Automated image data segmentation via self-iterative training

Jinyoon Kim
School of Science and Engineering
Penn State University Harrisburg
Middletown, PA, USA
juk481@psu.edu

Tianjie Chen
School of Science and Engineering
Penn State University Harrisburg
Middletown, PA, USA
tvc5586@psu.edu

Md Faisal Kabir
School of Science and Engineering
Penn State University Harrisburg
Middletown, PA, USA
mpk5904@psu.edu

***Abstract*—Image segmentation in computer vision, especially in tasks related to object detection, remains a significant challenge in achieving both accuracy and efficiency. This article introduces an automated segmentation algorithm that employs a synergistic approach, combining active learning techniques with YOLOv8 across various datasets and conditions. Additionally, this method is applied to OneFormer-a vision transformer model-to experiment with its validation across wider domains. The objective of this study is to automate the image segmentation process and enhance the performance of the object detection model. The proposed YOLOv8 Automated Image Segmentation (YOLOv8-AIS) algorithm operates by leveraging an active learning mechanism where it automatically labels a portion of the unlabeled data based on uncertainty scores, then integrating this data into the training dataset for the iterative training process. This approach aims to reduce the need for manual annotation, which is inefficient due to the reliance on manual effort. The effectiveness of the method was evaluated using three datasets: the tomato and apple leaf datasets from the Plant Village collection, and the HAM10000 skin lesion dataset, under various conditions.**

***Index Terms*—Automated Data Labeling, Active Learning, Semi-Supervised Learning, Instance Segmentation, Object Detection**

I. Introduction

Machine learning algorithms can be categorized as either supervised or unsupervised learning, depending on the task of the object [1]. In the case of supervised learning, the availability of extensive, high-quality labeled datasets plays a pivotal role in determining the performance of models. However, obtaining such extensive labeled data is challenging because manual annotation is a expensive and labor-intensive procedure. To mitigate these issues, various approaches have been developed, including Active Learning [2]–[4], Semi-Supervised Learning [5]–[7], and Automated Data Labeling [8]–[10].

Active Learning [11] refers to the use of learning algorithms that proactively choose which data to learn from to maximize the usefulness gained from each labeled instance. However, this method encounters obstacles such as inconsistent performance, sensitivity to hyperparameters, and potential difficulties with certain datasets and tasks.

On the other hand, Semi-Supervised Learning [12] presents a notable benefit in scenarios where labeled data is limited by utilizing both labeled and unlabeled data. It takes advantage of the abundance of unlabeled data, thus reducing the reliance on manual image segmentation.

Automated Data Labeling [13] aims to either fully or partially automate the image segmentation process to significantly reduce the time needed for manual annotation. An example of such a system is the Amazon Automated Data Labeling [10], which simplifies the creation and modification of training datasets. It underscores the drive to streamline the segmentation process in complex tasks like instance segmentation [14], significantly increasing the overall effectiveness of computer vision models.

The objective of this study is to incorporate the state-of-the-art YOLOv8 model [15] with an automated image segmentation algorithm, and to apply it to various datasets such as the plant leaves diseases dataset of PlantVillage and the skin lesion dataset from HAM10000 patients. Additionally, we applied a vision transformer model, OneFormer [16], to compare the performance of our algorithm on a different computer vision architecture. This tailored adaptation of the YOLOv8 model magnifies its capacity to discern and identify intricate patterns, greatly enhancing the efficacy of auto-segmentation algorithms.

To further strengthening our approach, we have synthesized various aspects of computer vision techniques into a single framework, resulting in the embedding of the Automated Image Segmentation (AIS) algorithm. Our iterative training algorithm yields a significant increase in the performance of the auto-segmentation process. This not only underscores the synergy of current technologies but also charts an interesting trajectory for future research in semi-supervised learning and computer vision.

The notable contributions of this study are as follows:

- Utilized a comprehensive dataset of over 20,000 plant samples for algorithmic training.
- Applied advanced object detection using YOLOv8, enhancing segmentation and accelerating training.
- Implemented a classification correction mechanism alongside an uncertainty data selection algorithm, optimizing model accuracy.
- Introduced semi-supervised learning, leveraging both labeled and unlabeled data for better model performance.

- Incorporated active learning to refine auto-segmentation and adjust model weights, ensuring adaptive learning from new data.
- The auto-segmentation system serves dual purposes: data annotation and model training, which boosts the accuracy and resilience of object detection tasks.
- Additional HAM10000 skin lesion dataset was used to test the validation of its performance across different datasets that present more challenging features for segmentation and are more severely skewed in class distribution.
- Evaluated the system's performance by exploring its capacity to utilize the minimum labeled dataset for its tasks and identified its limitations. Additionally, we proposed a new method to mitigate these problems.
- In addition to the YOLOv8 model, we also experimented the performance of a vision transformer model within our system.

The paper is divided into 5 sections. Following the introduction section, Section 2 discusses related works on semi-supervised learning, active learning, and the auto-segmentation system, offering an understanding of their impact on this research. Section 3 outlines the specific materials and methods employed, including the dataset used for the experiment and a detailed description of the algorithm's step-by-step process. The experimental outcomes and the environmental configuration are presented and analyzed in Section 4. Finally, Section 5 concludes the paper, summarizing the study and suggesting areas for future research improvement.

II. Related Works

This section embarks on a journey through various research studies and approaches that serve as the foundation for the proposed YOLOv8-AIS model. By understanding the valuable insights offered by these works, one can better appreciate the context and rationale behind the design choices in YOLOv8-AIS. The surveyed literature ranges from deep active learning techniques and semi-supervised learning to advanced object detection and weakly-supervised data creation methods.

The comprehensive paper, "A Comparative Survey of Deep Active Learning" [3], shapes the foundation of this research with pivotal DAL methods. Pseudo segmentation and uncertainty sampling, central techniques in our model, are further enriched by "Deep Active Learning for Named Entity Recognition" [2], despite not adopting its CNN-CNN-LSTM structure. In YOLOv8-AIS, these pseudo labels, produced from predictions, become interim ground truths, while uncertainty sampling zeroes in on complex cases. Seamlessly integrating with YOLOv8's instance segmentation, our model stands at the confluence of deep and active learning, enhancing performance and label accuracy. This unique blend elevates the efficiency of automated segmentation, marking a significant leap in machine learning.

