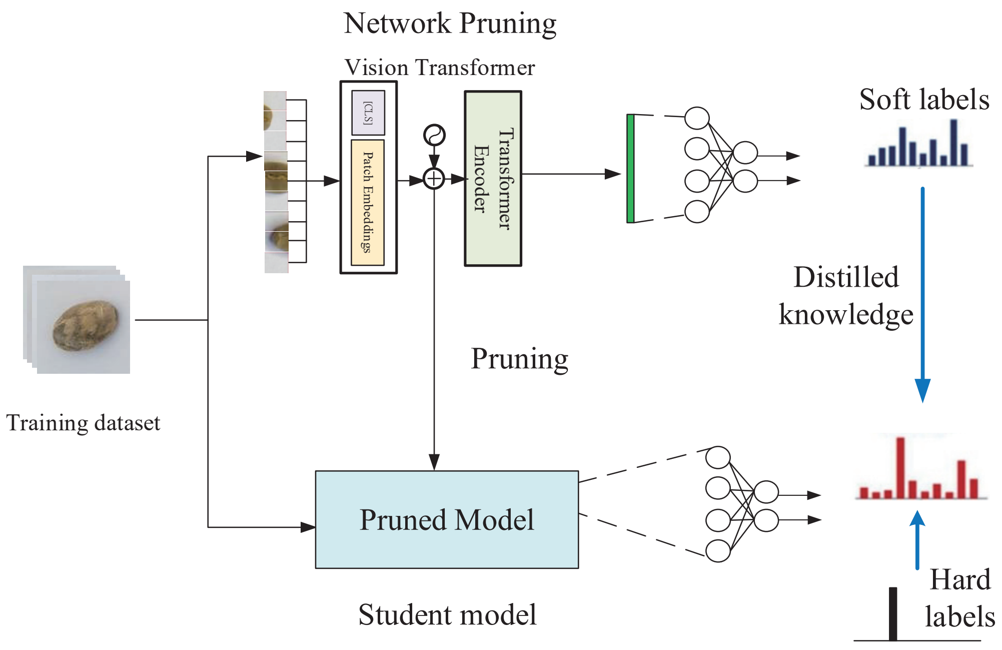


Fig. 8. (Color online) Lightweight processing procedure of the ViT-tiny model.

distributions and implicit knowledge from the teacher model, thereby further improving its classification accuracy.

4. Experimental Results and Discussion

In this study, the development of deep learning models was implemented using PyTorch 2.3.1 and Python 3.12.4. All experiments were conducted on a workstation equipped with an Intel® Core™ i7-9750H processor, 16 GB DDR4 RAM, and an NVIDIA GeForce RTX 2060 GPU to accelerate the computation.

4.1 Coffee bean dataset

In this study, the green coffee bean dataset from Coffee-cobra(23) was utilized as the source domain. It contains a total of 4,626 images of green coffee beans, including 2,149 images of normal beans and 2,477 images of defective beans. The defective beans include various types of defects such as floater beans, black beans, sour beans, and broken beans. To prevent the model from being biased towards defective samples during the training process, an up-sampling technique was applied to augment the number of normal bean images, ensuring a balanced distribution between normal and defective classes and improving the stability of classification. For the target domain, the roasted coffee bean dataset was collected from Aistudio-Baidu,(24) which contains 590 images of roasted beans, with 295 images each for normal and defective beans. Examples of the dataset samples are shown in Fig. 9. In general, green coffee beans exhibit a lighter color, a smooth surface, and relatively uniform shapes. In contrast, roasted

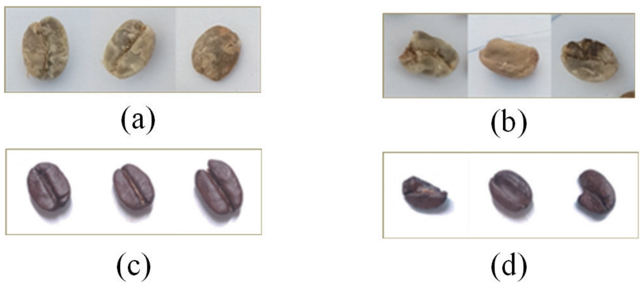


Fig. 9. (Color online) Sample images from the dataset: (a) normal green coffee beans, (b) defective green coffee beans, (c) normal roasted coffee beans, and (d) defective roasted coffee beans.

beans undergo significant changes after the roasting process, resulting in a darker color, oily and cracked surfaces, and highly diverse shapes. These substantial differences in appearance and characteristics between green and roasted beans pose significant challenges for cross-domain coffee bean quality inspection tasks.

