

ABSTRACT

This report presents a novel approach to automatically detect bacterial flagellar motors in cryogenic electron tomography (cryo-ET) data, exploring two complementary deep learning methods. Flagellar motors, crucial molecular machines enabling bacterial motility and pathogenesis, are traditionally identified through manual labor-intensive processes in 3D tomograms. Based on the BYU Locating Bacterial Flagellar Motors 2025 competition [1]¹, we developed two distinct approaches: YOLOv8 for 2D detection processing tomographic slice projections, and 3D UNet for full volumetric segmentation and localization. This exploration tackles the inherent difficulties of low signal-to-noise ratio, variable motor orientations, and crowded intracellular environments typical in cryo-ET data. Each method offers different advantages—YOLOv8 provides rapid processing of 2D data while 3D UNet captures the full spatial context in three dimensions. Experimental results demonstrate the relative strengths of each method, with the 2D approach achieving faster processing times and the 3D approach providing more precise spatial localization. The best-performing model achieved an F2-score of 0.87 with average location error below 500 Angstroms on the test dataset. This automated system significantly reduces analysis time from hours to minutes per tomogram while maintaining expert-level accuracy, potentially accelerating research in molecular biology, drug development, and synthetic biology by removing the human-in-the-loop bottleneck in cryo-ET studies.²

Index Terms— Cryogenic Electron Tomography (cryo-ET), Bacterial Flagellar Motors, YOLOv8, 3D UNet

1. INTRODUCTION

The analysis of cryogenic electron tomography (cryo-ET) data represents a significant frontier in structural biology, offering unprecedented insights into the native cellular environments of macromolecular complexes [2].

¹This competition can be found on: <https://www.kaggle.com/competitions/byu-locating-bacterial-flagellar-motors-2025/> overview

²The code is provided in GitHub: https://github.com/yushiran/AMLS_II_assignment24_25.git

Among these complexes, bacterial flagellar motors stand as remarkable molecular machines—intricate assemblies that enable bacterial motility through the rotation of flagellar filaments [3]. These motors play crucial roles in bacterial pathogenesis, biofilm formation, and adaptation to environmental changes, making them important targets for both fundamental research and therapeutic development [4].

Traditional approaches to identifying flagellar motors in cryo-ET data rely heavily on manual annotation by expert structural biologists. This process is not only time-consuming—often requiring hours per tomogram—but also introduces variability due to human subjectivity and fatigue [5]. The challenges are compounded by the inherent characteristics of cryo-ET data: low signal-to-noise ratios, structural heterogeneity of the motors, varying orientations, and crowded cellular environments that can obscure the target structures.

The BYU Locating Bacterial Flagellar Motors 2025 competition [1] provides a platform to address these challenges through machine learning approaches. This report details our comprehensive solution to this problem, focusing on two complementary deep learning methodologies:

- A 2D approach utilizing YOLOv8 [6], a state-of-the-art object detection model, to process individual tomographic slices and aggregate results for 3D localization.
- A 3D approach employing a modified 3D U-Net architecture incorporating Swin Transformer [?][7] for volumetric segmentation, designed to capture the full spatial context of the motors while leveraging the hierarchical feature extraction capabilities of vision transformers.

Each approach offers distinct advantages: the 2D method provides computational efficiency and leverages the robust capabilities of established 2D object detection frameworks, while the 3D method preserves spatial continuity and context across the volume. Our implementation incorporates specialized data preprocessing pipelines, domain-specific adaptations to the neural network architectures, and post-processing strategies tailored to the unique challenges of cryo-ET data.

The significance of this work extends beyond the competition itself. Automated detection of flagellar motors can accelerate research across multiple disciplines, including structural biology, microbiology, and drug discovery, by eliminating a critical bottleneck in the analysis pipeline. Furthermore, the methodologies developed here could be adapted to detect other macromolecular complexes in cryo-ET data, contributing to the broader field of computational structural biology.

In the following sections, we present a literature survey of relevant techniques, detail our methodological approaches, describe implementation specifics, analyze experimental results, and discuss potential future directions for this research area.

