SoyStageNet: Balancing Accuracy and Efficiency for Real-Time Soybean Growth Stage Detection

Abdellah Lakhssassi[1]*, Toqi Tahamid Sarker[1]*, Khaled Ahmed[1], Naoufal Lakhssassi[2], Khalid Meksem[1]
[1]Southern Illinois University Carbondale, USA
[2]Hampton University, USA
{abdellah.lakhssassi, toqitahamid.sarker, khaled.ahmed, meksem}@siu.edu
naoufal.lakhssassi@hamptonu.edu

Abstract

We present SoyStageNet, a novel transformer-based architecture for accurate detection of soybean growth stages. Precise growth stage identification is critical for optimizing agricultural interventions, yet existing solutions rely on conventional CNN architectures with limited accuracy and efficiency. Our approach integrates a lightweight Mix Vision Transformer backbone with a Neural Architecture Search-based Feature Pyramid Network and task-aligned detection head to effectively capture the subtle morphological differences between growth stages. Comprehensive experiments on our specially collected and annotated soybean dataset demonstrate that SoyStageNet achieves 83.2% AP with only 17.3M parameters and real-time inference speed of 28.5 frames per second, an 87% reduction in model size compared to DINO while maintaining competitive accuracy. This efficiency makes SoyStageNet particularly suitable for resource-constrained agricultural applications requiring real-time growth stage monitoring.

1. Introduction

Soybean is a crucial crop in today's agricultural businesses. It serves as a source of carbohydrates, sugar, oil, protein, and biofuel [5, 17]. Known for its high protein content, soybean is essential for food production and livestock feed. It typically contains approximately 36-40% protein, with some high-protein cultivars reaching up to 47% protein content [17]. Soy protein is complete, providing all nine essential amino acids crucial for human nutrition. Soybeans contain 18-24% oil [38], making soybean oil the second-highest annual yield behind palm oil and ahead of grape seed oil in the United States. In 2024, U.S. soybean production reached approximately 4.46 billion bushels [30], with the crop's total value exceeding $60.7 billion in 2023 [29]. Soybean ranks as the second-largest crop in the U.S. [30], primarily cultivated in the Midwest regions.

Understanding plant growth stages is important for optimizing agricultural practices, including decisions on irrigation, fertilization, pest control, and harvest timing [23]. Accurate knowledge of growth stages allows for targeted interventions that enhance crop quality and yield while minimizing chemical use and ecological impact. Chemical applications are most effective when properly timed, particularly to protect plants in early development.

Soybean growth can be broadly divided into Vegetative (V) and Reproductive (R) stages [23]. At VE (Emergence), seedlings break through the soil, beginning the Vegetative phase (VE-VN). By VC (Cotyledon Stage), the first unifoliolate leaves open, and each subsequent stage (V1-VN) represents an additional set of fully expanded tri-foliolate leaves. Once sufficiently developed, plants transition to the Reproductive stage (R1-R8), where yield is determined. The Reproductive phase progresses from flowering at R1 (Beginning Bloom) and R2 (Full Bloom), through pod development at R3-R4, to seed formation at R5-R6. By R7 (Beginning Maturity), pods begin turning yellow-brown, and at R8 (Full Maturity), plants are ready for harvest after seeds dry below 15% moisture.

Soybeans tolerate early interventions well; the early control of weeds and pests during vegetative growth spares resources for later yield formation [31]. The early reproductive growth stage (R1-R3) is an optimal window for insecticide use in soybean fields [19]. Spraying insecticides at the R3 stage when the canopy is sufficiently developed achieves good coverage. Some chemicals, such as certain triazole fungicides, cannot be applied after the R5 stage [18]. In other words, research consensus is that soybean pest and weed management should be synchronized with crop development: weed control should begin early by V1-V2, as maintaining a weed-free environment during this period is essential to prevent more than a 2.5% yield loss [31]. For

*Abdellah Lakhssassi and Toqi Tahamid Sarker contributed equally to this work.

insect and disease threats, treatment often begins at R1-R3 stages; however, during the late reproductive stage, such treatments should be discontinued to avoid chemical waste, yield damage, and residues [20].

Recent studies have demonstrated success in crop growth stage detection using deep learning. Object detection models (YOLOv5, RetinaNet, Faster R-CNN) have effectively identified weed growth stages [1], while specialized architectures like SGSNet achieved exceptional precision for strawberries [13]. CNN-based classifiers using VGG19 and DenseNet have successfully classified maize, wheat, and barley stages [25, 32], and hybrid models like MaizeHT reached 98% accuracy in field conditions [21].

Despite these advances, soybean growth stage detection remains underdeveloped. Current approaches rely on manual inspections, basic ML models, or satellite imagery [10], which lack real-time adaptability. Existing studies either use outdated CNN architectures [11] or focus on yield prediction [9] rather than precise stage detection [24]. This gap highlights the need for efficient transformer-based architectures for accurate soybean growth stage identification, critical for optimizing agricultural interventions. Moreover, enabling real-time inference on edge devices (e.g., drones or agricultural robots) enables timely and local decision-making without reliance on advanced and expensive cloud infrastructure.

The main contributions of this paper are as follows:

- We propose SoyStageNet, a transformer-based architecture specifically designed for soybean growth stage detection that balances accuracy and computational efficiency.
- We collected a comprehensive dataset of 8,953 labeled soybean images across different growth stages, captured under varying environmental conditions with rigorous annotation.
- Extensive ablation studies analyzing the impact of different detection heads, neck architectures, backbone capacities, and feature dimensions to identify optimal configurations for agricultural applications.

The rest of this paper is organized as follows: Section 2 describes our dataset. Section 3 presents the SoyStageNet architecture. Section 4 provides implementation details and experimental results. Section 5 concludes with findings and future directions.