To evaluate the classification performance of the proposed model, the green coffee bean dataset was partitioned into 80% for the training set and 20% for the testing set, ensuring a fair assessment of the model on unseen data. In addition, to improve the robustness of the model, various data augmentation techniques were applied during the training process. These techniques include random cropping, rotation, flipping, and color jittering, which effectively increase the diversity of the input data. Such augmentation strategies enhance the model's tolerance to data variations and improve its adaptability to different scenarios and cross-domain datasets.

4.2 Model architecture design and training strategy

To develop an efficient and generalizable coffee bean quality inspection model, we adopt the ViT-tiny model as the backbone network and design an integrated training strategy that combines UDA and ST. This approach is aimed at enhancing the model's adaptability to cross-domain scenarios. The following sections provide a detailed description of the architecture design of the backbone network and the implementation of the proposed training strategy.

4.2.1 Model architecture design

In this study, the ViT-tiny model, which contains a relatively small number of parameters, is adopted as the backbone network for feature extraction. To meet the requirements of the task, the input size of the ViT-tiny model is adjusted to $128 \times 128 \times 3$, with the patch size set to 8×8. Moreover, to facilitate the extraction of more discriminative feature representations, the original classification head of the ViT-tiny model is removed during model initialization. Instead, the feature vectors output from the intermediate layers are directly utilized as the input for the subsequent modules.

The label predictor is designed with five fully connected (FC) layers, each followed by a rectified linear unit (ReLU) activation function. This module takes the 192-dimensional feature vector extracted by the feature extractor as input and performs multiple nonlinear transformations to output a binary classification result for coffee bean quality assessment. Similarly, the domain classifier is also constructed with five FC layers, with each layer followed by batch normalization and a ReLU activation function to stabilize the training process and enhance the nonlinear representation capability. The domain classifier finally outputs a single scalar value, which is used to determine the domain membership of the input feature.

4.2.2 Model training strategy

The proposed model training process in this study is divided into three stages to progressively optimize domain-invariant representation learning and enhance cross-domain adaptability. In the first stage, the feature extractor and the label predictor are trained using the stochastic gradient descent (SGD) optimizer combined with the cross-entropy loss function, enabling the model to effectively learn classification on the source domain data. In the second stage, the domain confusion loss defined in Eq. (4) is introduced to train the domain classifier. At the same time, the feature extractor and the label predictor are fine-tuned using the overall loss function defined in Eq. (2) to further improve feature alignment between the source and target domains. In the third stage, using the confidence threshold defined in Eq. (5), the system selects pseudo-labeled samples with high prediction confidence from the target domain and incorporates them into the training dataset. This step is aimed at improving the classification performance on the target domain. The second and third stages are alternately performed iteratively, allowing the model to progressively strengthen its adaptability and generalization performance on the target domain by jointly leveraging domain feature alignment and the ST mechanism.

$$y^u = \arg\max_k \left(\mathrm{softmax}\left(\frac{z_u}{T} \right)^k \right) \tag{5}$$

Here, z_u denotes the predicted confidence score, and y^u represents the pseudo-label assigned to the target domain sample x^u, and k is the index of the class. In our experiments, the temperature coefficient T is empirically set to 3, and the confidence threshold for selecting high-confidence samples is set to 0.9.

4.3 Experimental evaluation of CUDA-ST for coffee bean quality inspection

To verify the effectiveness of the proposed CUDA-ST framework in cross-domain quality inspection tasks, we conduct generalization experiments using images of green coffee beans as the source domain and images of roasted coffee beans as the target domain. This experimental setting simulates a real-world domain shift scenario to evaluate the model's generalization capability on the target domain.

4.3.1 Effectiveness of CUDA-ST in coffee bean quality inspection

The aim of the experiment is to establish whether the proposed CUDA-ST framework can effectively learn domain-invariant representations through UDA while leveraging the ST mechanism to enhance class consistency. The goal is to address the domain shift problem between the green bean and roasted bean datasets. To determine the optimal weighting configuration, we adjusted the weight ratio between L and L_d in the total loss function defined in Eq. (2), and observed the performance of the model under adversarial learning. The experimental results are summarized in Table 1. When the weight parameter $\lambda = 0$, the model only minimizes the cross-entropy loss of the label predictor. In this case, the model achieves an accuracy of 99.07% on the source domain, but only 66.78% accuracy on the target domain, indicating that the model can correctly classify green beans but suffers from a significant domain shift when applied to roasted beans. As the value of λ increases, the model progressively emphasizes the domain confusion loss of the domain classifier, encouraging the feature extractor to learn domain-invariant representations. When $\lambda = 1.0$, the model maintains a high accuracy of 99.42% on the source domain, while the accuracy on the target domain is significantly improved from 66.78 to 96.95%. Further raising λ to 1.1 or 1.2 results in only minor fluctuations in source-domain accuracy without additional gains on the target domain. Taken together, these observations indicate that $\lambda = 1.0$ provides a well-balanced trade-off between classification fidelity and domain alignment, with the 96.95% target-domain accuracy appearing to approximate the model's practical performance ceiling under the present experimental setting.

