UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Yaroslav Prytula

IAUNet: Instance-Aware U-Net

Master's Thesis (30 ECTS)

Supervisor:
Dmytro Fishman, PhD

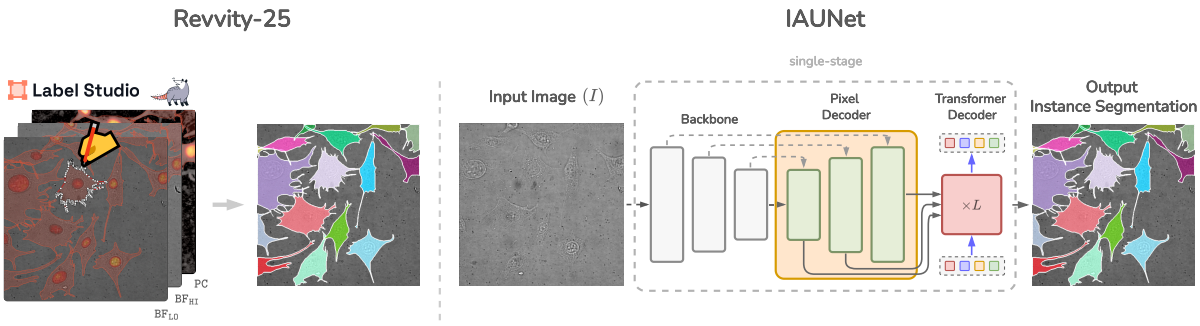
Tartu 2025

IAUNet: Instance-Aware U-Net

Abstract:

Instance segmentation is critical in biomedical imaging to accurately distinguish individual objects like cells, which often overlap and vary in size. Recent query-based methods, where object queries guide segmentation, have shown strong performance. While U-Net has been a go-to architecture for medical image segmentation, its potential in query-based approaches remains largely unexplored. In this work, we present IAUNet, a novel query-based U-Net architecture. The core design keeps the full U-Net structure and introduces a lightweight convolutional Pixel decoder that efficiently aggregates multi-scale features with minimal computational cost, making the model more efficient and reducing the number of parameters unlike its Transformer counterparts. On top of that, we propose a Transformer decoder with deep supervision that refines object-specific queries across multiple layers and resolutions, enabling precise instance-level segmentation. Finally, we introduce the Revvity-25 Full Cell Segmentation Dataset, a new 2025 benchmark featuring detailed annotations of overlapping cell cytoplasm in brightfield images. This dataset provides high-resolution labels with accurate instance boundaries and supports evaluation under both modal and amodal segmentation settings. Extensive experiments on multiple public datasets and Revvity-25, we show that IAUNet outperforms most state-of-the-art fully convolutional, transformer-based, and query-based models and cell segmentation-specific models, establishing a strong baseline for cell instance segmentation tasks.
Code is available at: https://github.com/SlavkoPrytula/IAUNet

Visual Abstract:



Keywords: Medical and Biological Vision, Cell Microscopy, Instance Segmentation, Deep Learning, Computer Vision

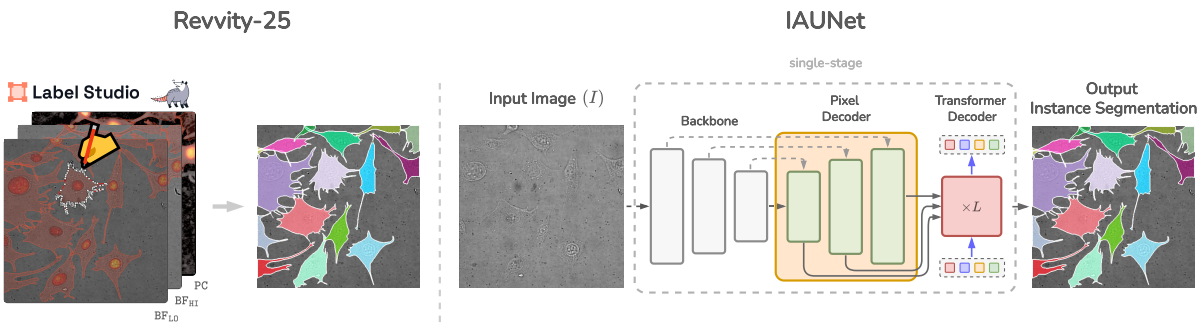
CERCS: T111 - Imaging, image processing; P176 - Artificial intelligence; B110 - Bioinformatics, medical informatics, biomathematics biometrics, P170 Computer science, numerical analysis, systems, control

IAUNet: Instance-Teadlik U-Net

Lühikokkuvõte:

Instance-segmentimine on biomeditsiinilises kujutises ülioluline, et täpselt eristada üksikuid objekte, näiteks rakke, mis sageli kattuvad ja varieeruvad suuruse poolest. Viimased päringupõhised meetodid, kus segmentimist juhivad objektipäringud, on näidanud tugevat jõudlust. Kuigi U-Net on olnud populaarne arhitektuur meditsiinilise kujutise segmenteerimiseks, on selle potentsiaal päringupõhistes lähenemistes jäänud suures osas uurimata. Käesolevas töös esitleme IAUNet-i – uut päringupõhist U-Net arhitektuuri. Selle põhistruktuur säilitab täielikult U-Neti ülesehituse ning lisab kerge konvolutsioonilise pikslidekodeerija, mis koondab mitmeskaalalisi tunnuseid efektiivselt ja väikese arvutusliku kuluga, muutes mudeli tõhusamaks ja väiksema mahtuvusega kui tüüpilised Transformer-põhised lähenemised. Lisaks esitame Transformer-põhise dekoodri koos sügava järelevalvemehhanismiga, mis täpsustab objektipõhiseid päringuid mitmes kihis ja eraldusvõimes, võimaldades täpset instance-segmentimist. Lõpuks toome välja uue 2025. aasta referentsandmestiku Revvity-25 Full Cell Segmentation Dataset, mis sisaldab detailseid märgendusi kattuvate rakkude tsütoplasmade kohta heleväljakujutistes. See andmestik pakub kõrgeresolutsioonilisi märgendusi täpsete piiridega ning võimaldab hindamist nii modaalses kui amodaalses segmentatsioonis. Ulatuslike katsetega mitmetel avalikel andmestikel ja Revvity-25 andmestikul näitame, et IAUNet ületab enamuse kaasaegsetest täiskonvolutsioonilistest, Transformer-põhistest ja päringupõhistest mudelitest ning spetsiaalselt rakusegmentatsiooniks loodud lähenemistest, seades tugeva aluse edasiseks arenguks rakutasemel instance-segmentimisel.
Lähtekood on kättesaadav aadressil: https://github.com/SlavkoPrytula/IAUNet

Visuaalne kokkuvõte:



Võtmesõnad: Meditsiiniline ja bioloogiline nägemine, Rakumikroskoopia, Isendite segmenteerimine, Süvaõpe, Arvutinägemine

CERCS: T111 - Pilditehnika; P176 - Tehisintellekt; B110 - Bioinformaatika, meditsiiniinformaatika, biomatemaatika, biomeetrika; P170 - Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Contents

1. Introduction

Accurate cell instance segmentation is crucial in biomedical imaging [1], as it enables the precise identification and analysis of individual cells. This capability is essential for understanding cellular behavior and disease mechanisms [2]. However, the diverse and irregular shapes of cells present significant challenges for segmentation algorithms [3, 4]. Variations in morphology, overlapping structures, and inconsistent imaging conditions often lead to segmentation errors [1]. Addressing these challenges requires advanced models that can effectively handle complex shapes and dense regions. Deep learning has driven substantial progress in cell segmentation, often outperforming traditional image processing techniques [5–7]. Still, segmentation remains a difficult task due to heterogeneous cell appearances, frequent overlaps, and highly varied object densities across different microscopy modalities. This makes it critical to design models that generalize well across conditions and are robust to such variability.

Brightfield microscopy, valued for its simplicity and affordability, presents unique challenges for segmentation [1]. Brightfield captures images by transmitting standard white light through the sample, and contrast is generated based on the natural absorption and scattering properties of cellular structures [8]. The resulting image is a 2D projection of light intensities, where denser regions appear darker. It does not require optical phase manipulation or fluorescent labeling, making it widely accessible with standard laboratory microscopes. In contrast, phase-contrast microscopy uses phase plates and annular diaphragms to shift the phase of light waves that pass through transparent specimens [9]. These phase shifts are then converted into intensity differences through optical interference, which enhances the visibility of internal cell structures such as membranes and organelles. Fluorescence microscopy captures images by exciting fluorescent molecules within the sample using specific wavelengths of light [8, 10]. These molecules emit light at longer wavelengths, which is collected through emission filters to isolate specific cellular components with high specificity and signal-to-noise ratio. The technique typically requires chemical or genetic labeling of target structures, along with specialized light sources and detectors. While fluorescence and phase-contrast modalities often require specialized equipment and sample preparation, brightfield imaging can be performed using standard light microscopes and unstained live cells. Its ease of acquisition, low cost, and compatibility with real-time observation make it a popular choice in both research and clinical settings [1, 11, 12]. However, brightfield images are inherently low-contrast, noisy, and visually variable [13], which makes precise cell segmentation particularly difficult and highlights the need for segmentation models tailored to this modality.

Many previous works have adapted instance segmentation models originally developed for natural images to the medical imaging domain without introducing model-specific modifications [14–16]. In contrast, U-Net [6] has long been the architecture of choice for semantic segmentation in biomedical applications. Its encoder-decoder structure with skip connections enables precise localization and detailed boundary reconstruction, making it especially effective for medical tasks where fine structures and object boundaries are critical. U-Net's lightweight design also allows for efficient training on smaller microscopy datasets, which are common in biomedical research. This is why we chose to focus on U-Net in our work, leveraging its proven success and compatibility with microscopy data.

More recently, the success of DETR [17] in object detection has led to a surge in query-based single-stage instance segmentation methods [18–23]. These models depart from traditional convolutional paradigms by employing attention mechanisms [24] and a fixed set of learnable object queries to predict instance masks and class labels directly in an end-to-end fashion. Despite their strong performance, these Transformer-based architectures typically rely on single-level or shallow features to initialize and refine queries. They often overlook the hierarchical and spatially rich representations provided by skip connections and decoder feature maps. This limits their ability to integrate multi-scale contextual information, which is crucial for accurately segmenting complex or overlapping structures in medical images.

To address these limitations, we bridge the gap between the widely adopted U-Net architecture in biomedical imaging and the task of instance segmentation. We present IAUNet, a novel instance-aware U-Net architecture that integrates query-based mechanisms for accurate instance-level predictions. IAUNet enhances the classical U-Net design with two key components: a lightweight convolutional Pixel decoder that efficiently aggregates multi-scale features, and a Transformer decoder that refines object-specific queries across multiple layers. This architecture enables the model to scale well with larger backbones and maintain strong performance on both small and large datasets, while remaining computationally efficient.

As part of our contributions, we also introduce the 2025 Revvity Full Cell Segmentation Dataset, specifically curated for benchmarking instance segmentation models in brightfield microscopy. It consists of high-resolution brightfield images with hundreds of hand-labeled cell instances, including challenging cases of overlapping cells. Each annotation has been carefully validated, making the dataset well-suited for evaluating the ability of models to capture fine boundaries and complex morphologies. It establishes a new benchmark for evaluating both modal and amodal segmentation performance.

Our main contributions are as follows:

- We introduce IAUNet, a novel model for cell instance segmentation that integrates a lightweight convolutional Pixel decoder and a Transformer decoder for efficient multi-scale object query refinement.
- We design a Transformer decoder that performs multi-layer query refinement and enables instance-aware segmentation with deep supervision.
- We present the 2025 Revvity Full Cell Segmentation Dataset, featuring detailed and validated annotations for evaluating segmentation models on brightfield images.
- We conduct an extensive evaluation of IAUNet across multiple public microscopy datasets and our own, demonstrating consistent state-of-the-art performance compared to fully convolutional, transformer-based, and specialized cell segmentation models.

2. Related Work

Early approaches to image segmentation in biomedical contexts were primarily based on fully convolutional networks developed for semantic segmentation. Among these, U-Net [**unet**] became the most widely used model due to its effectiveness in capturing fine structures with limited training data. However, unlike semantic segmentation, the task of instance segmentation requires distinguishing between individual object instances in addition to assigning pixel-level class labels. To address this, a range of instance segmentation methods have been developed and are generally categorized into region-based, query-based, and specialized approaches that often involve preprocessing steps.

2.1 U-Net

Early work in semantic segmentation adopted the Fully Convolutional Network (FCN) architecture [25], which replaced fully connected layers with convolutional layers to enable dense pixel-wise predictions. U-Net [6] builds on this concept and introduces a more elegant encoder-decoder structure designed specifically for biomedical image segmentation. The model adopts a symmetric encoder-decoder design with skip connections. The encoder consists of multiple convolutional blocks, each followed by a downsampling operation, progressively reducing the spatial resolution while increasing the number of feature channels. This compresses the input into a latent representation that captures contextual and semantic information. The decoder mirrors the encoder by applying upsampling and convolution operations to recover spatial resolution. Skip connections link each encoder stage to its corresponding decoder stage, allowing the model to reuse fine-grained spatial features that would otherwise be lost through downsampling. These connections concatenate high-resolution features from the encoder with the upsampled features in the decoder, allowing precise localization by preserving spatial detail that would otherwise be lost. This architecture allows U-Net to localize object boundaries with high spatial precision, which is particularly important in biomedical tasks where targets often have irregular shapes or closely packed structures [6, 7, 26, 27]. Despite its success in semantic segmentation, U-Net has not been widely adapted for instance segmentation tasks, where the goal is to distinguish and label individual object instances rather than predict a class label for each pixel.

2.2 Region-based Methods

A widely adopted solution to this challenge is the use of region-based methods [28]. Exemplified by Mask R-CNN [14, 29, 30], these models have set a standard in natural image segmentation with their proposal-based structure.

Mask R-CNN. Building on Faster R-CNN [29], Mask R-CNN adds a mask prediction branch for end-to-end instance segmentation. The model, by design, follows a two-stage architecture. First, given an input image, a convolutional backbone such as ResNet extracts intermediate feature maps at multiple stages, denoted as C_2 through C_5. These features are fed to a Feature Pyramid Network (FPN) [31], which constructs a top-down pathway with lateral connections, producing a pyramid of semantically rich feature maps P_2 through P_5, each capturing information at a different scale. Next, the Region Proposal Network (RPN) [29] uses the FPN outputs to generate a large set of candidate object regions (RoI). Since RPN often produces overlapping predictions, non-maximum suppression (NMS) is applied (Figure 1, RPN + NMS) to remove redundant

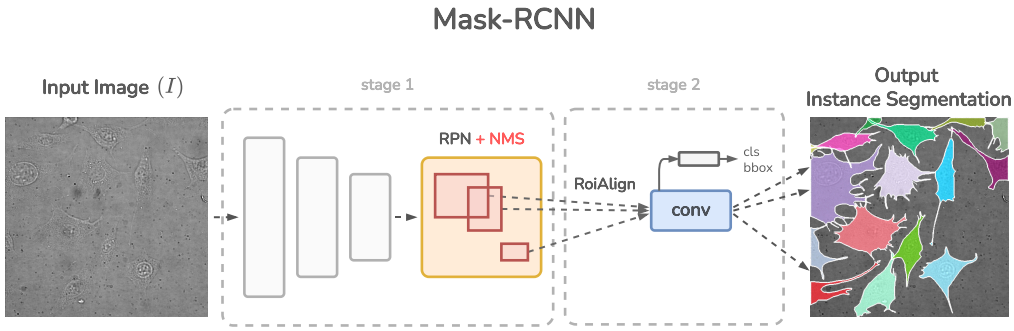


Figure 1. **Mask R-CNN architecture.** The model uses a two-stage design. In stage 1, a backbone and Feature Pyramid Network (FPN) extract multi-scale features from the input image. The Region Proposal Network (RPN) generates candidate regions, which are filtered using non-maximum suppression (NMS). In stage 2, Region of Interest (RoI) features are extracted using RoIAlign and passed to classification, bounding box regression, and mask prediction heads to produce final instance-level segmentation.

proposals with minimal overlap. Each RoI is then assigned to a feature pyramid level based on its scale. Then operations like RoI-Pooling [30] or RoI-Align [14] are used to extract a fixed-size 28×28 feature map from the corresponding level. Lastly, the extracted features are passed through three parallel heads: one for object classification, one for bounding box regression, and one for binary mask prediction. The mask head produces a pixel-level segmentation mask independently for each object class.

However, these two-stage methods often generate numerous redundant region proposals, reducing efficiency [22, 32]. Although they perform well on many benchmarks, their reliance on small RoI regions frequently leads to coarse mask predictions. Some methods focus on enhancing the precision of detected bounding boxes [33], while others, like PointRend [34], specifically address low-quality segmentation masks by refining boundaries at uncertain points to improve segmentation quality. PointRend refines coarse mask predictions by iteratively sampling uncertainty points and predicting each point independently using a small multilayer perceptron (MLP).

YOLOv8. While these methods improves mask quality through iterative refinement, they still rely on a two-stage pipeline with RoI-based feature extraction. To overcome such limitations and simplify the segmentation process, recent approaches have shifted toward fully convolutional, single-stage designs. Among them, YOLOv8 [15] represents a prominent example that unifies object detection and instance segmentation within a single-stage framework.

To simplify the pipeline and accelerate inference, YOLOv8 introduces a single-stage architecture that performs object detection and instance segmentation jointly, without region proposals or RoI operations.

Given an input image I, YOLOv8 extracts multi-scale feature maps using a convolutional backbone constructed primarily from C2f (Cross-Stage Partial Fusion) modules [15], which are a streamlined variant of the Cross-Stage Partial (CSP) [35] design used in earlier YOLO architectures. These feature maps, denoted as P_1 through P_5, capture both fine spatial details and

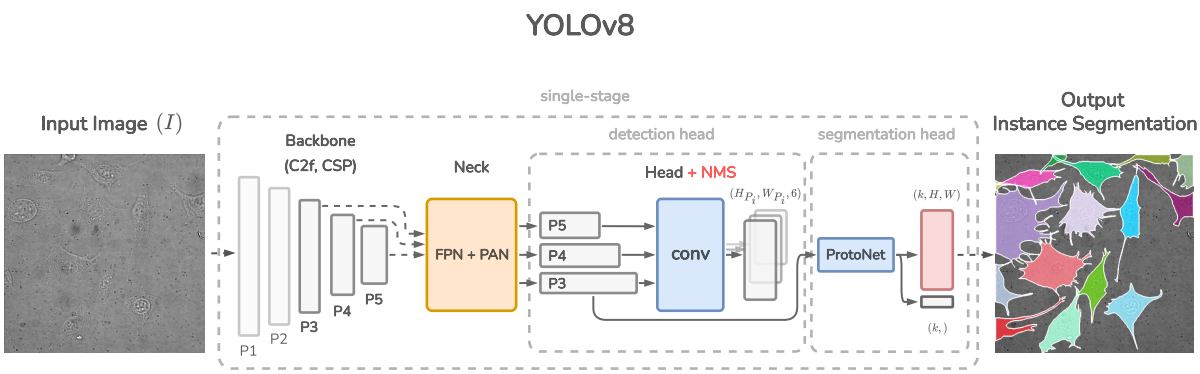


Figure 2. **YOLOv8 architecture.** A single-stage model for object detection and instance segmentation. The input image is processed by a C2f/CSP-based backbone to extract multi-scale features, which are refined by a Feature Pyramid Network (FPN) and Path Aggregation Network (PAN). The detection head predicts bounding boxes, class scores, and mask coefficients. The segmentation head (ProtoNet) generates global prototype masks, which are linearly combined using predicted coefficients to form instance masks. Final masks are cropped and thresholded.

high-level semantics across scales. The extracted features are passed through the neck module, which consists of a combination of Feature Pyramid Network (FPN) [31] and Path Aggregation Network (PAN) [36] structures. FPN propagates semantic information from deeper layers to shallower ones via top-down connections, while PAN enhances localization by strengthening bottom-up pathways. This produces refined multi-scale feature maps $\{P_3, P_4, P_5\}$ used by the prediction heads. The detection head applies convolutional layers to predict bounding boxes, class probabilities, and mask coefficients directly from the feature maps. Similar to Mask R-CNN, post-processing with non-maximum suppression (NMS) is applied to suppress redundant detections (Figure 2, detection head) and retain the most confident ones.