2. Dataset

In early September 2024, we planted 20 seeds from eight different wild-type soybean species in a controlled greenhouse environment. The environment was maintained at an average temperature of 27°C to ensure optimal growth. Initially, the seeds were sown in trays for early-stage monitoring and subsequently transplanted into pots at the VC growth stage. Details on species and number of seeds germinated can be found in Tab. 1. Regular irrigation was carried out every two to five days, depending on the development phase. We systematically gathered images of the plants from multiple angles every two to four days—typically just before watering—to capture various lighting conditions, backgrounds, plant orientations, and growth stages. All images were taken using a Samsung Galaxy S20FE, ensuring consistent image quality throughout the data collection period.

Species Planted	Number of Seeds	Seeds Germinated
Spencer	20	16
Forrest	20	14
PI567516C	20	14
PI407729	20	14
PI88788	20	12
Williams 82	20	18
Saluki	20	18
Essex	20	16

Table 1. Seeds planted for each species

Class	VE	VC	V1	V2	V3	VN
Number of Images	1487	890	2597	1035	1011	1933

Table 2. Dataset distribution by class

As the plants progressed through their growth stages, we followed a structured fertilization and irrigation strategy. We applied slow-release fertilizers at the first reproductive stage (R1) to promote healthy development. We observed that the PI567516C species displayed a faster growth rate than others, reaching the R7 stage significantly earlier. During the critical reproductive stages from R3 (Beginning Pod) to R6 (Full Seed), we increased irrigation to support pod development and seed filling. However, once the plants reached the R6 stage, we stopped watering to prevent excessive moisture and allow the seeds to dry and mature.

To ensure accurate annotations, we labeled the collected images using SAM 2 [26]. While SAM 2 provided initial segmentation, it struggled with accuracy due to environmental factors such as varying backgrounds, fluctuating lighting conditions, and reflections from sunlight on the plant leaves. To address these inconsistencies, we applied manual labeling to refine the bounding boxes and class labels, ensuring correct and precise annotations on all images. Our final dataset comprised 8,953 labeled images, with a distribution of classes detailed in Tab. 2.

For training and evaluation, we split the dataset into three subsets: 80% for training, 10% for validation, and 10% for testing. This standard split ensured a well-balanced learning process, where our model could generalize effectively across different growth stages and environmental conditions. Our structured approach to data collection, labeling, and splitting aimed to create a robust dataset for deep learning-based soybean growth stage classification.

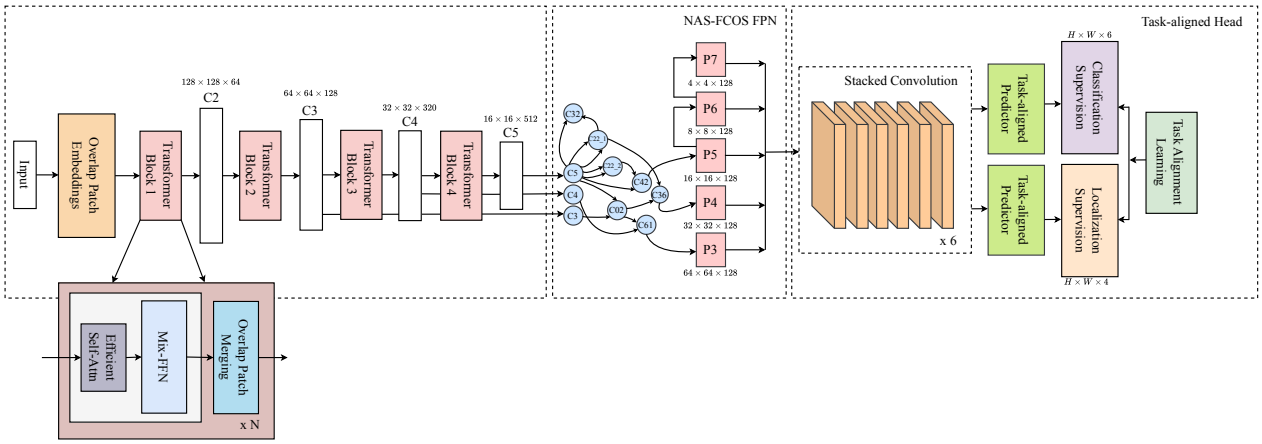

Figure 1. Architecture of SoyStageNet. Our model consists of three key components: (a) a SegFormer-based backbone (MiT-B1) that extracts hierarchical features (C2-C5) from input images through overlapped patch embeddings and transformer blocks; (b) a NAS-FCOS FPN that uses seven concatenation cells (C22_1, C22_2, C32, C02, C42, C36, C61) to fuse multi-scale features into uniform 128-channel feature maps (P3-P7); and (c) a Task-aligned head with six stacked convolutions and task decomposition modules that explicitly address classification-localization misalignment through adaptive feature selection and alignment learning.

3. Method

In this section, we introduce SoyStageNet, our novel object detection architecture for soybean growth stage detection.

3.1. Overall Architecture

Our detector consists of three key components: a Mix Vision Transformer [35] backbone for hierarchical feature extraction, a NAS-FCOS Feature Pyramid Network (FPN) [33] for multi-scale feature fusion, and a Task-aligned head [8] for accurate localization and classification. As a one-stage detector, SoyStageNet directly predicts object classes and bounding boxes in a single forward pass. Given an input image $\mathbf{I} \in \mathbb{R}^{H\times W\times 3}$, our model processes it through these components to generate predictions for six soybean growth stage classes:

$$\mathbf{Y} = \mathcal{H}(\mathcal{N}(\mathcal{B}(\mathbf{I}))) \tag{1}$$

where $\mathcal{B}$, $\mathcal{N}$, and $\mathcal{H}$ represent the backbone, neck, and head components. The complete architecture of SoyStageNet is illustrated in Fig. 1.

3.2. Mix Vision Transformer Backbone

We utilize a lightweight Mix Vision Transformer (MiT-B1) architecture [35] that processes input through four sequential stages, each generating feature maps at different scales with progressively increasing feature complexity and decreasing spatial resolution. This design effectively combines the global context modeling capability of transformers with the efficiency of hierarchical feature extraction, enabling our model to capture both local details and global context necessary for distinguishing subtle differences between growth stages. The MiT backbone consists of three key components that enhance its performance for soybean growth stage detection: overlapped patch merging, efficient self-attention, and Mix-FFN.