4.3.2 Feature distribution visualization analysis

To further validate whether the proposed CUDA-ST framework enables the model to learn domain-invariant representations and align the originally distinct feature distributions between the source and target domains, we conducted a feature distribution visualization analysis. Specifically, we extracted 192-dimensional feature vectors from the output of the last encoder layer of the ViT-tiny model. The t-SNE method was then employed to project the high-

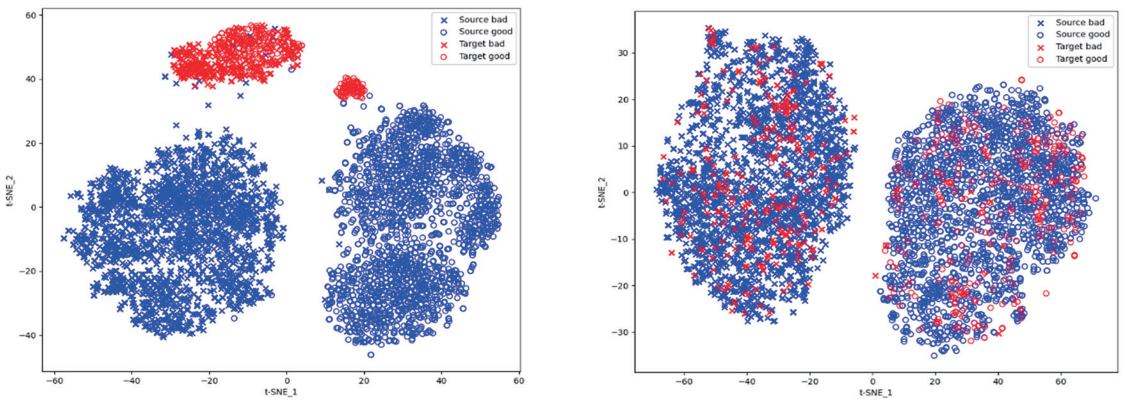
Table 1
Accuracy variation in coffee bean quality inspection with different λ values.

λ	Source data accuracy	Target data accuracy
0.0	99.07	66.78
0.1	99.07	64.92
0.2	98.37	72.03
0.3	98.49	89.66
0.4	98.49	93.56
0.5	99.42	94.92
0.6	98.95	96.95
0.7	99.19	95.59
0.8	99.30	96.78
0.9	99.30	96.10
1.0	99.42	96.95
1.1	99.30	96.95
1.2	99.53	96.95

dimensional feature distributions into a two-dimensional space for visualization, allowing us to observe the changes in feature distributions more intuitively. Figure 10 illustrates the distribution differences of green bean and roasted bean samples in the feature space before and after applying CUDA-ST. As shown in the figure, before applying CUDA-ST, the feature distributions of the two domains were clearly separated, indicating that the model failed to capture domain-shared representations effectively. In contrast, after applying CUDA-ST, the feature distributions of the two domains became highly overlapped in the feature space, with their classwise distributions tending to be more consistent. This result demonstrates that the proposed feature alignment and class consistency enhancement mechanisms in CUDA-ST successfully enabled the label predictor, originally trained only for green bean classification, to correctly recognize the quality of roasted beans as well. Consequently, the model achieved the goal of domain generalization.

4.4 Local interpretable model-agnostic explanations (LIME)-based explainable model analysis

Deep learning models typically consist of millions to billions of parameters, resulting in extremely complex internal mechanisms that are difficult for humans to intuitively understand. This inherent black-box property makes it challenging for users to grasp the decision-making logic of the model, thereby limiting its interpretability in practical applications. To address this issue, we utilized the LIME framework[25] to analyze the model's decision basis for coffee bean quality inspection. Through LIME analysis, we can better understand which visual features the model relies on when making quality predictions. Figures 11(a) and 11(b) respectively present the LIME-based interpretation results for green bean and roasted bean images. The analysis reveals that for high-quality coffee beans, the model primarily focuses on the overall shape and color



(a) (b)

Fig. 10. (Color online) Visualization of feature distributions for green bean and roasted bean samples before and after applying CUDA-ST. Blue circles represent high-quality green beans, red circles represent high-quality roasted beans, blue crosses represent defective green beans, and red crosses represent defective roasted beans. (a) Before applying CUDA-ST, there are significant differences in both feature distributions and classwise distributions between green beans and roasted beans. (b) After applying CUDA-ST, the feature distributions of green beans and roasted beans completely overlap, and the classwise distributions are well aligned.