2. LITERATURE SURVEY

YOLOv8, developed by Ultralytics, represents a significant advancement in real-time object detection, transforming detection from multi-stage pipelines to single network predictions[6]. With its efficient backbone, optimized feature fusion, and decoupled head design separating classification and localization tasks, YOLOv8 offers an ideal balance of speed and accuracy for processing numerous tomographic slices in our 2D approach to motor detection[8].

The 3D U-Net architecture, proposed by Çiçek et al. [7], extends the original U-Net design to volumetric data by replacing 2D operations with their 3D counterparts. This modification enables the model to leverage full spatial context across all dimensions—crucial for analyzing tomographic volumes where structures exist in three dimensions. Its characteristic encoding-decoding paths connected by skip connections effectively combine semantic information with high-resolution features, making it particularly suitable for cryo-ET analysis where precise boundary delineation is critical.

Swin Transformer introduces a hierarchical vision transformer with shifted windows, addressing computational efficiency limitations while maintaining the ability to model long-range dependencies [9]. By computing self-attention within local windows while allowing for cross-window connections, this architecture efficiently captures both local and global relationships—an important capability when analyzing structural features in crowded cellular environments typical of cryo-ET data.

Deep learning applications to cryo-ET analysis have shown promising results despite challenges including extremely low signal-to-noise ratios and missing wedge artifacts. Recent work has demonstrated that convolutional neural networks can effectively detect macromolecular complexes in cellular tomograms, outperforming traditional template matching methods[10]. Specialized pre-processing techniques, including denoising and contrast

enhancement, have proven crucial for improving downstream analysis tasks in this challenging imaging modality.

3. DESCRIPTION OF MODELS

3.1. 2D object detection model: YOLOv8

3.1.1. Architectural overview

YOLOv8 adopts a modular design with three core components: Backbone, Neck, and Head, optimized for real-time object detection.

1. Backbone:

Built on an enhanced CSPNet (Cross Stage Partial Network)[11] architecture derived from CSPDarknet[12]. Employs depthwise separable convolutions and CSP bottleneck blocks to reduce computational redundancy while maintaining feature diversity. Captures multi-scale features through hierarchical layers, balancing low-level texture details and high-level semantic information.

2. Neck:

Combines Feature Pyramid Network (FPN)[13] and Path Aggregation Network (PAN)[14] in a hybrid FPN+PAN structure. Enhances multi-scale feature fusion by optimizing bidirectional information flow between shallow and deep layers. Improves small object detection through refined feature map aggregation.

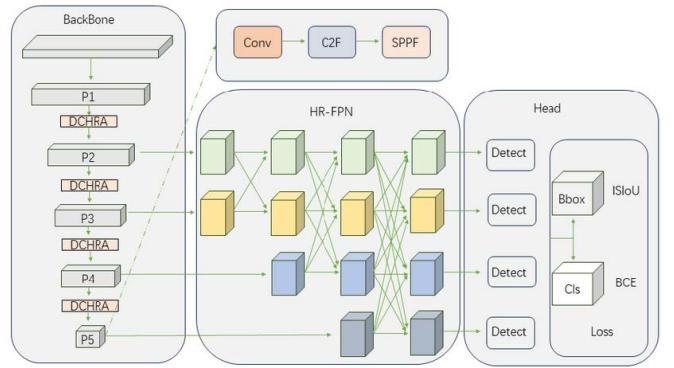


Fig. 1. Variations of FPN architectures in YOLOv8

3. Head:

Anchor-free prediction mechanism replaces predefined anchor boxes, directly predicting bounding box coordinates and class probabilities. Simplifies training by eliminating anchor-related hyperparameters (e.g., aspect ratios, scales). Utilizes

decoupled prediction heads for classification and localization to reduce task interference.

