DINO-Nucleus: Multi-Level Self-Supervised Features for Nucleus Instance Segmentation

Mohammad Sarhangzadeh[1(✉)], Elyar Esmaeilzadeh[1], and Doruk Öner[1]

Department of Computer Engineering, Bilkent University, Ankara, Turkey
{m.sarhangzadeh, elyar, doruk.oner}@bilkent.edu.tr

Abstract. Nucleus instance segmentation and counting are essential for quantitative microscopy, enabling reliable analysis of cell populations, tissue organization, and disease-related changes. However, accurately separating touching nuclei remains difficult, and many existing approaches rely on computationally expensive models or sensitive post-processing procedures. We present DINO-Nucleus, an efficient framework that leverages multi-level self-supervised DINOv2 representations for nucleus instance segmentation and counting. The model jointly predicts nucleus regions, pixel-wise instance flows, and explicit nucleus centers, while a consistency objective encourages the predicted flows to converge toward their corresponding centers. Multi-level transformer features provide complementary spatial and semantic information, and the pretrained backbone is adapted using lightweight LoRA modules to reduce fine-tuning cost. Experimental results show thatDINO-Nucleus achieves strong segmentation and counting performance while remaining computationally efficient. These findings demonstrate that parameter-efficient adaptation of multi-level self-supervised features can provide an effective solution for nucleus instance segmentation in microscopy images.

Keywords: Microscopy image analysis · Nucleus instance segmentation · Self-supervised features · Parameter-efficient fine-tuning

1 Introduction

Microscopy imaging plays a central role in studying cellular organization and tissue structure. Among the many cellular components visible in microscopy images, nuclei are particularly informative because their number, distribution, size, and morphology provide valuable clues about cell populations and biological conditions [2]. Reliable nucleus analysis is therefore important for tasks such as cell-density estimation, tissue characterization, disease assessment, and treatment-response monitoring. Automating this process requires not only detecting nucleus pixels, but also separating each nucleus as an individual instance so that accurate morphological measurements and cell counts can be obtained. Despite its importance, nucleus instance segmentation remains challenging because nuclei can vary considerably in shape, size, intensity, and appearance across images [10]. The problem becomes more difficult in densely populated regions,

where neighboring nuclei may touch or overlap and the boundary between them may be faint or entirely unclear [4,19]. In such cases, semantic segmentation can correctly identify the overall nucleus region while still merging several nuclei into a single object, so accurate counting depends on preserving instance-level structure rather than only distinguishing foreground from background.

Existing methods have substantially improved nucleus segmentation, yet reliable instance separation remains difficult in crowded images [4,8]. Modern convolutional and transformer-based models can produce accurate foreground predictions, but this does not necessarily translate into correct instance boundaries or counts when adjacent nuclei share similar appearance and weak separation cues. As a result, many pipelines still depend on carefully tuned post-processing or task-specific reconstruction rules, whose performance may vary with nucleus size, density, and image characteristics [21,19]. Pretrained vision transformers provide a promising alternative through transferable visual representations [13], but adapting them for dense prediction can require considerable computation and careful optimization, particularly when annotated microscopy data are limited [17]. Moreover, many fine-tuned architectures use only the final transformer output, even though earlier and intermediate layers retain complementary spatial information that may be important for resolving small structures and separating closely packed nuclei [1,17].

To address these limitations, we introduce **DINO-Nucleus**, a nucleus instance-segmentation and counting framework that makes use of the representations learned by DINOv2 [13]. Instead of relying only on the final transformer output, the method combines features from several depths of the backbone so that fine spatial cues are preserved alongside stronger semantic information. The pretrained encoder is fine-tuned through lightweight LoRA modules [7], reducing the computational cost of adaptation while retaining the benefits of large-scale visual pretraining. The instance-separation strategy is designed around the idea that every pixel belonging to a nucleus should point toward a common center [19,5]. The model therefore learns a pixel-wise flow field together with an explicit prediction of nucleus centers. These two outputs are not treated independently: a consistency objective encourages the predicted flows to lead toward the corresponding center regions. During inference, pixels that follow the same flow pattern are grouped into a single instance, allowing touching nuclei to be separated directly from the learned geometry without watershed-based reconstruction.