Overlapped Patch Embedding. Unlike traditional ViT [7] with non-overlapping patches, our approach uses overlapped patch embedding to preserve local continuity. For each stage i with output feature map $x_i \in \mathbb{R}^{H_i\times W_i\times C_i}$, we apply convolution with stride s_i and kernel size $k_i > s_i$ followed by normalization. Our implementation uses a 7×7 patch size with stride 4 for the first stage, followed by 3×3 patches with stride 2, creating multi-scale features at 1/4, 1/8, 1/16, and 1/32 of input resolution.

Efficient Self-Attention. Standard self-attention in transformers incurs quadratic complexity with sequence length, which is prohibitively expensive for high-resolution images. To reduce computational complexity while capturing long-range dependencies, we compute query vectors from original features and derive key/value vectors from spatially reduced features [34]:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}^Q, \quad \mathbf{K} = \mathbf{X}_r\mathbf{W}^K, \quad \mathbf{V} = \mathbf{X}_r\mathbf{W}^V \tag{2}$$

Here, $\mathbf{X}_r \in \mathbb{R}^{(HW/R^2)\times C}$ is obtained by applying a strided convolution with stride R followed by layer normalization. The attention operation is then computed as:

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \tag{3}$$

This spatial reduction approach reduces the computational complexity from $O(N^2)$ to $O(\frac{N^2}{R})$ with $R \in \{8, 4, 2, 1\}$ across stages.

Mix-FFN. Our backbone incorporates Mix-FFN, which eliminates explicit positional encoding through a depthwise 3×3 convolution between linear projections:

$$\text{Mix-FFN}(\mathbf{X}) = \text{MLP}_2(\text{DWConv}(\text{GELU}(\text{MLP}_1(\mathbf{X})))) \tag{4}$$

where MLP_1 expands channels by 4×, MLP_2 projects back to original dimensions, and DWConv captures position information from zero padding. This design avoids the need for explicit positional encoding while maintaining test resolution flexibility.

3.3. NAS-FCOS Feature Pyramid Network

For multi-scale feature fusion, we implement a NAS-FCOS FPN [33] that takes the outputs from stages 2-4 of the backbone. The FPN design is based on neural architecture search principles and incorporates deformable convolutions for enhanced geometric adaptation.

Feature Adaptation. The FPN first transforms the input features to a common dimension of 128 channels through 1×1 convolutions:

$$\mathbf{F}_i = \text{Conv1x1}(C_i), \quad i \in \{2, 3, 4\} \tag{5}$$

where C_i represents the output from stage $i + 1$ of the backbone (dimensions $64 \times 64 \times 128$, $32 \times 32 \times 320$, and $16 \times 16 \times 512$, respectively). This channel normalization ensures that features from different levels can be meaningfully combined in subsequent fusion operations. The 1×1 convolutions serve as bottleneck layers that perform dimensionality reduction or expansion while preserving spatial dimensions. Additionally, they include batch normalization and ReLU activation to enhance feature discrimination and introduce non-linearity. This adaptation step is essential for standardizing feature representations before the complex fusion operations in the concatenation cells.

Concatenation Cells. The fundamental building blocks of the NAS-FCOS FPN architecture are concatenation cells that efficiently fuse multi-scale features from the MiT-B1 backbone. Unlike conventional FPNs that rely on element-wise addition for feature aggregation [15], our approach uses a concatenation-based cells that preserve the distinct characteristics of input features while enabling more effective information flow. This design allows the network to selectively emphasize important features from each input while maintaining spatial alignment, which is particularly beneficial for dense prediction tasks. Each concatenation cell operates on a pair of input feature maps and can be formulated as:

$$\text{ConcatCell}(\mathbf{F}_i, \mathbf{F}_j) = \phi_{\text{out}}(\text{Concat}[\phi_i(\mathbf{F}_i), \phi_j(\mathbf{F}_j)]) \tag{6}$$

where ϕ_i and ϕ_j represent modulated deformable convolutions [6] applied to the respective feature maps, and ϕ_{out} is a grouped convolution for channel fusion.

The NAS-FCOS FPN incorporates seven concatenation cells with a specific connectivity pattern discovered through neural architecture search. These cells can be categorized into four functional types: (1) Adjacent-level fusion (c32), which combines features from levels 3 and 2; (2) Within-level refinement (c22_1, c22_2), which performs repeated transformation within level 2; (3) Long-range connections (c02, c42), which connect distant levels in the feature hierarchy; and (4) Cross-scale integration (c36, c61), which establishes pathways between intermediate features and deeply processed representations. This learned connectivity enables information to flow both horizontally (within the same resolution) and vertically (across different resolutions), progressively building rich feature representations. The naming convention directly indicates the source feature levels—for instance, c32 fuses features from levels 3 and 2, while c22_1 and c22_2 perform different refinement operations within level 2.

Feature Pyramid Output. The FPN produces five output feature levels: P3-P7. The first three levels (P3-P5) are generated by combining the fused features with upsampled features from the deepest level:

$$\mathbf{P}_i = \text{Upsample}(\mathbf{F}_{\text{idx}(i)} + \text{Upsample}(\mathbf{F}_5)), \quad i \in \{3, 4, 5\} \tag{7}$$

where $\text{idx}(i)$ maps the output level to the corresponding fused feature index. This addition of upsampled deeper features introduces semantic information from higher levels to lower levels, enhancing their representational capability. The upsampling is performed using bilinear interpolation to ensure smooth feature transitions. The additional levels P6 and P7 are generated through stride-2 convolutions:

$$\begin{aligned} \mathbf{P}_6 &= \text{Conv3x3}(\text{Norm}(\mathbf{P}_5), \text{stride} = 2) \\ \mathbf{P}_7 &= \text{Conv3x3}(\text{Norm}(\text{ReLU}(\mathbf{P}_6)), \text{stride} = 2) \end{aligned} \tag{8}$$

This results in five feature levels with dimensions: $\mathbf{P}_3 \in \mathbb{R}^{64\times64\times128}$, $\mathbf{P}_4 \in \mathbb{R}^{32\times32\times128}$, $\mathbf{P}_5 \in \mathbb{R}^{16\times16\times128}$, $\mathbf{P}_6 \in \mathbb{R}^{8\times8\times128}$, and $\mathbf{P}_7 \in \mathbb{R}^{4\times4\times128}$.