The pivotal studies "Semi-supervised Active Learning for Instance Segmentation via Scoring Predictions" [5] and "Learning from Noisy Large-Scale Datasets with Minimal Supervision" [6] have steered advancements in semi-supervised learning. [5] introduced a groundbreaking active learning framework that synergizes initial labeled data with self-labeled data for refined segmentation. Meanwhile, [6] emphasized useful insights for the semi-supervised learning, even though its core focus was on noisy annotations that were not concerned in this research. Building upon these foundations, the YOLOv8-AIS algorithm assimilates the robust object detection of YOLOv8 and the spirit of [5]. Unlike [6]'s dual network approach, YOLOv8-AIS champions a singular advanced model. Its active learning loop perpetually hones the model and improves labeled data quality, setting a novel paradigm in processing expansive datasets.

Snorkel, introduced in [9], facilitates rapid training data creation via weak supervision. By allowing users to design segmentation functions that generate noisy labels, Snorkel consolidates them into probabilistic labels through a generative model, reducing manual annotation efforts for large datasets. Conversely, our method taps into semi-supervised learning, capitalizing on the YOLOv8 model and the abundance of unlabeled data for training. This direct approach, unlike Snorkel's reliance on human expertise for multiple segmentation functions, is more straightforward and demands less human intervention. With the advanced learning capabilities of the recent object detection models and minimal labeled data, our method mostly offers precise results, promising a balanced solution in terms of human effort, data utility, and model performance, potentially edging out frameworks like Snorkel in specific contexts.

OneFormer [16], experimented alongside the YOLOv8 object detection model in this research, introduces a novel approach to universal image segmentation. It employs a single transformer model that achieves state-of-the-art performance across semantic, instance, and panoptic segmentation tasks through a single training process. The framework's task-conditioned joint training strategy incorporates a task token, dynamically adjusting to the specific segmentation task at hand. Additionally, it utilizes a query-text contrastive loss to improve task and class distinction. This innovative design significantly reduces resource requirements, making high-quality segmentation more accessible and efficient. Such advancements mark a significant step towards achieving truly universal image segmentation.

In overview, this research reflects the integration and adaptation of several existing methodologies in the fields of active learning, semi-supervised learning, and advanced object detection models with instance segmentation. It draws from the strengths of these methods, while addressing their limitations to optimize the use of plentiful unlabeled data. The model thus aims to create a balanced solution, reducing the human effort required in image segmentation while acquiring satisfactory performance. Ultimately, it contributes a new perspective to the ongoing discussions around automated image segmentation and machine learning.

III. Materials and Methods

A. YOLOv8

The You Only Look Once (YOLO) models [17], [18] revolutionized object detection by unifying location identification and classification, both of which were traditionally separated from each other. This approach allows YOLO models to analyze an entire image during training and prediction in one go. This global view facilitates the recognition of overall contextual patterns within the image, enhancing both the accuracy of object detection and classification. The standard architecture of YOLO models comprises three main parts: the backbone for extracting low-level features, the neck for multiscale feature fusion, and the head for final object detection predictions.

Building on this architecture, YOLOv8 [19] introduces notable enhancements. It incorporates an advanced loss function blending Mean Squared Error (MSE) for bounding box regression and Binary Cross-Entropy (BCE) for objectness. It also adopts a novel neural network architecture that leverages both Feature Pyramid Network (FPN) and Path Aggregation Network (PAN) to significantly boost predictive accuracy. YOLOv8 offers five model sizes catered to various task requirements and computational resources. In this study, YOLOv8s was employed because of its high performance and computational speed. Moreover, YOLOv8 supports rectangular input shapes and auto-adjusts input size according to the device, enhancing speed for both training and inference. These improvements in YOLOv8 enhance its scalability and adaptability, making it an integral part of the YOLOv8-AIS algorithm for efficient automated image segmentation.

B. OneFormer

Vision transformers have recently gained prominence for their effectiveness across various vision tasks. Following this trend, we also tested our proposed algorithm using a vision transfoermer model built on a framework called the OneFormer. Originated from the success of transformers in natural language processing, vision transformers adapt their architecture for visual data to process images as sequences of patches. OneFormer builds on this foundation by introducing significant advancements such as a unified framework for addressing multiple segmentation tasks (semantic, instance, panoptic) simultaneously with a single model. Compared to the Mask2Former framework, it offers improved performance in task adaptability and segmentation precision through innovations include task-conditioned training and query-text contrastive loss [20].

C. Dataset Description

The PlantVillage dataset is a publicly accessible collection of leaf images representing various plant species, with each image labeled with specific disease conditions or as healthy. As detailed in Table I, both PlantVillage Tomato (PV-Tomato) and PlantVillage Apple (PV-Apple) datasets, derived from the Plant Village [21] dataset, are used in the experiment. The PV-Apple dataset, manually annotated in its entirety, serves as a performance benchmark. In contrast, only the initial training and test subsets of the PV-Tomato dataset are manually annotated, leaving the rest for the algorithm to label. Both datasets have been instance segmented using polygons via the data annotation tool ”Roboflow” [22] as represented in Fig 1. This format of annotation, commonly utilized across various studies, was employed to bolster the final performance of the model after the training process and enable the creation of a comprehensive auto-labeled dataset in the format of instance segmentation. This achievement was made possible because of the inherent instance segmentation capabilities of the YOLOv8 model.

The HAM10000 dataset paper presents an extensive collection of dermatoscopic images designed to enhance the automated diagnosis of pigmented skin lesions through machine learning. It contains over 10,000 images from varied sources, carefully curated to encompass a broad spectrum of pigmented lesions. As shown in Table II, the dataset includes an imbalanced distribution across different classes. Unlike the PlantVillage dataset, which required manual annotation, the HAM10000 dataset comes with pre-existing segmentation masks, facilitating easier conversion for experimental use. This

Dataset	PV-Tomato dataset	PV-Apple dataset
Manual Annotation	Only initial train and test	Entire dataset
Number of images	18160 images	3164 images
Proportion of initial training	25/100 images per class (250/1000 images)	50 images per class (200 images)
Proportion of Active Learning	15348 images	2649 images
Proportion of test data	10% of total (1812 images)	10% of total (315 images)