We evaluate DINO-Nucleus on the BBBC038 2018 dataset [2]. By combining multi-level DINOv2 features, parameter-efficient LoRA fine-tuning, and center-consistent flow learning, the proposed method achieves strong nucleus instance-segmentation and counting performance while requiring only a small fraction of the backbone parameters to be updated, remaining computationally efficient. Together, these components provide a unified framework in which transferable representations, efficient adaptation, and instance-aware geometric learning are optimized jointly. These results highlight the value of adapting pretrained visual representations in a lightweight manner while explicitly preserving instance-level structure.

2 Related Work

2.1 Nucleus Instance Segmentation

Early deep-learning approaches to nucleus segmentation were largely based on U-Net-style architectures that predict foreground masks and recover individual nuclei through connected components, contour processing, distance transforms, or watershed [15,10,20]. Within this family, methods such as TC-UNet++ [21] improve semantic segmentation and instance separation through stronger feature extraction and watershed-based reconstruction. Proposal-based methods such as Mask R-CNN [6] take a different approach by detecting candidate objects and predicting a separate mask for each proposal, although densely packed and small nuclei can remain difficult to localize reliably. Nucleus-specific methods have also introduced geometric representations that encode instance structure more directly [5]. StarDist [16] represents each nucleus using a center probability and radial distances that form a star-convex polygon, while HoVer-Net predicts horizontal and vertical center-relative maps to separate touching nuclei [4]. CellViT combines pretrained transformer encoders with a HoVer-style decoder, demonstrating how stronger visual representations can be integrated with vector-based nucleus reconstruction [8]. Flow-based methods provide another alternative by assigning each foreground pixel a direction toward a shared convergence region. Cellpose popularized this strategy for biological instance segmentation and reconstructs instances by grouping pixels according to their predicted trajectories [19]. However, the nucleus center is generally represented only implicitly through the learned flow field [17].

2.2 DINO and Parameter-Efficient Fine-Tuning

Pretrained vision transformers provide transferable representations that are particularly useful when task-specific annotations are limited [12,17,11]. Self-supervised models such as DINO [3] and DINOv2 [13] learn without manual class labels and capture meaningful object-level structure in their attention and feature representations. Features from different transformer depths provide complementary information: intermediate layers preserve finer spatial organization, whereas deeper layers encode more abstract semantic content [14].

Adapting these large backbones nevertheless involves a trade-off between accuracy and efficiency. Full fine-tuning requires substantial memory and computation, while freezing the encoder can limit adaptation to microscopy images. Parameter-efficient fine-tuning methods address this issue by updating only a small subset of parameters. LoRA, learns low-rank updates to selected transformer projections while leaving the pretrained weights unchanged [7].

Building on both lines of work,DINO-Nucleus combines multi-level DINOv2 representations with LoRA-based adaptation for efficient feature transfer. It further connects these representations to flow-based nucleus reconstruction by jointly predicting pixel-wise flows and explicit centers and enforcing consistency between them. In this way, the framework brings together efficient pretrained feature adaptation and center-guided instance separation.

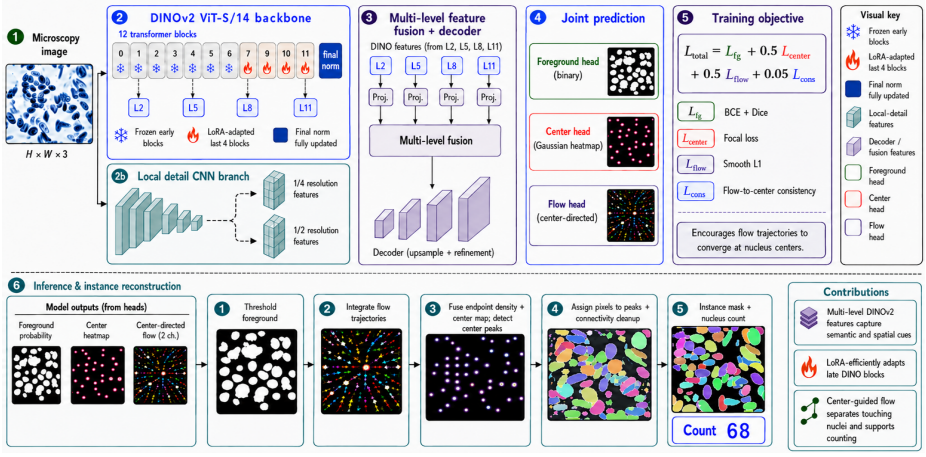


Fig. 1. Overview of DINO-Nucleus, showing multi-level DINOv2 feature fusion, joint foreground, center, flow prediction, and center-guided instance reconstruction.