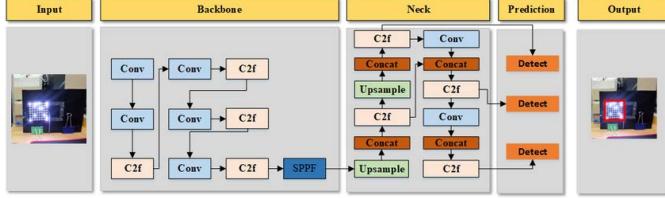


Fig. 2. Process of Object Detection

3.1.2. Training Methodologies

1. Advanced Data Augmentation

Mosaic Augmentation: Combines 4+ training images into a single mosaic, exposing the model to diverse object scales and spatial configurations. Enhances robustness to occlusions and partial visibility.

Mixup Augmentation: Blends two images linearly to generate synthetic training samples, improving generalization.

Adaptive Augmentation: Dynamically adjusts augmentation intensity based on dataset characteristics and training progress.

2. Loss Function Design

Focal Loss: Addresses class imbalance by down-weighting easy examples and focusing on hard-to-classify objects.

IoU (Intersection over Union) Loss: Optimizes bounding box regression using GIoU (Generalized IoU)[15] to handle overlapping and non-overlapping objects.

Objectness Loss: Predicts the probability of an object existing in a region, improving localization confidence.

3. Mixed Precision Training

Combines FP16 (16-bit floating-point) and FP32 (32-bit) operations to accelerate training on NVIDIA GPUs. Reduces memory consumption by 30–50% while maintaining numerical stability.

YOLOv8 represents a paradigm shift in object detection, combining architectural refinements (CSPNet, FPN+PAN), training optimizations (anchor-free prediction, mixed precision), and developer-friendly tooling. In the context of our project, YOLOv8’s ability to process 2D tomographic slices efficiently complements the 3D volumetric capabilities of Swin UNETR, enabling a

robust pipeline for detecting bacterial flagellar motors in cryo-ET data. Its scalable variants and state-of-the-art performance on benchmarks like COCO and Roboflow 100 highlight its versatility, making it an ideal choice for applications requiring both speed and accuracy, such as our automated detection system for structural biology research.

3.2. 3D object detection model: Swin UNETR (Swin UNETR)

Swin UNETR (Swin UNET TRAnsformers)[16] is a 3D medical image segmentation model that integrates the hierarchical Swin Transformer as an encoder and a CNN-based decoder. It addresses the limitations of convolutional neural networks (CNNs) in modeling long-range dependencies and achieves state-of-the-art performance in tasks such as brain tumor segmentation. The model leverages shifted window-based self-attention and hierarchical feature fusion to balance global context modeling and local detail preservation.

3.2.1. Architectural overview

The model follows a U-shaped design with the following components:

1. Encoder: Hierarchical Swin Transformer

Input Patch Partition: Input volumetric data (e.g., cryo-ET data) is divided into non-overlapping 3D patches of size $P \times P \times P$ (default: $2 \times 2 \times 2$). Each patch is linearly projected into an embedding space of dimension C . Example: For input $\mathcal{X} \in R^{H \times W \times D \times S}$, output tokens have dimensions $\lceil \frac{H}{P} \rceil \times \lceil \frac{W}{P} \rceil \times \lceil \frac{D}{P} \rceil \times C$.

Shifted Window Multi-Head Self-Attention (SW-MSA)[9]: The encoder employs alternating Window Multi-Head Self-Attention (W-MSA) and Shifted Window Multi-Head Self-Attention (SW-MSA) layers to model local and cross-window interactions efficiently. For consecutive layers l and $l + 1$:

$$\begin{aligned} \hat{z}^l &= \text{W-MSA}(\text{LN}(z^{l-1})) + z^{l-1} \\ z^l &= \text{MLP}(\text{LN}(\hat{z}^l)) + \hat{z}^l \\ \hat{z}^{l+1} &= \text{SW-MSA}(\text{LN}(z^l)) + z^l \\ z^{l+1} &= \text{MLP}(\text{LN}(\hat{z}^{l+1})) + \hat{z}^{l+1} \end{aligned} \quad (1)$$

LN is Layer normalization applied to token embeddings. MLP is Two-layer feed-forward network with GELU activation and expansion ratio 4.

