2024 IEEE First International Conference on Artificial Intelligence for Medicine, Health and Care (AIMHC) | 979-8-3503-7198-7/24/$31.00 ©2024 IEEE | DOI: 10.1109/AIMHC59811.2024.00040

Deploying a Sustainable Deep Learning Pipeline for Poison Ivy Image Classification

Wonjun Park*
Computer Science and Engineering, Konkuk University
Seoul, Republic of Korea
kuwjjgjk@konkuk.ac.kr

Sumin Cho†
Human Centered Artificial Intelligence, Sangmyung University
Seoul, Republic of Korea
201910837@sangmyung.kr

Subin Kim‡
Computer Software Engineering, Soonchunhyang University
Asan, Republic of Korea
kim1q345789@sch.ac.kr

Jiyeon Lee§
Computer Science and Engineering, Kyonggi University
Suwon, Republic of Korea
dkdljy0112@kyonggi.ac.kr

Jack Mahedy¶
Computer and Information Technology
Purdue University
West Lafayette, IN, USA
jmahedy@purdue.edu

Nebey Gebresalssie‖
Computer and Information Technology
Purdue University
West Lafayette, IN, USA
ngebresl@purdue.edu

Minji Lee**
Computer and Information Technology
Purdue University
West Lafayette, IN, USA
lee3450@purdue.edu

***Abstract*—Poison ivy, causing irritated skin in humans due to the urushiol oil it contains, is a well-known plant mainly distinguishable by the traits of its leaves. Nevertheless, according to the statistical data, a significant number of people are still affected by the plant. This research postulates that this notorious result stemmed from the scarce analysis about the plant. The paper proposes an automated pipeline, functioning Continuous Integration, Deployment, and Training on Kubeflow so that the harmful impact of the plant decreases, leading the public health outcomes to enhance. The pipeline contains a dataset composed of both a set of leaves and a stem images which are manually collected from a local farm and a devised model which considers both features to classify Western poison ivy. With the pipeline, the research contributes to decrease the number of suffering people from the plant as promptly reacting to seasonal changes.**

***Index Terms*—Plant Image Classification, Deep Learning, Transfer Learning, Machine Learning Operations**

I. Introduction

Poison ivy, widely found in North America and East Asia, poses significant health risks due to its urushiol oil, which causes reddish rashes on human skin. Despite traditional identification methods focusing on its distinctive three broad, tear-shaped leaves, the American Skin Association [1] reports about fifty million annual harm cases, with 10-15% being severe, requiring substantial medical intervention. This occurs despite efforts by the Centers for Disease Control and Prevention (CDC), which allocated twenty-five million dollars in 2022 to manage toxic substances and environmental public health [2]. This paper tackles the plant's stem, especially in Western poison ivy, a sub-species that poses risks through stems easily broken by animals, transferring toxins to humans. Concurrently, advancements in Machine Learning (ML), particularly Convolutional Neural Networks (CNN), have revolutionized image classification. This research incorporates both leaf and stem images in the classification process, utilizing a dataset manually collected from Jackson Township, IN, and comparing model scores to validate the dataset. Additionally, to address the fickle plants depending on time passes, this work applies Machine Learning Operations (MLOps), a methodology combining Development and Operations (DevOps) with ML, using Kubeflow to streamline data preparation, training, testing, and deployment.

In conclusion, the paper contributes to the following parts; 1) diversifying available datasets with the dataset compounding stem features, 2) introducing a novel model architecture in plant classification, 3) implementing an automated ML workflow pipeline that enables to react to seasonal changes rapidly, and 4) ultimately aiming to reduce the number of people who suffer from Poison ivy.

II. Literature Review

Extensively researched in the 1900s, Poison ivy is known for causing skin irritation due to urushiol oil, affecting 50-75% of the U.S. population according to Yesul [3]. The CDC [4] describes it as having a trio of green, shiny leaves with smooth or slightly notched edges. Western poison ivy [5] differs as a low shrub, unlike the climbing vine of Eastern poison ivy. Its leaves turn red or orange in fall and disappear in winter and early spring, necessitating alternative identification methods.