3 Method

Given an RGB microscopy image $\mathbf{X} \in \mathbb{R}^{3\times H\times W}$, the objective is to predict an instance map $\hat{\mathbf{M}} \in \{0, 1, \ldots, \hat{N}\}^{H\times W}$, where zero denotes background and each positive label identifies one nucleus. The final nucleus count $\hat{N}$ is obtained from the number of unique non-background labels in $\hat{\mathbf{M}}$.

As shown in Fig. 1, DINO-Nucleus does not predict the instance map directly. Instead, the network produces a foreground probability map $\hat{\mathbf{Y}}_{\text{fg}} \in [0, 1]^{H\times W}$, a center heatmap $\hat{\mathbf{Y}}_{\text{c}} \in [0, 1]^{H\times W}$, and a two-channel center-directed flow field $\hat{\mathbf{Y}}_{\text{f}} \in \mathbb{R}^{2\times H\times W}$. The foreground map identifies nucleus pixels, the center heatmap provides explicit instance locations, and the flow field directs each foreground pixel toward the center of its corresponding nucleus.

The input image is encoded by a pretrained DINOv2 backbone adapted with LoRA, while features from multiple transformer depths are retained. A lightweight CNN branch also extracts higher-resolution local information from the input. The projected transformer and CNN features are fused through a shared decoder and passed to separate foreground, center, and flow heads. Training combines foreground segmentation, center localization, and flow regression. During inference, the three outputs are jointly used to integrate pixel trajectories, detect nucleus centers, assign foreground pixels to instances, and obtain the final instance map and count.

3.1 Encoding and Feature Fusion

Given an input image $\mathbf{X}$, we use a pretrained DINOv2 ViT-S/14 encoder and retain features from different stages of the transformer to capture information from its early, intermediate, and final layers. Among the layer combinations considered

during model development, layers 2, 5, 8, and 11 provided the best validation performance and were therefore selected. For each layer l, the feature map is written as $\mathbf{F}_l \in \mathbb{R}^{C \times H_p \times W_p}$, where H_p and W_p denote the patch-grid dimensions. Although these features share the same spatial resolution, they represent different levels of abstraction: earlier layers retain more local structure, while deeper layers provide stronger semantic context. Each feature is projected into a common channel space using a layer-specific 1×1 convolution, $\mathbf{P}_l = \phi_l(\mathbf{F}_l)$, before the projected features are combined into a multi-level representation.

To adapt the pretrained encoder efficiently, all early transformer blocks remain frozen, while the final four blocks are adapted using LoRA modules. The original weights of these blocks remain fixed, and only the low-rank updates are optimized. The final normalization layer is fully trainable to further align the adapted representations with the microscopy task. Since the DINOv2 features remain on the patch grid, a lightweight CNN branch also extracts higher-resolution local features $\mathbf{L}_{1/4}$ and $\mathbf{L}_{1/2}$ from the input image. The fused transformer representation is then progressively upsampled and combined with these local features through the decoder, producing a shared full-resolution representation $\mathbf{Z}$ for the foreground, center, and flow prediction heads.

3.2 Center-Flow Learning and Instance Reconstruction

A foreground map identifies nucleus regions but cannot reliably separate touching nuclei once they form a connected component. We therefore complement it with an explicit center heatmap and a pixel-wise center-directed flow field. Together, $\hat{\mathbf{Y}}_{\mathrm{fg}}$, $\hat{\mathbf{Y}}_{\mathrm{c}}$, and $\hat{\mathbf{Y}}_{\mathrm{f}}$ describe nucleus extent, instance locations, and pixel-to-instance associations.