TABLE I: PlantVillage Dataset Structure Used for Experiment

Classes	Number of Images
Actinic keratoses	327
Basal cell carcinoma	514
Benign keratosis-like lesions	1099
Dermatofibroma	115
Melanoma	1113
Melanocytic nevi	6705
TOTAL	9758

TABLE II: Utilized Image Data Distribution for each class from HAM10000 Dataset

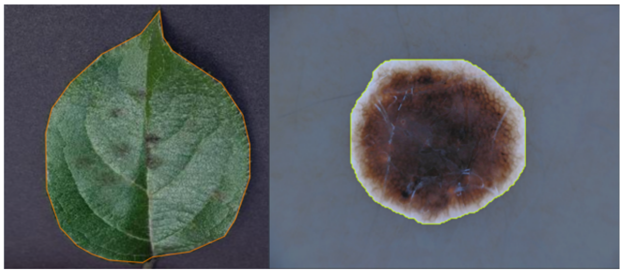

Fig. 1: Instance Segmentation annotations precise polygon shapes, highlighting the distinct texture of the object

feature, coupled with its varied nature compared to baseline datasets like PV-Tomato and PV-Apple, offers a comprehensive view of the system's effectiveness across different dataset types and data distribution scenarios, highlighting its utility in diversifying research approaches.

D. Methodology

The YOLOv8-AIS methodology utilizes an iterative active learning process. In each cycle of training, prediction, and segmentation, the model refines its object identification and segmentation capabilities. This process progressively enriches the labeled dataset and thereby boosts the model's performance.

Algorithm 1 Confidence score based YOLOv8-AIS Algorithm Pseudo Code

```
Input: Labeled data $D_L$, unlabeled data $D_U$, test data $D_T$,
    YOLOv8 model $M$, AIS parameters, $\lambda$
Output: Trained model $M$, metrics, auto-labels for dataset
    LOOP Process :
for each cycle do
    Train $M$ on $D_L$
    Predict labels for $D_U$ with confidence scores
    Define heap size for classes as $\lambda$ percent of unlabeled
    images
    for each class do
        Maintain a min-heap for predictions
        Replace heap top for predictions with higher confidence
    end for
    Auto-label images from heap using YOLOv8 segmentation
    Correct misclassified labels then update $D_L$ and $D_U$
    Assess $M$ on $D_T$
end for
return $M$, metrics, auto-labels
```

1) Active Learning Process and Prediction: As illustrated in Figure 2 and detailed in Algorithm 1, the proposed model for automated image segmentation employs an iterative active learning approach. The initial model is trained on manually segmented data, then performs inference on all unlabeled images in the dataset to calculate their confidence scores. During each inference, images are processed in batches, with the YOLOv8 model, trained on the available labeled data, predicting labels for these images. Each image is assigned with a confidence score, which serves as an indicator of the model's certainty regarding the assigned label. These confidence scores play a crucial role in the auto-segmentation process as they help determine which images are selected for segmentation in each iteration. Based on the confidence scores, the algorithm then selects a (λ) percentage of the inference results with the highest uncertainty scores. This (λ) variable is determined by the user and is used to establish the size of the heap for each class in the dataset. The results with the highest uncertainty scores are then added to the training and validation splits of the dataset, completing one iteration of the training process.

2) Heap Structure, Auto-segmentation, and Label Correction: The algorithm processes images and maintains a heap-a structure that stores inference results such as confidence scores with predicted classes and the actual class information-for each class. The heap size is determined by the λ percent calculated earlier. As predictions are being made, predicted labels with the lowest confidence scores, which signify a high level of uncertainty, are added to the heap. These labels can replace existing entries with higher confidence scores if the heap reaches its maximum capacity. Once the prediction phase is completed, all labels within the heap are used for auto-segmentation. This process produces detailed instance segmentation with polygons accurately fitting the shapes of leaf objects in the images by capitalizing on the segmentation masks generated based on the robustness of the YOLOv8 model. For each misclassification of the class information of the selected segmentation labels, the output class is corrected by reverting it to the original class according to its saved actual class information, ensuring the accuracy of the labels. Next, the freshly corrected and auto-labeled images are then merged into the current labeled dataset for future training iterations, directing the active learning mechanism of the AIS algorithm towards the most complex instances.

3) Model Retraining, Performance Evaluation, and Stopping Conditions: Following the auto-segmentation and correction process, the labeled images are removed from the unlabeled data pool. The model is then retrained on the updated labeled dataset. With each iteration, the quantity and diversity of the labeled data used for training increase, thereby allowing the model to continually improve its predictive accuracy. After each retraining cycle, the model's performance is evaluated using a test dataset, providing key performance metrics. These metrics often reveal an improvement in the model's performance over time due to the growing size and diversity of the labeled dataset. The segmentation and learning process is repeated until a stopping condition is met. This could occur when there are no more unlabeled images, or when the model is no longer capable of making further progress in detecting the remaining unlabeled images.

4) Simple Data Duplication Method: The Simple Data Duplication method involves duplicating the images from the given initial dataset. Since augmentation occurs automatically within the YOLOv8 training process, this method could help the model better learn the common shapes within the dataset from an enlarged initial dataset. Consequently, this can enhance the model's performance in subsequent training iterations. This method is applied to the dataset under specific conditions: 1) when the collapse of model performance is expected because of the initial size of the dataset being too small, or 2) when the data from the dataset itself has unclear and vague shapes or features, requiring the model to learn from a more numerous initial dataset.