Many CNN variants have been proposed in plant image classification. Noor et al. [6] combined SVM and CNN with transformer models to classify Arabian Peninsula plants, and Gui [7] used a similar approach for soybean disease detection.

979-8-3503-7198-7/24/$31.00 ©2024 IEEE
DOI 10.1109/AIMHC59811.2024.00040

These studies primarily focus on leaf features. However, Zahan et al. [8] classified toxic mushrooms with their whole pictures, inspiring in that our research examines both leaves and stems.

The field of MLOps has evolved from DevOps, which integrates the work continuously and automatically. MLOps extends this integration to ML workflows with Continuous Integration (CI), Continuous Deployment (CD), and Continuous Training (CT). Google [9] categorizes pipelines based on automation levels, and virtualization or containerization assists various stakeholders in cohesive yet independent development and deployment. Kreuzberger et al. [10] highlighted the significance of MLOps components, workflows, and roles, underlining the need for pipelines. Nevertheless, practical examples in academia are so rare that the instant usages of proposed ML models still suffer from obstacles.

III. Methodology

A. Dataset

In previous research on plant classification, images of leaves from plants in their grown state were predominantly used [11]–[13]. This study collected clear data of leaves and stems from Poison Ivy and other plants. The data was gathered from March to June 2023 at a private farm (112,785m²) in Jackson Township, Indiana, and within the Purdue University campus. The usage of diverse cameras is expected to positively influence the reproducibility of our future models. Furthermore, to independently train the model for classifying leaves and stems, Semantic Segmentation was conducted to separate leaves and stems from the collected overall plant images. The Segment Anything Model (SAM), developed by Meta AI and capable of zero-shot generalization, is applied to achieve this goal [14]. We preprocessed two separate images for leaves and stem, from a plant image, for which we set up the appropriate labeling. In conclusion, a dataset consisted of 1028 images each of leave, stem, and overall plant for Poison Ivy and others.

B. Model Architecture

This study develops a robust Poison ivy classification model, addressing limitations in traditional plant classification relying solely on leaves. The proposed model architecture includes:

(1) Generation and Storage of Classification Models: Two models are trained and stored. The first model is a binary classifier distinguishing Poison ivy leaves from those of other plants, using leaf images for training. The second model is another binary classifier focusing on Poison ivy stems, utilizing stem images for training. (2) Utilization of Classification Models: The trained models are employed to predict probabilities for test data. (3) Calculation of Leaves and Stem Occupancy: Leaf and stem occupancy is determined by calculating pixels occupied by each component in the entire image. (4) Calculation of Weighted Averages: Using occupancies as weights, a weighted average is applied to prediction probabilities, generating a final prediction for Poison ivy based on both leaves and stems in the test data.

Performance evaluation metrics include accuracy, precision, recall, and F1 score, derived from the confusion matrix. [15] This provides a comprehensive analysis of the model's performance and an objective assessment of experimental results.

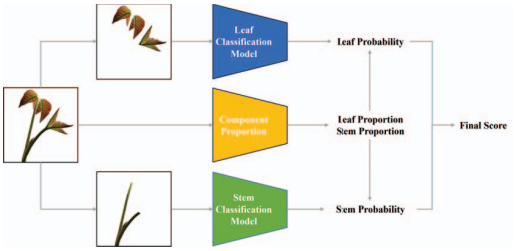


Fig. 1: Model Architecture.

C. Deep Learning Pipeline

The components of the pipeline were containerized by Docker images, and each image was pushed and managed in the Azure Container Registry (ACR). The containerization allowed each stage to be separately managed and executed. The pipeline consisted of three stages: data preparation, model training, and model registration. In the stage of data preparation, a dataset consisting of leaves and stem images was prepared to train the model. In model training, models were trained by the prepared data. In the final, model registration, the trained models were rolled into the Azure Machine Learning Workspace (AML).