The supervision targets are generated from the ground-truth instance masks. The foreground target $\mathbf{Y}_{\mathrm{fg}}$ is obtained by merging all nucleus instances into a binary mask. For each nucleus, its center c_k is selected as the point with the largest Euclidean distance from the boundary, ensuring that it lies inside the instance even for irregular shapes. To provide denser and more stable center supervision, a Gaussian response is placed around each center:

$$\mathbf{Y}_{\mathrm{c}}(p) = \max_k \exp\left(-\frac{\|p - c_k\|_2^2}{2\sigma_k^2}\right), \tag{1}$$

where σ_k controls the spatial extent of the response for nucleus k. For each pixel p belonging to nucleus k, the flow target is defined as a normalized vector pointing toward its center:

$$\mathbf{Y}_{\mathrm{f}}(p) = \frac{c_k - p}{\|c_k - p\|_2 + \epsilon}. \tag{2}$$

Background flow is set to zero. This representation allows pixels from the same nucleus to converge toward a common location, while pixels from touching nuclei are directed toward different centers. The foreground branch is trained using binary cross-entropy and Dice loss, such that $\mathcal{L}_{\mathrm{fg}} = \mathcal{L}_{\mathrm{BCE}} + \mathcal{L}_{\mathrm{Dice}}$. The

center branch uses Gaussian focal loss, $\mathcal{L}_{\mathrm{c}} = \mathcal{L}_{\mathrm{GF}}(\hat{\mathbf{Y}}_{\mathrm{c}}, \mathbf{Y}_{\mathrm{c}})$. Flow regression is restricted to foreground pixels:

$$\mathcal{L}_{\mathrm{f}} = \frac{\sum_p \mathbf{Y}_{\mathrm{fg}}(p) \sum_{d\in\{x,y\}} \mathrm{SmoothL1}\left(\hat{\mathbf{Y}}_{\mathrm{f}}^{d}(p), \mathbf{Y}_{\mathrm{f}}^{d}(p)\right)}{\sum_p \mathbf{Y}_{\mathrm{fg}}(p) + \epsilon}, \tag{3}$$

preventing the much larger background region from dominating the objective. Local flow regression alone does not ensure that the predicted trajectories terminate at valid nucleus centers. We therefore introduce endpoint consistency. Starting from a foreground pixel p, the predicted flow field is differentiably integrated over T steps:

$$q_p^0 = p, \qquad q_p^{t+1} = q_p^t + s\,\rho(q_p^t)\frac{\hat{\mathbf{Y}}_{\mathrm{f}}(q_p^t)}{\|\hat{\mathbf{Y}}_{\mathrm{f}}(q_p^t)\|_2 + \epsilon}, \tag{4}$$

where s is the integration step size, $\rho(q_p^t)$ denotes foreground support sampled at current trajectory position, and q_p^T is the resulting endpoint. The center reference combines the ground-truth heatmap with detached center prediction:

$$\mathbf{Y}_{\mathrm{ref}} = \eta\mathbf{Y}_{\mathrm{c}} + (1-\eta)\,\mathrm{sg}\left(\hat{\mathbf{Y}}_{\mathrm{c}}\right), \tag{5}$$

where $\mathrm{sg}(\cdot)$ prevents gradients from this term from updating the center branch. Endpoint consistency is then defined as

$$\mathcal{L}_{\mathrm{end}} = \frac{\sum_p \mathbf{Y}_{\mathrm{fg}}(p)\left[-\log\left(\mathbf{Y}_{\mathrm{ref}}(q_p^T) + \epsilon\right)\right]}{\sum_p \mathbf{Y}_{\mathrm{fg}}(p) + \epsilon}. \tag{6}$$

While $\mathcal{L}_{\mathrm{f}}$ supervises local flow directions, $\mathcal{L}_{\mathrm{end}}$ encourages foreground trajectories to terminate within valid nucleus-center. The complete objective is

$$\mathcal{L} = \lambda_{\mathrm{fg}}\mathcal{L}_{\mathrm{fg}} + \lambda_{\mathrm{c}}\mathcal{L}_{\mathrm{c}} + \lambda_{\mathrm{f}}\mathcal{L}_{\mathrm{f}} + \lambda_{\mathrm{end}}r(e)\mathcal{L}_{\mathrm{end}}, \tag{7}$$

where the λ coefficients balance the loss terms and $r(e)$ gradually introduces endpoint consistency during training. Training begins with a decoder-only warm-up, after which LoRA parameters and final normalization layer are optimized.