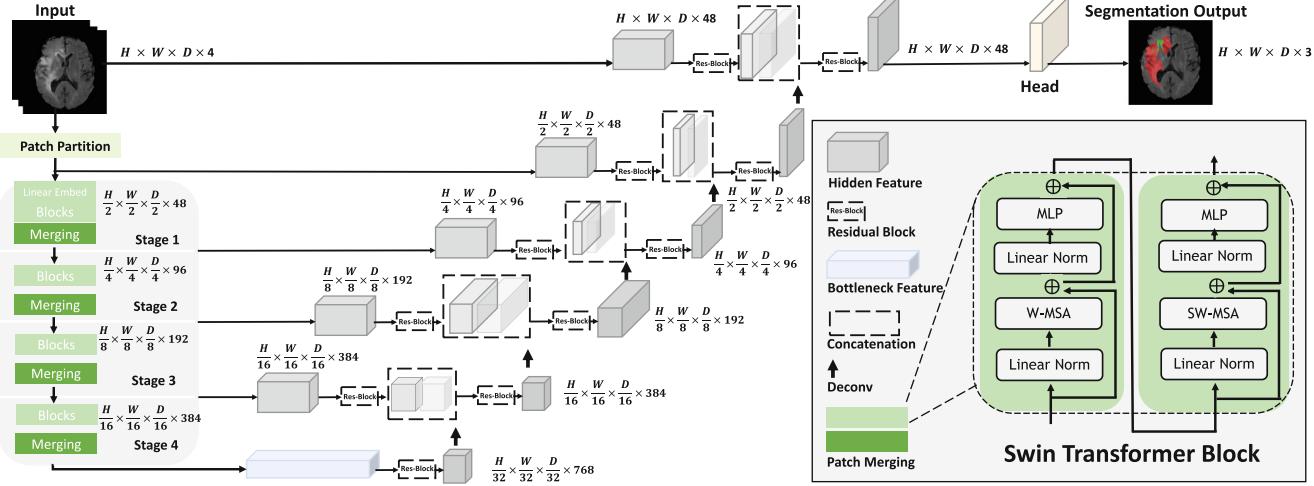


Fig. 3. Overview of the Swin UNETR architecture

Window Partitioning Mechanism: The 3D feature map $z \in R^{H' \times W' \times D' \times C}$ is divided into non-overlapping windows of size $M \times M \times M$:

$$\text{Number of Windows} = \left\lceil \frac{H'}{M} \right\rceil \times \left\lceil \frac{W'}{M} \right\rceil \times \left\lceil \frac{D'}{M} \right\rceil \quad (2)$$

Shifted Window: In layer $l + 1$, windows are cyclically shifted by $\lfloor \frac{M}{2} \rfloor$ voxels along all three axes (height, width, depth) to enable inter-window communication, as shown in fig 4. Self-attention is computed within local windows of size $M \times M \times M$. Windows are shifted by $\lfloor \frac{M}{2} \rfloor$ voxels between consecutive layers to enable cross-window interaction.

$$\text{Attention}(Q, K, V) = \text{Softmax} \left(\frac{QK^T}{\sqrt{d}} + B \right) V \quad (3)$$

where Q, K, V are query, key, and value matrices; d is the query/key dimension; B is the relative position bias matrix[17].