3.4. Task-aligned Head

The detection head in our architecture is based on the TOOD architecture [8], which addresses the task misalignment issue in one-stage detectors through explicit task decomposition and alignment mechanisms. Traditional one-stage detectors often suffer from inconsistency between classification and localization tasks, where high classification scores may not correlate with accurate bounding box predictions. Task-aligned head (T-head) explicitly resolves this issue through adaptive feature decomposition, task-specific prediction, and alignment-based training strategies.

Feature Extractor. For each feature level l from the FPN, the head applies six stacked convolutional layers with 128 channels:

$$\mathbf{F}_l^{(j)} = \text{Conv}^{(j)}(\mathbf{F}_l^{(j-1)}), \quad j \in \{1, 2, \ldots, 6\} \tag{9}$$

where $\mathbf{F}_l^{(0)} = \mathbf{P}_l$ is the input feature from the FPN, and $\mathbf{F}_l^{(j)}$ is the output of the j-th convolutional layer. These convolutional layers extract rich features while maintaining the spatial dimensions. Each convolution uses a 3×3 kernel with padding to preserve spatial dimensions, followed by normalization and ReLU activation. The stacked design progressively increases the receptive field and introduces

multiple non-linearities, allowing the network to learn more complex feature representations. Unlike traditional detection heads that only use the final layer's features for prediction, T-head retains all intermediate features, allowing task-specific modules to selectively utilize different layers.

Task-Aligned Predictor. We implement Task-Aligned Predictors for classification and localization tasks that consist of task decomposition, task-specific prediction, and spatial alignment components:

1) Task Decomposition: The task decomposition module adaptively aggregates features from all convolutional layers using learnable attention weights. First, we concatenate features from all six layers along the channel dimension:

$$\mathbf{F}_l^{\text{cat}} = [\mathbf{F}_l^{(1)}, \mathbf{F}_l^{(2)}, \ldots, \mathbf{F}_l^{(6)}] \in \mathbb{R}^{H_l \times W_l \times 768} \tag{10}$$

Task-specific layer attention mechanism first applies global average pooling (GAP) to reduce spatial dimensions, then uses a two-layer MLP to compute weights for each convolutional layer:

$$\mathbf{A}_l^{\text{task}} = \sigma(\text{MLP}(\text{GAP}(\mathbf{F}_l^{\text{cat}}))) \in \mathbb{R}^6 \tag{11}$$

where the MLP reduces dimensions from $768 \to 32 \to 6$, and σ is the sigmoid function. The resulting 6-dimensional weight vector corresponds to the importance of each convolutional layer for the specific task. These weights are then used to combine the features, followed by a 1×1 convolution that reduces the channel dimension:

$$\mathbf{F}_l^{\text{task}} = \text{Conv1x1}\left(\sum_{j=1}^{6} \mathbf{A}_l^{\text{task}}[j] \cdot \mathbf{F}_l^{(j)}\right) \in \mathbb{R}^{H_l \times W_l \times 128} \tag{12}$$

This adaptive aggregation is computed separately for classification (task=cls) and regression (task=reg). The weights are data-dependent, allowing the network to selectively emphasize different layers for each task. For example, earlier layers with finer spatial details might be weighted more heavily for localization, while deeper layers with richer semantic information might contribute more to classification.

2) Prediction and Alignment: To further enhance task alignment, our predictor integrates spatial probability maps and offset maps into the final predictions:

$$\begin{aligned} \mathbf{C}_l &= \text{Conv}_{cls}(\mathbf{F}_l^{\text{cls}}) \cdot \text{Prob}(\mathbf{F}_l^{\text{cat}}) \\ \mathbf{B}_l &= \text{Scale}_l(\text{Conv}_{reg}(\mathbf{F}_l^{\text{reg}})) \odot \Delta(\mathbf{F}_l^{\text{cat}}) \end{aligned} \tag{13}$$

where Conv_{cls} and Conv_{reg} are task-specific prediction heads, $\text{Prob}(\cdot)$ is a spatial probability map that estimates classification confidence, $\Delta(\cdot)$ is a spatial offset map that refines localization predictions, and Scale_l is a learnable scaling factor (initialized to 1.0 for each level) that adaptively adjusts the magnitude of regression outputs.

The spatial probability map modulates the classification logits, focusing the network on discriminative regions for classification. Similarly, the spatial offset map enables refinement of bounding box coordinates by identifying precise boundary regions. Together, these mechanisms can be viewed as implementing a form of spatial attention that enhances task-specific feature representation while maintaining alignment between the two tasks.

Head	Anchor	AP	AP_{50}	AP_{75}	AP_M	AP_L	H (M)	GFLOPs	FPS
RTM	×	55.4	79.6	66.0	34.7	56.1	**2.38**	**13.48**	57.4
FCOS	×	63.2	88.5	75.7	37.3	63.7	4.92	23.35	43.5
VFNet	×	69.7	91.1	82.1	36.5	70.3	5.34	20.45	34.8
YOLOX	×	51.2	82.2	58.4	18.0	51.6	11.82	13.52	**62.5**
T-head	×	**71.0**	88.6	82.3	**44.8**	**71.6**	4.65	21.96	35.4
GFL	✓	68.1	90.1	80.0	40.2	68.3	5.06	23.91	44.0
RetinaNet	✓	69.9	**92.6**	**82.9**	36.4	70.3	4.93	23.36	49.0

Table 3. Impact of Detection Head. All configurations use MiT-B0 backbone (3.32M params) with FPN neck (3.07M params) and vary only the detection head. H (M) denotes head parameters in millions.