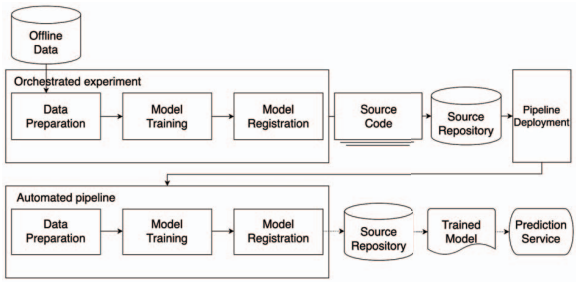


Fig. 2: Automated Deep Learning Pipeline.

IV. Implementation

A. Dataset Quality Inspection

Before model training, ensuring the high quality of data is imperative. High-quality data not only enhances model performance but also ensures the reliability of results, which holds significance for future research on Poison ivy and plant stems. Inter-rater consistency was evaluated using Fleiss' Kappa among five annotators, yielding a substantial agreement value of 0.9, indicating consistent labeling based on shared criteria [16]. Data completeness was achieved by addressing missing images and removing duplicates. The final dataset comprised 1028 uniquely identified Poison ivy leaves and stems. The same process was applied to other plants to ensure

dataset consistency. Visualization results are presented in Fig. 3. After confirming data quality, we trained a Scratch CNN and observed stable convergence during training.

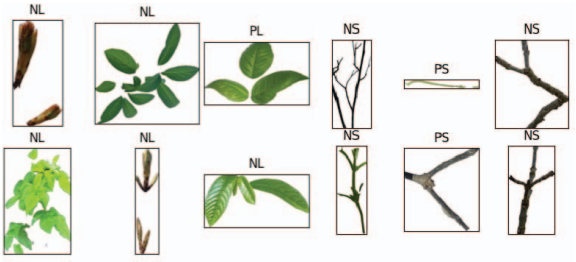


Fig. 3: Poison ivy and non-poison ivy leaves and stem dataset.

B. Experiments of the model performance

The Scratch CNN model faced problems in effectively capturing high-level features from both leaves and stems. The paper addressed them by applying data augmentation and fine-tuning advanced CNNs pre-trained on a vast dataset.

Experimental Design A dataset of 1028 images per class was divided into 928 for training and 100 for final evaluation. The 928 training images were split into an 8:1:1 ratio for training, validation, and testing. Image preprocessing used TensorFlow's ImageDataGenerator [17] to address data shortage, introducing variability in natural environmental conditions. The experiment utilized the Keras applications [18], offering weights for over 40 CNNs pre-trained on the ImageNet dataset [19]. Four criteria, including top-1 and top-5 accuracy, shallowness, and fewer parameters, guided the selection of CNN models, as outlined in the official Keras documentation [20]. Shallow models with fewer parameters were chosen to bring diversity in outcomes, given that models with high top-1 and top-5 accuracy tend to be deep and have many parameter counts. Pretrained ImageNet feature extractors were frozen, and only classifiers were retrained. Control variables like Adam optimizer, Binary Cross-Entropy loss, and hyperparameters were consistent for uniform evaluation.

Experiment result The performance of 31 selected models, based on four criteria, was evaluated on leaf and stem data Table I. In leaf and stem model, relatively shallow models with fewer parameters exhibited outstanding performance across Accuracy, Loss, Precision, Recall, and F1-Score. These results suggest that simpler models were less prone to overfitting due to the limited dataset size. For the Leaf model, the Top 3 models achieved an average Accuracy of 0.93, with Precision (NL/PL) at 0.933/0.924, Recall (NL/PL) at 0.915/0.939, and F1-Score (NL/PL) at 0.924/0.932, outperforming the Scratch CNN by more than 0.5. Despite limited visible features in stems, the Top 3 models achieved an average Accuracy of 0.844. Precision (NL/PL) was 0.855/0.834, Recall (NL/PL) was 0.827/0.86, and F1-Score (NL/PL) was 0.841/0.846. While these scores were lower than those of leaves, they indicated the potential for plant classification using stems. Table III presents the final scores for each of the Top 3 models.