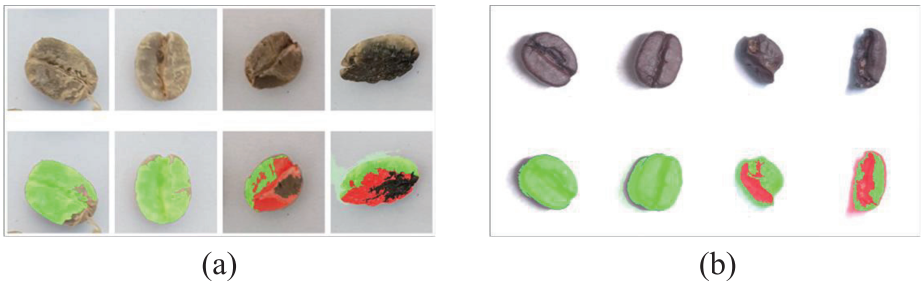


Fig. 11. (Color online) Visualization of model explanations generated by LIME for coffee bean quality inspection. (a) Green coffee bean image. (b) Roasted coffee bean image.

Table 2
Classification accuracy comparison of the model before and after lightweighting.

Pruning ratio	Vol (MB)	Parameters (M)	Fine-tune with KD	
			Source Acc (%)	Target Acc (%)
Base	21.75	5.43	99.42	96.95
0.1	19.48	4.86	97.33	96.10

uniformity of the beans. In contrast, for low-quality coffee beans, the model tends to concentrate on local defects, such as surface blemishes, discoloration, or abnormal shapes.

4.5 Model lightweighting

To further reduce model complexity and improve inference efficiency, we applied structured pruning to the ViT-tiny model to obtain a more compact and practical version for deployment. Considering the strong interlayer dependences inherent in transformer architectures, we adopted the dependency graph (DepGraph) method(26) to analyze the dependence relationships between different layers of the model and to perform groupwise pruning of tightly coupled parameters. The experimental results are summarized in Table 2. When the pruning ratio was set to 0.1, both the model size and the number of parameters were significantly reduced. After applying KD for fine-tuning, the classification accuracy on both the source and target domains was maintained at over 96%. These results demonstrate that even on the already lightweight ViT-tiny model, the combination of DepGraph-based pruning and KD fine-tuning can effectively reduce model complexity while preserving excellent classification performance.

5. Conclusions

In this study, we proposed a deep learning framework for coffee bean quality inspection that adopts the ViT-tiny model as its backbone and integrates UDA with an ST strategy. The framework effectively narrows the feature distribution gap between green and roasted beans while reinforcing class-level alignment, thereby improving generalization in cross-domain settings. To enable deployment on embedded and real-time platforms, a lightweight scheme combining structured pruning and KD was introduced to compress the ViT-tiny model, achieving a favorable balance between classification accuracy and inference speed. Experimental results show that the framework raises roasted bean classification accuracy from 66.78 to 96.95%; even after compression, it maintains an accuracy greater than 96% for both bean types, confirming its practicality for field deployment.

Potential directions for future research include the integration of multispectral and hyperspectral imaging to uncover chemical and structural cues beyond the reach of conventional RGB sensors, which could further enhance the sensitivity and precision of coffee bean defect detection. In addition, deploying the compressed inspection network on resource-constrained edge accelerators and subjecting it to real-time, in-line evaluations on industrial conveyor belts would permit a rigorous characterization of latency, throughput, and long-term stability under production conditions.