Task Alignment Learning. To further enhance alignment between classification and localization tasks, we implement Task Alignment Learning (TAL). This approach focuses optimization on anchors that perform well on both tasks simultaneously, improving detection quality. We define an anchor alignment metric that combines classification and localization performance:

$$t = s^{\alpha} \cdot u^{\beta} \tag{14}$$

where s represents the classification score, u denotes the IoU between the predicted box and ground truth, and $\alpha = 1$ and $\beta = 6$ are parameters that balance the influence of each task. For sample assignment, we select the top-k anchors with the highest alignment scores as positive samples for each ground truth. Our implementation uses a progressive schedule: ATSS [37] with topk=9 for the initial 4 epochs, followed by our task-aligned assignment with topk=13. This dynamic assignment continuously adapts as the network improves, focusing training on increasingly well-aligned anchors. This unified approach to task alignment ensures that the detector produces spatially consistent predictions, which is particularly important for distinguishing between visually similar soybean growth stages while maintaining accurate plant localization.

4. Experiment

4.1. Implementation Details

We implement our experiments using PyTorch [22] and MMDetection [4] frameworks. All training is conducted on a server equipped with an Intel Xeon Gold 6240 CPU (2.60GHz), an NVIDIA Tesla V100S GPU (32GB memory), and 128GB RAM. For comparative evaluation, we train several established object detection architectures alongside our proposed SoyStageNet. We adopt a deliberate training strategy based on model initialization: DETR [3], DINO [36], RetinaNet [16], Faster-RCNN [27], and TOOD [8] are initialized with COCO-pretrained weights and trained for 20 epochs, while SoyStageNet is initialized with ImageNet weights and trained for 50 epochs. This

Neck	AP	AP_{50}	AP_{75}	AP_M	AP_L	N (M)	GFLOPs	FPS
FPN	71.0	88.6	82.3	44.8	71.6	**3.07**	21.96	**35.4**
NAS-FPN	82.2	93.5	88.6	46.6	82.6	29.20	41.45	24.1
NAS-FCOS	**82.9**	**95.0**	**90.8**	**47.7**	**83.6**	9.90	**21.11**	28.9

Table 4. Impact of Neck Architecture. All configurations use MiT-B0 backbone (3.32M params) with T-head (4.65M params) and vary only the neck architecture. N (M) denotes neck parameters in millions.

Head	AP	AP_{50}	AP_{75}	AP_M	AP_L	H (M)	GFLOPs	FPS
T-head	**82.9**	**95.0**	**90.8**	**47.7**	**83.6**	4.65	21.11	28.9
NAS-FCOS	76.5	94.1	86.9	41.3	76.9	**4.20**	**10.68**	**30.0**

Table 5. Impact of Task-Aligned vs. NAS-FCOS Detection Head. Both configurations use MiT-B0 backbone (3.32M params) with NAS-FCOS FPN neck (9.90M params) and vary only the detection head. H (M) denotes head parameters in millions.

methodological distinction is significant as COCO pretraining provides models with strong prior knowledge of object boundaries, contextual relationships, and scale variations directly applicable to detection tasks. In contrast, ImageNet pretraining primarily encodes classification features without spatial localization capabilities. The extended training regimen for SoyStageNet (50 epochs versus 20) compensates for this fundamental difference in initialization, ensuring fair comparison while acknowledging the different starting points in optimization trajectories. We use the AdamW optimizer with an initial learning rate of 1×10^{-4} and weight decay of 1×10^{-4}. Our learning rate schedule incorporates a linear warm-up phase for the first 1,000 iterations, followed by a multi-step decay with milestones at the 8th and 11th epochs (decay factor 0.1). All models are trained with a consistent batch size of 4 images per GPU. For data preprocessing, images are resized to 512 × 512 pixels while preserving aspect ratio to minimize distortion. We utilize the Albumentations [2] library for comprehensive data augmentation, including random horizontal flipping, random scaling, and photometric distortions (brightness, contrast, saturation, and hue adjustments). These augmentations enhance model generalization by simulating the variability of field conditions and viewing angles encountered in agricultural environments. We use Quality Focal Loss [12] for classification and GIoU Loss [28] for bounding box regression, which effectively address class imbalance and improve localization accuracy, respectively. Validation is performed every 10 epochs, with model checkpoints saved based on COCO mAP metrics [14].

4.2. Ablation Studies

To thoroughly evaluate the effectiveness of our proposed SoyStageNet architecture, we conduct extensive ablation studies on key components and design choices. All experiments follow the same training protocol described in Sec. 4.1, using standard COCO-style evaluation metrics.

Backbone	AP	AP_{50}	AP_{75}	AP_M	AP_L	B (M)	GFLOPs	FPS
MiT-B0	82.9	95.0	90.8	47.7	83.6	**3.32**	**21.11**	**28.9**
MiT-B1	84.5	95.4	**93.3**	48.0	85.0	13.15	26.12	27.8
MiT-B2	84.8	95.3	91.8	46.2	85.5	24.20	32.73	22.1
MiT-B3	85.3	**96.2**	91.9	**50.1**	85.9	44.07	44.10	14.2
MiT-B4	84.8	96.1	91.9	44.8	85.7	60.84	55.38	8.5
MiT-B5	**85.6**	96.1	93.1	43.3	**86.3**	81.44	66.33	10.0

Table 6. Impact of backbone Scaling. All configurations use NAS-FCOS FPN neck (10.02M params) and T-head (4.65M params) and vary only the backbone scale. B (M) denotes backbone parameters in millions.