TABLE I: Leaf and Stem Model Test Performance

	Leaf				Stem			
Model	**Accuracy**	**Precision (NL/PL)**	**Recall (NL/PL)**	**F1-Score (NL/PL)**	**Accuracy**	**Precision (NS/PS)**	**Recall (NS/PS)**	**F1-Score (NS/PS)**
ConvNeXtBase	0.91	0.95 / 0.89	0.86 / 0.96	0.90 / 0.92	0.81	0.79 / 0.82	0.83 / 0.78	0.81 / 0.80
ConvNeXtLarge	0.92	0.92 / 0.92	0.91 / 0.93	0.91 / 0.92	0.79	0.79 / 0.79	0.80 / 0.78	0.79 / 0.79
ConvNeXtXLarge	0.92	0.93 / 0.90	0.91 / 0.94	0.92 / 0.92	0.78	0.82 / 0.75	0.72 / 0.84	0.77 / 0.79
DenseNet121	0.93	0.94 / 0.92	0.91 / 0.95	0.92 / 0.94	0.84	0.85 / 0.84	0.84 / 0.85	0.84 / 0.84
DenseNet201	0.94	0.93 / 0.93	0.93 / 0.93	0.93 / 0.93	0.86	0.86 / 0.86	0.86 / 0.86	0.86 / 0.86
EfficientNetB0	0.47	0.47 / 1.00	1.00 / 0.00	0.64 / 0.00	0.50	1.00 / 0.50	0.00 / 1.00	0.00 / 0.67
EfficientNetB1	0.63	0.74 / 0.60	0.32 / 0.90	0.45 / 0.72	0.52	0.70 / 0.51	0.08 / 0.97	0.14 / 0.67
EfficientNetB2	0.63	0.73 / 0.60	0.33 / 0.89	0.46 / 0.72	0.58	0.61 / 0.56	0.42 / 0.73	0.50 / 0.63
EfficientNetB3	0.63	0.77 / 0.60	0.31 / 0.92	0.44 / 0.73	0.66	0.65 / 0.67	0.69 / 0.62	0.67 / 0.64
EfficientNetB4	0.62	0.81 / 0.59	0.25 / 0.95	0.39 / 0.73	0.60	0.70 / 0.57	0.35 / 0.85	0.47 / 0.68
EfficientNetB5	0.74	0.76 / 0.72	0.63 / 0.83	0.69 / 0.77	0.55	0.54 / 0.56	0.67 / 0.43	0.60 / 0.49
EfficientNetB6	0.67	0.63 / 0.73	0.75 / 0.61	0.68 / 0.66	0.59	0.59 / 0.58	0.57 / 0.60	0.58 / 0.59
EfficientNetB7	0.76	0.86 / 0.72	0.59 / 0.92	0.70 / 0.80	0.62	0.69 / 0.59	0.45 / 0.79	0.55 / 0.68
EfficientNetV2B0	0.70	0.71 / 0.69	0.61 / 0.78	0.65 / 0.73	0.57	0.74 / 0.54	0.22 / 0.92	0.33 / 0.68
EfficientNetV2B1	0.63	0.73 / 0.61	0.34 / 0.89	0.47 / 0.72	0.50	0.50 / 0.00	0.99 / 0.00	0.66 / 0.00
EfficientNetV2B2	0.70	0.70 / 0.70	0.62 / 0.77	0.66 / 0.73	0.55	0.68 / 0.53	0.20 / 0.90	0.31 / 0.67
EfficientNetV2B3	0.73	0.85 / 0.68	0.52 / 0.92	0.64 / 0.78	0.60	0.65 / 0.57	0.42 / 0.77	0.51 / 0.66
EfficientNetV2L	0.69	0.63 / 0.77	0.79 / 0.60	0.70 / 0.67	0.68	0.74 / 0.65	0.57 / 0.79	0.64 / 0.72
EfficientNetV2M	0.48	0.47 / 0.63	0.97 / 0.05	0.63 / 0.09	0.57	0.58 / 0.55	0.46 / 0.67	0.51 / 0.60
EfficientNetV2S	0.81	0.81 / 0.81	0.77 / 0.84	0.79 / 0.82	0.69	0.66 / 0.74	0.78 / 0.60	0.72 / 0.66
InceptionV3	0.89	0.88 / 0.91	0.90 / 0.89	0.89 / 0.90	0.74	0.76 / 0.73	0.71 / 0.77	0.73 / 0.75
MobileNet	0.93	0.93 / 0.92	0.91 / 0.94	0.92 / 0.93	0.77	0.78 / 0.76	0.74 / 0.80	0.76 / 0.77
MobileNetV2	0.91	0.91 / 0.91	0.91 / 0.91	0.91 / 0.91	0.80	0.81 / 0.78	0.77 / 0.82	0.79 / 0.80
NASNetLarge	0.89	0.90 / 0.88	0.86 / 0.92	0.88 / 0.90	0.79	0.82 / 0.76	0.73 / 0.84	0.77 / 0.79
NASNetMobile	0.89	0.86 / 0.92	0.92 / 0.87	0.89 / 0.90	0.83	0.86 / 0.80	0.78 / 0.87	0.82 / 0.84
ResNet152V2	0.89	0.91 / 0.87	0.84 / 0.93	0.87 / 0.90	0.77	0.79 / 0.76	0.74 / 0.81	0.77 / 0.78
ResNet50	0.74	0.73 / 0.75	0.70 / 0.78	0.72 / 0.76	0.68	0.65 / 0.73	0.78 / 0.58	0.71 / 0.65
ResNet50V2	0.90	0.90 / 0.89	0.87 / 0.92	0.89 / 0.91	0.76	0.78 / 0.75	0.74 / 0.78	0.76 / 0.77
VGG16	0.91	0.90 / 0.92	0.91 / 0.91	0.90 / 0.91	0.79	0.76 / 0.82	0.84 / 0.74	0.80 / 0.78
VGG19	0.89	0.87 / 0.91	0.90 / 0.88	0.88 / 0.89	0.79	0.81 / 0.77	0.75 / 0.83	0.78 / 0.80
Xception	0.90	0.91 / 0.90	0.89 / 0.92	0.90 / 0.91	0.81	0.85 / 0.78	0.76 / 0.86	0.80 / 0.82