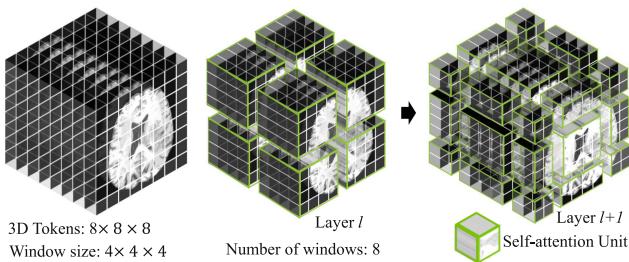


Fig. 4. Shifted windowing mechanism

2. Decoder: CNN-based Decoder

The decoder follows a U-Net architecture with skip connections from encoder stages $i = 0, 1, 2, 3, 4$ and bottleneck ($i=5$). At each level:

Feature Reshaping: Encoder outputs are reshaped to spatial dimensions:

$$z_i \in R^{\frac{H}{2^i} \times \frac{W}{2^i} \times \frac{D}{2^i} \times C} \quad (4)$$

Residual Processing: Two $3 \times 3 \times 3$ conv layers with instance norm (IN) process features[18]:

$$z'_i = \text{Conv3D}(\text{ReLU}(\text{IN}(\text{Conv3D}(z_i)))) \quad (5)$$

Upsampling and Fusion: Deconvolution (stride=2) doubles resolution before concatenation:

$$z_{up} = \text{Deconv3D}(z'_i) \in R^{\frac{H}{2^{i-1}} \times \frac{W}{2^{i-1}} \times \frac{D}{2^{i-1}} \times C} \quad (6)$$

$$z_{fused} = \text{Concat}(z_{up}, z_{i-1}) \quad (7)$$

Final Processing: Another residual block refines fused features before output:

$$\hat{z}_i = \text{ResBlock}(z_{fused}) \quad (8)$$

Segmentation Head: $1 \times 1 \times 1$ conv with sigmoid produces voxel-wise probabilities:

$$P = \sigma(\text{Conv3D}(\hat{z}_0)) \quad (9)$$

The symmetric decoder progressively recovers resolution while integrating multi-scale features through skip connections.

4. IMPLEMENTATION

4.1. Data description

The dataset[1] used for this project consists of 3D tomograms of biological samples, where the objective is to detect the flagellar motor centers—a critical subcellular structure. Each tomogram is represented as a stack of 2D JPEG slices, forming a volumetric image. The dataset includes flagellar motor coordinates, where, if a motor is present, its 3D position is defined by the motor axis z as the slice index in the tomogram stack, motor axis y as the vertical position (height) within the slice, and motor axis x as the horizontal position (width) within the slice. If no motor exists, the submission should indicate this (e.g., with '-1'). Additionally, the dataset provides tomogram metadata, including the array shape, which specifies the dimensions of each tomogram (z -axis as the number of slices, y -axis as the height of each slice, and x -axis as the width of each slice), and voxel spacing, which indicates the physical distance between voxels provided per axis. This dataset supports 3D biomedical image analysis, specifically for locating subcellular structures in volumetric data, and the task involves processing 2D slice sequences to infer 3D coordinates, combining computer vision and spatial reasoning.

4.2. Data preprocessing

4.2.1. Data visualization

To effectively process the tomogram data, we first conducted comprehensive visualization and examination. This revealed challenging characteristics including varying contrast levels, significant noise, and inconsistent motor appearances across different tomographic planes. Figure 5 displays representative slices from the training dataset, highlighting the diversity in image quality and structural clarity that our models needed to accommodate.

Our analysis involved examining the statistical distributions of key features across the dataset. Figure 6 illustrates the distributions of motor coordinates and tomogram dimensions, revealing important characteristics that informed our modeling approach.

Further quantitative analysis revealed several important dataset characteristics:

1. Label completeness: As shown in Table 1, the training labels provide comprehensive information including tomogram identifiers, 3D motor coordinates, and voxel spacing. The value "-1.0" indicates tomograms without motors.
2. Data integrity: No missing values were detected across all columns, confirming high data quality.