FPN Channels	AP	AP_{50}	AP_{75}	AP_M	AP_L	N (M)	H (M)	FPS
64	79.2	92.5	87.5	46.3	79.7	**0.96**	**0.30**	**28.8**
128	83.2	95.1	92.0	46.2	83.9	2.95	1.17	28.5
256	**84.5**	**95.4**	**93.3**	**48.0**	**85.0**	10.02	4.65	27.8

Table 7. Impact of FPN Channel Dimensions. All configurations use MiT-B1 backbone (13.15M params) with NAS-FCOS FPN neck and T-head with different channel dimensions. N (M) and H (M) denote neck and head parameters in millions, respectively.

Impact of Detection Head. First, we investigate different detection heads while keeping the MiT-B0 backbone and FPN neck fixed. Table 3 shows the comparative results across seven detection heads. The T-head achieves the best overall performance (71.0% AP) among all tested heads, with significant advantages for medium-sized instances (44.8% AP_M). While RetinaNet shows stronger performance at lower IoU thresholds with 92.6% AP_{50}, T-head excels at higher IoU thresholds (82.3% AP_{75}), indicating superior localization precision. This suggests that T-head's explicit task alignment mechanism effectively addresses the classification-localization misalignment problem in one-stage detectors. Anchor-free options like FCOS and VFNet show competitive results, but cannot match the overall effectiveness of T-head.

Impact of Neck Architecture. Next, we investigate the influence of different neck architectures for multi-scale feature fusion, maintaining the MiT-B0 backbone and T-head detector. Table 4 presents comparative results. The neural architecture search-based NAS-FCOS demonstrates superior performance (82.9% AP), significantly outperforming the standard FPN (71.0% AP) with a remarkable +11.9% absolute improvement. This substantial gain is achieved while maintaining comparable computational efficiency (21.11G vs. 21.96G FLOPs). While NAS-FPN also shows strong performance (82.2% AP), it requires nearly three times more parameters (29.20M vs. 9.90M) with double the computational cost (41.45G vs. 21.11G FLOPs).

Task-aligned Head vs. NAS-FCOS Head. We further compare our chosen T-head with the NAS-FCOS head using the optimal NAS-FCOS FPN neck and MiT-B0 backbone. Table 5 presents the comparative results. T-head significantly outperforms the NAS-FCOS head (+6.4% AP),

Model	Accuracy (%)					Parameters (M)					Efficiency	
	AP	AP_{50}	AP_{75}	AP_M	AP_L	Total	B	N	E+D	H	GFLOPs	FPS
Faster R-CNN	90.0	98.3	97.5	53.8	90.5	60.0	41.9	3.3	-	14.5	62.9	24.3
RetinaNet	91.5	98.0	96.9	54.8	92.0	55.1	41.9	8.0	-	4.9	51.4	22.9
DETR	43.7	59.3	55.5	0.0	43.9	41.6	23.2	0.5	17.4	**0.1**	17.9	**46.0**
DINO	**95.8**	**99.0**	**98.3**	**58.2**	**96.3**	208.0	195.2	0.7	11.5	0.7	237.0	10.4
TOOD	94.1	98.6	97.9	58.0	94.6	53.4	44.2	3.9	-	5.2	35.3	26.3
SoyStageNet	83.2	95.1	92.0	46.2	83.9	**17.3**	**13.2**	**2.9**	-	1.2	**12.2**	28.5

Table 8. Comparison with state-of-the-art object detectors on soybean growth stage detection. B, N, E+D, and H denote backbone, neck, encoder-decoder, and head parameters in millions, respectively.

with particularly substantial gains at higher IoU thresholds (+3.9% AP_{75}) and for medium-sized instances (+6.4% AP_M). While the NAS-FCOS head offers computational advantages (10.68G vs. 21.11G FLOPs, 30.0 vs. 28.9 FPS), the substantial performance improvement justifies the additional computation for T-head. This confirms our design choice of prioritizing task alignment for accurate soybean growth stage detection.

Impact of Backbone Capacity. We investigate the effect of scaling the MiT backbone capacity while maintaining the NAS-FCOS FPN neck and T-head detector. Table 6 presents results across six backbone variants. Performance generally increases with backbone capacity, from 82.9% AP with MiT-B0 to 85.6% AP with MiT-B5, demonstrating the benefits of stronger feature extraction. However, the gains diminish with larger models—MiT-B1 to MiT-B2 shows only +0.3% AP improvement despite a 84% increase in parameters. Interestingly, scale-specific performance reveals different patterns: AP_M peaks at MiT-B3 (50.1%) before declining with larger models, while AP_L continues improving up to MiT-B5 (86.3%). This suggests that very large backbones may overfit to large instances at the expense of medium-sized objects. For SoyStageNet, we identify MiT-B1 as the optimal backbone, achieving 84.5% AP and exceptional high-IoU performance (93.3% AP_{75}) with reasonable computational requirements (13.15M parameters, 27.8 FPS). This configuration achieves 98.7% of MiT-B5's accuracy while using only 16% of the parameters.

Impact of Feature Channel Dimensions. Finally, we examine the effect of feature channel dimensionality in the NAS-FCOS FPN with the MiT-B1 backbone and T-head detector. Table 7 presents comparative results. Increasing channel dimensions from 64 to 128 yields substantial improvements (+4.0% AP, +4.5% AP_{75}), while further increasing to 256 channels shows diminishing returns (+1.3% AP) at significantly higher computational cost. Both neck and head parameters scale quadratically with channel dimensions—the 256-channel configuration requires 3.4× more neck parameters and 4.0× more head parameters than the 128-channel variant. The 128-channel configuration achieves 98.5% of the 256-channel variant's performance while using only 28% of the combined neck and head parameters. This confirms that moderate feature dimensionality provides sufficient representational capacity while maintaining computational efficiency.

Final Architecture Configuration. Based on these comprehensive ablation studies, we select the MiT-B1 backbone with NAS-FCOS FPN (128 channels) and T-head as our final SoyStageNet architecture, offering an excellent balance between accuracy (83.2% AP) and efficiency (28.5 FPS) for practical agricultural applications. This configuration reduces the total model parameters by 38% compared to the 256-channel variant while sacrificing only 1.3 percentage points in AP.