YOLOv8 shares ideas with methods like YOLACT [37], using a prototype-based strategy for segmentation (Figure 2, segmentation head). For this, the model includes a parallel segmentation head that uses a lightweight ProtoNet module. ProtoNet generates a set of k global prototype masks $P \in \mathbb{R}^{k \times H \times W}$, independent of specific object instances. For each detected object, the model predicts a set of coefficients $c \in \mathbb{R}^k$. The final instance mask $M \in \mathbb{R}^{H \times W}$ is obtained by a linear combination:

$$M = \sum_{i=1}^{k} c_i \cdot P_i \tag{1}$$

The resulting mask is then cropped to the predicted bounding box and thresholded to produce the final binary segmentation.

However, even with these advancements, traditional region-based methods face limitations in biomedical image segmentation [23], where objects have complex shapes, orientations, and sizes. In these settings, traditional axis-aligned bounding boxes struggle to capture detailed contours, particularly for irregular and overlapping cellular structures [38, 39]. Additionally, while NMS

helps reduce redundant predictions, it can remove valid overlapping instances, further limiting segmentation accuracy in densely packed or occluded regions.

2.3 Specialized Cell Instance Segmentation Methods

Specialized methods such as StarDist [40] segment biomedical images by representing objects as star-convex polygons, predicting distances from a central point to boundaries in multiple directions. This method, along with other similar approaches like DeepWatershed [41] and Micro-Net [42], works well for star-shaped or rounded cells but struggles with irregular, elongated shapes and overlapping cells.

CellPose [43], by contrast, similar to Hover-Net [44], uses a U-Net to predict horizontal and vertical gradients alongside a binary cell map, creating a vector field that directs pixels toward the cell center. In addition, CellPose extracts style vectors (Figure 3, style vector) from intermediate U-Net activations, specifically from the global average pooled output of a mid-level feature map. These style vectors are 256-dimensional and summarize the overall appearance and texture of the input image. During inference, they are used to assign images to style-based clusters, allowing the system to automatically select from a pool of specialized models trained for different visual domains.

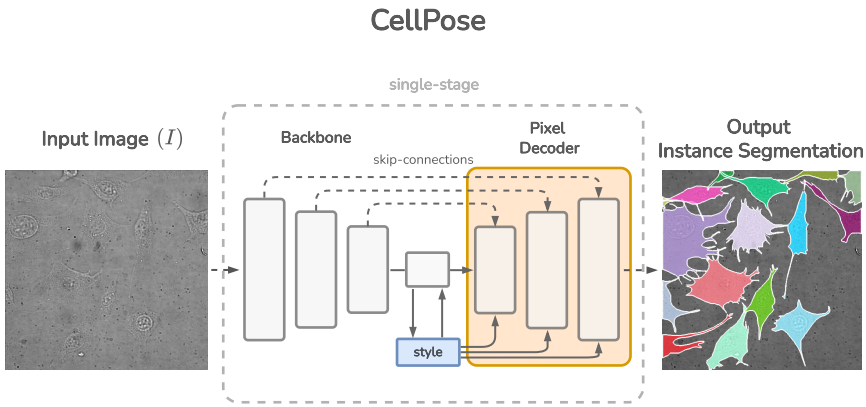


Figure 3. **CellPose architecture.** A single-stage segmentation model based on a U-Net backbone with skip connections. The network predicts per-pixel vector fields that direct pixels toward the centers of individual cells. Additionally, a 256-dimensional style vector is extracted from mid-level feature maps to capture global image characteristics. This style embedding is used during inference by the size model (SM) to predict an average object diameter for an image and resize it accordingly.

While this method effectively separates individual cells, it often relies on an additional size model [45] to estimate object diameters before inference. This scalar diameter prediction is used to rescale input images such that cells appear to be approximately the same size as those seen during training. Although this rescaling step improves segmentation consistency, it assumes that all objects in an image are of roughly uniform size. In cases where cells exhibit high size variability within the same field of view, the single diameter estimate can lead to under-segmentation of large cells or over-segmentation of small ones. Although these methods offer advancements

over traditional techniques, they remain limited in accurately segmenting overlapping cells and handling complex cellular morphologies.

2.4 Query-based Methods

Query-based Methods have gained popularity since the introduction of DETR [17], which demonstrated the potential of Transformer-based architectures for instance segmentation. Unlike traditional region-based models, query-based methods use object queries to directly predict object instances, removing the need for predefined bounding boxes.

DETR formulates object detection as a set prediction task using a single-stage Transformer-based encoder-decoder architecture. The model takes an input image $I \in \mathbb{R}^{H_0 \times W_0 \times 3}$, which is processed by a convolutional backbone, typically ResNet [46], to produce a feature map $f \in \mathbb{R}^{H \times W \times C}$, where $C = 2048$, $H = H_0/32$, and $W = W_0/32$. A 1×1 convolution reduces the channel dimension from C to a smaller value D, resulting in a feature map $x \in \mathbb{R}^{H \times W \times D}$. The feature map x is then flattened across spatial dimensions into a sequence of length HW, with each element of dimension D.

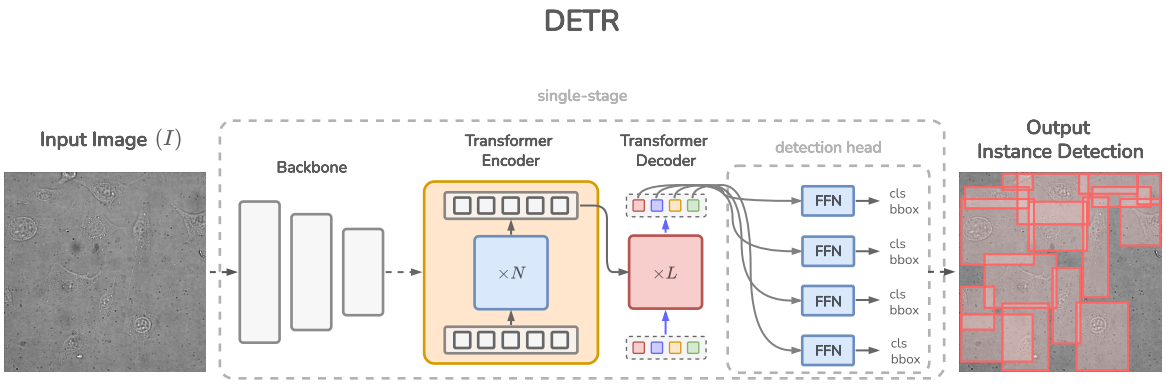


Figure 4. **DETR architecture.** DETR casts object detection as a set prediction task using a Transformer-based encoder-decoder. A CNN backbone extracts features, which are flattened and passed to the Transformer encoder. A fixed set of learned object queries interact with encoded features in the Transformer decoder to predict object instances. Each query output is fed to a prediction head for class and bounding box regression. Duplicate predictions are avoided through bipartite matching with the Hungarian algorithm during training.

The image features are passed to the Transformer encoder, which consists of multiple layers with multi-head self-attention and feedforward networks (FFN) (Figure 4). Lastly, fixed positional encodings are added at each layer to preserve spatial structure. The decoder uses a fixed set of learned object queries $q \in \mathbb{R}^{N \times D}$ together with the encoded image features x. These queries interact with the encoder output via cross-attention and produce N output embeddings, each representing a potential object instance. Each embedding is passed through a prediction head that outputs a class label and bounding box. DETR uses bipartite matching with the Hungarian algorithm [47] to assign predictions to ground truth targets in a one-to-one manner, avoiding duplicate detections. This eliminates the need for hand-crafted components such as anchor boxes, non-maximum suppression, or region proposals.

Mask2Former. Building on DETR, models like Mask2Former [19] and FastInst [22] introduced masked attention to improve convergence and segmentation precision. These models heavily rely on producing fine features using multi-scale deformable attention (MSDeformAttn) [48] Pixel decoder. Specifically, in Mask2Former (Figure 5), a CNN backbone extracts multi-scale feature maps, which are progressively refined by the Pixel decoder using MSDeformAttn to enhance spatial detail.

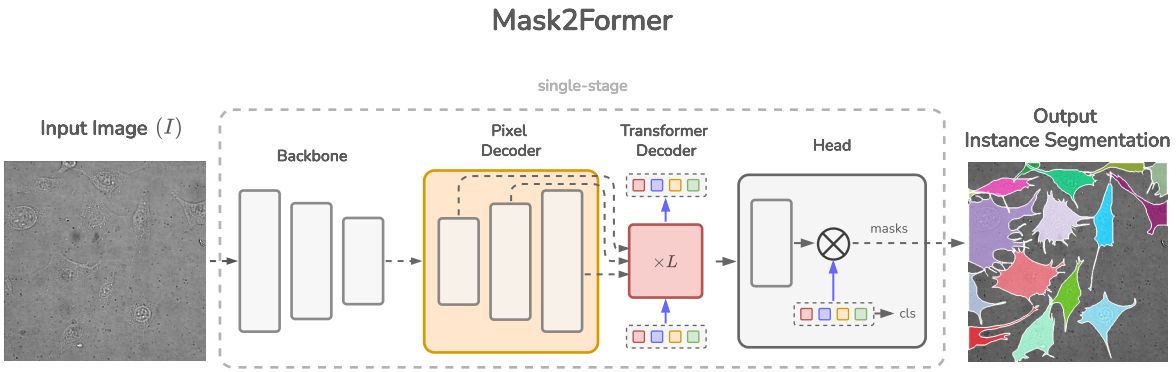


Figure 5. **Mask2Former architecture.** The model combines a CNN backbone, a Pixel decoder with multi-scale deformable attention (MSDeformAttn), and a Transformer decoder for instance segmentation. A fixed set of learnable object queries is refined through masked cross-attention across decoder layers. Queries attend to features from the Pixel decoder layers, and the output mask embeddings are projected onto high-resolution features to generate final instance masks.

As in DETR, a fixed set of $N = 100$ randomly initialized learnable object queries $q \in \mathbb{R}^{N \times D}$ is used to decode instance-specific representations. These queries are processed by a Transformer decoder composed of $L = 3$ repeated blocks, each containing three layers, for a total of nine layers. In each layer, the queries perform masked cross-attention with the Pixel decoder features, attending only to the spatial regions indicated by their predicted masks from the previous stage. The attended features are fused with the query embeddings and passed through a feedforward network (FFN). The model uses a Round-Robin scheme, where queries are passed sequentially through the multi-scale feature maps from the Pixel decoder in each of the L decoder blocks. Final masks are obtained by projecting the refined mask embeddings onto the high-resolution output of the Pixel decoder.

MaskDINO [20] further advances instance segmentation by adding a segmentation branch that generates high-resolution binary masks from object queries. The model builds on previous works such as DINO [49], Conditional DETR [50], and Deformable DETR [51] by combining unified query selection, joint denoising training, and improved positional priors for segmentation. The architecture consists of a single-stage Transformer encoder and decoder, and applies three prediction heads for classification, bounding box regression, and segmentation to both encoder and decoder outputs.

First, from the encoder, the model obtains dense features $x \in \mathbb{R}^{HW \times D}$. Each token is scored using the classification head by computing its probability of corresponding to an object class. The top-k scoring tokens are selected as content queries $q \in \mathbb{R}^{N \times D}$, where each query is represented as a tuple of a content vector and a positional vector (Figure 6, arrows from Transformer Encoder

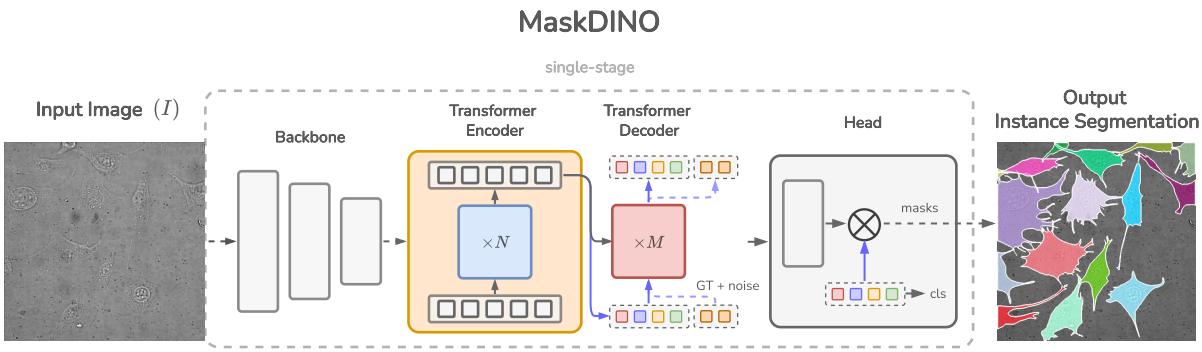


Figure 6. **MaskDINO architecture.** The model combines a Transformer encoder-decoder architecture with unified query selection and joint denoising training. Queries are initialized from top-scoring encoder tokens and perturbed ground-truth (GT) labels with added noise. The Transformer decoder refines these queries over multiple layers using masked cross-attention with multi-scale image features. Final masks are obtained via a point-wise product between queries and pixel features. MaskDINO uses a hybrid matching strategy for supervision across classification, box, and mask outputs.

to Transformer Decoder). Following Conditional DETR, the content vector captures semantic intent, while the positional vector $b \in \mathbb{R}^{N\times 4}$ is a learnable box coordinate used as an explicit spatial prior. These positional components are used to initialize anchor boxes and are refined during decoding. The selected encoder features are also passed through decoder heads to produce initial bounding boxes and mask embeddings, which are supervised and serve as priors for the decoder.

MaskDINO extends the query denoising mechanism introduced in DINO by applying it to both detection and segmentation. Noised versions of ground-truth boxes and labels (Figure 6, GT + noise) are encoded as positional and content queries and passed alongside the selected ones. For segmentation, the model treats the perturbed boxes as coarse spatial inputs and learns to reconstruct detailed instance masks. This allows the model to learn to localize fine-grained masks from imprecise spatial signals. The Transformer decoder refines both selected and denoising queries over multiple layers. At each layer, queries attend to multi-scale pixel features via masked cross-attention. These pixel features are extracted from the pixel decoder and concatenated across scales. The decoder head at each layer predicts a refinement $\Delta b_i^{(t)}$, and the updated box is computed as $b_i^{(t+1)} = b_i^{(t)} + \Delta b_i^{(t)}$. This approach incorporates explicit spatial priors into query refinement. The decoder outputs include updated class logits, bounding boxes, and mask embeddings.

Lastly, MaskDINO uses a hybrid matching strategy during training. Predictions are matched to ground-truth objects using a cost that combines classification, bounding box, and mask losses. This enforces alignment between outputs and ensures that each object is assigned to a unique query.

Cell-DETR. Recently, adaptations of query-based models have also emerged in the biomedical domain. For example, Cell-DETR [52] adapts DETR specifically for cell segmentation by leveraging queries to detect individual instances. The model uses the final feature map of the

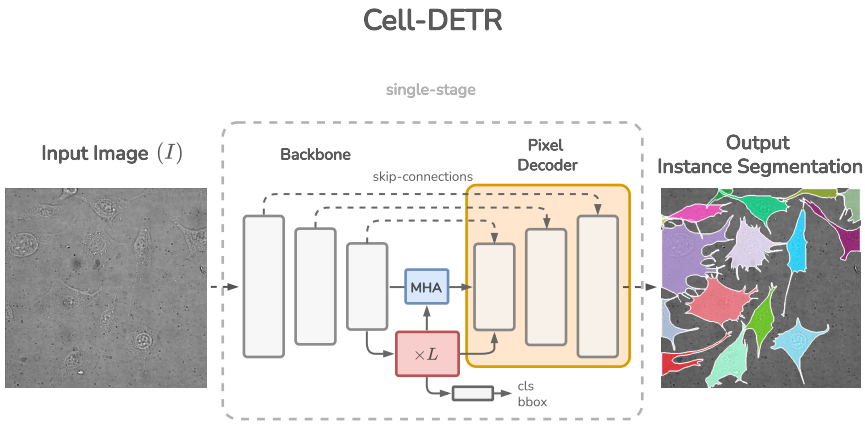


Figure 7. **Cell-DETR architecture.** Cell-DETR adapts the DETR framework for cell segmentation by initializing queries from the encoder's final feature map and applying multi-head attention (MHA) between the encoder and the Transformer decoder features. The model fuses queries with pixel features only at the lowest decoder layer. The CNN-based Pixel Decoder is responsible for decoding instances as separate channels.

encoder for query initialization, limiting multi-scale query refinement across decoder features. Its segmentation head applies multi-head attention between encoder and decoder features, followed by a CNN decoder. However, it merges queries with decoder features only at the lowest layer, forcing the CNN decoder (Figure 7, Pixel Decoder) to handle most of the instance separation. This makes the model inefficient for high-resolution inputs with many queries. Additionally, Cell-DETR applies softmax to suppress overlapping predictions, reducing its ability to segment occluding cells effectively. Recent work, such as PCTrans [23], built on Mask2Former, introduces a position-guided transformer with a query contrastive loss. Similar to DETR, position guidance is done by predicting the normalized center coordinates of each object. While natural objects are often convex, cells present more complex shapes, with centers that often fall outside boundaries, particularly in elongated structures [53], making mask representation less effective.

Segment Anything Model (SAM) [54] introduces a prompt-driven segmentation framework. Unlike DETR-based models [17, 19, 20, 22, 49] that use object queries to discover objects, SAM segments regions based on external prompts, which act similar to object queries. The architecture consists of three components: a ViT-based image encoder [55], a prompt encoder, and a mask decoder. First, the large image encoder produces dense image embeddings (Figure 8). Then, each prompt is embedded into a 256-dimensional embedding using positional encodings [56] and learned type-specific vectors. Finally, the image and prompt embeddings are passed to a two-layer Transformer decoder. Each layer performs self-attention over the tokens, followed by cross-attention from tokens to image embeddings, then updates the tokens via an MLP, and finally applies cross-attention from image embeddings to tokens, injecting prompt information into the spatial features. The updated image embeddings are then upsampled by $4\times$ using two transposed convolutional layers. The final mask is predicted via a spatial point-wise product between the upsampled image embedding and the prompt embeddings.

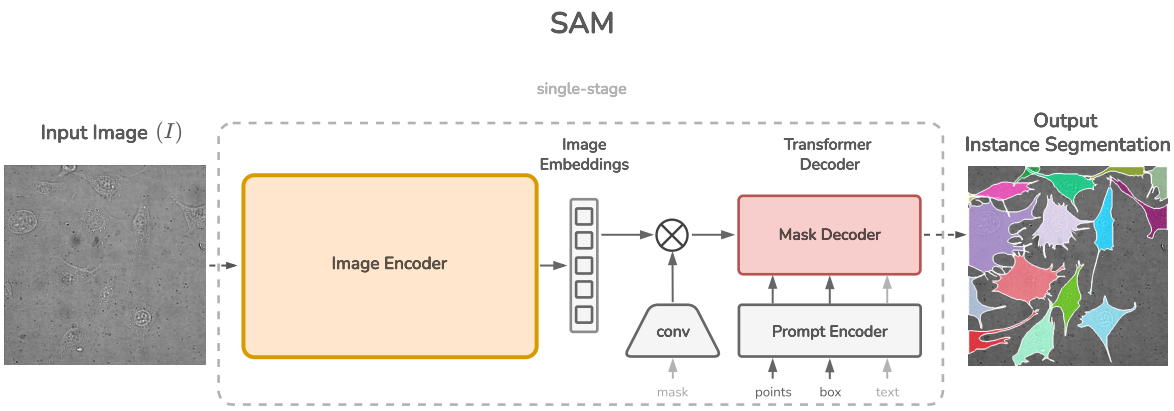


Figure 8. **Segment Anything Model (SAM) architecture.** SAM segments regions based on external prompts rather than learned object queries. The architecture includes a ViT-based image encoder, a prompt encoder for encoding points, boxes, or masks, and a lightweight Transformer-based mask decoder. The decoder injects prompt information into the image features, which are then upsampled and used to produce masks via a point-wise projection.

While SAM demonstrates strong zero-shot generalization due to pretraining on the 1 billion-image SA-1B dataset, it requires precise input prompts to segment objects and lacks native instance-aware modeling. This limits its performance in crowded or overlapping scenes.

All previous query-based models [17, 19, 20, 23] have been designed around the idea of a Transformer-based Pixel decoder, which raises concerns about scalability to smaller datasets. Unlike these models, we propose a lightweight Pixel decoder that improves performance on smaller datasets. In Table 2, we show that IAUNet consistently outperforms state-of-the-art models across different backbones while maintaining strong results on large-scale datasets (Table 1). Our experiments show that IAUNet outperforms most alternatives while using fewer parameters and achieving higher efficiency.