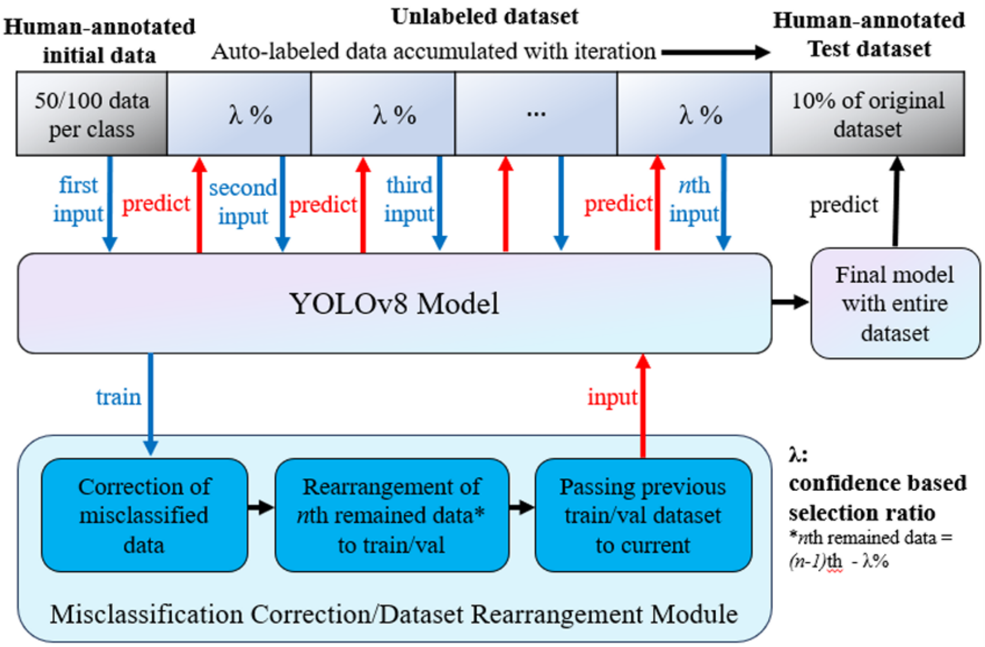
Human-annotated initial data
Unlabeled dataset
Auto-labeled data accumulated with iteration
Human-annotated Test dataset
50/100 data per class
λ %
λ %
...
λ %
10% of original dataset
first input
predict
second input
predict
third input
predict
nth input
predict
YOLOv8 Model
Final model with entire dataset
train
input
Correction of misclassified data
Rearrangement of nth remained data* to train/val
Passing previous train/val dataset to current
Misclassification Correction/Dataset Rearrangement Module
λ: confidence based selection ratio
*nth remained data = (n-1)th - λ%

Fig. 2: An overview of the overall architecture.

E. *OneFormer model application*

After a series of experiments with YOLOv8-AIS, we applied our algorithm to the OneFormer vision transformer model to experiment with the performance of transformer models on this task. The key question is whether it can adapt to the initial dataset enough to capture the common shapes and features. This adaptation would allow it to utilize the learned weights for the next iteration of the training process, making successful inferences to integrate them into the existing training dataset, as we did with the YOLOv8 version. The OneFormer model is fundamentally different from the YOLOv8 model and originates from a different framework. Consequently, the iterative training process must be explicitly implemented in a manner distinct from that used with YOLOv8, albeit to perform a similar function. The dataset, initially in the instance segmentation annotation format of YOLOv8 (text files), must be converted into the semantic segmentation mask format to be utilized by the OneFormer model. Once trained on these image and mask pairs, the model is then used to infer mask annotations for a λ proportion of unannotated data, which are subsequently combined into the existing training dataset.

IV. EXPERIMENTS

A. *Environmental Set up*

1) Experimental Environment: This study utilized the NVIDIA GeForce RTX 3070 Laptop GPU for the YOLOv8 model. For the OneFormer vision transformer model, it employed online graphic card-equipped environments from VAST.AI because of the significant computational resources required by transformer models. The research focuses on refined and selected Apple and Tomato leaf image data from the Plant Village dataset, alongside the HAM10000 skin lesion dataset. All data preprocessing and the implementation of YOLOv8-AIS were carried out in Python using PyCharm, facilitating the initiation of the algorithm's active learning cycle for the auto-segmentation process. Conversely, for the AIS implementation of the OneFormer model, a Jupyter notebook wa utilized to run the processes server-side on VAST.AI. The server had 48GB of disk space for storing data and model run information and 2 NVIDIA RTX 4090 graphics cards to utilize PyTorch-based machine learning environments.

2) Dataset Construction: Each of the PV-Tomato, PV-Apple, and HAM10000 datasets shares the following settings: 10% of the dataset was separated as test split at the start, then it was assigned its own chosen initial dataset size that they initially train on, and the remaining data works as the active learning part which does not have any segmentation information but only have classified information.

3) Class-Wise Initial Dataset Sizes and Their Purposes: Different training set sizes (25 and 100 images per class) were used for the Tomato dataset to assess how the initial size influences the algorithm's performance. The PV-Apple dataset, initialized with 50 images per class, served as a performance benchmark against fully manually annotated datasets. A balance between the classes in the training dataset was maintained to avoid model bias and enhance generalization capabilities. For the HAM10000 dataset, 60 images per class were used to investigate how the model performs on datasets with more vague features and shapes, as well as significantly imbalanced data distribution compared to the Tomato and Apple datasets.

4) λ Variable Analysis: The impact of varying λ values (10% and 20%) on the performance of the model was examined using both the PV-Tomato and PV-Apple baseline datasets. The variable λ represents the fraction of the unlabeled dataset that is auto-labeled and added to the training data at each iteration. By manipulating λ, we could observe how different proportions of additional data per iteration influence the model's performance enhancement as the training progresses.

For the experiments with minimal initial dataset sizes using the HAM10000 dataset, the λ values were fixed at 30%. This decision was based on observations that variations in the λ values do not sufficiently impact the training process to make a meaningful difference in the baseline test cases, where the model was tested on the baseline PV-Tomato and PV-Apple datasets without modifications. Therefore, we chose to maintain λ at a common maximum threshold of new labeled data that the model can incorporate per iteration of training, rather than observing variations of them in these later experiments.

5) Evaluating Performance with Minimal Initial Dataset Sizes Under Varied Conditions: In the second stage of the experiments, which tests the critical threshold of the initial data amount that can compromise the performance of the model's iterative training system, the initial data size for each class was reduced to 20 for both the PV-Tomato and PV-Apple datasets. The PV-Apple dataset was subjected to more detailed and varied conditions in addition to the minimal initial dataset size condition, including cases where specific classes' image data were entirely missing. This included conditions with only the minimal size initial dataset (ALL), missing half of the classes that showed the lowest accuracy (TOP2), and missing every class except the one which showed the highest accuracy (TOP1), to evaluate whether the model can overcome this missing data utilizing the misclassification correction algorithm.

The PV-Tomato dataset experiment focused on how the dataset could be augmented with the Simple Data Duplication method to mitigate the decrease in model's performance when the minimal initial dataset size crosses the critical threshold, which prevents the model from learning adequately from its initial size and thereby affects the training process outcomes negatively. There were two cases for the PV-Tomato dataset: one with only the minimal initial dataset size of 20 images per class (ALL) and another with the Simple Data Duplication method applied (ALL_DUPLICATED).