TABLE II: Final Performance Score

Model		**Evaluation Matrix**			
Leaf	***Stem***	***Accuracy***	***Precision***	***Recall***	***F1 Score***
DenseNet121	DenseNet121	0.815	0.805	0.83	0.817
	DenseNet201	0.815	0.800	0.84	0.819
	NASNetMobile	0.82	0.826	0.81	0.818
DenseNet201	DenseNet121	0.815	0.805	0.83	0.817
	DenseNet201	0.805	0.790	0.83	0.809
	NASNetMobile	0.82	0.826	0.81	0.818
MobileNet	DenseNet121	0.785	0.782	0.79	0.786
	DenseNet201	0.775	0.761	0.8	0.780
	NASNetMobile	0.795	0.804	0.78	0.791

C. Running The Pipeline On Kubeflow

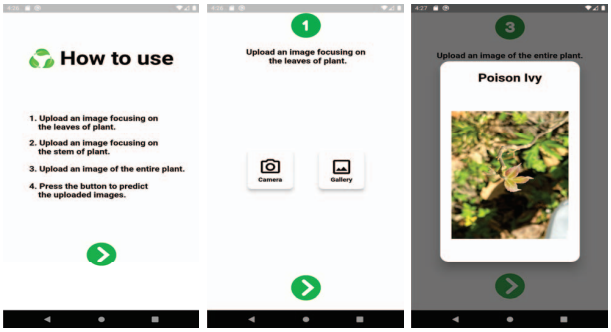
Kubeflow was deployed and executed on Azure Kubernetes Service (AKS) to build DL pipeline. Kubeflow 1.7 version was used. Each component of the pipeline was containerized independently by Docker images. Dockerfiles were defined with a base image, TensorFlow 2.12.0. Further packages were appended which were required by each component. The constructed images were pushed to ACR.