During inference, the foreground map is thresholded to identify candidate nucleus pixels, whose trajectories are then integrated through the predicted flow field. Their endpoints are accumulated into an endpoint-density map, where high-density regions indicate likely nucleus centers. This density is combined with the predicted center heatmap, and local peaks in the fused response are selected as instance centers.

Each foreground pixel is assigned to a detected center according to its trajectory endpoint. Disconnected regions sharing the same provisional label are separated, and small isolated fragments are removed. The remaining connected regions form the final instance map $\hat{\mathbf{M}}$, from which nucleus count $\hat{N}$ is obtained.

4 Experiments

We describe the dataset, evaluation protocol, metrics, and implementation settings, followed by qualitative analysis of the learned representations, center heatmaps, and flow fields. We then compare DINO-Nucleus with TC-UNet++, Cellpose3, StarDist, and CellViT [21,18,16,8] in terms of segmentation, counting, and parameter efficiency. Finally, we compare it with a basic DINOv2 watershed baseline [13] to assess the contributions of center-flow consistency.

Dataset and preprocessing. We use BBBC038, containing 670 microscopy images with instance-level nucleus annotations [2]. Using a fixed seed, images are stratified by nucleus count and mean intensity into 468 training, 67 validation, and 135 test samples. The validation set is used for model selection and post-processing calibration, while the test set is reserved for final evaluation. Training images are aspect-ratio resized and zero-padded to 448×448. Native-resolution test images are processed using sliding-window inference.

Evaluation metrics. We evaluate foreground segmentation using Dice and IoU, and instance segmentation using object-level F1, panoptic quality (PQ), and aggregated Jaccard index (AJI) [9,10]. Instances are matched at IoU > 0.45. Counting is evaluated using mean absolute error (MAE) and Pearson correlation.

Implementation details. DINO-Nuclei uses a pretrained DINOv2 ViT-S/14 encoder [13] and fuses features from blocks 2, 5, 8, and 11 through a 128-channel decoder. Rank-8 LoRA adapters with scaling 16 and dropout 0.05 are applied to the final four blocks [7]. Foreground, center, flow, and endpoint-consistency losses are weighted by 1.0, 0.5, 0.5, and 0.05, respectively, with consistency enabled after five warm-up epochs. During inference, foreground pixels follow the predicted flow and are grouped by their converged endpoints [19]. Reconstruction and post-processing parameters are calibrated on the validation set and fixed for testing. All baselines use the same splits, augmentations, epoch budget, and evaluation protocol while retaining architecture-specific training settings. Experiments are implemented in PyTorch and run on aone NVIDIA A100 GPU.

Feature analysis. Figure 2 examines the pretrained DINOv2 features used by DINO-Nuclei. The PCA maps show that nucleus regions form clear patterns that differ from the surrounding background. This is also visible in the prototype-score maps, where nucleus locations appear nucleus-like across all four layers. The t-SNE plots provide another view of the feature space: background patches often form separate groups, while nucleus and boundary patches remain more closely related. This is expected because boundary patches contain both nucleus and background pixels. Taken together, the PCA maps show that DINOv2 preserves the spatial structure of nucleus regions, while the t-SNE plots show that nucleus-dominant patches occupy feature-space regions that are largely separate from background patches, with boundary patches naturally appearing between them.

Comparison with existing methods. Table 1 compares our method with existing nucleus segmentation approaches and the DINO baseline on BBBC038. Our method achieves competitive foreground segmentation, despite adapting

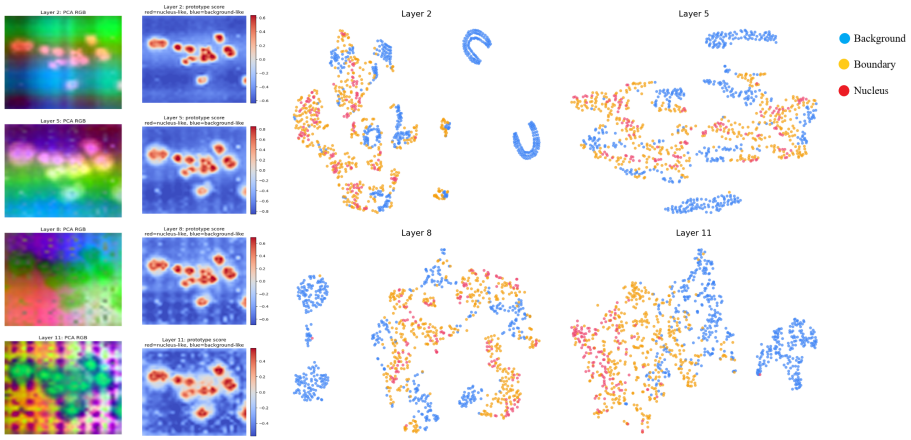


Fig. 2. Pretrained DINOv2 features from layers 2, 5, 8, and 11. PCA maps visualize spatial token organization, prototype scores highlight nucleus (red) and background (blue) regions. T-SNE plots show background, boundary, and nucleus-dominant patches.