6) Evaluating Model Performance on a New and Ambiguously Shaped Dataset: To explore whether the Simple Data Duplication method could also enhance the model's predictive accuracy when the dataset itself presents critical challenges that could drastically reduce performance, the HAM10000 dataset was subjected to this mitigation approach because of its imbalanced distribution of classes and vague shapes and features.

7) OneFormer settings: Because of its significant computational resource requirements, the OneFormer model version was experimented with solely on the PV-Apple dataset to conduct the experiment efficiently. This choice was also influenced by the verified performance of the PV-Apple dataset on the YOLOv8-AIS architecture under various conditions. The λ is fixed at 30% for the OneFormer application version. The training dataset was shared with the minimal size PV-Apple dataset testcase of the YOLOv8-AIS model but converted into the semantic segmentation mask format. The first experiment was conducted with 10 epochs without changing the minimal size PV-Apple dataset. The second experiment used the same dataset, but with a Simple Data Duplication method applied to triple its size. This was done to assist the OneFormer model in better capturing the shapes from the initial dataset because of its unpredictable performance results at first glance. Unlike YOLOv8, OneFormer does not use a validation split of the training dataset. Therefore, it started with an initial dataset consisting only of the training split and continuously adds newly segmented data into the training split.

8) Performance Evaluation: The performance metric employed is the mAP@0.5 score, which is widely acknowledged in the fields of object detection and instance segmentation because of its comprehensive performance evaluation capabilities. The score is measured for the models that were trained on every iteration of the training process, based on the test split of the dataset from the start.

B. Experiment on Semi-Supervising Capability

Initial data	25 per class		100 per class	
Dataset	λ=0.1	λ=0.2	λ=0.1	λ=0.2
Initial	0.6966	0.6966	0.9165	0.9165
iteration 1	0.9176	0.941	0.9359	0.9486
iteration 2	0.959	0.9557	0.9609	0.9574
iteration 3	0.9395	**0.9631**	0.9494	0.9574
iteration 4	0.9546	0.9613	0.9574	0.9615
iteration 5	0.9457	0.9628	0.9619	**0.9639**
iteration 6	0.9604		0.9598	
iteration 7	0.9622		0.9598	
iteration 8	**0.9637**		0.9598	
iteration 9	0.9625		0.9621	
iteration 10	0.9552		**0.9654**	

TABLE III: mAP@0.5 scores for Tomato(λ =0.1), Tomato(λ =0.2) on each initial data type(25/100)

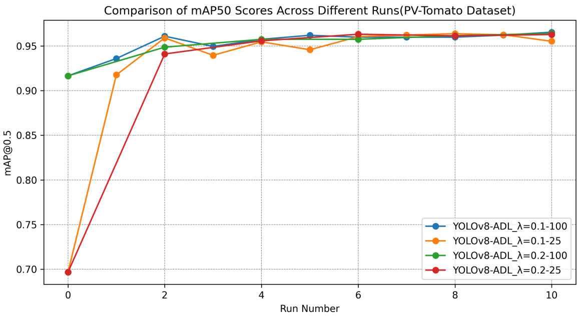

Fig. 3: YOLOv8-AIS performance on PV-Tomato

The adaptability and performance of the YOLOv8 AIS model under different initial conditions and λ values were examined using the PV-Tomato dataset. Initial training conditions included sets of 25 and 100 images per class and λ values at 0.1(10%) and 0.2(20%). The mAP@0.5 scores, which are reflective of the model's performance, showed initial results dependent on the size of the training set. The model trained with 100 images per class had a higher initial mAP@0.5 score than the model trained with 25 images per class because of the larger amount of labeled data available for learning.

As the active learning iterations progressed, all models showed steady improvement in their mAP@0.5 scores. Remarkably, the model trained with 25 images per class caught up with the performance of the model trained with 100 images per class, demonstrating the ability of the YOLOv8 AIS algorithm to make effective use of unlabeled data and improve its performance significantly, even when starting with a relatively small amount of labeled data. Specifically, the best performance of the model trained with 25 images per class, as indicated by the highest mAP@0.5 score, was 0.9637 and 0.9631 for λ values of 0.1 and 0.2, respectively. The model trained with 100 images per class achieved a mAP@0.5 score of 0.9654 for λ=0.1 and 0.9639 for λ=0.2. Initially, variations in the λ parameter were anticipated to introduce notable disparities in training progression and accuracy. However, the derived results suggest that changes in λ values had no significant differential impacts, emphasizing the model's resilience to such parameter alterations.

These results, encapsulated in Table III and visualized in Figure 3, underline the robustness and adaptability of the proposed algorithm in semi-supervised environments. They underscore the impact of the chosen λ value on the rate of improvement and the number of iterations required for the mAP@0.5 score to converge.

C. Comparison to Human Annotation

Dataset	Apple(λ=0.1)	Apple(λ=0.2)	Apple(manual)
initial	0.9083	0.9083	0.9649
iteration 1	0.9690	0.9709	
iteration 2	0.9743	**0.9756**	
iteration 3	**0.9751**	0.9625	
iteration 4	0.9728	0.9750	
iteration 5	0.9673	0.9731	
iteration 6	0.9682		
iteration 7	0.9722		
iteration 8	0.9624		
iteration 9	0.9729		
iteration 10	0.9688		

TABLE IV: mAP@0.5 scores for Apple(λ =0.1), Apple(λ =0.2) and Apple(manual) with fixed initial data(50)

The analysis was extended to the PV-Apple dataset, which allowed for an interesting comparison with manually annotated labels. This dataset offered a distinct benchmark to evaluate the performance of the YOLOv8 AIS model. In this scenario, the model began with an initial training set of 50 images per class, using λ values of 0.1 and 0.2, and the mAP@0.5 score was used as a performance metric. The model trained on the fully labeled dataset provided a solid performance with an mAP@0.5 score of 0.9649. This score represents the level of accuracy typically obtained with meticulous human annotation. However, the YOLOv8 AIS model, starting from a lower score, demonstrated a significant ability to improve its performance through active learning. Specifically, the algorithm, when implemented with a λ value of 0.1, progressed to achieve an mAP@0.5 score of 0.9751. An even higher score of 0.9756 was achieved with a λ value of 0.2. These results showcase the capacity of the YOLOv8 AIS model to not only automate the manual annotation process but also to enhance the overall performance of the object detection task.