3. Data and Methods

Instance segmentation is essential for analyzing cell morphology in microscopy images. While datasets like LIVECell [57] and EVICAN [58] provide annotated cell images, they often lack the precision needed for supervised instance-level learning or the granularity required for evaluating detailed instance-level performance. For instance, the EVICAN dataset includes partially annotated training images and test images with coarse polygon annotations and only a few instances per sample. LIVECell, while extensive, also provides coarse annotations in the phase-contrast modality and lacks detailed instance-level masks. Most importantly, these datasets were not specifically designed to handle overlapping cell structures, thus limiting their use for that purpose. To our knowledge, there is currently no publicly available dataset of brightfield microscopy images of human cancer cells that includes precise instance-level annotations. As highlighted in [59, 60], while some efforts exist for yeast or other model organisms, accurate annotation of overlapping cells in brightfield images remains an unresolved challenge in the context of microscopy data.

Previous datasets serve as valuable benchmarks for cell instance segmentation, but they often fall short when it comes to annotation precision, full per-cell labeling, or the accurate representation of overlapping structures. In our work, we extend this landscape by introducing a novel dataset that combines fully annotated, high-resolution, and high-precision segmentation of complex and overlapping cell structures. One of our key contributions is a new cell instance segmentation dataset named **Revvity-25**.

On the other hand, to approach the difficulty of training and scaling instance segmentation models on datasets of various sizes, we propose a novel instance segmentation U-Net architecture. Our model reorients the traditional purpose of U-Net by integrating a Pixel-Transformer decoder, which includes a lightweight convolutional Pixel decoder and a multi-scale Transformer decoder. This design enables refinement of object-specific features across multiple scales and explicitly targets segmentation of overlapping instances. Later, we show that our method, while maintaining simplicity, outperforms most of the state-of-the-art models.

3.1 Data

We evaluate all the methods on several publicly available cell instance segmentation datasets, as well as our own. In what follows, we describe the datasets used in our experiments, including their characteristics, annotation protocols, and the preprocessing steps.

3.1.1 LIVECell

The LIVECell dataset [57] is one of the most extensive resources available for instance segmentation of cells. It contains 5,239 high-resolution phase-contrast microscopy images (520×704 pixels), with a total of over 1.6 million manually annotated cell instances. The annotations include tight per-cell instance masks verified by multiple experts, covering eight cancer cell lines: A172 (glioblastoma), BT474 (breast cancer), BV2 (microglia), Huh7 (hepatocellular carcinoma), MCF7 (breast cancer), SH-SY5Y (neuroblastoma), SkBr3 (breast cancer), and SKOV3 (ovarian cancer). These lines span a wide range of shapes, adhesion properties, and growth densities, providing a challenging testbed for generalization across cell morphology.

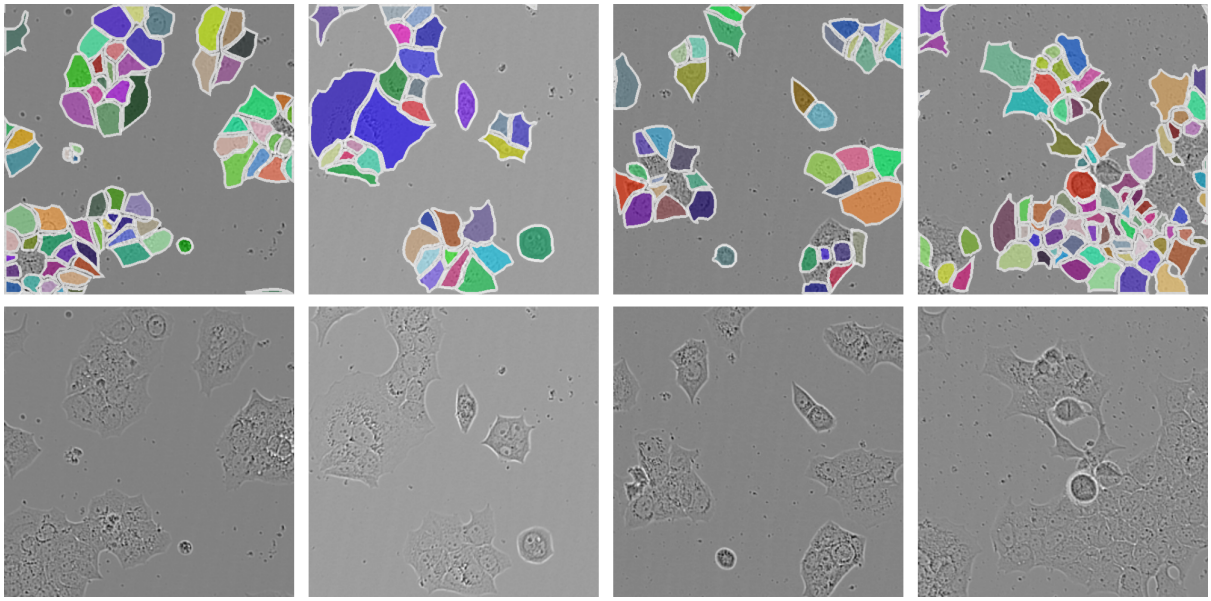


Figure 9. LIVECell Dataset.

We begin by processing the dataset by generating random 256×256 crops from each image and resizing them to 512×512 to maintain consistency in input resolution across all datasets. We empirically find that 256×256 crops are well-suited for extracting up to 100 cells per image in the LIVECell dataset. This approach ensures uniformity in prediction scale and prevents performance bias from overly dense regions.

3.1.2 EVICAN

The EVICAN2 dataset [58] provides a large-scale, heterogeneous benchmark for cell and nucleus segmentation across brightfield (BF) and phase-contrast (PhC) microscopy. Cells were cultured at 37°C and 90% humidity in media supplemented with Penicillin/Streptomycin, fetal calf serum (FCS), and nonessential amino acids (NEAA). Twenty-four hours prior to imaging, cells were seeded into 96-well plates at 30%, 50%, or 100% confluency, fixed with 4% paraformaldehyde, and stained using CellMask Orange for cell membranes and DAPI for nuclei. Imaging was performed using four different microscope platforms: Opera Phenix (Perkin Elmer), AF7000 (Leica), IX81 (Olympus), and Biorevo BZ-9000 (Keyence), with objective lenses ranging from 10× to 63×. Grayscale BF and PhC images were annotated using region-matched fluorescence overlays in FIJI, enabling accurate delineation of both cellular and nuclear structures.

The dataset consists of over 4,600 partially segmented images, collected using multiple imaging setups and annotated by seven trained personnel. Each image contains between 3 and 10 labeled objects, including both cell and nucleus instances, selected at random and verified by experts. EVICAN is provided in two labeling schemes: EVICAN60, where each structure and cell line is treated as a unique class (60 in total), and EVICAN2, which groups all structures into two general categories: cell and nucleus. We use EVICAN2 for all experiments in this work. The training set includes 3,714 partially annotated images with 42,317 instances, and the validation set includes 926 images with 10,642 instances, totaling 52,959 labeled objects. An additional 98 fully annotated images constitute the evaluation set used to test the performance of the models. We report metrics as the mean over both annotated classes: cell and nucleus.

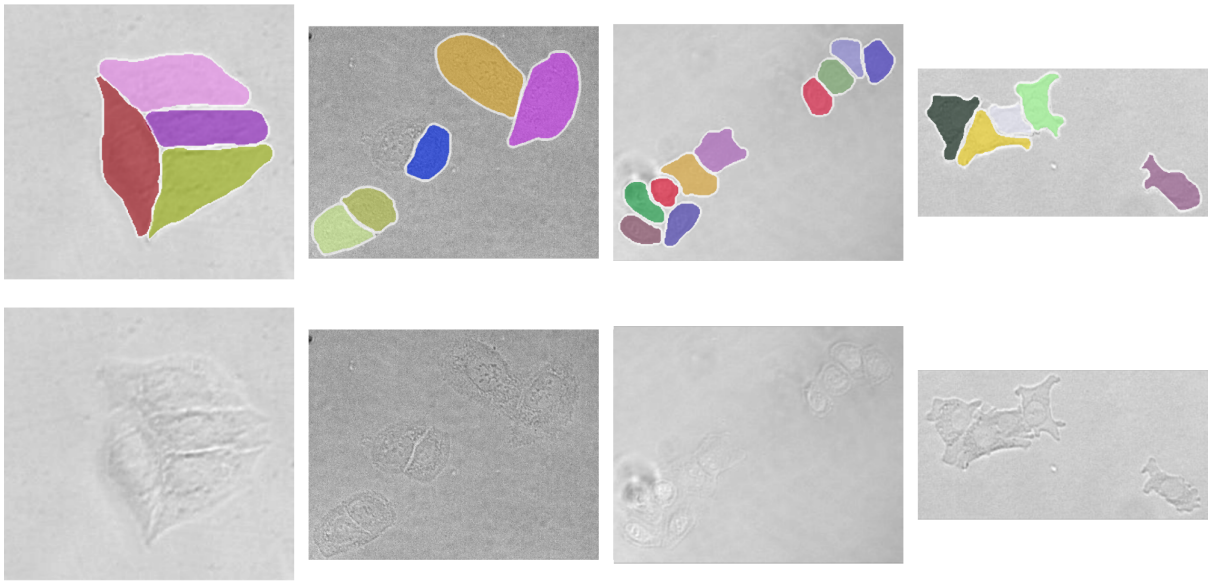


Figure 10. EVICAN Dataset.

For evaluation, we follow the same setup described in the original publication. Specifically, we use the three predefined subsets to assess the performance of the models under different levels of imaging complexity, contrast, and segmentation difficulty:

- **Easy**: 33 images, 1,084 instances. These images are well-focused, primarily in phase-contrast, and depict 2D monolayers with minimal cell-cell contact. Cell boundaries are clear and most nuclei are visible.
- **Medium**: 33 images, 1,036 instances. These images are slightly defocused and include a mix of BF and PhC modalities. Cells show more interaction, and some nuclei are no longer clearly visible.
- **Difficult**: 32 images, 1,102 instances. These images are primarily brightfield and often defocused, with frequent cell overlap, 3D-like colony formation, and limited visibility of nuclei due to poor contrast.

3.1.3 ISBI2014

The ISBI2014 dataset originates from the Overlapping Cervical Cytology Image Segmentation Challenge [61], which focuses on the segmentation of overlapping cervical cells in cytology specimens. The primary objective is to detect and segment individual nuclei and cytoplasm regions, even when multiple layers of cells are present in a single image. This is a common occurrence in real-world cervical smear analysis, where cells in the upper layers can partially occlude those beneath. To address this complexity, the dataset uses extended depth-of-field (EDF) imaging, which combines multiple focal planes into a single, in-focus image. This simplifies image acquisition while preserving the challenge of analyzing overlapping cellular structures.

The dataset consists of 945 synthetic grayscale images and 16 real EDF cervical cytology images. The synthetic images were designed to simulate realistic variations in cell shape, density, and overlap, and are accompanied by high-quality pixel-level annotations for both cytoplasm and

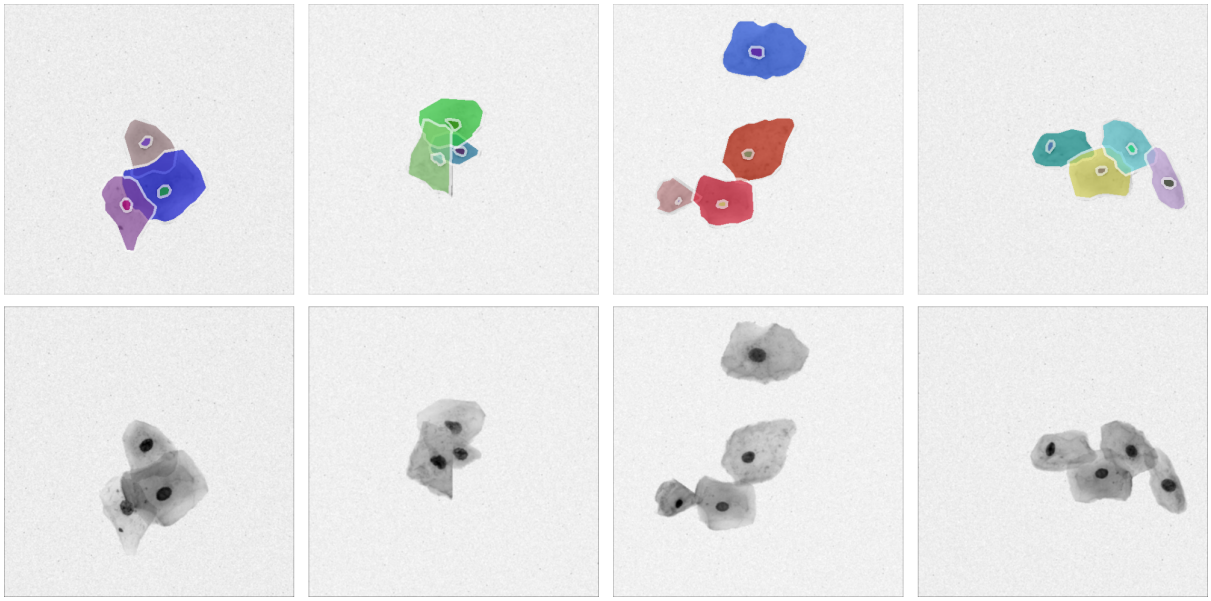


Figure 11. ISBI2014 Dataset.

nucleus. These annotations make the dataset particularly valuable for training and evaluating models in scenarios involving cell overlap. This allows to evaluate the model even further on the specific challenge of overlapping instance segmentation, which remains difficult in clinical settings. The real EDF images are intended for qualitative assessment, reflecting practical microscopy scenarios. Following the official challenge protocol, we use 45 synthetic images for training, 90 for validation, and 810 for testing. The real EDF images are not used for quantitative evaluation. For all experiments, we report results as the mean over both cytoplasm and nucleus classes.

3.1.4 Revvity-2025

Revvity-25 is a high-resolution, multi-cell line dataset collected to advance cell segmentation research in biomedical imaging. It includes seven distinct cell lines from human, dog, and mouse origins, capturing a broad range of cellular morphologies. The human cell lines include breast cancer (MCF7), fibrosarcoma (HT1080), cervical cancer (HeLa), hepatocellular carcinoma (HepG2), and alveolar basal epithelial cells (A549). Additionally, the dataset features dog kidney epithelial cells (MDCK) and mouse embryonic fibroblast cells (NIH3T3), which further enhance structural diversity. Cells were seeded into 384-well Collagen Type I-coated CellCarrier Ultra Microplates (Revvity, Waltham, MA; catalog no. 6057700). To improve image contrast, cells were stained with 10µg/ml Hoechst 33342 (Thermo Fisher, Waltham, MA; catalog no. H3570) and fixed using a 3.7% formaldehyde solution (Sigma Aldrich, St. Louis, MO; catalog no. 252549). All the images were acquired using the Opera Phenix™ high-content screening system in confocal mode with a 20× water immersion objective. Each image has a native resolution of 1080×1080 pixels and a physical pixel size of 0.59µm. All images are stored in 16-bit format to preserve maximum quality. For each cell line, 432 fields of view were captured, totaling 3024 microscopy images.

Revvity-25 comprises 110 brightfield images selected for annotation. Each image contains, on average, 27 manually labeled and expert-validated cells, resulting in a total of 2937 annotated

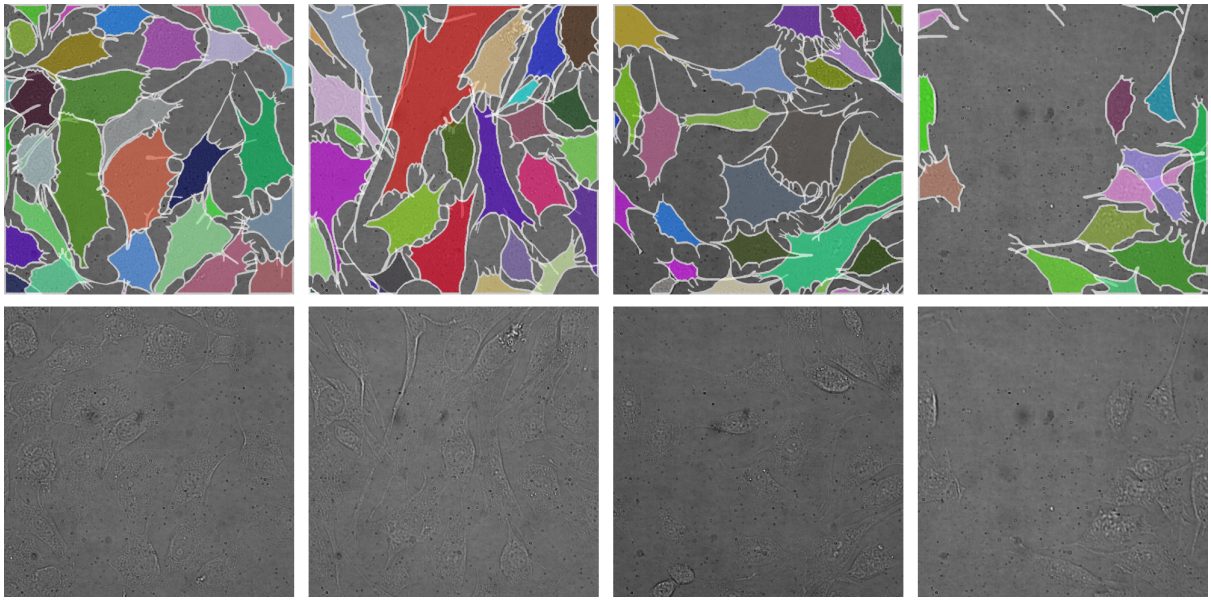


Figure 12. Revvity-25 Dataset.

cell instances. Each cell is represented using high-fidelity polygon masks, with an average of 60 points per cell and up to 400 points for cells with complex morphology. An example of annotated cell instances is shown in Figure 12. To our knowledge, Revvity-25 is the first publicly available dataset to combine high-resolution brightfield imaging with precise instance-level annotations that account for overlaps and complex boundaries. This dataset establishes a new benchmark for both modal and amodal instance segmentation in biomedical imaging. For benchmarking, we divide the Revvity-25 dataset equally into train and test sets, each containing 55 images.

Each image in Revvity-25 was acquired at 63× magnification and includes a pair of brightfield images taken at different focal planes (we refer to these as lower and higher planes). Within the original dataset provided by Revvity company, we had access to corresponding phase-contrast and fluorescence images for each brightfield image. These additional modalities were used solely to support the annotation process and are not included in the final dataset. The full unlabeled dataset comprises 11,808 images of six different cell lines in the brightfield modality. To obtain accurate instance-level annotations, we start by manually labeling a subset of the dataset, totaling 110 images, using the LabelStudio tool [62]. Each cell is labeled individually to capture precise boundaries, including complex morphologies and overlapping regions. We found the annotation process to take approximately 40–60 minutes per image.

To improve annotation visibility, we developed a preprocessing strategy that combines multiple imaging modalities. First, we discarded samples with dense cell clusters or excessive occlusion, as these tend to reduce annotation quality. For the remaining images, we superimpose each brightfield plane with its corresponding fluorescence channel and apply color enhancement to highlight the nuclei channel. This visualization provided clearer structural and visual cues for distinguishing separate instances and guiding boundary placement. We use phase-contrast images as an additional reference throughout the annotation process due to their superior signal-to-noise ratio and enhanced contrast.

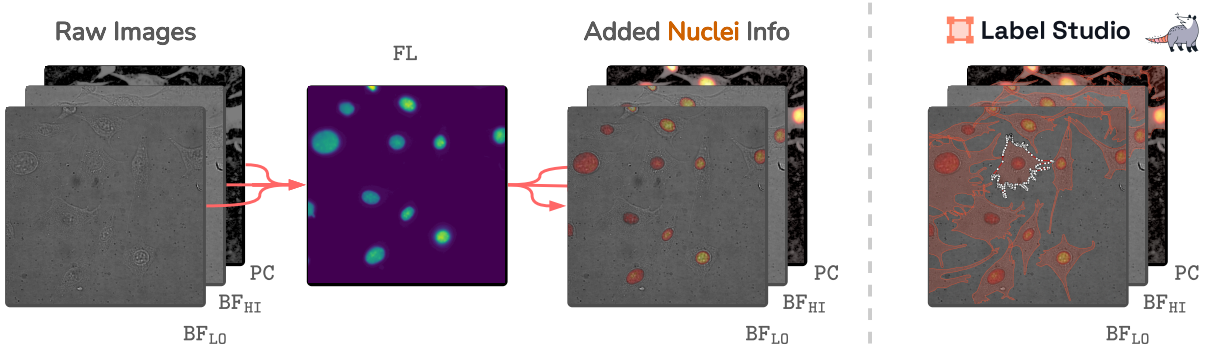


Figure 13. Multimodal annotation workflow for the Revvity-25 dataset. Each image includes a lower ($\texttt{BF}_{\texttt{LO}}$) and higher ($\texttt{BF}_{\texttt{HI}}$) focal plane brightfield image, along with a corresponding phase-contrast image (PC). These three channels are jointly used during annotation in Label Studio [62], with each instance validated across all modalities. The fluorescence image (FL), containing nuclear signal, is overlaid to enhance visibility and assist in resolving ambiguous boundaries and overlapping structures.