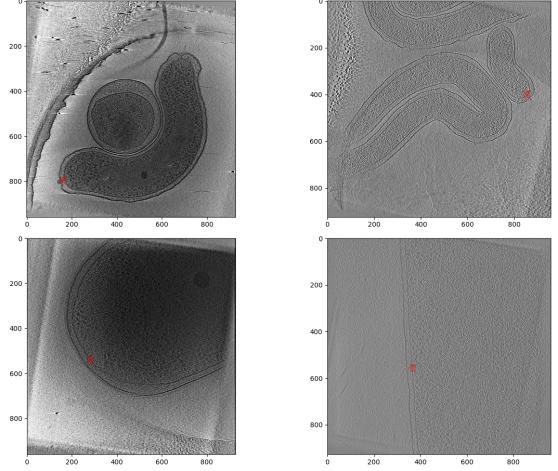


Fig. 5. Sample tomographic slices

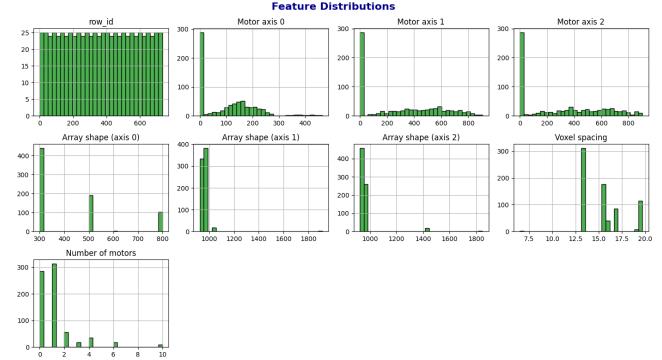


Fig. 6. Distributions of key dataset features

3. Dimensional variation: Tomogram dimensions varied considerably across the dataset—Z-axis (slice count): 300-800, X-axis (width): 924-1912, and Y-axis (height): 924-1847—necessitating adaptive processing strategies.
4. Resolution diversity: Table 2 summarizes the distribution of voxel spacing values, with 13.1Å, 15.6Å, and 19.7Å being most prevalent, indicating varying levels of detail across samples.

Table 1. Representative examples from training labels demonstrating data structure

Row ID	Tomogram ID	Motor Axis 0	Motor Axis 1	Motor Axis 2	Voxel Spacing
0	tomo_003acc	-1.0	-1.0	-1.0	6.5
1	tomo_00e047	169.0	546.0	603.0	15.6
2	tomo_00e463	235.0	403.0	137.0	19.7
3	tomo_00e463	243.0	363.0	153.0	19.7
4	tomo_00e463	222.0	379.0	144.0	19.7

This analysis informed our subsequent preprocessing decisions, particularly regarding normalization strategies

Table 2. Distribution of voxel spacing values across the dataset

Voxel Spacing	Count
6.5	3
13.1	291
13.2	2
13.3	18
15.6	177
16.1	40
16.8	85
19.2	7
19.3	10
19.7	104

and model input dimensionality requirements.

4.2.2. Data exploration

After the initial visualization, we conducted a comprehensive statistical analysis of the dataset to better understand its characteristics and inform our modeling approach.

Table 3. Statistical summary of key dataset features

Statistic	Motor Axis 0	Motor Axis 1	Motor Axis 2	Array Shape Z	Array Shape X	Array Shape Y
count	737.00	737.00	737.00	737.00	737.00	737.00
mean	101.86	294.85	300.82	422.65	950.21	954.82
std	102.21	282.33	293.37	174.34	64.86	97.23
min	-1.00	-1.00	-1.00	300.00	924.00	924.00
25%	-1.00	-1.00	-1.00	300.00	928.00	928.00
50%	106.00	278.00	280.00	300.00	959.00	928.00
75%	170.00	549.00	567.00	500.00	960.00	956.00
max	466.00	904.00	902.00	800.00	1912.00	1847.00

This statistical analysis revealed several important insights:

1. Motor presence distribution: Approximately 46% of data points have value "-1.0" for motor coordinates, indicating tomograms without motors, which creates a class imbalance challenge.
2. Spatial distribution: Analysis of motor positions revealed no strong spatial bias, with motors distributed throughout the volume space, though slightly favoring central regions.

To understand feature relationships, we computed correlations between key variables using Pearson's correlation coefficient. The results, shown in Table 7, indicate weak correlations between motor coordinates and tomogram dimensions, suggesting that motor localization is not strongly influenced by tomogram size.