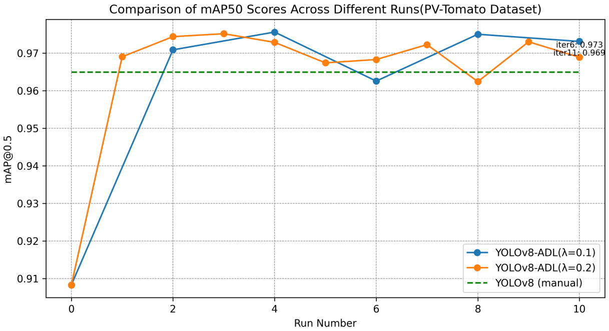

Fig. 4: YOLOv8-AIS and YOLOv8(manual) performance on PV-Apple

The variation in the size of the initial dataset once again demonstrated its influence at the beginning of the training process, whereas changes in the λ value did not significantly affect the system's performance. The specifics of these observations are detailed in Table IV and illustrated in Figure 4. These findings compellingly underscore the model's capacity to effectively learn from semi-supervised datasets by incorporating auto-labeled data into the training set over time, thus consistently enhancing its overall performance. This insight into the effects of the parameters, coupled with the system's successful performance on the PV-Apple dataset — even in comparison to human annotations — provides a clear direction for the next phase of research: investigating whether the system can maintain its efficacy in scenarios characterized by even smaller datasets, imbalanced datasets, and datasets with more complex features and textures.

D. Experiments on Minimal Dataset Sizes and Dataset Imbalance

Loop	**ALL**	**TOP1**	**TOP2**
Loop1	0.1917	0.1930	0.1886
Loop2	0.8847	0.7779	0.8896
Loop3	0.9777	0.9830	0.9749
Loop4	0.9936	0.9925	0.9910

TABLE V: mAP@0.5 scores for ALL, TOP1, and TOP2 test cases.

New research objective aimed to test the system's adaptability to datasets with minimal initial data or imbalance by missing class instances. Initially, the PV-Apple dataset comprised 50 images per class, which was reduced to 20 images per class for the initial training phase. The model trained on this reduced dataset demonstrated a significant

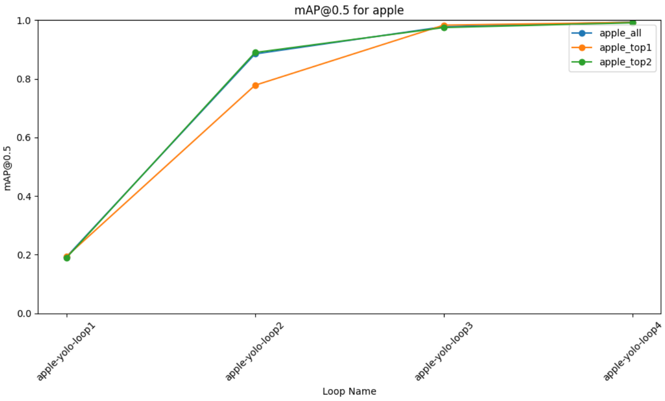

Fig. 5: YOLOv8-AIS on PV-Apple minimal and imbalanced version

decrease in performance initially, compared to its performance when trained on the full dataset. However, as the training progressed, the models trained on iteratively labeled datasets exhibited remarkable The improvements are presented in Table V under the 'ALL' column and illustrated in Figure 5 as 'apple-all'(ALL). This indicates that, even in scenarios with extremely low data availability, the model can maintain high performance levels if the data is sufficiently clear and distinct for learning. This underscores the efficacy of the YOLOv8 model in leveraging minimal datasets effectively.

Following experimentation with minimal datasets, we also explored the impact of intentionally imbalanced datasets. Specifically, we excluded classes that exhibited the lowest mAP@50 scores during minimal dataset testing. Our experimentation involved two scenarios: one where half of the classes were removed from a set of four (TOP2), and another where all but the top-performing class were excluded (TOP1). Consistent with previous findings, both scenarios initially demonstrated low performance. However, a rapid recovery in model performance was observed as the training iterations progressed. This suggests that if the model can accurately recognize the object shapes within the initial dataset, and if the dataset's classes share similar features and shapes, the model is capable of learning features from missing class data through the training process. This learning is facilitated by our correction algorithm, which adjusts the class of model inferences during each iteration. Although the model initially generates segmentations only for classes it recognizes, it acquires segmentation annotations for previously unrecognized classes in subsequent training iterations. The outcomes of both test cases—the minimal size dataset and the missing classes dataset—are detailed in Table V and illustrated in Figure 5.

E. Mitigating Performance Collapse Beyond Critical Thresholds through Data Duplication

Following the successful adaptation to minimal and imbalanced datasets, we also investigated scenarios where the system failed to acquire the capability to accurately segment common shapes within the dataset from the initial data. Consequently, it could not achieve significant improvements in subsequent training iterations. This issue was encountered during experiments with the PV-Tomato dataset, where the number of images per class was reduced from 25/100 to 20. Performance improvements were observed across iterations, but due to the model's inability to accurately segment shapes based on the initial dataset, the final results remained unsatisfactory when compared to previous experiments.

Loop	ALL	ALL_DUPLICATED
Loop1	0.1471	0.2871
Loop2	0.8004	0.6114
Loop3	0.8239	0.8977
Loop4	0.9080	0.9466
Loop5	0.8970	0.9651
Loop6	0.9132	0.9654
Loop7	0.9340	0.9667
Loop8	0.9145	*N/A*

TABLE VI: mAP@0.5 scores for ALL and ALL_DUPLICATED test cases in the tomato version.