4.3. Comparison with State-of-the-Art Methods

We evaluate SoyStageNet against established state-of-the-art object detectors including transformer-based models (DETR, DINO) and CNN-based detectors (RetinaNet, Faster R-CNN, TOOD). All models are trained on our soybean growth stage dataset using identical train-test splits and optimization settings. Quantitative results are presented in Tab. 8. Transformer-based DINO achieves the highest accuracy (95.8% AP) across all metrics, with particularly strong performance on medium-sized instances (58.2% AP_M) which are often challenging in agricultural settings. However, this comes at substantial computational cost (208M parameters, 237G FLOPs) and reduced inference speed (10.4 FPS), making it impractical for deployment on resource-constrained agricultural platforms. Similarly, CNN-based TOOD achieves excellent accuracy (94.1% AP) but with high computational requirements (53.4M parameters).

In contrast, SoyStageNet achieves a highly competitive 83.2% AP and 46.2% AP_M with only 17.3M parameters and 12.2G FLOPs—representing an 87% reduction in parameters and 95% reduction in FLOPs compared to DINO. Our model operates at 28.5 FPS, nearly 3× faster than DINO and faster than all tested CNN-based detectors. This significant efficiency advantage comes from our optimized architecture with task-aligned transformer components and carefully balanced feature dimensionality. The performance gap between SoyStageNet and larger models is most pronounced on medium-sized instances (46.2% vs. 58.2% AP_M for DINO), while maintaining strong performance on large instances (83.9% AP_L).

For real-world agricultural applications requiring on-field processing with resource constraints, SoyStageNet of-

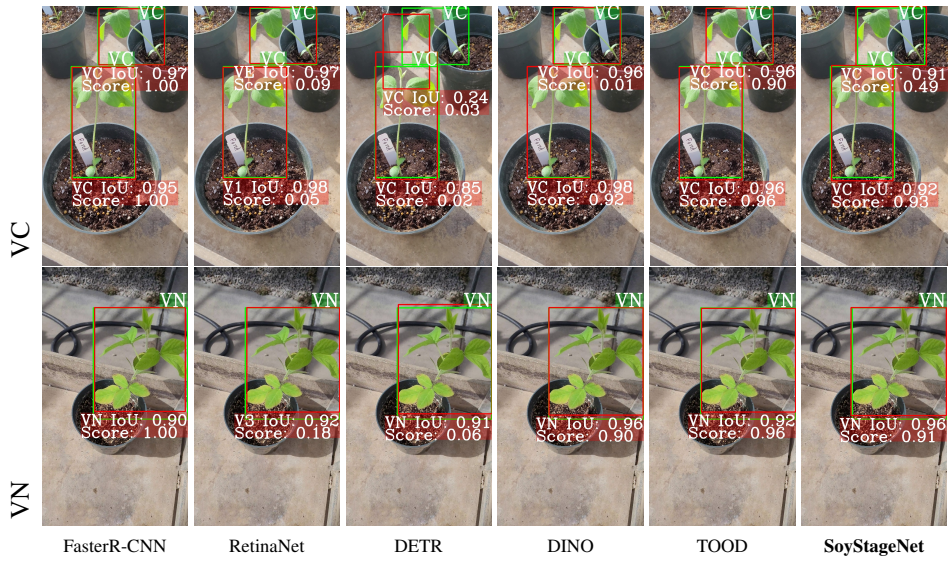

Figure 2. Qualitative comparison of detection results on soybean growth stages (VC and VN). Red boxes show model predictions, green boxes indicate ground truth annotations. Despite its efficiency advantage, SoyStageNet achieves accurate detection with high IoU values comparable to more complex models.

fers the optimal balance between accuracy and efficiency. While larger models achieve marginally higher accuracy, SoyStageNet's substantially lower computational footprint and higher inference speed make it far more suitable for practical deployment in precision agriculture scenarios where real-time growth stage monitoring is essential for timely interventions.

4.4. Qualitative Comparison

Figure 2 presents qualitative detection results comparing SoyStageNet with state-of-the-art models on two representative soybean growth stages (VC and VN). While Faster R-CNN and TOOD display high confidence scores (1.00 and 0.96 respectively) in VC, they require substantially more parameters. RetinaNet achieves high IoU values but struggles with low confidence scores (0.09, 0.05), indicating uncertainty in classification despite accurate localization. DETR demonstrates poor confidence in its predictions (0.03, 0.06) despite reasonable IoU, suggesting difficulties in distinguishing between similar growth stages. Our SoyStageNet achieves competitive detection quality with high IoU values (0.91-0.96) and robust confidence scores (0.49-0.96) across both growth stages. This performance is particularly noteworthy considering SoyStageNet's significantly smaller model size (17.3M parameters versus 208M for DINO). These results demonstrate SoyStageNet's ability to balance detection quality and efficiency, making it ideal for real-time agricultural monitoring applications.

5. Conclusion

We have demonstrated that transformer-based architectures can be effectively optimized for agricultural applications through our SoyStageNet model. By addressing the task-specific challenges of soybean growth stage detection through careful component selection and optimization, we have shown that deep learning can provide practical solutions for precision agriculture. Our extensive ablation studies reveal important design considerations for deploying computer vision systems in agricultural settings, where computational resources are often limited. The principles demonstrated in this work extend beyond soybeans and could be applied to other crops where timely interventions depend on accurate growth stage identification. Future work will focus on extending the model to additional crop varieties and exploring knowledge distillation techniques to further reduce computational requirements without sacrificing detection accuracy. Additionally, we plan to expand our dataset, which currently covers only six vegetative classes, by labeling more growth stages — including the reproductive growth stages — to improve the model's generalization across the full soybean growth cycle.

Acknowledgment

This work is supported by the Soybean Center at the Southern Illinois University Carbondale.