During the annotation process, we validate the accuracy of each labeled instance by copying and overlaying the segmentation mask across all available modalities. This step helps confirm boundaries in regions where one modality may be noisy or blurred and ensures consistency across visual contexts. By cross-referencing cell outlines in brightfield, phase-contrast, and fluorescence images, we maintain high annotation quality throughout the dataset.

3.2 U-Net

For biomedical datasets such as ours, models are typically expected to handle fine structural details, varying cell shapes, and dense spatial arrangements. A widely used architecture in this domain is U-Net [6], which has become a standard baseline for semantic segmentation tasks in biomedical imaging. The U-Net architecture is commonly implemented as a fully convolutional neural network designed for dense pixel-wise prediction tasks, particularly in biomedical image segmentation.

The architecture follows a symmetric encoder-decoder structure designed to learn both global context and fine spatial details. Given an input image $I \in \mathbb{R}^{H \times W \times 3}$, the model first processes it through an encoder path composed of a sequence of convolutional blocks. Each block consists of two 3×3 convolutions, each followed by a ReLU activation [63] and a batch normalization layer [64]. A 2×2 max pooling operation is applied after each block to reduce spatial resolution. As the image is progressively downsampled, the number of feature channels is typically doubled at each level, resulting in a coarse but semantically rich representation at the bottom of the network. This lowest-level feature map has the smallest spatial resolution but the largest number of channels, enabling the model to capture high-level contextual features.

The decoder path mirrors the encoder. Each stage begins with an upsampling operation, typically implemented using either bilinear interpolation or transposed convolution, to increase spatial resolution. This is followed by two 3×3 convolutions with ReLU activations and batch normalization. At each level, the upsampled feature map is concatenated channel-wise with

the corresponding feature map from the encoder via skip connections. This mechanism helps preserve fine spatial detail by reintroducing high-resolution features that were lost during the downsampling process. The final layer of U-Net is a 1×1 convolution that projects the decoder output to the desired number of classes per pixel. For binary segmentation tasks, the model learns to classify each pixel as foreground or background using a sigmoid activation and the binary cross-entropy loss [65]:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)\right], \tag{2}$$

where $y_i \in \{0, 1\}$ is the ground truth label, and $\hat{y}_i \in [0, 1]$ is the predicted probability for pixel i. For multi-class segmentation, the model applies a softmax activation and uses the categorical cross-entropy loss [65]:

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{i,k} \log \hat{y}_{i,k}, \tag{3}$$

where K is the number of classes, $y_{i,k} \in \{0, 1\}$ is the one-hot ground truth vector, and $\hat{y}_{i,k}$ is the predicted softmax probability for class k at pixel i.

Nevertheless, while U-Net is highly effective for semantic segmentation, the architecture was not specifically designed to handle instance segmentation, which limits its ability to separate individual object instances of the same class.

3.3 IAUNet

In this work, we present a novel instance segmentation model, IAUNet (Instance-Aware U-Net), which extends the U-Net architecture with a lightweight convolutional Pixel decoder and a Transformer decoder.

IAUNet follows the U-Net structure, as illustrated in Figure 14. Given an input image $I \in \mathbb{R}^{H\times W\times 3}$, the encoder extracts multi-scale feature maps at resolutions $1/4$, $1/8$, $1/16$, and $1/32$ of the input. These feature maps are utilized as skip connections in the decoder. The Pixel decoder starts from the lowest-resolution feature map and progressively upsamples it through a sequence of decoder stages. At each stage, it incorporates the corresponding skip connection and applies a lightweight convolutional mask branch to produce intermediate mask features X_m. This branch consists of a series of depth-wise separable [66] coordinate convolutions [67] followed by a Squeeze-and-Excitation (SE) block [68], allowing the model to focus on informative spatial channels with low computational cost.

These mask features serve as input to the Transformer decoder, which operates on a fixed set of learnable object queries $q \in \mathbb{R}^{N\times D}$. The Transformer decoder consists of L blocks per layer. At each layer, queries interact with the mask features via cross-attention and are updated using a feedforward network (FFN). We adopt a sequential (seq.) update strategy, where object queries are refined through all the Transformer decoder blocks within one layer before being passed to the next one. This process is iterative, with updated queries passing through each decoder stage.

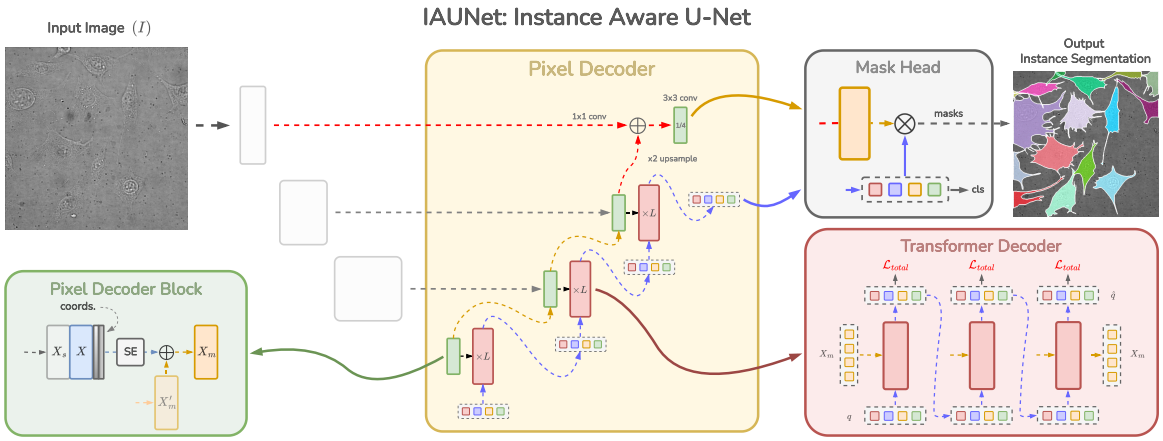


Figure 14. **Model overview.** Overview of the IAUNet architecture, highlighting the Pixel and Transformer Decoder stages. Given an input image I, the encoder extracts multi-scale features as skip connections for the Pixel decoder. At each decoder block, we add skip connections X_s to the main features X and inject normalized coordinate features for CoordConv. [67]. Stacked depth-wise convolutions [66] with an Squeeze-and-Excitation (SE) block [68] refine spatial information, generating mask features X_m. The Transformer decoder then processes learnable queries q through three Transformer blocks per layer, iteratively refining them with X_m. Deep supervision loss is applied after each Transformer block using updated queries $\hat{q}$ and high-resolution mask features.

In the final stage, the mask head combines mask features and instance queries to produce output instance masks.

3.4 Pixel Decoder

In the biomedical domain, U-Net [6], with all its variants [7, 26, 27, 69], remains one of the most effective architectures for accurate segmentation. This is primarily due to the design of U-Net's decoder, which maintains high spatial and semantic fidelity by using skip connections. Inspired by this design, we include a convolutional decoder, referred to as the Pixel decoder. Our Pixel decoder (Figure 14, middle panel) operates on two feature types: main decoder features X and auxiliary mask features X_m. The main features play a similar role to those in standard U-Net, aggregating multi-scale spatial information through skip connections X_s, while the mask features are dedicated to enriching the semantic context used by the Transformer decoder (Section 3.5).

A key difference in our decoder is the consistent use of high-dimensional feature channels across all spatial scales. This is necessary to provide rich semantic input to the Transformer decoder, which depends on these features to enable instance-level separation. However, maintaining constant channel dimensionality across all decoder levels places a high computational burden, as the spatial resolution increases at each stage. Since the convolutional cost grows quadratically with spatial size, this results in an increasing number of floating-point operations per second (FLOPs) at higher decoder layers. To mitigate this issue, we design a lightweight Pixel decoder architecture that minimizes computational overhead while preserving representational capacity.

At each decoder level, the skip connection X_s is first projected to 256 channels using a 1×1 convolution. This standardizes the feature dimensionality before concatenation. The projected

skip feature is concatenated with the upsampled decoder feature from the previous level X' and passed through a lightweight double 3×3 depth-wise convolution [66], batch normalization [64], and ReLU block [63], denoted as G_x (Eq. (4)). Due to the complex and spatially variable structure of biomedical images, we enhance the decoder with Coordinate Convolution (CoordConv) [67], which adds two additional channels representing normalized spatial coordinates to the feature maps. This injects explicit positional information into the decoder without increasing computational complexity. Experimentally, we find this modification to be critical for improving the model's ability to separate adjacent and overlapping instances, leading to substantial gains in segmentation performance. The resulting output is further refined using a Squeeze-and-Excitation (SE) block [68] to produce the updated main features X:

$$X = SE\Big(G_x([X_s, X']) + X'\Big) \tag{4}$$

To preserve semantic alignment with the Transformer decoder, we simultaneously update the mask features. The upsampled mask features X'_m are combined with the current main features X, and the result is processed by a separate mask branch G_m, consisting of two stacked 3×3 convolutions:

$$X_m = G_m\Big(X_m + X\Big) \tag{5}$$

The updated mask features X_m are used for query refinement in the Transformer decoder. Finally, both X and X_m are upsampled using bilinear interpolation and passed to the next decoder layer.

3.4.1 CoordConv

To enhance spatial reasoning in the decoder, we incorporate Coordinate Convolution (CoordConv) [67] (Figure 14, green Pixel Decoder Block), which allows the model to explicitly condition feature learning on spatial position. Unlike standard convolutions that are translation-invariant, CoordConv extends the input by embedding absolute coordinate information, enabling the network to better distinguish objects based on their spatial context. Given an input feature map $X \in \mathbb{R}^{D \times H \times W}$, we augment it by concatenating two additional channels: X_{coord} and Y_{coord}, which contain normalized horizontal and vertical coordinates, respectively:

$$X = [X, X_{\text{coord}}, Y_{\text{coord}}] \tag{6}$$

where X_{coord} and Y_{coord} are additional channels encoding normalized spatial coordinates in the range $[-1, 1]$. At every decoder stage, we concatenate these coordinate planes to the feature tensor $X \in \mathbb{R}^{D \times H \times W}$, resulting in an augmented representation $X \in \mathbb{R}^{(D+2) \times H \times W}$.

This augmentation preserves the original semantics while providing explicit positional cues. CoordConv is applied at each decoder stage before convolution, and we find it to be particularly beneficial in separating spatially adjacent and overlapping instances. Despite its simplicity, this addition introduces negligible computational overhead and yields consistent improvements in instance segmentation performance.

3.5 Transformer Decoder

Object queries are central to state-of-the-art instance segmentation architectures [19, 21, 22, 70], where each query serves as a learnable embedding that represents an individual object. Typically, these are D-dimensional vectors, which encode the semantic identity of a potential object and interact with pixel features via cross-attention to aggregate object-specific context. Transformer-based models such as DETR [17], Deformable DETR [51], MaskFormer [18], and Mask2Former [19] rely heavily on queries to directly predict object masks or bounding boxes in an end-to-end fashion. In our model, we use a set of N learnable queries $q \in \mathbb{R}^{N \times 256}$, where each query encodes the semantic representation of a potential object instance. These queries are progressively refined across multiple stages of the Transformer decoder using cross-attention with the corresponding mask features X_m. The mask features are flattened to a sequence of length $L = H_l \times W_l$, resulting in a representation $X_m \in \mathbb{R}^{L \times 256}$ for the decoder layer at resolution level $l \in [1, L]$. At each decoder stage, we apply three successive Transformer decoder blocks to refine the queries (see Figure 15). This iterative process updates both semantic and spatial information in the query embeddings, improving their ability to represent distinct object instances. The overall design is illustrated in Figure 14, red block.

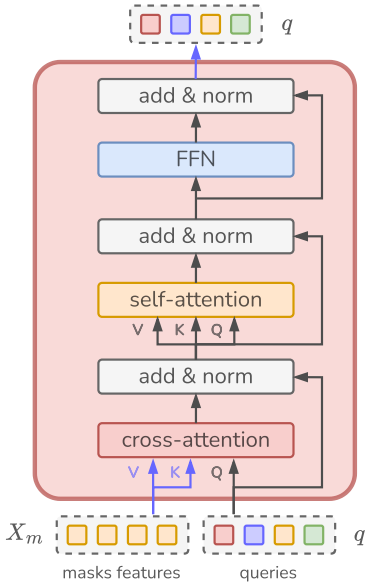


Figure 15. **IAUNet Transformer decoder block.** Each block consists of a cross-attention layer between object queries q and flattened pixel features X_m, followed by self-attention among queries and a feed-forward network (FFN). All sub-layers are followed by residual connections and layer normalization. This structure is applied iteratively across decoder layers to progressively refine instance-level query embeddings.

3.5.1 Positional Embeddings

Transformers are inherently permutation-invariant [24] and lack an inductive bias for spatial structure [55, 71, 72]. To compensate for this, positional information must be explicitly encoded in both the input pixel features and the object queries to maintain spatial awareness. In our model, we apply separate positional embeddings to mask features X_m and the object queries q.

For the mask features, we use the 2D sinusoidal positional encoding scheme introduced in [17, 55]. Given a feature map $X_m \in \mathbb{R}^{D\times H\times W}$, we generate two coordinate grids corresponding to the row and column indices of each spatial position. Each index is then mapped to a $D/2$-dimensional sinusoidal vector using:

$$PE_{(\text{pos},2i)} = \sin\left(\frac{\text{pos}}{10000^{\frac{2i}{D}}}\right), \quad PE_{(\text{pos},2i+1)} = \cos\left(\frac{\text{pos}}{10000^{\frac{2i}{D}}}\right)$$

where $i \in [0, D/2-1]$, and pos denotes the absolute position in either the vertical or horizontal direction. The final encoding $E_{\text{pos}} \in \mathbb{R}^{D\times H\times W}$ is the concatenation of these vectors and is added to the mask feature map:

$$X_m = X_m + E_{\text{pos}} \tag{7}$$

Each position in the feature map receives a unique embedding based on its absolute row and column indices, using sine and cosine functions at multiple frequencies. These encodings are fixed and remain unchanged during training, which supports generalization to unseen input sizes.

For the object queries $q \in \mathbb{R}^{N\times D}$, we assign a learnable positional embedding $P_q \in \mathbb{R}^{N\times D}$. The final representation passed to the Transformer decoder is:

$$q = q + P_q \tag{8}$$

The learnable embeddings provide spatial priors [20, 49, 73] that are refined through training, allowing the decoder to associate each object query with distinct spatial regions in the image [17, 50, 51].

3.5.2 Instance Queries Update

At each decoder layer $l \in \{1, \ldots, L\}$, a set of instance queries $q \in \mathbb{R}^{N\times D}$, where $D = 256$, is refined using the flattened mask features $X_m \in \mathbb{R}^{H_l W_l \times D}$. Each decoder block (see Figure 15) consists of a cross-attention layer, followed by a self-attention layer, and a feed-forward network (FFN).

The multi-head cross-attention mechanism is defined as:

$$q \leftarrow \text{MHAttn}(Q, K, V) + q \tag{9}$$

where

$$Q = qW_Q, \quad K = X_m W_K, \quad V = X_m W_V \tag{10}$$

and $W_Q, W_K, W_V \in \mathbb{R}^{D\times D}$ are learned projection matrices.

The output of attention is computed as:

$$\text{MHAttn}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h) W_O \tag{11}$$

with each head given by:

$$\text{head}_i = \text{softmax}\left(\frac{Q_i K_i^T}{\sqrt{D_h}}\right) V_i \tag{12}$$

where $D_h = D/h$, and $Q_i, K_i, V_i \in \mathbb{R}^{N \times D_h}$ are the linearly projected inputs for the i-th head. The projection $W_O \in \mathbb{R}^{D \times D}$ maps the concatenated heads back to the query dimensionality.

Next, the queries undergo multi-head self-attention:

$$q \leftarrow \text{MHAttn}(Q', K', V') + q \tag{13}$$

where $Q' = K' = V' = qW'$, with shared projection matrix $W' \in \mathbb{R}^{D \times D}$.

After attention, the queries are processed by a feed-forward network:

$$q \leftarrow \text{FFN}(q) + q \tag{14}$$

where

$$\text{FFN}(q) = \text{ReLU}(qW_1 + b_1)W_2 + b_2 \tag{15}$$

with $W_1 \in \mathbb{R}^{D \times 4D}$, $W_2 \in \mathbb{R}^{4D \times D}$, and biases b_1, b_2. Here, $4D$ corresponds to a feed-forward dimension scaled relative to the embedding size $D = 256$. We apply Layer Normalization [74] before each sub-layer and use residual connections after each operation. This sequential refinement strategy is applied over L decoder layers, with each layer updating the queries based on the progressively refined pixel and object-level features. The updated queries are then passed to the next decoder layer. During training, we supervise all the Transformer decoder layers with deep supervision loss. Specifically, at each layer, we consider the newly refined queries q and the high-resolution pixel features described in Section 3.5.3 to predict instance masks. We then compute the loss on these predictions (as described in Section 3.6), effectively propagating gradients through all layers of the Transformer decoder.

3.5.3 Mask Head

To keep the prediction process lightweight without compromising performance, we restrict mask decoding to high-resolution features. As shown in Figure 14 (red arrows), we follow a similar design proposed in MaskDINO [20] and construct a dense pixel embedding map by combining the $1/4$ resolution backbone feature map X_b with the upsampled $1/8$ resolution mask features X_m from the Pixel decoder. This fusion is designed to preserve fine spatial details from the high-resolution encoder while enriching them with refined semantic information from the decoder.

The fusion proceeds in two steps. First, we apply a convolutional layer $\mathcal{F}$ to X_b to align its channel dimension with the Transformer hidden size. Next, the mask features X_m are upsampled by a factor of 2 using bilinear interpolation, denoted by $\mathcal{U}$. The resulting tensors are element-wise added and passed to a segmentation head $\mathcal{M}$, consisting of a 3×3 convolution layer, batch normalization, and ReLU activation block, which produces the final pixel embedding map used for instance prediction. Each refined query $q \in \mathbb{R}^{N \times D}$ is linearly projected into the mask embedding vector $q_c \in \mathbb{R}^{N \times D}$, and a vector of class logits. Mask prediction is formulated as a dot product between the mask embeddings and the fused pixel features:

$$m = q_c \otimes \mathcal{M}\left(\mathcal{F}(X_b) + \mathcal{U}(X_m)\right), \tag{16}$$

where $m \in \mathbb{R}^{N \times H \times W}$ is the predicted mask tensor for all N instances.

In parallel, each query also predicts a class probability distribution over $K + 1$ classes, where the extra category denotes the "no object" ($\varnothing$) case. These logits are passed through a softmax layer to obtain the final class probabilities $c_i \in [0, 1]$. During inference, we re-rank the predicted masks using the maskness score [19]. This score captures the average activation of a predicted mask and is defined as:

$$p_i = \frac{1}{H \cdot W} \sum_{(h,w)} m_i(h, w), \tag{17}$$

where m_i is the sigmoid-activated mask prediction for query i. The final confidence score for each predicted instance combines class confidence and maskness:

$$\hat{c}_i = c_i \cdot p_i, \tag{18}$$

where $\hat{c}_i$ represents the re-scored confidence for instance i. This formulation ensures that instances with both high semantic certainty and spatial coherence are prioritized in the final predictions.

3.6 Mask Level Matching

During training, the model outputs $\{M_n\}_{n=1}^{N}$ predicted masks, where $N > M$, the number of ground truth masks $\{G_m\}_{m=1}^{M}$. To align predictions with ground truth targets, we adopt a bipartite matching strategy based on the Hungarian algorithm [47], following the formulation introduced in DETR [17]. The objective is to find an optimal one-to-one matching that minimizes the total assignment cost. Formally, we define a cost matrix $C \in \mathbb{R}^{N \times M}$, where each element $C(n, m)$ measures the dissimilarity between the predicted mask M_n and ground truth mask G_m. The assignment problem is defined as:

$$\min_{X} \sum_{n=1}^{N} \sum_{m=1}^{M} C(n, m) \cdot X(n, m) \tag{19}$$

subject to:

$$\begin{aligned} \sum_{n=1}^{N} X(n, m) &= 1 \quad \forall m \\ \sum_{m=1}^{M} X(n, m) &= 1 \quad \forall n \\ X(n, m) &\in \{0, 1\} \quad \forall n, m \end{aligned}$$

Alternatively, we can express the optimal permutation σ as:

$$\sigma = \arg\min_{\sigma \in S} \sum_{i=1}^{M} \mathcal{L}_{\text{match}}(M_{\sigma(i)}, G_i) \tag{20}$$

Each cost entry is computed as a weighted combination of classification and mask similarity losses:

$$C(n,m) = \lambda_{\text{cls}} \cdot \mathcal{L}_{\text{cls}}(c_n, c_m) + \lambda_{\text{bce}} \cdot \mathcal{L}_{\text{bce}}(M_n, G_m) + \lambda_{\text{dice}} \cdot \mathcal{L}_{\text{dice}}(M_n, G_m) \tag{21}$$

For the classification loss, we use standard cross-entropy [65]:

$$\mathcal{L}_{\text{cls}}(c_n, c_m) = -\sum_{k=1}^{K} y_m^{(k)} \log \hat{y}_n^{(k)} \tag{22}$$

where $y_m^{(k)} \in 0, 1$ is the one-hot encoded ground truth class and $\hat{y}_n^{(k)}$ is the predicted probability. The ”no object” class is down-weighted by a factor of 0.1.