Table 1. Comparison of nucleus segmentation and counting performance.

Method	Foreground		Instance			Counting	
	Dice ↑	IoU ↑	Obj.-F1 ↑	PQ ↑	AJI ↑	MAE ↓	Pearson ↑
TC-UNet++	**92.46**	**88.51**	83.64	72.33	77.59	9.19	89.70
Cellpose3	88.71	81.34	88.52	76.69	83.13	5.97	93.39
StarDist	85.94	81.07	83.06	70.05	75.93	9.72	87.62
CellViT	90.22	84.35	<u>90.75</u>	<u>77.40</u>	<u>85.35</u>	<u>5.63</u>	<u>94.58</u>
DINOv2	<u>90.98</u>	<u>86.16</u>	83.51	74.88	79.28	7.47	89.20
Ours	90.45	85.30	**92.38**	**79.33**	**88.97**	**3.11**	**98.75**

only a small part of the pretrained encoder through LoRA. Together with the feature analysis in Fig. 2, this indicates that pretrained DINOv2 features already provide strong information for locating nucleus regions. The main advantage of our method appears in instance segmentation and counting, where it consistently outperforms all competing methods. Compared with the DINOv2 baseline using watershed reconstruction, the foreground results remain similar, as the center-flow formulation is not mainly designed to improve foreground detection. In contrast, the large improvements in instance segmentation and counting show that the predicted centers, center-directed flows, and their consistency are effective for separating neighboring nuclei and reconstructing individual instances.

We next compare the model sizes and number of trainable parameters in Table 2. Although DINO-Nuclei uses a transformer encoder, most pretrained parameters remain frozen, and only the decoder, LoRA adapters, and final normalization layer are updated. This allows the model to achieve strong performance with limited backbone adaptation.

Table 2. Parameter comparison between the evaluated methods

Method	Total (M) ↓	Trainable (M) ↓	Trainable (%) ↓
TC-UNet++	9.22	9.22	100.00
Cellpose3	<u>6.60</u>	6.60	100.00
StarDist	**1.99**	**1.99**	100.00
CellViT	39.95	39.95	100.00
DINOv2	24.28	9.32	<u>38.40</u>
Ours	24.35	<u>4.07</u>	**16.72**

5 Conclusion

In this paper, we introduced DINO-Nucleus, a parameter-efficient framework for nucleus instance segmentation and counting. By leveraging multi-level DINOv2 features adapted via lightweight LoRA modules, the model effectively combines fine spatial details with robust semantic context. Our approach jointly predicts foreground regions, explicit nucleus centers, and pixel-wise flows, utilizing a consistency objective that directs pixel trajectories toward valid centers. This center-guided formulation accurately separates touching nuclei without relying on sensitive watershed post-processing. Experiments on the BBBC038 dataset demonstrate that DINO-Nucleus significantly outperforms existing methods in both instance segmentation and counting accuracy. Ultimately, this highlights the strong potential of combining efficiently adapted self-supervised representations with geometric learning for reliable quantitative microscopy.