References

1 D. Mandal: Appl. Syst. Innov. **1** (2018) 19. https://doi.org/10.3390/asi1020019
2 E. R. Arboleda, A. C. Fajardo, and R. P. Medina: Proc. 2018 IEEE Int. Conf. Innovative Research and Development (IEEE, 2018) 1–5. https://doi.org/10.1109/ICIRD.2018.8376326
3 C.-H. Hsia, Y.-H. Lee, and C.-F. Lai: Appl. Sci. **12** (2022) 10966. https://doi.org/10.3390/app122110966
4 E. Hassan: Neural Comput. Appl. **36** (2024) 9023. https://doi.org/10.1007/s00521-024-09623-z
5 P.-Y. Yang, S.-Y. Jhong, and C.-H. Hsia: Proc. 2021 IEEE Int. Conf. Consumer Electronics-Taiwan (IEEE, 2021) 1–2. https://doi.org/10.1109/ICCE-TW52618.2021.9603134
6 P.-H. Chen, S.-Y. Jhong, and C.-H. Hsia: Proc. 2022 IEEE Int. Conf. Consumer Electronics-Taiwan (IEEE, 2022) 411–412. https://doi.org/10.1109/ICCE-Taiwan55306.2022.9869187
7 L.-Y. Ke, E. Chen, and C.-H. Hsia: Proc. IEEE 6th Int. Conf. Knowledge Innovation and Invention (IEEE, 2023) 430–431. https://doi.org/10.1109/ICKII58656.2023.10332580
8 S. Kumari and P. Singh: Comput. Biol. Med. **170** (2024) 107912. https://doi.org/10.1016/j.compbiomed.2023.107912
9 H.-W. Lin, Y. Tsai, H. J. Lin, C.-H. Yu, and M.-H. Liu: AIMS Math. **9** (2024) 6628. https://doi.org/10.3934/math.2024323
10 S. Cui, S. Wang, J. Zhuo, C. Su, Q. Huang, and Q. Tian: Proc. 2020 IEEE/CVF Conf. Computer Vision and Pattern Recognition. (IEEE, 2020) 12452–12461. https://doi.org/10.1109/CVPR42600.2020.01247
11 C. Xu, M. Zhou, T. Ge, Y. Jiang, and W. Xu: Proc. 2023 IEEE/CVF Conf. Computer Vision and Pattern Recognition. (IEEE, 2023) 10114–10123. https://doi.org/10.1109/CVPR52729.2023.00975
12 E. Zhao, A. Liu, A. Anandkumar, and Y. Yue: Proc. Int. Conf. Artificial Intelligence and Statistics (2021) 3412–3420.
13 F. D. Johansson, D. Sontag, and R. Ranganath: Proc. Int. Conf. Artificial Intelligence and Statistics (2019) 527–536.
14 B. Li, Y. Wang, S. Zhang, D. Li, K. Keutzer, T. Darrell, and H. Zhao: Proc. 2021 IEEE/CVF Conf. Computer Vision and Pattern Recognition (IEEE, 2021) 1104. https://doi.org/10.1109/CVPR46437.2021.00116
15 Z. Pei, Z. Cao, M. Long, and J. Wang: Proc. AAAI Conf. Artif. Intell. **32** (2018) 3934. https://doi.org/10.1609/aaai.v32i1.11767
16 S. Jung, J. Lee, N. Kim, A. Shaban, B. Boots, and J. Choo: Proc. 2023 IEEE/CVF Int. Conf. Computer Vision (IEEE, 2023) 19014–19025. https://doi.org/10.1109/ICCV51070.2023.01747
17 A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin: Adv. Neural Inf. Process. Syst. **30** (2017) 5998.
18 A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby: arXiv (2020). https://arxiv.org/abs/2010.11929
19 Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky: J. Mach. Learn. Res. **17** (2016) 1.
20 S. M. Xie, A. Kumar, R. Jones, F. Khani, T. Ma, and P. Liang: Proc. 9th Int. Conf. Learning Representations (ICLR, 2021) 1–13.
21 S. Srinivas and R. V. Babu: Proc. British Machine Vision Conf. (2015) 31.1–31.12.
22 P. Luo, Z. Zhu, Z. Liu, X. Wang, and X. Tang: Proc. AAAI Conf. Artif. Intell. **30** (2016) 3560.
23 Edgeryders: https://github.com/edgeryders/coffee-cobra (accessed March 2021).

24 Baidu AI Studio: https://aistudio.baidu.com/datasetdetail/150459 (accessed June 2022).
25 J. Hu, K. Zhu, S. Cheng, N. M. Kovalchuk, A. Soulsby, M. J. H. Simmons, O. K. Matar, and R. Arcucci: Chem. Eng. J. **481** (2024) 148465.
26 G. Fang, X. Ma, M. Song, M. B. Mi, and X. Wang: Proc. 2023 IEEE/CVF Conf. Computer Vision and Pattern Recognition (IEEE, 2023) 16091–16101. https://doi.org/10.1109/CVPR52729.2023.01544