The binary cross-entropy loss [65] between predicted and ground truth masks is defined as:

$$\mathcal{L}_{\text{bce}}(M_n, G_m) = -\frac{1}{N} \sum_{i=1}^{N} \left[G_m^{(i)} \log M_n^{(i)} + (1 - G_m^{(i)}) \log(1 - M_n^{(i)}) \right] \tag{23}$$

where $N = H \cdot W$ is the number of pixels in the mask.

To assess overlap quality, we use the Dice loss [75]:

$$\mathcal{L}_{\text{dice}}(M_n, G_m) = 1 - \frac{2 \cdot \sum_{i=1}^{N} M_n(i) G_m(i)}{\sum_{i=1}^{N} M_n(i)^2 + \sum_{i=1}^{N} G_m(i)^2 + \epsilon} \tag{24}$$

where ϵ is a small constant for numerical stability.

Following [19], we set the weights as $\lambda_{\text{cls}} = 1.0$, $\lambda_{\text{bce}} = 5.0$, and $\lambda_{\text{dice}} = 2.0$. The final loss over matched pairs can be defined as a combination of mask and classification losses:

$$\mathcal{L} = \lambda_{\text{cls}} \cdot \mathcal{L}_{\text{cls}} + \lambda_{\text{bce}} \cdot \mathcal{L}_{\text{bce}} + \lambda_{\text{dice}} \cdot \mathcal{L}_{\text{dice}} \tag{25}$$

This joint formulation ensures that classification accuracy and spatial alignment of masks are optimized simultaneously in an end-to-end manner.

3.7 Alternative Approaches

To compare our approach for instance segmentation, we evaluate a set of representative models from the three main categories discussed earlier in Section 2: region-based, query-based, and specialized biomedical segmentation methods. For each model, we adopt publicly available implementations and configurations, adjusting them minimally to fit the datasets used in our experiments.

Mask R-CNN is a region-based two-stage model that extends Faster R-CNN by adding a mask prediction branch for pixel-level segmentation. We use the MMDetection [76] implementation of the model with both pretrained on ImageNet [77] ResNet [46] and Swin Transformer [78]

backbones. For training, we use Stochastic Gradient Descent (SGD) [79] with a learning rate of 0.02, momentum of 0.9, and weight decay of 1e-4. The learning rate schedule includes a linear warm-up phase with a start factor of 0.001 for the first 500 iterations, followed by a multi-step decay at epochs 800 and 900 with a gamma of 0.1. During inference, the Region Proposal Network (RPN) selects up to 1000 candidate regions, applying non-maximum suppression (NMS) with an IoU threshold of 0.7. The RoI head processes these proposals, applies a classification score threshold of 0.05, performs NMS with an IoU threshold of 0.5, and keeps the top 100 outputs. Predicted masks are binarized using a threshold of 0.5.

PointRend is a refinement-based extension of Mask R-CNN that improves mask quality by adaptively sampling and refining uncertain regions at higher resolution. It retains the two-stage architecture of Mask R-CNN but adds a lightweight point-based prediction head that iteratively samples points near object boundaries and predicts their labels using an MLP. We use the MMDetection implementation [76] and follow the same training setup as for Mask R-CNN, using SGD with a learning rate of 0.02, momentum of 0.9, and weight decay of 1e-4. We use the same linear scheduler setup with warm-up and step-based decay.

Mask2Former is a single-stage instance segmentation model that uses a Transformer decoder and masked attention to iteratively refine object queries over multi-scale features. We use the MMDetection implementation [76] with pretrained ResNet and Swin Transformer backbones. More specifically, we train the model using the AdamW optimizer with a base learning rate of 1e-4, weight decay of 0.05, and gradient clipping with a max norm of 0.01. For the backbone, we use a learning rate multiplier of 0.1 and set weight decay to zero.

MaskDINO adopts the same single-stage Transformer-based architecture as DINO [49], extending it with a segmentation head that predicts instance masks directly from object queries. We use the official Detectron2 [80] implementation and follow the same training configuration as Mask2Former. Specifically, we use the AdamW optimizer with a base learning rate of 1e-4 and weight decay of 0.05. The backbone is trained with a reduced learning rate multiplier of 0.1. Similar to previous models, we use both ResNet and Swin Transformer backbones initialized from pretrained weights. For each dataset, we train two variants of the model using 100 and 300 object queries, and compare their performance to our proposed implementation.

CellPose is a U-Net-based instance segmentation model that predicts spatial gradients and a binary cell mask to generate vector fields pointing toward object centers. We use the official CellPose implementation [43] and train the model using the Adam optimizer with a learning rate of 1e-4 and a weight decay of 1e-5. The network includes residual connections with four convolutional layers per downsampling stage and uses style-based skip connections in the decoder. Alongside the main model, we train the corresponding size model (SM) [45], a linear regression module that estimates the average object diameter in an image based on extracted style features. At inference, we use the learned size model to estimate the diameter of cells in each image and rescale accordingly before feeding the image to the CellPose model. We also compare the performance of the model, setting the diameter of cells to be the same as the mean diameter from the training set. We apply the default flow threshold of 0.4 to determine convergence in the predicted vector fields during mask reconstruction.

Cell-DETR adapts the DETR [17] architecture for instance segmentation in biomedical images by incorporating a Transformer decoder and CNN-based mask head. We use the official implementation [52], which employs a ResNet-34 backbone for feature extraction. To ensure consistency with other query-based models in our benchmark, we extend the number of object queries from the default to 100. The model is trained using the AdamW optimizer with a learning rate of 1e-4 and weight decay of 1e-6. A MultiStep learning rate scheduler is applied with milestones at epochs 50 and 100 and a decay factor of 0.1.

YOLOv8 and YOLOv9. We train YOLOv8 and YOLOv9 models using the official Ultralytics implementation [81]. Both models are optimized using AdamW with a base learning rate of 1e-3 and weight decay of 5e-4. A cosine learning rate schedule [82] is applied with linear warmup over the first 10 epochs. Momentum is set to 0.937, with warmup momentum initialized at 0.8. We apply HSV-based color augmentation with hue, saturation, and value jittering factors of 0.015, 0.7, and 0.4, respectively. Geometric augmentations include random translation ($+/-10\%$), and scaling ($+/-50\%$), with horizontal and vertical flipping applied with 50% probability. Additional augmentations include random erasing with 40% probability and mosaic augmentation. We use the RandAugment policy for automated augmentation during classification training. During inference, we retain a maximum of 100 predictions per image, apply non-maximum suppression with an IoU threshold of 0.5, and threshold predicted instance masks at 0.5 to produce final binary segmentations.

SAM. We evaluate the pretrained SAM-B and SAM-L models using the Ultralytics implementation [81]. For all experiments, input images are resized to 512×512, and instance masks are thresholded at 0.5. A confidence threshold of 0.25 is used to filter low-confidence predictions. We test SAM with two prompt modalities: *point* prompts and *box* prompts. For point prompts, we use the center of each ground truth bounding box as input. For box prompts, the full ground truth bounding box is provided as input. Each instance is prompted individually, and the model returns a binary mask per object conditioned on the given prompt.

3.8 Writing Aid

We used ChatGPT [83] and Grammarly [84], and Overleaf [85] to assist in structuring text, refining grammar, and helping with LaTeX editing and formatting of figures and tables. These tools supported the writing process and improved the overall clarity and presentation of the manuscript.

4. Experiments and Results

In this section, we evaluate our proposed IAUNet (Section 3.3) on four diverse datasets: EVICAN2 [58], the ISBI 2014 Overlapping Cervical Cytology dataset [61], LIVECell [57], and our novel Revvity-25 dataset. These benchmarks include fluorescent and brightfield microscopy images as well as synthetic cytology images with overlapping structures. For benchmarking, we evaluate all models introduced in Section 3.7 and compare them to our model using consistent training and inference protocols. In addition, we conduct detailed ablation studies to assess the contribution of each architectural component in IAUNet and its effect on segmentation performance.

4.1 Implementation Details

All experiments were implemented in Python 3.9.19 using the PyTorch deep learning framework [86]. Data processing and image transformations were handled using NumPy [87], SciPy [88], and OpenCV [89], while Matplotlib [90] was used to visualize results and generate plots.

We used Hydra [91] to manage experiment configurations, ensuring a modular and flexible setup across all datasets and models. Experiment tracking and logging were performed using Weights & Biases (WandB) [92], enabling consistent monitoring of training metrics, losses, and outputs.

4.2 Training Details

All experiments were conducted on a single Tesla V100 GPU with 32GB of memory, provided by the High-Performance Computing Center at the Institute of Computer Science, University of Tartu. We follow a unified training protocol aligned with prior work [19] to ensure comparability across models.

To train our IAUNet model, we use the AdamW optimizer [93] with an initial learning rate of 1e-4, weight decay of 0.05, and the CosineAnnealingLR scheduler [82] with a minimum learning rate of 1e-6. All models are trained to full convergence with a batch size of 8. For data augmentation, we adopt the Albumentations library [94]. We apply longest-side resizing to 512 $\times$ 512 pixels while preserving the original aspect ratio. This is followed by scale jittering [95] in the range $[0.8, 1.5]$, fixed-size cropping to 512 $\times$ 512, and random horizontal and vertical flipping. During inference, we apply longest-side resizing to 512 $\times$ 512 pixels as input to the model. Unless specified, this resizing procedure is consistently used across all evaluations. For evaluation, we remove any padding introduced during longest-side resizing and resize the predicted masks back to the original input dimensions. Predictions are then binarized using a fixed threshold of 0.5. Since the models in Section 3.7 come from different codebases, we perform inference using a unified implementation to ensure consistency across model outputs.

For the ISBI2014 dataset [61], we follow the official train, validation, and test splits. All models except CellPose [43] are trained to jointly segment both the cell and nucleus classes. As CellPose does not support multi-class segmentation, we train two separate models, one for each class, and report the average performance across the two.

4.3 Metrics

To evaluate instance segmentation performance, we adopt the mean Average Precision (mAP) metric, which is widely used in object detection and segmentation benchmarks [19, 96], particularly in the COCO evaluation protocol [96]. mAP measures both the accuracy of object localization and segmentation quality, making it well-suited for evaluating models that predict instance-level masks.

Given a set of predicted masks $\{M_n\}_{n=1}^{N}$ and a set of ground truth masks $\{G_m\}_{m=1}^{M}$, we first compute pairwise Intersection-over-Union (IoU) scores between them. The Average Precision (AP) is computed as the area under the Precision-Recall (PR) curve for a given Intersection-over-Union (IoU) threshold. First, the predicted masks are ranked and sorted by their confidence scores $\hat{c}_n$ (see Section 3.6). A predicted mask $M_n \in \{M_n\}_{n=1}^{N}$ is considered a true positive (TP) if it matches a ground truth mask $G_m \in \{G_m\}_{m=1}^{M}$ with IoU above the given threshold (Eq. (26)), and if it is the highest-scoring unmatched prediction for that ground truth. All unmatched predictions are counted as false positives (FP), while unmatched ground truths are considered false negatives (FN).

The IoU between a predicted mask M and a ground truth mask G is defined as:

$$\text{IoU}(M, G) = \frac{|M \cap G|}{|M \cup G|}. \tag{26}$$

Using the matched predictions, we compute:

$$\text{Precision}(p) = \frac{\text{TP}}{\text{TP} + \text{FP}}, \qquad \text{Recall}(r) = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{27}$$

where TP, FP, and FN denote the number of true positives, false positives, and false negatives, respectively.

The AP is defined as the area under the Precision-Recall curve:

$$\text{AP}_\tau = \int_0^1 p(r)dr, \tag{28}$$

where τ is the IoU threshold.

In practice, AP is approximated by sampling precision at 101 evenly spaced recall values from 0 to 1 [96]. For each recall threshold r, the maximum precision over all $\tilde{r} \geq r$ is used. These values are then averaged to compute AP:

$$\text{AP}_\tau \approx \frac{1}{101} \sum_{r \in [0.00, \ldots, 1.00]} \max_{\tilde{r} \geq r} p(\tilde{r}). \tag{29}$$

The mean Average Precision can then mathematically be defined as the average over multiple IoU thresholds:

$$\text{mAP} = \frac{1}{T} \sum_{t=1}^{T} \text{AP}_{\tau_t}, \quad \text{where } \tau_t \in 0.50, 0.55, \ldots, 0.95. \tag{30}$$

We compute mAP under the following COCO-style configurations:

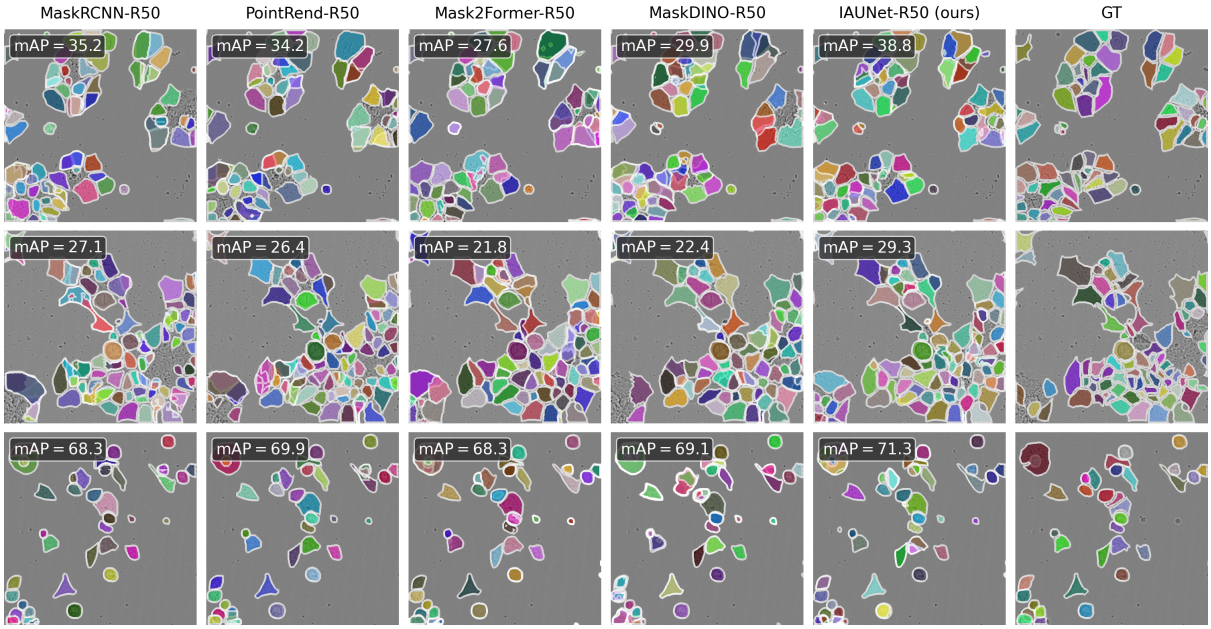


Figure 16. **LIVECell**. Visualization of instance segmentation predictions on the LIVECell dataset across different state-of-the-art models (using ResNet50 backbone). We also report per-image AP score. Last columns shows ground-truth annotations.

- mAP@[0.5:0.95]: averaged over 10 IoU thresholds from 0.50 to 0.95 in steps of 0.05, capturing performance across varying levels of strictness.
- AP@50 and AP@75: single-threshold scores at IoU = 0.50 and IoU = 0.75, respectively, reflecting more relaxed or strict overlap requirements.
- AP_S, AP_M, AP_L: evaluated on small (area $< 32^2$), medium ($32^2 \leq$ area $< 96^2$), and large (area $\geq 96^2$) objects.

As mentioned in Section 4.2, we use a unified implementation to evaluate all model outputs. Specifically, we adapt the MMDetection implementation [76] of the COCO instance segmentation evaluator into a more flexible pipeline for our benchmarks.

4.4 Main Results

In this subsection, we benchmark IAUNet across multiple datasets and compare its performance with a diverse set of state-of-the-art instance segmentation models introduced in Section 3.7. All models are evaluated using the unified evaluation framework described in Section 4.3, and a maximum of 100 object queries is used unless otherwise noted. Table 1 summarizes the instance segmentation performance across five public biomedical datasets: LIVECell, EVICAN2-Easy, EVICAN2-Medium, EVICAN2-Difficult, and ISBI2014. We report both AP, which is a $mAP_{50:95}$, and AP_{50} scores. Additionally, we include the number of model parameters and (Giga)FLOPs to provide a comprehensive comparison of model efficiency and scale.

4.4.1 Models with Convolution-Based Backbones

IAUNet consistently achieves top performance across convolutional and transformer backbones while maintaining a low parameter count and FLOPs. With a ResNet-50 backbone, IAUNet