Data Availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

[1] Abeer M Almalky and Khaled R Ahmed. Real time deep learning algorithm for counting weed's growth stages. In

2023 IEEE 15th International Symposium on Autonomous Decentralized System (ISADS), pages 1–6. IEEE, 2023. 2

[2] Alexander Buslaev, Vladimir I Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A Kalinin. Albumentations: fast and flexible image augmentations. *Information*, 11(2):125, 2020. 6

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 5

[4] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Y Cao, Y Xiong, X Li, S Sun, W Feng, Z Liu, J Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. arxiv 2019. *arXiv preprint arXiv:1906.07155*, 1906. 5

[5] Kuan-I Chen, Mei-Hui Erh, Nan-Wei Su, Wen-Hsiung Liu, Cheng-Chun Chou, and Kuan-Chen Cheng. Soyfoods and soybean products: from traditional use to modern applications. *Applied microbiology and biotechnology*, 96:9–22, 2012. 1

[6] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 4

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3

[8] Chengjian Feng, Yujie Zhong, Yu Gao, Matthew R Scott, and Weilin Huang. Tood: Task-aligned one-stage object detection. In *ICCV*, 2021. 3, 4, 5

[9] Haotian He, Xiaodan Ma, Haiou Guan, Feiyi Wang, and Panpan Shen. Recognition of soybean pods and yield prediction based on improved deep learning model. *Frontiers in plant science*, 13:1096619, 2023. 2

[10] Deepak R Joshi, Sharon A Clay, Prakriti Sharma, Hossein Moradi Rekabdarkolaee, Tulsi Kharel, Donna M Rizzo, Resham Thapa, and David E Clay. Artificial intelligence and satellite-based remote sensing can be used to predict soybean (glycine max) yield. *Agronomy Journal*, 116(3):917–930, 2024. 2

[11] Jinyang Li, Qingda Li, Chuntao Yu, Yan He, Liqiang Qi, Wenqiang Shi, and Wei Zhang. A model for identifying soybean growth periods based on multi-source sensors and improved convolutional neural network. *Agronomy*, 12(12): 2991, 2022. 2

[12] Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. *Advances in neural information processing systems*, 33:21002–21012, 2020. 6

[13] Zhiyu Li, Jianping Wang, Guohong Gao, Yufeng Lei, Chenping Zhao, Yan Wang, Haofan Bai, Yuqing Liu, Xiaojuan Guo, and Qian Li. Sgsnet: a lightweight deep learning model for strawberry growth stage detection. *Frontiers in Plant Science*, 15:1491706, 2024. 2

[14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pages 740–755. Springer, 2014. 6

[15] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 4

[16] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2017. 5

[17] Jelena Medic, Christine Atkinson, and Charles R Hurburgh. Current knowledge in soybean composition. *Journal of the American oil chemists' society*, 91:363–384, 2014. 1

[18] Daren Mueller. Pre-harvest restrictions for fungicides, 2008. 1

[19] Scott W Myers, David B Hogg, and John L Wedberg. Determining the optimal timing of foliar insecticide applications for control of soybean aphid (hemiptera: Aphididae) on soybean. *Journal of Economic Entomology*, 98(6):2006–2012, 2005. 1

[20] Matheus Mereb Negrisoli, Flávio Nunes da Silva, Raphael Mereb Negrisoli, Lucas da Silva Lopes, Francisco de Sales Souza Júnior, Bianca Rezende de Freitas, Edivaldo Domingues Velini, and Carlos Gilberto Raetano. Impact of fungicide application timing based on soybean rust prediction model on application technology and disease control. *Agronomy*, 12(9):2119, 2022. 2

[21] Xindong Ni, Faming Wang, Hao Huang, Ling Wang, Changkai Wen, and Du Chen. A cnn-and self-attention-based maize growth stage recognition method and platform from uav orthophoto images. *Remote Sensing*, 16(14):2672, 2024. 2

[22] A Paszke. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019. 5

[23] Larry C Purcell, Montserrat Salmeron, and Lanny Ashlock. Soybean growth and development. *Arkansas soybean production handbook*, 197:1–8, 2014. 1

[24] Osvaldo Pérez, Brian Diers, and Nicolas Martin. Maturity prediction in soybean breeding using aerial images and the random forest machine learning algorithm. *Remote Sensing*, 16(23), 2024. 2

[25] Sanaz Rasti, Chris J Bleakley, Guénolé CM Silvestre, Nicholas M Holden, David Langton, and Gregory MP O'Hare. Crop growth stage estimation prior to canopy closure using deep learning algorithms. *Neural Computing and Applications*, 33(5):1733–1743, 2021. 2

[26] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 2

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 5

[28] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019. 6

[29] Ondulla T. Toomer, Edgar Orlando Oviedo-Rondón, Muhammad Ali, Michael Joseph, Thien Vu, Ben Fallen, and Rouf Mian. Full-fat soybean meals as an alternative poultry feed ingredient—feed processing methods and utilization—review and perspective. *Animals*, 14(16), 2024. 1

[30] U.S. Department of Agriculture, National Agricultural Statistics Service. Crop Production, 2024. 1

[31] Rene C Van Acker, Clarence J Swanton, and Stephan F Weise. The critical period of weed control in soybean [glycine max (l.) merr.]. *Weed Science*, 41(2):194–200, 1993. 1

[32] Chunying Wang, Xubin Song, Weiting Pan, Haixia Yu, Xiang Li, and Ping Liu. Determination of wheat growth stages using image sequences and deep learning. *Agronomy*, 15(1), 2025. 2

[33] Ning Wang, Yang Gao, Hao Chen, Peng Wang, Zhi Tian, Chunhua Shen, and Yanning Zhang. Nas-fcos: Fast neural architecture search for object detection. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11943–11951, 2020. 3, 4

[34] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 568–578, 2021. 3

[35] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in neural information processing systems*, 34: 12077–12090, 2021. 3

[36] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M. Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection, 2022. 5

[37] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9759–9768, 2020. 5

[38] Ying Zhou, Weiwei Zhao, Yong Lai, Baohong Zhang, and Dangquan Zhang. Edible plant oil: global status, health issues, and perspectives. *Frontiers in Plant Science*, 11:1315, 2020. 1