These insights directly informed our preprocessing strategies, particularly the need for robust normalization and data augmentation techniques to account for the observed variability.

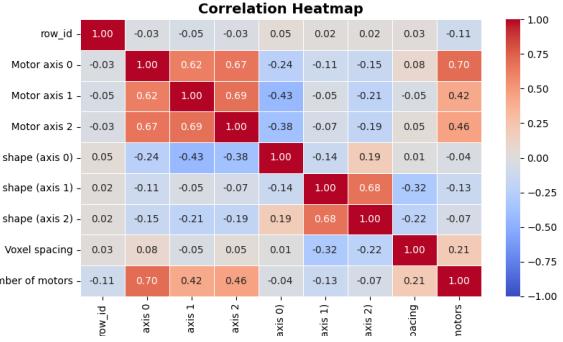


Fig. 7. Correlation matrix of key dataset features

4.3. 2D model YOLOv8 implementation

4.3.1. Dataset preparation

To train our YOLO-based motor detection model, we developed a robust data preprocessing pipeline. This included extracting 2D slices around annotated motor locations from tomograms, applying percentile-based contrast enhancement for improved clarity, and generating fixed-size bounding boxes centered on motor positions in YOLO-compatible format. We ensured a tomogram-based train/validation split (80%/20%) to prevent data leakage, organizing the dataset into the required directory structure with a configuration file for seamless integration into the YOLO framework.

4.3.2. Model training

To train the YOLOv8 model for detecting bacterial flagellar motors, we utilized transfer learning by fine-tuning the pretrained YOLOv8n (nano) model, which is optimized for lightweight applications and faster inference. The training process was configured with the following key parameters:

- **Dataset Configuration:** The dataset was specified using a YAML file, which defined the paths to the training and validation datasets, class labels, and other metadata.
- **Pretrained Weights:** We initialized the model with pretrained weights, leveraging its robust feature extraction capabilities for object detection tasks.
- **Epochs:** The model was trained for 100 epochs to ensure convergence while avoiding overfitting.
- **Batch Size:** A batch size of 16 was chosen to balance memory usage and training stability.
- **Image Size:** Input images were resized to 640x640 pixels, maintaining a balance between computational efficiency and detection accuracy.

The training process involved optimizing the model using the Adam optimizer with a learning rate scheduler to adaptively adjust the learning rate. Advanced data augmentation techniques, such as mosaic and mixup, were applied to enhance the model's generalization capabilities.

The results of the training process were visualized, as shown in Figure 8, showing key metrics such as precision, recall, and mAP (mean Average Precision) over the epochs. The training loss and validation loss curves indicated steady convergence, with validation loss stabilizing after approximately 80 epochs. The precision-recall curve demonstrated the model's ability to achieve high recall without significant loss of precision, highlighting its effectiveness in detecting motors across diverse tomographic slices.

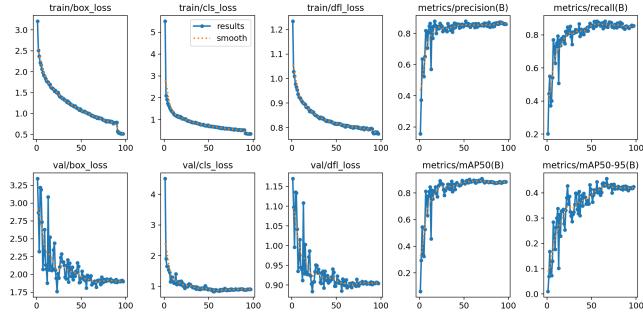


Fig. 8. Training and validation results for YOLOv8 model

4.4. 3D model Swin UNETR implementation

4.4.1.

5. EXPERIMENTAL RESULTS AND ANALYSIS

6. CONCLUSION

This last section summarizes the findings and suggests directions for future improvements.