			LIVECell		*EVICAN2$_E$*		*EVICAN2$_M$*		*EVICAN2$_D$*		*ISBI2014*			
Models	backbones	num_queries	AP	AP$_{50}$	AP	AP$_{50}$	AP	AP$_{50}$	AP	AP$_{50}$	AP	AP$_{50}$	#params.	FLOPs
Models with Convolution-Based Backbones														
Mask R-CNN [14]	R50	100	44.7	74.2	48.1	75.9	20.7	42.5	19.1	39.8	58.9	**88.7**	44M	115G
PointRend [34]	R50	100	44.0	73.5	26.6	47.9	18.0	38.5	13.4	28.3	**60.0**	**88.7**	56M	66G
Mask2Former [19]	R50	100	43.7	73.8	53.4	89.1	29.1	54.9	24.2	**50.4**	58.5	87.5	44M	67G
MaskDINO [20]	R50	100	43.3	73.5	50.7	83.9	29.3	57.9	22.0	41.9	55.4	86.8	44M	64G
IAUNet (ours)	R50	100	**45.3**	**75.3**	**58.0**	**91.8**	**32.1**	**59.0**	**24.9**	45.4	56.0	85.0	39M	49G
Mask R-CNN [14]	R101	100	44.2	73.2	41.5	69.9	23.3	46.9	17.8	36.7	**60.7**	88.8	63M	134G
PointRend [34]	R101	100	44.0	73.7	41.3	65.2	20.2	39.3	14.8	32.1	60.3	**89.2**	75M	86G
Mask2Former [19]	R101	100	44.0	73.5	54.4	87.8	27.1	51.7	20.4	42.4	59.5	88.6	63M	86G
MaskDINO [20]	R101	100	43.4	73.6	53.7	85.0	31.8	59.2	**27.1**	**51.3**	55.7	87.4	63M	84G
IAUNet (ours)	R101	100	**45.4**	**75.5**	**58.3**	**92.7**	**32.9**	**59.6**	26.9	50.0	56.5	87.1	58M	69G
Models with Transformer-Based Backbones														
Mask R-CNN [14]	Swin-S	100	44.3	73.3	52.6	91.7	27.0	59.2	20.2	50.2	61.9	90.7	69M	141G
PointRend [34]	Swin-S	100	43.9	73.5	55.1	89.2	30.1	61.6	24.4	54.6	**62.1**	**91.0**	81M	93G
Mask2Former [19]	Swin-S	100	44.6	74.3	**65.2**	**96.8**	**36.2**	66.7	**30.9**	62.7	57.1	87.3	69M	93G
MaskDINO [20]	Swin-S	100	43.9	73.8	57.0	86.9	33.6	64.9	27.6	56.9	52.7	85.3	71M	181G
MaskDINO [20]	Swin-S	300	44.8	75.1	56.5	91.8	35.0	**70.7**	30.2	**64.3**	51.2	83.4	71M	187G
IAUNet (ours)	Swin-S	100	45.4	75.4	58.8	93.1	32.2	61.9	27.7	54.1	61.1	90.1	64M	76G
IAUNet (ours)	Swin-S	300	**45.6**	**76.4**	60.9	93.6	33.2	62.0	29.6	58.0	61.8	89.8	64M	87G
Mask R-CNN [14]	Swin-B	100	44.2	73.1	52.0	89.0	26.7	60.3	24.8	55.5	62.4	**91.5**	107M	186G
PointRend [34]	Swin-B	100	44.0	73.7	58.6	91.0	34.1	64.6	25.8	52.0	62.7	**91.5**	119M	137G
Mask2Former [19]	Swin-B	100	44.9	74.7	55.0	92.5	31.4	60.9	27.7	56.6	58.1	88.4	107M	138G
MaskDINO [20]	Swin-B	100	44.3	74.1	57.3	91.1	37.3	75.7	30.1	65.6	53.5	86.6	110M	226G
MaskDINO [20]	Swin-B	300	45.2	75.8	57.9	91.6	**39.1**	**78.8**	**34.0**	**72.3**	53.3	84.8	110M	232G
IAUNet (ours)	Swin-B	100	45.5	75.6	59.6	93.5	34.2	65.7	28.9	56.9	61.5	90.8	102M	120G
IAUNet (ours)	Swin-B	300	**45.8**	**76.7**	**61.2**	**94.8**	38.0	69.6	30.7	59.9	**63.0**	**91.5**	102M	132G
Specialized Cell Segmentation Methods														
CellPose [43]		-	34.5	60.1	0.9	2.8	0.1	0.3	0.0	0.0	40.5	69.3	6.6M	163.6G
CellPose + SM [45]		-	34.9	60.4	8.7	16.8	1.6	4.4	2.3	6.8	41.6	70.4	6.6M	163.6G
CellDETR [52]	R34	100	13.9	32.7	0	0.1	0.0	0.0	0.0	0.0	0.046	0.135	57M	3.6T
IAUNet (ours)	R50	100	**45.3**	**75.3**	**58.0**	**91.8**	**32.1**	**59.0**	**24.9**	**45.4**	**56.0**	**85.0**	39M	49G
YOLO Family														
YOLOv8-M [15]		-	37.5	72.2	43.8	82.3	27.5	57.1	20.0	46.2	54.9	90.7	27.2M	110.4G
YOLOv8-L [15]		-	40.5	72.5	44.7	83.1	28.1	58.2	20.3	46.1	55.1	91.1	45.9M	220.8G
YOLOv8-X [15]		-	41.1	73.1	45.8	85.6	28.9	59.2	20.7	47.3	55.3	**91.4**	71.8M	344.5G
IAUNet (ours)	Swin-S	100	**45.4**	**75.4**	**58.8**	**93.1**	**32.2**	**61.9**	**27.7**	**54.1**	**61.1**	90.1	64M	76G
YOLOv9-E [16]		-	41.2	73.2	45.6	84.4	27.2	57.9	20.1	47.3	53.3	**91.6**	27.8M	159.1G
YOLOv9-C [16]		-	41.4	73.1	45.9	85.6	28.3	59.8	22.2	49.9	55.7	91.1	60.5M	248.1G
IAUNet (ours)	Swin-S	100	**45.4**	**75.4**	**58.8**	**93.1**	**32.2**	**61.9**	**27.7**	**54.1**	**61.1**	90.1	64M	76G
SAM Family														
SAM-B *(points)* [54]		-	5.0	12.4	28.4	56.0	5.4	13.8	3.2	7.2	33.8	51.8	90M	742G
SAM-B *(boxes)* [54]		-	24.3	56.9	55.0	**96.6**	**38.6**	**91.2**	**34.8**	**82.3**	59.6	**92.8**	90M	742G
IAUNet (ours)	Swin-S	100	**45.4**	**75.4**	**58.8**	93.1	32.2	61.9	27.7	54.1	**61.1**	90.1	64M	76G
SAM-L *(points)* [54]		-	6.3	13.6	28.1	54.1	4.9	12.4	3.2	7.5	32.8	51.0	308M	2.6T
SAM-L *(boxes)* [54]		-	29.2	65.2	57.2	**96.6**	**45.8**	**95.3**	**39.7**	**88.6**	60.8	**93.6**	308M	2.6T
IAUNet (ours)	Swin-B	300	**45.8**	**76.7**	**61.2**	94.8	38.0	69.6	30.7	59.9	**63.0**	91.5	102M	132G

Table 1. **Instance segmentation on LIVECell [57], EVICAN2 (Easy, Medium, Difficult test subsets) [58], and ISBI2014 [61].** IAUNet outperforms strong query-based Mask2Former [19] and MaskDINO [20] baselines for both AP and AP$_{50}$ when training with fewer parameters. For a fair comparison, we only consider single-scale inference and models trained until full convergence. IAUNet remains efficient across different backbones.

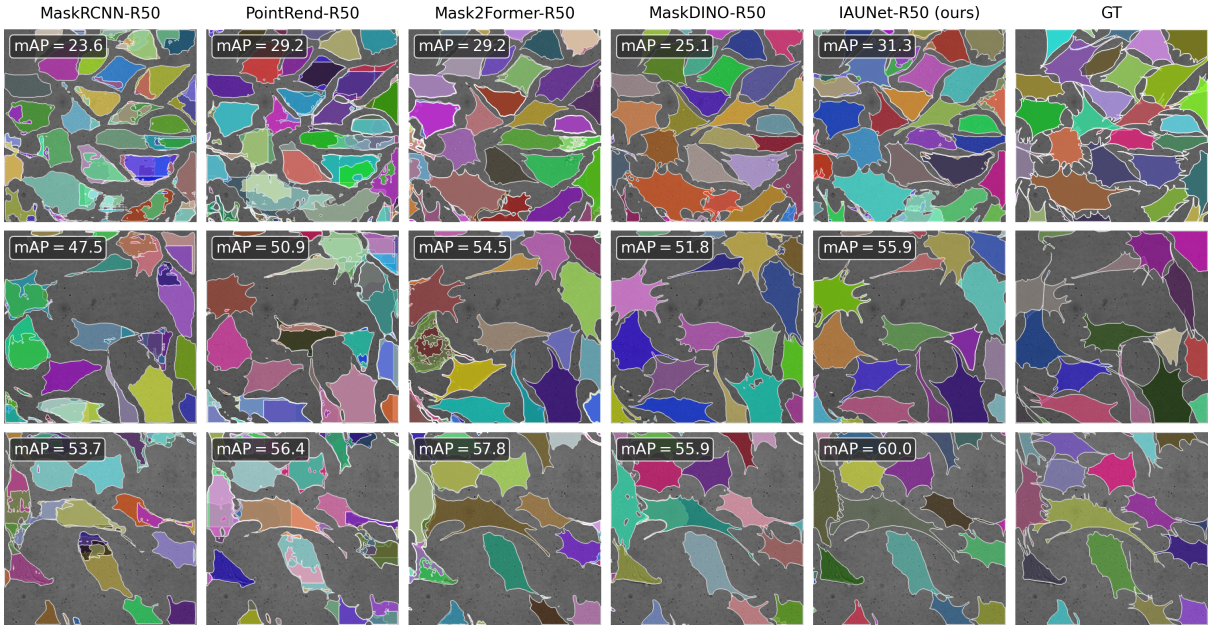


Figure 17. **Revvity-25**. Visualization of instance segmentation predictions on the Revvity-25 dataset across different state-of-the-art models (using ResNet50 backbone). Last columns shows ground-truth annotations. IAUNet as well as MaskDINO show good generalization across tiny details and overlaping instances. We also report per-image AP score.

achieves an AP of 45.3 and AP_{50} of 75.3 on LIVECell, outperforming Mask R-CNN, PointRend, Mask2Former, and MaskDINO while using fewer parameters (39M) and lower computational cost (49G FLOPs). On the same dataset with ResNet-101, it achieves a slightly improved AP of 45.4 and AP_{50} of 75.5. IAUNet maintains comparable or better accuracy than other methods while remaining efficient across various convolutional backbones.

4.4.2 Models with Transformer-Based Backbones

IAUNet also scales better compared to MaskDINO when using transformer-based backbones. With Swin-B and 100 object queries, IAUNet achieves 45.5 AP and 75.6 AP_{50} on LIVECell, outperforming Mask R-CNN (44.2 AP, 73.1 AP_{50}) and MaskDINO (44.3 AP, 74.1 AP_{50}), while using significantly fewer FLOPs (120G vs. 186G and 226G, respectively). This efficiency is largely attributed to IAUNet's lightweight Pixel decoder (see Section 3.4), which aggregates encoder features in a computationally lighter way compared to the multi-scale attention mechanisms used in MaskDINO. IAUNet remains computationally efficient even when increasing the number of queries to 300, further improving performance across most datasets without incurring the high inference cost seen in other models. Similar trends are observed with the smaller Swin-S backbone, where IAUNet continues to outperform comparable models in both accuracy and computational efficiency.

On ISBI2014, IAUNet performs slightly below some baselines, such as PointRend and Mask R-CNN, which achieve higher AP_{50}. We attribute this to the small number of object instances in each image, which can lead to under-utilization of the full query capacity and occasional duplicate predictions. Nevertheless, IAUNet remains competitive, achieving strong AP and robust performance across diverse segmentation challenges.

Revvity-25

Models	backbones	num_queries	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	#params.	FLOPs
Models with Convolution-Based Backbones										
Mask R-CNN [14]	R50	100	39.7	77.2	37.4	0.6	19.0	44.6	44M	115G
PointRend [34]	R50	100	42.2	79.4	40.9	0.4	21.7	47.3	56M	66G
Mask2Former [19]	R50	100	46.4	79.8	49.9	0.7	25.7	52.8	44M	67G
MaskDINO [20]	R50	100	45.6	80.4	48.2	**1.8**	22.3	51.8	44M	64G
IAUNet (ours)	R50	100	**49.7**	**82.1**	**54.8**	0.6	**27.3**	**56.0**	39M	49G
Mask R-CNN [14]	R101	100	40.7	77.5	39.9	0.4	20.1	45.8	63M	134G
PointRend [34]	R101	100	42.9	79.3	42.5	0.0	18.4	48.9	75M	86G
Mask2Former [19]	R101	100	47.2	80.1	51.8	**1.7**	25.7	53.3	63M	86G
MaskDINO [20]	R101	100	47.3	81.0	50.4	0.9	23.0	53.5	63M	84G
IAUNet (ours)	R101	100	**51.5**	**84.7**	**56.1**	**1.7**	**29.2**	**57.8**	58M	69G
Models with Transformer-Based Backbones										
Mask R-CNN [14]	Swin-S	100	24.7	63.4	12.5	0.0	7.3	28.9	69M	141G
PointRend [34]	Swin-S	100	43.6	80.0	43.0	0.5	21.5	48.9	81M	93G
Mask2Former [19]	Swin-S	100	51.2	83.3	56.4	2.7	27.7	58.0	69M	93G
MaskDINO [20]	Swin-S	100	50.3	83.2	53.9	**4.7**	27.6	56.1	71M	181G
MaskDINO [20]	Swin-S	300	49.4	83.6	53.3	2.9	25.8	55.3	71M	187G
IAUNet (ours)	Swin-S	100	53.0	85.7	57.0	1.3	**29.7**	59.1	64M	76G
IAUNet (ours)	Swin-S	300	**53.3**	**86.0**	**59.6**	1.6	29.4	**59.8**	64M	87G
Mask R-CNN [14]	Swin-B	100	27.1	64.9	17.2	0.1	9.7	31.2	107M	186G
PointRend [34]	Swin-B	100	45.2	80.1	47.9	0.1	23.0	50.9	119M	137G
Mask2Former [19]	Swin-B	100	52.0	83.6	58.4	1.1	27.8	59.0	107M	138G
MaskDINO [20]	Swin-B	100	50.5	83.5	54.9	**2.0**	27.1	56.4	110M	226G
MaskDINO [20]	Swin-B	300	50.4	84.3	54.8	0.8	26.3	56.6	110M	232G
IAUNet (ours)	Swin-B	100	53.5	86.1	**59.4**	0.8	**30.5**	59.7	102M	120G
IAUNet (ours)	Swin-B	300	**53.7**	**86.5**	**59.4**	1.0	30.0	**60.3**	102M	132G
YOLO Family										
YOLOv8-M [15]		-	37.5	72.2	35.6	32.8	38.3	46.0	27.2M	110.4G
YOLOv8-L [15]		-	40.5	72.5	42.6	37.4	41.3	46.9	45.9M	220.8G
YOLOv8-X [15]		-	41.1	73.1	43.2	**38.2**	**41.6**	47.6	71.8M	344.5G
IAUNet (ours)	Swin-S	100	**53.0**	**85.7**	**57.0**	1.3	29.7	**59.1**	64M	76G
YOLOv9-E [16]		-	41.2	73.2	43.6	**38.9**	**41.9**	46.0	27.8M	159.1G
YOLOv9-C [16]		-	41.4	73.1	43.9	38.6	**41.9**	47.4	60.5M	248.1G
IAUNet (ours)	Swin-S	100	**53.0**	**85.7**	**57.0**	1.3	29.7	**59.1**	64M	76G
SAM Family										
SAM-B *(points)* [54]		-	5.0	12.4	3.3	14.9	5.5	0.1	90M	742G
SAM-B *(boxes)* [54]		-	24.3	56.9	18.4	**29.1**	24.0	18.9	90M	742G
IAUNet (ours)	Swin-S	100	**53.0**	**85.7**	**57.0**	1.3	**29.7**	**59.1**	64M	76G
SAM-L *(points)* [54]		-	6.3	13.6	5.5	21.4	7.0	0.3	308M	2.6T
SAM-L *(boxes)* [54]		-	29.2	65.2	22.8	**32.4**	29.0	24.8	308M	2.6T
IAUNet (ours)	Swin-B	300	**53.7**	**86.5**	**59.4**	1.0	**30.0**	**60.3**	102M	132G

Table 2. **Instance segmentation on our Revvity-25 dataset (see Section 3.1.4).** IAUNet outperforms strong query-based Mask2Former [19] and MaskDINO [20] baselines as well as other state-of-the-art models when training with fewer parameters. For a fair comparison, we only consider single-scale inference and models trained until full convergence. IAUNet also efficiently scales with more queries while remaining efficient.

4.4.3 Specialized Cell Segmentation Methods

We also compare our IAUNet to specialized cell segmentation methods. While CellPose is specifically designed for cellular imagery, it struggles when object sizes vary significantly between training and test sets, as is the case with EVICAN2, due to its reliance on object

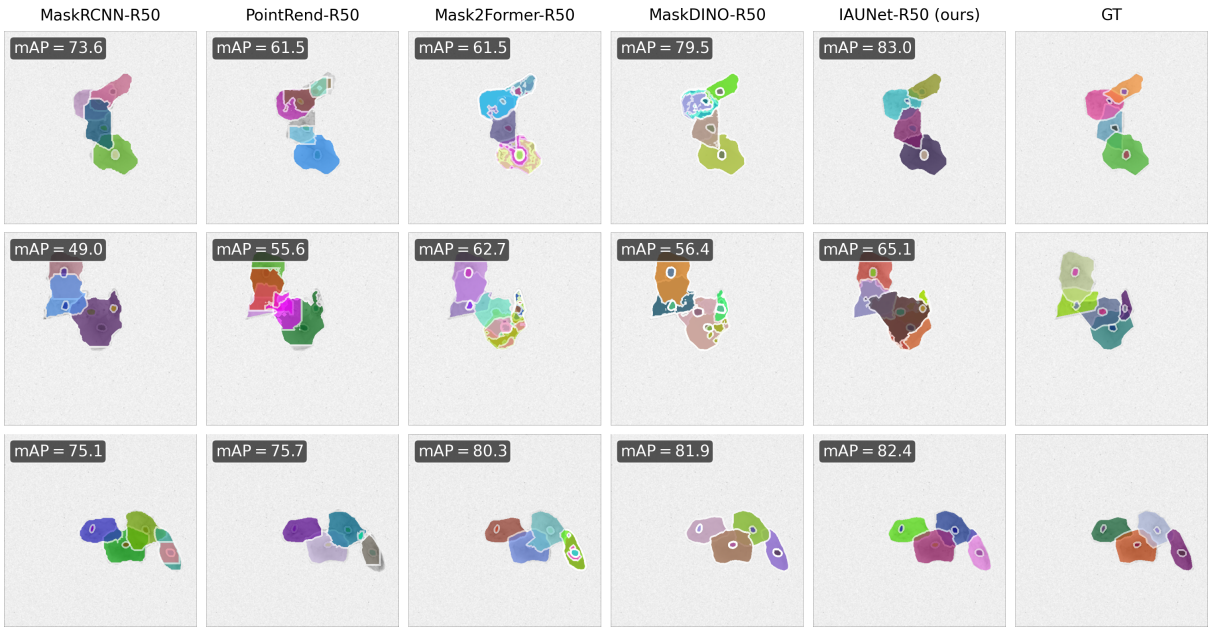


Figure 18. **ISBI2014**. Visualization of instance segmentation predictions on the ISBI2014 dataset across different state-of-the-art models (using ResNet50 backbone). Last columns shows ground-truth annotations. IAUNet as well as MaskDINO show good generalization across tiny details and overlaping instances. We also report per-image AP score.

diameter predictions with the Size Model (SM) [45] for post-processing. CellDETR, on the other hand, does not use object queries as in DETR [17], but instead predicts instances as separate channels using a convolutional head with a softmax activation. Originally, CellDETR was evaluated on 128×128 images. When scaling to high-resolution 512×512 images and using 100 object queries, the approach becomes computationally prohibitive due to the cost of large convolutional operations. Among these specialized methods, IAUNet achieves the best overall results, outperforming CellPose, CellPose + SM, and CellDETR across nearly all datasets, while being significantly more efficient.

4.5 Ablation Studies

In this section, we present an ablation study to evaluate the impact of key components in our model architecture. We focus on understanding how the design choices within the Pixel decoder and Transformer decoder contribute to the overall performance of IAUNet. All experiments are conducted on the LIVECell dataset using a ResNet-50 backbone unless otherwise stated.

Skip Connections in the Pixel Decoder. Since IAUNet extends the U-Net paradigm, skip connections play a central role in integrating multi-scale spatial features. Table 3 evaluates different skip configuration designs. The best performance is achieved using full skip connections over main features X without channel reduction. To improve efficiency, we reduce skip feature channels to 256 via 1×1 convolutions and test two fusion strategies: concatenation and addition. Concatenation maintains performance while reducing FLOPs to 135G. Addition introduces an FPN-like structure but leads to a further performance drop. Finally, we add a lightweight mask head for high-resolution feature refinement, reducing computational cost to 42G with only a minor decrease in AP (43.8).

Pixel Decoder	AP	AP_{50}	AP_{75}	FLOPs
+ full skip	**44.7**	**73.9**	**48.9**	146G
+ 1 × 1 skip concat	44.2	73.8	48.3	135G
+ 1 × 1 skip add	44.3	73.3	48.2	132G
+ light mask head	43.8	73.1	47.4	42G

Table 3. **Pixel Decoder Variants (Skip Connections).** We retain skip connection concatenation as in Eq. (4) and introduce a lightweight mask head Section 3.5.3. These modifications significantly lower the computational cost of the model without much drop in performance.

Pixel Decoder Components. Further, Table 4 presents a step-by-step study of the Pixel decoder components. First, we decouple the main features X into a dedicated mask feature branch X_m (see Section 3.4), improving the alignment of features used in mask prediction. Next, we reduce the feedforward network dimensionality from 2048 to 1024, saving parameters while preserving performance. Adding a Squeeze-and-Excitation (SE) block [68] enhances the channel-wise feature recalibration and improves AP. We also test the inclusion of CoordConv [67], which adds explicit spatial coordinates to the input features (see Section 3.4.1). This modification improves spatial awareness and increases AP to 44.7, showing better localization in cluttered cellular scenes.

Decoder	AP	AP_{50}	AP_{75}	#params.	FLOPs
IAUNet (R50)	43.8	73.1	47.4	34M	42G
+ mask branch X_m	44.0	73.2	47.9	34M	42G
+ FFN (2048 → 1024)	44.1	73.2	48.0	32M	42G
+ SE block [68]	44.2	73.3	48.1	32M	42G
+ CoordConv [67]	44.7	74.1	48.7	32M	42G
+ L (1 → 3) (round-robin.)	44.3	74.0	48.1	39M	49G
+ L (1 → 3) (seq.)	45.1	74.4	**49.4**	39M	49G
+ deep_supervision	**45.3**	**75.3**	**49.4**	39M	49G

Table 4. **Decoder**. We investigate the benefit of adding different decoder components. Adding CoordConv [67] improves object localization. Scaling the Transformer decoder with deep supervision shows the best performance.

Transformer Decoder Structure. We further examine the impact of scaling and structuring the Transformer decoder (Table 4). Increasing the number of decoder layers from $L = 1$ to $L = 3$, we compare two update strategies: cycle-based (round-robin.) and sequential (seq.). The sequential approach, where queries are refined within all blocks of a given layer before progressing, outperforms the round-robin strategy used in Mask2Former [19], achieving 45.1 AP. When combined with deep supervision, where each decoder layer is individually supervised using updated queries and high-resolution mask features X_m, we observe a further improvement to 45.3 AP. These results confirm that deeper supervision and sequential decoding promote more robust instance representation.