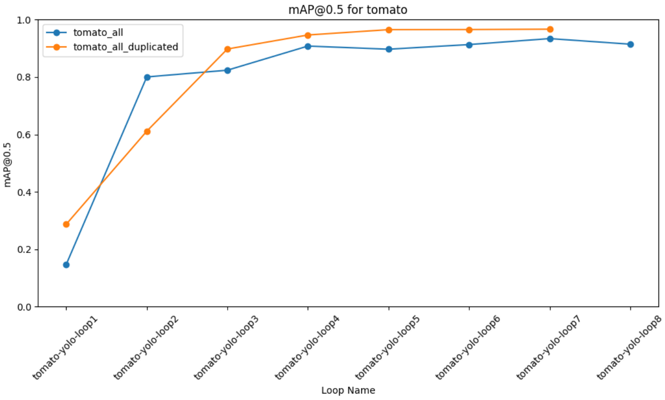

Fig. 6: YOLOv8-AIS on PV-Apple minimal and imbalanced version

In response to the challenges encountered with the minimal and imbalanced dataset scenarios, we utilized the Simple Data Duplication method to duplicate initial dataset. This approach, which does not involve adding new segmented data, allows the duplicated data to provide sufficient information for the model to learn the common shapes and features across classes. Models trained on datasets augmented by this method of duplication—beginning with the initial duplicated dataset—did not reach the performance levels observed in previous tests conducted with the PV-Tomato dataset. However, their performance still improved significantly compared to the baseline established with the original minimal dataset (ALL_DUPLICATED). These performance improvements are detailed in Table VI and illustrated in Figure 6.

This method encounters limitations when the shapes of objects vary significantly among classes or are ambiguous, compared to other datasets with specific and clear shapes. Therefore, the inherent characteristics of the dataset itself are not well-suited for iterative training based on a small initial dataset proportion. These limitations were revealed

Loop	First Set	Second Set
Loop1	0.1878	0.2083
Loop2	0.1859	0.2912
Loop3	0.2697	0.3235
Loop4	0.2866	0.3424
Loop5	0.2651	0.3473
Loop6	0.2730	0.3432
Loop7	0.2851	0.3570
Loop8	0.2777	0.3695
Loop9	0.2829	0.3167

TABLE VII: Iteration performance comparison for two sets of the HAM10000 dataset version.

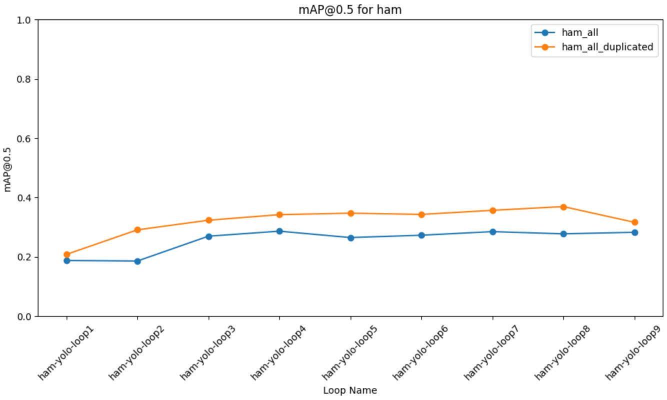

Fig. 7: YOLOv8-AIS on PV-Apple minimal and imbalanced version

when YOLOv8-AIS was tested on the HAM10000 dataset. This dataset contains classes with varying and ambiguous segmentation processes, alongside a natural data distribution imbalance among the classes. This led to varied performance in the YOLOv8-AIS model. While the model excelled in classes with high data distribution and clear shapes and features, it performed poorly in classes with low data distribution and ambiguous shapes and features, which are difficult to capture from the small initial dataset. These disparities resulted in catastrophic performance across all classes, yielding poor average results. The performance gap between the classes can be reviewed in Table VIII. Unfortunately, this poor performance could not be significantly improved even through the data duplication method. The iterative training processes' results for both test cases—one for the baseline (ALL) and another for the version with the duplicated method (ALL_DUPLICATED)—are detailed in Table VII, and their visualizations as graphs are illustrated in Figure 7.

Class	mAP50
All	0.317
Actinic keratoses	0.138
Basal cell carcinoma	0.157
Benign keratosis-like lesions	0.272
Dermatofibroma	0.098
Melanoma	0.31
Melanocytic nevi	0.925

TABLE VIII: mAP@0.5 scores by class.

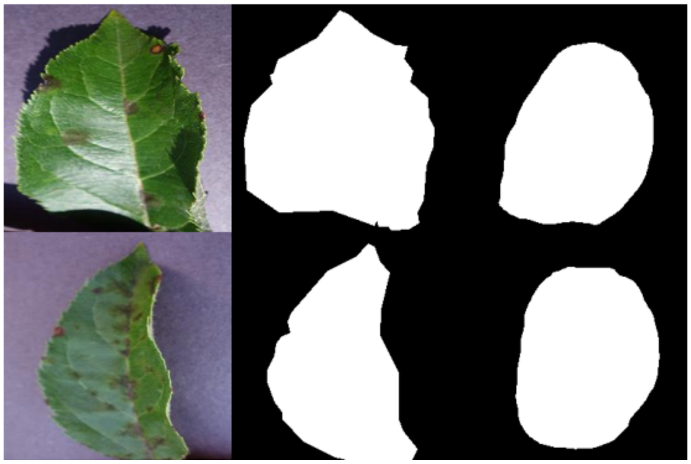

Fig. 8: Inference Sample Analysis for the Oneformer Model: Showcased from left to right are the original image (left), the ground truth segmentation mask (middle), and the model's predictive output (right))

F. Experiments on the application of the system on the OneFormer model

While the YOLOv8-AIS has demonstrated impressive performance across various conditions and datasets, it had not been previously tested whether our AIS algorithm could extend beyond YOLOv8 instance segmentation models. The OneFormer application of the model was experimented with in two test cases, as introduced in the OneFormer settings section. However, the results presented in Figure 8 have shown that the transformer-based model is not as versatile as YOLOv8 in environments where it must make accurate predictions based on a very limited proportion of the entire dataset. This limitation resulted in the segmentation performed in later iterations failing to capture the basic features and shapes of objects in the images in both test cases.

Model	Total Inference Time (seconds)
OneFormer	17.0774
YOLOv8	7.0720

TABLE IX: Inference times of OneFormer and YOLOv8 models on 100 sample images.

The failure of the OneFormer model on the PV-Apple dataset starkly contrasts with the performance of the YOLOv8-AIS model, which excelled under various conditions of the PV-Apple dataset, even with a minimal initial dataset size, similar epoch numbers, and, most importantly, significantly lower computational resource requirements for short inference times. The difference in inference time is illustrated in Table IX, highlighting that the gap in inference time between the two models widens as the sample image size increases and suggesting that the actual difference in inference times could be even larger. This underscores the limitations in data efficiency of vision transformers and highlights how YOLOv8 can outperform other segmentation models in learning from a small amount of data while still effectively capturing significant features at the current stage of machine learning model development.