The pipeline was defined as a series of individual components, each with its metadata and interface. The first component, data preparation, involved the arguments of the dataset. In the second component, model training, the arguments of epochs, batch size, and learning rate were utilized to train the model, and the output path was specified to save the model. Model registration, for the last, included the information of the service principal to access Azure resources [21].

To run the pipeline, it was compiled by Kubeflow Pipelines SDK and was enrolled as user interface [22]. The containers were pulled from ACR while the pipeline executed. Consequently, two classification models for leaves and stem were registered in AML.

D. Application

The research developed the mobile application which classifies Western poison ivy from a plants image. The UI was built by Flutter. FastAPI is adopted to build a RestAPI server. In Fig. 4(a), the descriptions of the User's guide were presented. Plant images, focusing on the leaves, the stem, and the entire plant, are required for predictions. As shown Fig. 4(b), Once the images are uploaded to the server, the server calculated and responded the result with the registered model. The application displayed the output as shown in Fig. 4(c). If the model



(a) Instructions (b) Upload page (c) Predict page

Fig. 4: Application pages

continuously deployed as the DL pipeline, the qualified service of classifying Western poison ivy can be provided without suspending services.

V. Conclusion

This research paper rigorously explores the development of a sustainable and autonomous MLOps pipeline, utilizing ML models that classify Western poison ivy. With the concept of MLOps, the study pioneers an approach where each stage of the pipeline—from data preparation to deploying—is autonomously managed and executed through Kubeflow Pipeline and Docker. The research involved compiling a dataset from the local farm, which includes both leaves and stems of the plant, with each object segmented using the SAM model. Despite achieving significant advancements, the research remains areas requiring further exploration. One key challenge is the manual preprocessing of the dataset, attributed to the limitations of SAM in natural settings. Fine-tuning the SAM model could enhance the automation process from an MLOps perspective. As a result, this research not only contributes to the field of artificial intelligence in natural plant recognition but also holds the promise of reducing the incidence and associated costs of poison ivy afflictions.

Acknowledgment

This research was supported by the MSIT (Ministry of Science and ICT), South Korea, under the National Program for Excellence in SW, supervised by the IITP (Institute for Information & communication Technology Planning & evaluation) in 2023 (No. 2018-0-00213, 2021-0-01399, 2021-0-01393, 2019-0-01880). We would like to acknowledge the support and resources facilitated the completion of this work, including the faculty and staff of Computer and Information Technology (CNIT) in Purdue University for their assistance and guidance throughout the project.

References

[1] American Skin Association. ”Poison Ivy, Sumac and Oak.” AmericanSkin.org. https://www.americanskin.org/resource/poisonivy.php (accessed May. 30, 2023).

[2] Centers for Disease Control and Prevention. ”Centers for Disease Control and Prevention Fiscal Year 2022 Grants Summary Profile Report,” 2022. [Online]. Available: https://fundingprofiles.cdc.gov/Report_Docs/PDFDocs/Rpt2022/All-US-States-2022.pdf (accessed Sep. 12, 2023).

[3] Y. Kim, A. Flamm, M. A. Elsohly, D. H. Kaplan, R. J. Hage, C. P. Jr. Hamann and J. G. Jr. Marks, ”Poison , Ivy, Oak, and Sumac Dermatitis: What Is Known and What Is New?,” Dermatitis : contact, atopic, occupational, drug, 30(3), 183–190, doi: https://doi.org/10.1097/DER.0000000000000472

[4] Centers for Disease Control and Prevention. ”Poisonous Plants.” CDC.gov. https://www.cdc.gov/niosh/topics/plants/default.html (accessed Oct. 13, 2023).