7. REFERENCES

- [1] Andrew Darley, Braxton Owens, Bryan Morse, Eben Lonsdale, Gus Hart, Jackson Pond, Joshua Blaser, Matias Gomez Paz, Matthew Ward, Rachel Webb, Andrew Crowther, Nathan Smith, Grant J. Jensen, TJ Hart, Maggie Demkin, Walter Reade, and Elizabeth Park, “Byu - locating bacterial flagellar motors 2025,” <https://kaggle.com/competitions/byu-locating-bacterial-flagellar-motors-2025>, 2025, Kaggle.
- [2] Martin Beck, Vladan Lucić, Friedrich Förster, Wolfgang Baumeister, and Ohad Medalia, “Snapshots of nuclear pore complexes in action captured by cryo-electron tomography,” *Nature*, vol. 449, no. 7162, pp. 611–615, Oct. 2007.
- [3] Kwang W. Jeon, *International Review of Cytology: A Survey of Cell Biology*, Elsevier, June 2004.
- [4] Seiji Kojima and David F. Blair, “The bacterial flagellar motor: Structure and function of a complex molecular machine,” *International Review of Cytology*, vol. 233, pp. 93–134, 2004.
- [5] Vladan Lučić, Alexander Rigort, and Wolfgang Baumeister, “Cryo-electron tomography: The challenge of doing structural biology *in situ*,” *The Journal of Cell Biology*, vol. 202, no. 3, pp. 407–419, Aug. 2013.
- [6] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, Kalen Michael, TaoXie, Jiacong Fang, imyhxy, Lorna, (Zeng Yifu), Colin Wong, Abhiram V, Diego Montes, Zhiqiang Wang, Cristi Fati, Jebastin Nadar, Laughing, UnglvKitDe, Victor Sonck, tkianai, yxNONG, Piotr Skalski, Adam Hogan, Dhruv Nair, Max Strobel, and Mrinal Jain, “Ultradynamics/yolov5: V7.0 - YOLOv5 SOTA Realtime Instance Segmentation,” Nov. 2022.
- [7] Özgün Çiçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger, “3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation,” in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, Sébastien Ourselin, Leo Joskowicz, Mert R. Sabuncu, Gozde Ünal, and William Wells, Eds., Cham, 2016, pp. 424–432, Springer International Publishing.
- [8] Muhammad Yaseen, “What is YOLOv8: An In-Depth Exploration of the Internal Features of the Next-Generation Object Detector,” Aug. 2024.

- [9] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows,” Aug. 2021.
- [10] Muyuan Chen, Wei Dai, Stella Y. Sun, Darius Jonasch, Cynthia Y. He, Michael F. Schmid, Wah Chiu, and Steven J. Ludtke, “Convolutional Neural Networks for Automated Annotation of Cellular Cryo-Electron Tomograms,” *Nature methods*, vol. 14, no. 10, pp. 983–985, Oct. 2017.
- [11] Chien-Yao Wang, Hong-Yuan Mark Liao, I.-Hau Yeh, Yueh-Hua Wu, Ping-Yang Chen, and Jun-Wei Hsieh, “CSPNet: A New Backbone that can Enhance Learning Capability of CNN,” Nov. 2019.
- [12] Alexey Bochkovskiy, Chien-Yao Wang, and H. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” ArXiv, Apr. 2020.
- [13] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie, “Feature pyramid networks for object detection,” 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 936–944, 2016.
- [14] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia, “Path aggregation network for instance segmentation,” 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8759–8768, 2018.
- [15] Seyed Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian D. Reid, and Silvio Savarese, “Generalized intersection over union: A metric and a loss for bounding box regression,” 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 658–666, 2019.
- [16] Ali Hatamizadeh, V. Nath, Yucheng Tang, Dong Yang, Holger R. Roth, and Daguang Xu, “Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images,” ArXiv, vol. abs/2201.01266, 2022.
- [17] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin, “Attention is all you need,” in *Neural Information Processing Systems*, 2017.
- [18] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky, “Instance normalization: The missing ingredient for fast stylization,” ArXiv, vol. abs/1607.08022, 2016.