References

1. Archit, A., Freckmann, L., Nair, S., Khalid, N., Hilt, P., Rajashekar, V., Freitag, M., Teuber, C., Spitzner, M., Tapia Contreras, C., Buckley, G., von Haaren, S., Gupta, S., Grade, M., Wirth, M., Schneider, G., Dengel, A., Ahmed, S., Pape, C.: Segment anything for microscopy. Nature Methods **22**(3), 579–591 (2025)
2. Caicedo, J.C., Goodman, A., Karhohs, K.W., Cimini, B.A., Ackerman, J., Haghighi, M., Heng, C., Becker, T., Doan, M., McQuin, C., Rohban, M., Singh, S., Carpenter, A.E.: Nucleus segmentation across imaging experiments: the 2018 data science bowl. Nature Methods **16**(12), 1247–1253 (2019)
3. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9650–9660 (2021)
4. Graham, S., Vu, Q.D., Raza, S.E.A., Azam, A., Tsang, Y.W., Kwak, J.T., Rajpoot, N.: Hover-net: Simultaneous segmentation and classification of nuclei in multi-tissue histology images. Medical image analysis **58**, 101563 (2019)
5. He, H., Huang, Z., Ding, Y., Song, G., Wang, L., Ren, Q., Wei, P., Gao, Z., Chen, J.: CDNet: Centripetal direction network for nuclear instance segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4026–4035 (2021)

6. He, K., Gkioxari, G., Dollar, P., Girshick, R.: Mask R-CNN . vol. 42, pp. 386–397. IEEE Computer Society, Los Alamitos, CA, USA (Feb 2020)
7. Hu, E.J., yelong shen, Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: LoRA: Low-rank adaptation of large language models. In: International Conference on Learning Representations (2022)
8. Hörst, F., Rempe, M., Heine, L., Seibold, C., Keyl, J., Baldini, G., Ugurel, S., Siveke, J., Grünwald, B., Egger, J., Kleesiek, J.: Cellvit: Vision transformers for precise cell segmentation and classification. Medical Image Analysis **94**, 103143 (2024)
9. Kirillov, A., He, K., Girshick, R., Rother, C., Dollár, P.: Panoptic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9404–9413 (2019)
10. Kumar, N., Verma, R., Sharma, S., Bhargava, S., Vahadane, A., Sethi, A.: A dataset and a technique for generalized nuclear segmentation for computational pathology. IEEE Transactions on Medical Imaging **36**(7), 1550–1560 (2017)
11. Lavaee, A., Jain, A., Scanavachi Moreira Campos, G., Costa-Filho, J.I., Ingemansson, A., Kirchhausen, T.: Spatialdino: A self-supervised 3d vision transformer that enables segmentation and tracking in crowded cellular environments. bioRxiv (2026)
12. Moutakanni, T., Couprie, C., Yi, S., Doron, M., Chen, Z.S., Moshkov, N., Gardes, E., Caron, M., Touvron, H., Joulin, A., Bojanowski, P., Pernice, W.M., Caicedo, J.C.: Cell-dino: Self-supervised image-based embeddings for cell fluorescent microscopy. PLOS Computational Biology **21**(12), 1–23 (12 2026)
13. Oquab, M., Darcet, T., Moutakanni, T., Vo, H.V., 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., Bojanowski, P.: DINOv2: Learning robust visual features without supervision (2024)
14. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12179–12188 (2021)
15. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. pp. 234–241. Springer International Publishing (2015)
16. Schmidt, U., Weigert, M., Broaddus, C., Myers, G.: Cell detection with star-convex polygons. In: Frangi, A.F., Schnabel, J.A., Davatzikos, C., Alberola-López, C., Fichtinger, G. (eds.) Medical Image Computing and Computer Assisted Intervention – MICCAI 2018. pp. 265–273. Springer International Publishing, Cham (2018)
17. Stillwagon, K., VandeLoo, A.D., Magondu, B., Forest, C.R.: Self-supervised pretraining of cell segmentation models (2026)
18. Stringer, C., Pachitariu, M.: Cellpose3: one-click image restoration for improved cellular segmentation. Nature Methods **22**(3), 592–599 (2025)
19. Stringer, C., Wang, T., Michaelos, M., Pachitariu, M.: Cellpose: a generalist algorithm for cellular segmentation. Nature Methods **18**(1), 100–106 (2021)
20. Wazir, S., Kim, D.: Rethinking the nested u-net approach: Enhancing biomarker segmentation with attention mechanisms and multiscale feature fusion. In: Proceedings of 2024 International Conference on Medical Imaging and Computer-Aided Diagnosis, MICAD 2024. pp. 175–186 (2025)
21. Zheng, K., Pan, J., Jia, Z., Xiao, S., Tao, W., Zhang, D., Li, Q., Pan, L.: A method of nucleus image segmentation and counting based on tc-unet++ and distance watershed. Medical Engineering Physics **133**, 104244 (2024)