[5] USDA. ”Western Poison-ivy.” USDA.gov. https://www.fs.usda.gov/wildflowers/plant-of-the-week/toxicodendron_rydbergii.shtml (accessed Oct. 13, 2023).

[6] Noor, Talal H., Ayman Noor, and Mahmoud Elmezain. ”Poisonous Plants Species Prediction Using a Convolutional Neural Network and Support Vector Machine Hybrid Model.” Electronics 11.22 (2022): 3690.

[7] Gui, Jiangsheng, et al. ”Grading method of soybean mosaic disease based on hyperspectral imaging technology.” Information Processing in Agriculture 8.3 (2021): 380-385.

[8] Zahan, Nusrat, et al. ”A deep learning-based approach for edible, inedible and poisonous mushroom classification.” 2021 International Conference on Information and Communication Technology for Sustainable Development (ICICT4SD). IEEE, 2021.

[9] Google. “MLOps: Continuous delivery and automation pipelines in machine learning.” cloud.google.com. https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning (accessed Oct. 1, 2023).

[10] Kreuzberger, Dominik, Niklas Kühl, and Sebastian Hirschl. ”Machine learning operations (mlops): Overview, definition, and architecture.” IEEE Access (2023).

[11] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk and D. Stefanovic, ”Deep neural networks based recognition of plant diseases by leaf image classification,” Computational Intelligence and Neuroscience, vol. 2016, May. 2016, doi: https://doi.org/10.1155/2016/3289801.

[12] S. G. Wu, F. S. Bao, E. Y. Xu, Y. -X. Wang, Y. -F. Chang and Q. -L. Xiang, ”A Leaf Recognition Algorithm for Plant Classification Using Probabilistic Neural Network,” presented at the 2007 IEEE International Symposium on Signal Processing and Information Technology, Giza, Egypt, Dec. 15-18, 2007.

[13] S. H. Lee, C. S. Chan, S. J. Mayo and P. Remagnino, ”How deep learning extracts and learns leaf features for plant classification,” Pattern Recognition, vol. 71, pp. 1-13, Nov. 2017, doi: https://doi.org/10.1016/j.patcog.2017.05.015.

[14] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W. Lo, P. Dollár and R. Girshick, ”Segment anything,” arXiv preprint arXiv:2304.02643, Apr. 5, 2023.

[15] Pan, Haihong, et al. ”A new image recognition and classification method combining transfer learning algorithm and mobilenet model for welding defects.” IEEE Access 8 (2020): 119951-119960

[16] andis, J. Richard, and Gary G. Koch., “The Measurement of Observer Agreement for Categorical Data,” presented at the Wiley, International Biometric Society, 1977.

[17] TensorFlow. “tf.keras.preprocessing.image.ImageDataGenerator.” tensorflow.org. https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/Im (aaccessed Sep. 1, 2023).

[18] keras. “Keras Applications.” keras.io. https://keras.io/api/applications/ (aaccessed Sep. 1, 2023).

[19] J. Deng, W. Dong, R. Socher, L. -J. Li, Kai Li and Li Fei-Fei, ”ImageNet: A large-scale hierarchical image database,” *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA, 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

[20] keras. “Keras Applications.” keras.io. https://keras.io/api/applications/ (aaccessed Sep. 1, 2023).

[21] The Kubeflow Authors. ”Component Specification.” Kubeflow.org. https://www.kubeflow.org/docs/components/pipelines/v1/reference/component-spec/ (accessed Oct. 1, 2023).

[22] The Kubeflow Authors. ”Quick Start.” Kubeflow.org. https://www.kubeflow.org/docs/components/pipelines/v1/overview/quickstart/ (accessed Oct. 1, 2023).