V. Conclusion

This study demonstrates the effectiveness of the proposed automated image segmentation method in enhancing the performance of object detection models like YOLOv8. The results showed that by incorporating a proportion of auto-labeled data into the training set at each iteration, the model's performance can be significantly improved. It was also observed that the rate of improvement is more influenced by the initial dataset size, which provides the initial training weights and accuracy of the model, than by the λ value, which dictates the proportion of data to be auto-labeled in each cycle. This implies that the success of the model in this research heavily depends on the model's ability to capture the common features and shapes of the object from the given initial dataset at the start of the training process.

The current implementation of the proposed method with the YOLOv8 model has been proven to succeed in both improving model performance and creating an automatically and accurately segmented dataset for the clear dataset. This enhanced performance persisted even in scenarios where the initial dataset's size was reduced to a minimum of 20 per class or even further, with missing data for certain classes. In instances where the model failed to capture the common features of the dataset due to its reduced size, the Simple Data Duplication method enabled the model to learn from an increased size of the initial dataset from the minimal dataset, and performance was mostly restored.

However, there were still limitations when our algorithm was utilized across a broader range of datasets. These limitations became apparent during our experiments with the HAM10000 dataset. As previously discussed, if the dataset is severely skewed or characterized by ambiguous shapes and features, there is a high likelihood that models performing well on certain dataset classes might fail to generate accurate segmentation data for subsequent training iterations. This implies that the algorithm's success largely depends on the inherent performance of the model on the dataset. Therefore, if the model struggles with a dataset that is difficult to comprehend, the effectiveness of our algorithm is also likely to be compromised, causing it to fail to perform its task with great accuracy and efficiency. These limitations might be mitigated by utilizing our method exclusively for clean datasets or by improving the model's structure so that it can inherently perform better on specific datasets.

Additionally, the application of our algorithm to the OneFormer model has shown that the successful performance of our algorithm also largely depends on the models that demonstrate great efficiency in capturing the shapes and features of the image data with a small amount of data given at the start of the training process. Because vision transformer models require a comparatively larger amount of data and computational resources for their learning process, they struggle to capture sufficient features and shapes from the smaller-sized initial datasets used in this research's experiments. In contrast, the state-of-the-art CNN architecture models were able to successfully perform on these tasks. This might improve as vision transformer models enhance both their efficiency and performance through further development in the near future, thereby enabling better training even with smaller sizes of datasets and improving their training efficiency.

To summarize, the performance of our algorithm demonstrates an efficient method for creating segmentation of image data with minimal manual effort. Our method even improves the detection performance of the model on the dataset with this self-generated dataset. This holds true even for datasets containing unique categories that are not learned from its pretrained weights but are newly learned. However, there are still some limitations. Notably, the algorithm's success largely depends on the performance of the model it utilizes and the clean status of the dataset that the model can inherently comprehend. This limitation might be mitigated by improving the quality of the dataset through preprocessing the image data, or by further improvement of the vision models themselves by making modifications to adapt to the specific dataset in the future researches.

References

[1] M. F. Kabir and S. A. Ludwig, "Enhancing the performance of classification using super learning," *Data-Enabled Discovery and Applications*, vol. 3, pp. 1–13, 2019.

[2] Y. Shen, H. Yun, Z. C. Lipton, Y. Kronrod, and A. Anandkumar, "Deep active learning for named entity recognition," *arXiv preprint arXiv:1707.05928*, 2017.

[3] X. Zhan, Q. Wang, K. Huang, H. Xiong, D. Dou, and A. B. Chan, "A comparative survey of deep active learning," 2022.

[4] P. F. Jacobs, G. M. de Buy Wenniger, M. Wiering, and L. Schomaker, "Active learning for reducing labeling effort in text classification tasks," 2021.

[5] J. Wang, S. Wen, K. Chen, J. Yu, X. Zhou, P. Gao, C. Li, and G. Xie, "Semi-supervised active learning for instance segmentation via scoring predictions," 2020.

[6] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie, "Learning from noisy large-scale datasets with minimal supervision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 839–847.

[7] X. Yang, Z. Song, I. King, and Z. Xu, "A survey on deep semi-supervised learning," 2021.

[8] X. Chen, B. Mersch, L. Nunes, R. Marcuzzi, I. Vizzo, J. Behley, and C. Stachniss, "Automatic labeling to generate training data for online lidar-based moving object segmentation," 2022.

[9] A. Ratner, S. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: Rapid training data creation with weak supervision," in *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, vol. 11, no. 3. NIH Public Access, 2017.

[10] "Amazon automate data labeling," 2023, retrieved May 5, 2023. [Online]. Available: https://docs.aws.amazon.com/sagemaker/latest/dg/sms-automated-labeling.html

[11] H. Hino, "Active learning: Problem settings and recent developments," 2020.

[12] Y. Ouali, C. Hudelot, and M. Tami, "An overview of deep semi-supervised learning," 2020.

[13] H. Vishwakarma, H. Lin, F. Sala, and R. K. Vinayak, "Good data from bad models : Foundations of threshold-based auto-labeling," 2022.

[14] A. M. Hafiz and G. M. Bhat, "A survey on instance segmentation: State of the art," *International Journal of Multimedia Information Retrieval*, 2020.

[15] Roboflow, "Yolov8," retrieved May 11, 2023. [Online]. Available: https://roboflow.com/model/yolov8

[16] J. Jain, J. Li, M. Chiu, A. Hassani, N. Orlov, and H. Shi, “Oneformer: One transformer to rule universal image segmentation,” 2022.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” 2016.

[18] J. Terven and D. Cordova-Esparza, “A comprehensive review of yolo: From yolov1 and beyond,” 2023.

[19] D. Reis, J. Kupec, J. Hong, and A. Daoudi, “Real-time flying object detection with yolov8,” 2023.

[20] B. Cheng, A. G. Schwing, and A. Kirillov, “Per-pixel classification is not all you need for semantic segmentation,” 2021.

[21] Kaggle, “Plant village dataset,” retrieved May 11, 2023.

[22] R. Inc., “Polygon tool,” retrieved May 12, 2023.