Effect of the Number of Queries. Finally, in Table 5, we evaluate how the number of object queries influences segmentation performance. We observe consistent gains when increasing

the query count from 100 to 300 and 500, peaking at 46.1 AP and 76.8 AP_{50} with 500 queries. However, using 1000 queries results in performance saturation, with increased computational cost (104G FLOPs) and no performance gain. These results show that IAUNet effectively utilizes additional queries up to a point, making it adaptable to complex datasets with many object instances while maintaining efficient inference at smaller scales. They also indicate that the approach could be extended and studied further to develop more effective matching strategies, potentially improving the handling of duplicate predictions.

num_queries	AP	AP_{50}	AP_{75}	FLOPs
100	45.3	75.3	49.4	49G
300	<u>45.9</u>	<u>76.5</u>	<u>50.4</u>	61G
500	**46.1**	**76.8**	**50.8**	73G
1000	45.3	76.3	50.0	104G

Table 5. **Num. queries**. Scaling the number of object queries benefits the model.

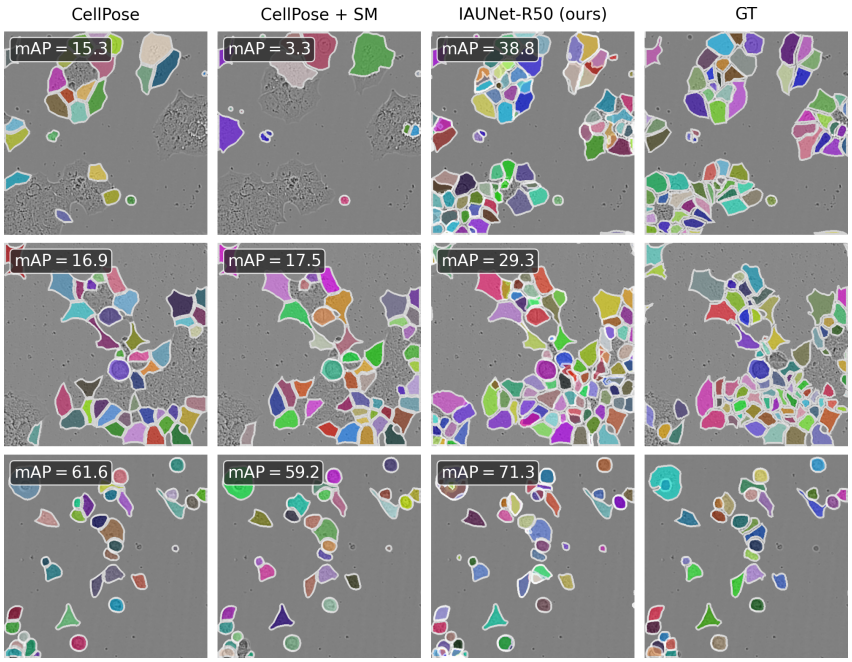


Figure 19. **LIVECell**. Visualization of instance segmentation predictions on the LIVECell dataset across specialized cell segmentation models (using ResNet50 backbone). We also report per-image AP score. Last columns shows ground-truth annotations.

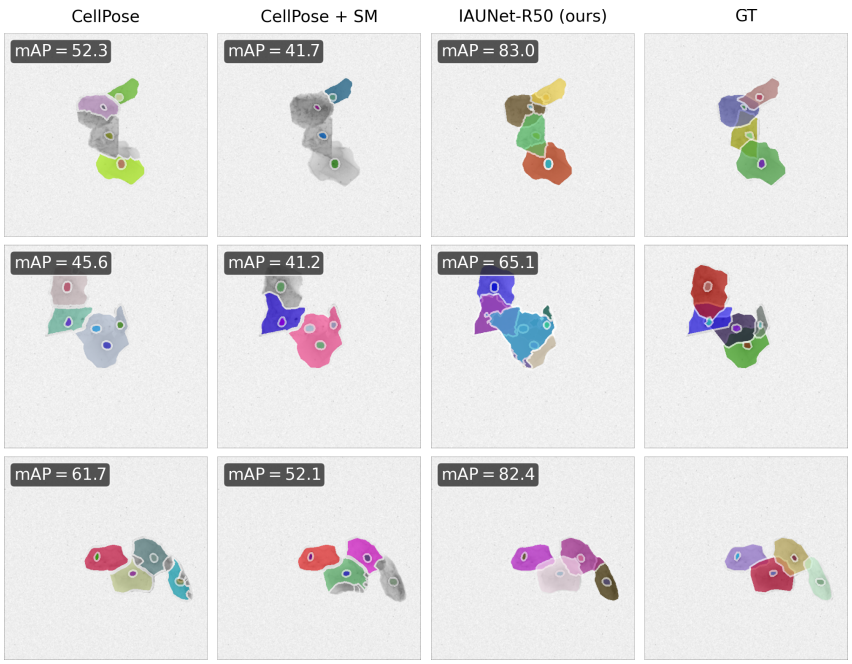


Figure 20. **ISBI2014**. Visualization of instance segmentation predictions on the ISBI2014 dataset across specialized cell segmentation models (using ResNet50 backbone). We also report per-image AP score. Last columns shows ground-truth annotations.

5. Discussion

IAUNet achieves strong instance segmentation performance across diverse microscopy datasets while maintaining a lower computational cost. On LIVECell and Revvity-25, it outperforms previous models across both convolutional and transformer backbones (Table 1 and Table 2). The model scales well with the number of object queries, with performance peaking at 500 queries before saturating (Table 5). Compared to heavier architectures like MaskDINO, IAUNet reduces FLOPs significantly through a lightweight Pixel decoder that aggregates multi-scale features efficiently (Table 3) without loss in performance. Visualization results in Figure 16 and Figure 17 show that IAUNet better resolves overlapping instances and captures object boundaries with higher fidelity.

Ablations in Table 4 show that CoordConv enhances spatial awareness, and deep supervision across Transformer decoder layers yields consistent performance gains, allowing the architecture to scale the number of decoder layers effectively. As a limiting factor, IAUNet shows a drop in AP_{50} on ISBI2014 compared to some baselines, likely due to sparse object counts per image and the reduced utility of dense query sets. Despite this, IAUNet remains efficient and accurate across a wide range of instance sizes and imaging conditions.

5.1 Future Work

While IAUNet achieves strong performance across diverse biomedical segmentation benchmarks, several limitations remain. Our current approach is specialized for instance segmentation. However, query-based methods offer a unified formulation that can be extended to address other segmentation tasks, including binary and multi-class semantic segmentation. In future work, we aim to adapt IAUNet into a general-purpose segmentation framework that can handle multiple task types within a single model.

Another direction involves improving the model's ability to handle overlapping structures, which remains a key challenge in microscopy data. We plan to extend the architecture to explicitly reason about occluded and visible parts of objects. We will utilize our Revvity-25 dataset to assess the model's ability to segment overlapping structures and improve amodal segmentation performance.

Additionally, the approach can be extended by incorporating query guidance or query selection strategies [20, 22], and by adopting hybrid matching schemes similar to those introduced in Co-DETR [97]. Finally, integrating a strong backbone from a foundation model such as SAM2 [54, 98] or DINOv2 [99, 100] could enable a promptable, zero-shot version of IAUNet, potentially improving its transferability to new domains without retraining.

6. Conclusions

In this work, we focus on the task of instance segmentation in microscopy images. Our contributions are twofold.

First, we introduced IAUNet, a novel query-based U-Net architecture for instance segmentation. We propose a hybrid model that integrates a lightweight convolutional Pixel decoder (Section 3.4) with a Transformer decoder (Section 3.5). The Pixel decoder aggregates encoder features at multiple scales, while the Transformer decoder refines object-specific queries across multiple layers. Each layer is supervised using deep supervision, allowing the model to efficiently learn instance representations. As shown in Table 1, IAUNet consistently outperforms existing models across datasets while maintaining low parameter counts and computational cost. The design of the Pixel decoder enables better scalability with Transformer backbones and higher query counts without significantly increasing inference cost.

Second, we introduce Revvity-25 (see Section 3.1.4), a new benchmark dataset for cell instance segmentation. It consists of 110 brightfield microscopy images at 1080×1080 resolution, each containing an average of 27 manually labeled, expert-validated cell instances. Each cell is annotated with high-fidelity polygon masks that capture detailed boundaries and overlapping structures. To our knowledge, Revvity-25 is the first publicly available dataset to combine high-resolution brightfield imaging with precise instance-level annotations that support both modal and amodal segmentation evaluation.

We extensively evaluate IAUNet across several datasets (Section 3.1) and benchmark multiple state-of-the-art models, described in Section 3.7, on our proposed Revvity-25 dataset, establishing a strong foundation for future research in cell segmentation.

This work has undergone peer review and has been accepted to CVPR 2025. It will be presented at the Computer Vision for Microscopy Image Analysis (CVMI) Workshop in Seattle.

7. Acknowledgments

I want to express immense gratitude to the Armed Forces of Ukraine and the bravery of the Ukrainian people for enabling a secure working environment, without which this work would not have been possible.

I would also like to express my deep appreciation to my supervisor and a great friend Dmytro Fishman, who has guided and supported me throughout my bachelor's studies at the Ukrainian Catholic University, my master's program at the University of Tartu, and through the publication of our work at CVPR and the completion of my master's thesis. I am equally grateful to the Biomedical Computer Vision Lab and all its members for their invaluable support, mentorship, and collaboration.

I want to extend a heartfelt thank you to all the professors, tutors, and assistants at the University of Tartu. Special thanks to Dmytro Fishman, Marharyta Domnich, Raul Vicente, Mark Fišel, Jaan Aru, Vitaly Skachek, Kallol Roy, and Jaak Vilo for their guidance, feedback on my presentations and lectures, and engaging discussions.

I am also deeply thankful to Oles Dobosevych, Dean of the Faculty of Applied Sciences at the Ukrainian Catholic University, with whom this entire journey began. Thank you for your motivation, conversations, and friendship. I am further grateful to the tutors at the Ukrainian Catholic University, including Stepan Fedynyak, Rostyslav Hryniv, Oleksii Molchanovskyi, Oleg Farenyuk, and others, for their continued support and openness to dialogue.

Finally, I wish to express my immense gratitude to my family and friends, whose encouragement and belief in me have carried me through every step of this academic journey.

This work was supported by Revvity and funded by the TEM-TA101 grant "Artificial Intelligence for Smart Automation." Computational resources were provided by the High-Performance Computing Cluster at the University of Tartu.

References

[1] Ali M. A. S., Hollo K., Laasfeld T., Torp J., Tahk M.-J., Rinken A., Palo K., Parts L., and Fishman D. ArtSeg—Artifact segmentation and removal in brightfield cell microscopy images without manual pixel-level annotations. *Scientific Reports* 12.1 (July 2022), p. 11404.

[2] Kherlopian A. R., Song T., Duan Q., Neimark M. A., Po M. J., Gohagan J. K., and Laine A. F. A review of imaging techniques for systems biology. *BMC systems biology* 2 (2008), pp. 1–18.

[3] Moen E., Bannon D., Kudo T., Graf W., Covert M., and Van Valen D. Deep learning for cellular image analysis. *Nature Methods* 16.12 (Dec. 2019), pp. 1233–1246.

[4] Shrestha P., Kuang N., and Yu J. Efficient end-to-end learning for cell segmentation with machine generated weak annotations. *Communications Biology* 6.1 (Mar. 2023), p. 232.

[5] He K., Zhang X., Ren S., and Sun J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. 2015. arXiv: 1502.01852 [cs.CV]. https://arxiv.org/abs/1502.01852.

[6] Ronneberger O., Fischer P., and Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation. 2015. arXiv: 1505.04597 [cs.CV]. https://arxiv.org/abs/1505.04597.

[7] Zhou Z., Siddiquee M. M. R., Tajbakhsh N., and Liang J. UNet++: A Nested U-Net Architecture for Medical Image Segmentation. 2018. arXiv: 1807.10165 [cs.CV]. https://arxiv.org/abs/1807.10165.

[8] Selinummi J., Ruusuvuori P., Podolsky I., Ozinsky A., Gold E., Yli-Harja O., Aderem A., and Shmulevich I. Bright field microscopy as an alternative to whole cell fluorescence in automated analysis of macrophage images. en. *PLoS One* 4.10 (Oct. 2009), e7497.

[9] Zernike F. Phase contrast, a new method for the microscopic observation of transparent objects. *Physica* 9.7 (1942), pp. 686–698. DOI: https://doi.org/10.1016/S0031-8914(42)80035-X. https://www.sciencedirect.com/science/article/pii/S003189144280035X.

[10] Sanderson M. J., Smith I., Parker I., and Bootman M. D. Fluorescence microscopy. en. *Cold Spring Harb Protoc* 2014.10 (Oct. 2014), db.top071795.

[11] Morrison L. E., Lefever M. R., Behman L. J., Leibold T., Roberts E. A., Horchner U. B., and Bauer D. R. Brightfield multiplex immunohistochemistry with multispectral imaging. *Laboratory Investigation* 100.8 (2020), pp. 1124–1136. DOI: https://doi.org/10.1038/s41374-020-0429-0. https://www.sciencedirect.com/science/article/pii/S0023683722003798.

[12] Wang G. and Fang N. Detecting and tracking nonfluorescent nanoparticle probes in live cells. en. *Methods Enzymol* 504 (2012), pp. 83–108.

[13] Dimopoulos S., Mayer C. E., Rudolf F., and Stelling J. Accurate cell segmentation in microscopy images using membrane patterns. *Bioinformatics* 30.18 (May 2014), pp. 2644–2651. DOI: 10.1093/bioinformatics/btu302. eprint: https://academic.oup.com/bioinformatics/article-pdf/30/18/2644/48929266/bioinformatics_30_18_2644.pdf. https://doi.org/10.1093/bioinformatics/btu302.

[14] He K., Gkioxari G., Dollár P., and Girshick R. Mask R-CNN. 2018. arXiv: 1703.06870 [cs.CV]. https://arxiv.org/abs/1703.06870.

[15] Reis D., Kupec J., Hong J., and Daoudi A. Real-Time Flying Object Detection with YOLOv8. 2024. arXiv: 2305.09972 [cs.CV]. https://arxiv.org/abs/2305.09972.

[16] Wang C.-Y., Yeh I.-H., and Liao H.-Y. M. YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. 2024. arXiv: 2402.13616 [cs.CV]. https://arxiv.org/abs/2402.13616.

[17] Carion N., Massa F., Synnaeve G., Usunier N., Kirillov A., and Zagoruyko S. End-to-End Object Detection with Transformers. 2020. arXiv: 2005.12872 [cs.CV]. https://arxiv.org/abs/2005.12872.

[18] Cheng B., Schwing A. G., and Kirillov A. Per-Pixel Classification is Not All You Need for Semantic Segmentation. 2021. arXiv: 2107.06278 [cs.CV]. https://arxiv.org/abs/2107.06278.

[19] Cheng B., Misra I., Schwing A. G., Kirillov A., and Girdhar R. Masked-attention Mask Transformer for Universal Image Segmentation. 2022. arXiv: 2112.01527 [cs.CV]. https://arxiv.org/abs/2112.01527.

[20] Li F., Zhang H., xu H., Liu S., Zhang L., Ni L. M., and Shum H.-Y. Mask DINO: Towards A Unified Transformer-based Framework for Object Detection and Segmentation. 2022. arXiv: 2206.02777 [cs.CV]. https://arxiv.org/abs/2206.02777.

[21] Fang Y., Yang S., Wang X., Li Y., Fang C., Shan Y., Feng B., and Liu W. Instances as Queries. 2021. arXiv: 2105.01928 [cs.CV]. https://arxiv.org/abs/2105.01928.

[22] He J., Li P., Geng Y., and Xie X. FastInst: A Simple Query-Based Model for Real-Time Instance Segmentation. 2023. arXiv: 2303.08594 [cs.CV]. https://arxiv.org/abs/2303.08594.

[23] Chen Q., Huang W., Liu X., Li J., and Xiong Z. PCTrans: Position-Guided Transformer with Query Contrast for Biological Instance Segmentation. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*. Oct. 2023, pp. 3903–3912.

[24] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L., and Polosukhin I. Attention Is All You Need. 2023. arXiv: 1706.03762 [cs.CL]. https://arxiv.org/abs/1706.03762.

[25] Long J., Shelhamer E., and Darrell T. Fully Convolutional Networks for Semantic Segmentation. 2015. arXiv: 1411.4038 [cs.CV]. https://arxiv.org/abs/1411.4038.

[26] Hatamizadeh A., Tang Y., Nath V., Yang D., Myronenko A., Landman B., Roth H., and Xu D. UNETR: Transformers for 3D Medical Image Segmentation. 2021. arXiv: 2103.10504 [eess.IV]. https://arxiv.org/abs/2103.10504.

[27] Cao H., Wang Y., Chen J., Jiang D., Zhang X., Tian Q., and Wang M. Swin-Unet: Unet-like Pure Transformer for Medical Image Segmentation. 2021. arXiv: 2105.05537 [eess.IV]. https://arxiv.org/abs/2105.05537.

[28] Gould S., Gao T., and Koller D. Region-based Segmentation and Object Detection. *Advances in Neural Information Processing Systems*. Ed. by Bengio Y., Schuurmans D., Lafferty J., Williams C., and Culotta A. Vol. 22. Curran Associates, Inc., 2009. https://proceedings.neurips.cc/paper_files/paper/2009/file/a7aeed74714116f3b292a982238f83d2-Paper.pdf.

[29] Ren S., He K., Girshick R., and Sun J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. 2016. arXiv: 1506.01497 [cs.CV]. https://arxiv.org/abs/1506.01497.

[30] Girshick R., Donahue J., Darrell T., and Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. 2014. arXiv: 1311.2524 [cs.CV]. https://arxiv.org/abs/1311.2524.

[31] Lin T.-Y., Dollár P., Girshick R., He K., Hariharan B., and Belongie S. Feature Pyramid Networks for Object Detection. 2017. arXiv: 1612.03144 [cs.CV]. https://arxiv.org/abs/1612.03144.

[32] Cheng T., Wang X., Chen S., Zhang W., Zhang Q., Huang C., Zhang Z., and Liu W. Sparse Instance Activation for Real-Time Instance Segmentation. 2022. arXiv: 2203.12827 [cs.CV]. https://arxiv.org/abs/2203.12827.

[33] Cai Z. and Vasconcelos N. Cascade R-CNN: High Quality Object Detection and Instance Segmentation. 2019. arXiv: 1906.09756 [cs.CV]. https://arxiv.org/abs/1906.09756.

[34] Kirillov A., Wu Y., He K., and Girshick R. PointRend: Image Segmentation as Rendering. 2020. arXiv: 1912.08193 [cs.CV]. https://arxiv.org/abs/1912.08193.

[35] Wang C.-Y., Liao H.-Y. M., Yeh I.-H., Wu Y.-H., Chen P.-Y., and Hsieh J.-W. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. 2019. arXiv: 1911.11929 [cs.CV]. https://arxiv.org/abs/1911.11929.

[36] Liu S., Qi L., Qin H., Shi J., and Jia J. Path Aggregation Network for Instance Segmentation. 2018. arXiv: 1803.01534 [cs.CV]. https://arxiv.org/abs/1803.01534.

[37] Bolya D., Zhou C., Xiao F., and Lee Y. J. YOLACT: Real-time Instance Segmentation. 2019. arXiv: 1904.02689 [cs.CV]. https://arxiv.org/abs/1904.02689.

[38] Follmann P. and König R. Oriented Boxes for Accurate Instance Segmentation. 2020. arXiv: 1911.07732 [cs.CV].

[39] Kirillov A., Levinkov E., Andres B., Savchynskyy B., and Rother C. InstanceCut: from Edges to Instances with MultiCut. 2016. arXiv: 1611.08272 [cs.CV].

[40] Schmidt U., Weigert M., Broaddus C., and Myers G. Cell Detection with Star-Convex Polygons. *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*. Springer International Publishing, 2018, pp. 265–273. DOI: 10.1007/978-3-030-00934-2_30. http://dx.doi.org/10.1007/978-3-030-00934-2_30.

[41] Bai M. and Urtasun R. Deep Watershed Transform for Instance Segmentation. 2017. arXiv: 1611.08303 [cs.CV]. https://arxiv.org/abs/1611.08303.

[42] Raza S. E. A., Cheung L., Shaban M., Graham S., Epstein D., Pelengaris S., Khan M., and Rajpoot N. M. Micro-Net: A unified model for segmentation of various objects in microscopy images. *Medical Image Analysis* 52 (Feb. 2019), pp. 160–173. DOI: 10.1016/j.media.2018.12.003. http://dx.doi.org/10.1016/j.media.2018.12.003.

[43] Stringer C., Wang T., Michaelos M., and Pachitariu M. Cellpose: a generalist algorithm for cellular segmentation. *Nature Methods* 18.1 (Jan. 2021), pp. 100–106.

[44] Graham S., Vu Q. D., Raza S. E. A., Azam A., Tsang Y. W., Kwak J. T., and Rajpoot N. HoVer-Net: Simultaneous Segmentation and Classification of Nuclei in Multi-Tissue Histology Images. 2019. arXiv: 1812.06499 [cs.CV]. https://arxiv.org/abs/1812.06499.

[45] Pachitariu M. and Stringer C. Cellpose 2.0: how to train your own model. *Nature Methods* 19.12 (Dec. 2022), pp. 1634–1641. DOI: 10.1038/s41592-022-01663-4. https://doi.org/10.1038/s41592-022-01663-4.

[46] He K., Zhang X., Ren S., and Sun J. Deep Residual Learning for Image Recognition. 2015. arXiv: 1512.03385 [cs.CV]. https://arxiv.org/abs/1512.03385.

[47] Stewart R. and Andriluka M. End-to-end people detection in crowded scenes. 2015. arXiv: 1506.04878 [cs.CV].

[48] Zhu X., Su W., Lu L., Li B., Wang X., and Dai J. Deformable DETR: Deformable Transformers for End-to-End Object Detection. 2021. arXiv: 2010.04159 [cs.CV]. https://arxiv.org/abs/2010.04159.

[49] Zhang H., Li F., Liu S., Zhang L., Su H., Zhu J., Ni L. M., and Shum H.-Y. DINO: DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection. 2022. arXiv: 2203.03605 [cs.CV]. https://arxiv.org/abs/2203.03605.

[50] Meng D., Chen X., Fan Z., Zeng G., Li H., Yuan Y., Sun L., and Wang J. Conditional DETR for Fast Training Convergence. 2023. arXiv: 2108.06152 [cs.CV]. https://arxiv.org/abs/2108.06152.

[51] Zhu X., Su W., Lu L., Li B., Wang X., and Dai J. Deformable DETR: Deformable Transformers for End-to-End Object Detection. 2021. arXiv: 2010.04159 [cs.CV]. https://arxiv.org/abs/2010.04159.

[52] Prangemeier T., Reich C., and Koeppl H. Attention-Based Transformers for Instance Segmentation of Cells in Microstructures. *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, Dec. 2020. DOI: 10.1109/bibm49941.2020.9313305. http://dx.doi.org/10.1109/BIBM49941.2020.9313305.

[53] Cutler K. J., Stringer C., Lo T. W., Rappez L., Stroustrup N., Brook Peterson S., Wiggins P. A., and Mougous J. D. Omnipose: a high-precision morphology-independent solution for bacterial cell segmentation. *Nature Methods* 19.11 (Nov. 2022), pp. 1438–1448. DOI: 10.1038/s41592-022-01639-4. https://doi.org/10.1038/s41592-022-01639-4.

[54] Kirillov A., Mintun E., Ravi N., Mao H., Rolland C., Gustafson L., Xiao T., Whitehead S., Berg A. C., Lo W.-Y., Dollár P., and Girshick R. Segment Anything. 2023. arXiv: 2304.02643 [cs.CV]. https://arxiv.org/abs/2304.02643.

[55] Dosovitskiy A., Beyer L., Kolesnikov A., Weissenborn D., Zhai X., Unterthiner T., Dehghani M., Minderer M., Heigold G., Gelly S., Uszkoreit J., and Houlsby N. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. 2021. arXiv: 2010.11929 [cs.CV]. https://arxiv.org/abs/2010.11929.

[56] Tancik M., Srinivasan P. P., Mildenhall B., Fridovich-Keil S., Raghavan N., Singhal U., Ramamoorthi R., Barron J. T., and Ng R. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. 2020. arXiv: 2006.10739 [cs.CV]. https://arxiv.org/abs/2006.10739.

[57] Edlund C., Jackson T. R., Khalid N., Bevan N., Dale T., Dengel A., Ahmed S., Trygg J., and Sjögren R. LIVECell—A large-scale dataset for label-free live cell segmentation. *Nature Methods* 18.9 (Sept. 2021), pp. 1038–1045. DOI: 10.1038/s41592-021-01249-6. https://doi.org/10.1038/s41592-021-01249-6.

[58] Schwendy M., Unger R. E., and Parekh S. H. EVICAN—a balanced dataset for algorithm development in cell and nucleus segmentation. *Bioinformatics* 36.12 (Apr. 2020),

pp. 3863–3870. DOI: 10.1093/bioinformatics/btaa225. eprint: https://academic.oup.com/bioinformatics/article-pdf/36/12/3863/50750214/bioinformatics_36_12_3863.pdf. https://doi.org/10.1093/bioinformatics/btaa225.

[59] Salem D., Li Y., Xi P., Phenix H., Cuperlovic-Culf M., and Kærn M. YeastNet: Deep-Learning-Enabled Accurate Segmentation of Budding Yeast Cells in Bright-Field Microscopy. *Applied Sciences* 11.6 (2021). DOI: 10.3390/app11062692. https://www.mdpi.com/2076-3417/11/6/2692.

[60] Raufeisen J., Xie K., Hörst F., Braunschweig T., Li J., Kleesiek J., Röhrig R., Egger J., Leibe B., Hölzle F., Hermans A., and Puladi B. Cyto R-CNN and CytoNuke Dataset: Towards reliable whole-cell segmentation in bright-field histological images. *Computer Methods and Programs in Biomedicine* 252 (July 2024), p. 108215. DOI: 10.1016/j.cmpb.2024.108215. http://dx.doi.org/10.1016/j.cmpb.2024.108215.

[61] Lu Z., Carneiro G., and Bradley A. P. An Improved Joint Optimization of Multiple Level Set Functions for the Segmentation of Overlapping Cervical Cells. *IEEE Transactions on Image Processing* 24.4 (2015), pp. 1261–1272. DOI: 10.1109/TIP.2015.2389619.

[62] Tkachenko M., Malyuk M., Holmanyuk A., and Liubimov N. Label Studio: Data labeling software. Open source software available from https://github.com/heartexlabs/label-studio. 2020-2022. https://github.com/heartexlabs/label-studio.

[63] Krizhevsky A., Sutskever I., and Hinton G. E. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*. Ed. by Pereira F., Burges C., Bottou L., and Weinberger K. Vol. 25. Curran Associates, Inc., 2012. https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[64] Ioffe S. and Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. 2015. arXiv: 1502.03167 [cs.LG]. https://arxiv.org/abs/1502.03167.

[65] Yi-de M., Qing L., and Zhi-bai Q. Automated image segmentation using improved PCNN model based on cross-entropy. *Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing, 2004.* 2004, pp. 743–746. DOI: 10.1109/ISIMP.2004.1434171.

[66] Chollet F. Xception: Deep Learning With Depthwise Separable Convolutions. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[67] Liu R., Lehman J., Molino P., Such F. P., Frank E., Sergeev A., and Yosinski J. An Intriguing Failing of Convolutional Neural Networks and the CoordConv Solution. 2018. arXiv: 1807.03247 [cs.CV]. https://arxiv.org/abs/1807.03247.

[68] Hu J., Shen L., Albanie S., Sun G., and Wu E. Squeeze-and-Excitation Networks. 2019. arXiv: 1709.01507 [cs.CV]. https://arxiv.org/abs/1709.01507.

[69] Chen J., Lu Y., Yu Q., Luo X., Adeli E., Wang Y., Lu L., Yuille A. L., and Zhou Y. TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation. 2021. arXiv: 2102.04306 [cs.CV]. https://arxiv.org/abs/2102.04306.

[70] Dong B., Zeng F., Wang T., Zhang X., and Wei Y. SOLQ: Segmenting Objects by Learning Queries. 2021. arXiv: 2106.02351 [cs.CV]. https://arxiv.org/abs/2106.02351.

[71] Raghu M., Unterthiner T., Kornblith S., Zhang C., and Dosovitskiy A. Do Vision Transformers See Like Convolutional Neural Networks? 2022. arXiv: 2108.08810 [cs.CV]. https://arxiv.org/abs/2108.08810.

[72] Khan S., Naseer M., Hayat M., Zamir S. W., Khan F. S., and Shah M. Transformers in Vision: A Survey. *ACM Computing Surveys* 54.10s (Jan. 2022), pp. 1–41. DOI: 10.1145/3505244. http://dx.doi.org/10.1145/3505244.

[73] Liu S., Li F., Zhang H., Yang X., Qi X., Su H., Zhu J., and Zhang L. DAB-DETR: Dynamic Anchor Boxes are Better Queries for DETR. 2022. arXiv: 2201.12329 [cs.CV]. https://arxiv.org/abs/2201.12329.

[74] Ba J. L., Kiros J. R., and Hinton G. E. Layer Normalization. 2016. arXiv: 1607.06450 [stat.ML]. https://arxiv.org/abs/1607.06450.

[75] Milletari F., Navab N., and Ahmadi S.-A. V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. 2016. arXiv: 1606.04797 [cs.CV]. https://arxiv.org/abs/1606.04797.

[76] Chen K., Wang J., Pang J., Cao Y., Xiong Y., Li X., Sun S., Feng W., Liu Z., Xu J., Zhang Z., Cheng D., Zhu C., Cheng T., Zhao Q., Li B., Lu X., Zhu R., Wu Y., Dai J., Wang J., Shi J., Ouyang W., Loy C. C., and Lin D. MMDetection: Open MMLab Detection Toolbox and Benchmark. 2019. arXiv: 1906.07155 [cs.CV]. https://arxiv.org/abs/1906.07155.

[77] Russakovsky O., Deng J., Su H., Krause J., Satheesh S., Ma S., Huang Z., Karpathy A., Khosla A., Bernstein M., Berg A. C., and Fei-Fei L. ImageNet Large Scale Visual Recognition Challenge. 2015. arXiv: 1409.0575 [cs.CV]. https://arxiv.org/abs/1409.0575.

[78] Liu Z., Lin Y., Cao Y., Hu H., Wei Y., Zhang Z., Lin S., and Guo B. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. 2021. arXiv: 2103.14030 [cs.CV]. https://arxiv.org/abs/2103.14030.

[79] Ruder S. An overview of gradient descent optimization algorithms. 2017. arXiv: 1609.04747 [cs.LG]. https://arxiv.org/abs/1609.04747.

[80] Wu Y., Kirillov A., Massa F., Lo W.-Y., and Girshick R. Detectron2. https://github.com/facebookresearch/detectron2. 2019.

[81] Jocher G., Chaurasia A., Laughing-Q, Fang J., Wong A., Kwon Y., and Luo T. Ultralytics. https://github.com/ultralytics/ultralytics. Accessed: 2025-05-03. 2023.

[82] Loshchilov I. and Hutter F. SGDR: Stochastic Gradient Descent with Warm Restarts. 2017. arXiv: 1608.03983 [cs.LG]. https://arxiv.org/abs/1608.03983.

[83] OpenAI. ChatGPT. https://chat.openai.com. 2024.

[84] Grammarly Inc. Grammarly Writing Assistant. https://www.grammarly.com. 2024.

[85] Overleaf. Overleaf - The Online LaTeX Editor. https://www.overleaf.com. 2024.

[86] Paszke A., Gross S., Massa F., Lerer A., Bradbury J., Chanan G., Killeen T., Lin Z., Gimelshein N., Antiga L., Desmaison A., Köpf A., Yang E., DeVito Z., Raison M., Tejani A., Chilamkurthy S., Steiner B., Fang L., Bai J., and Chintala S. PyTorch: An Imperative Style, High-Performance Deep Learning Library. 2019. arXiv: 1912.01703 [cs.LG]. https://arxiv.org/abs/1912.01703.

[87] Harris C. R., Millman K. J., Walt S. J. van der, Gommers R., Virtanen P., Cournapeau D., Wieser E., Taylor J., Berg S., Smith N. J., Kern R., Picus M., Hoyer S., Kerkwijk M. H.

van, Brett M., Haldane A., Río J. F. del, Wiebe M., Peterson P., Gérard-Marchant P., Sheppard K., Reddy T., Weckesser W., Abbasi H., Gohlke C., and Oliphant T. E. Array programming with NumPy. *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. https://doi.org/10.1038/s41586-020-2649-2.

[88] Virtanen P., Gommers R., Oliphant T. E., Haberland M., Reddy T., Cournapeau D., Burovski E., Peterson P., Weckesser W., Bright J., van der Walt S. J., Brett M., Wilson J., Millman K. J., Mayorov N., Nelson A. R. J., Jones E., Kern R., Larson E., Carey C. J., Polat İ., Feng Y., Moore E. W., VanderPlas J., Laxalde D., Perktold J., Cimrman R., Henriksen I., Quintero E. A., Harris C. R., Archibald A. M., Ribeiro A. H., Pedregosa F., van Mulbregt P., and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[89] Bradski G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* (2000).

[90] Hunter J. D. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.1109/MCSE.2007.55.

[91] Yadan O. Hydra - A framework for elegantly configuring complex applications. Github. 2019. https://github.com/facebookresearch/hydra.

[92] Biewald L. Experiment Tracking with Weights and Biases. Software available from wandb.com. 2020. https://www.wandb.com/.

[93] Loshchilov I. and Hutter F. Decoupled Weight Decay Regularization. 2019. arXiv: 1711.05101 [cs.LG]. https://arxiv.org/abs/1711.05101.

[94] Buslaev A., Iglovikov V. I., Khvedchenya E., Parinov A., Druzhinin M., and Kalinin A. A. Albumentations: Fast and Flexible Image Augmentations. *Information* 11.2 (2020). DOI: 10.3390/info11020125. https://www.mdpi.com/2078-2489/11/2/125.

[95] Ghiasi G., Cui Y., Srinivas A., Qian R., Lin T.-Y., Cubuk E. D., Le Q. V., and Zoph B. Simple Copy-Paste is a Strong Data Augmentation Method for Instance Segmentation. 2021. arXiv: 2012.07177 [cs.CV]. https://arxiv.org/abs/2012.07177.

[96] Lin T., Maire M., Belongie S. J., Bourdev L. D., Girshick R. B., Hays J., Perona P., Ramanan D., Doll'a r P., and Zitnick C. L. Microsoft COCO: Common Objects in Context. *CoRR* abs/1405.0312 (2014). arXiv: 1405.0312. http://arxiv.org/abs/1405.0312.

[97] Zong Z., Song G., and Liu Y. DETRs with Collaborative Hybrid Assignments Training. 2023. arXiv: 2211.12860 [cs.CV]. https://arxiv.org/abs/2211.12860.

[98] Ravi N., Gabeur V., Hu Y.-T., Hu R., Ryali C., Ma T., Khedr H., Rädle R., Rolland C., Gustafson L., Mintun E., Pan J., Alwala K. V., Carion N., Wu C.-Y., Girshick R., Dollár P., and Feichtenhofer C. SAM 2: Segment Anything in Images and Videos. 2024. arXiv: 2408.00714 [cs.CV]. https://arxiv.org/abs/2408.00714.

[99] Caron M., Touvron H., Misra I., Jégou H., Mairal J., Bojanowski P., and Joulin A. Emerging Properties in Self-Supervised Vision Transformers. 2021. arXiv: 2104.14294 [cs.CV]. https://arxiv.org/abs/2104.14294.

[100] Oquab M., Darcet T., Moutakanni T., Vo H., Szafraniec M., Khalidov V., Fernandez P., Haziza D., Massa F., El-Nouby A., Assran M., Ballas N., Galuba W., Howes R., Huang P.-Y., Li S.-W., Misra I., Rabbat M., Sharma V., Synnaeve G., Xu H., Jegou H., Mairal J., Labatut P., Joulin A., and Bojanowski P. DINOv2: Learning Robust Visual

Features without Supervision. 2024. arXiv: 2304.07193 [cs.CV]. https://arxiv.org/abs/2304.07193.

Appendices

One of our key contributions in the paper is the **2025 Revvity Full Cell Segmentation Dataset (Revvity-25)**. This dataset is designed specifically for cell instance segmentation in brightfield images, capturing diverse cellular morphologies across seven distinct cell lines. Here we provide additional samples from the dataset.

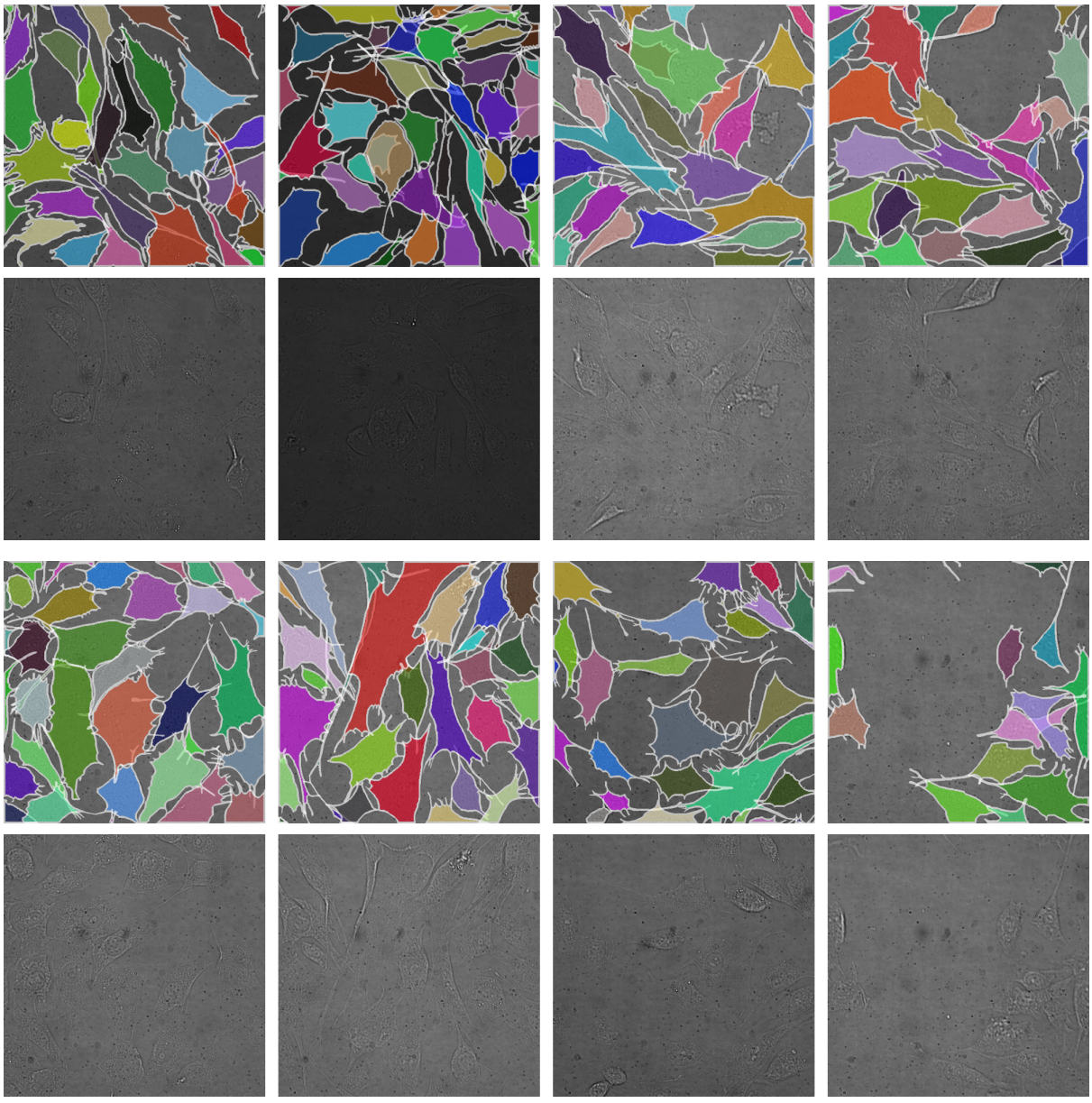


Figure 21. **2025 Revvity Full Cell Segmentation Dataset samples.**

License

Non-exclusive licence to reproduce the thesis and make the thesis public

I, **Yaroslav Prytula**,

1. grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the digital archives of the University of Tartu until the expiry of the term of copyright, my thesis

   IAUNet: Instance-Aware U-Net,

   supervised by Dmytro Fishman;

2. grant the University of Tartu a permit to make the thesis specified in point 1 available to the public via the web environment of the University of Tartu, including via the digital archives, under the Creative Commons licence CC BY NC ND 4.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright;

3. am aware of the fact that the author retains the rights specified in points 1 and 2;

4. confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Yaroslav Prytula
15/05/2025