**Course**: BINP37

**Credits**: 15 (10 weeks)

**Project Name**: Development of a 2D nuclei segmentation model for the detection and visualisation of poly-aneuploid cancer cells in pre-treatment breast cancer tissue images

**Student**: Elijah Ugoh

**Email**: ugohelijah@gmail.com

**Supervisor(s)**: Emma Hammerlund, Tissue Development and Evolution (TiDE) Group, Lund University Cancer Centre, BMC B11, Lund University, Sweden.

**Email**: [emma.hammarlund@med.lu.se](mailto:emma.hammarlund@med.lu.se)

Arthur Boffelli Castro, Tissue Development and Evolution (TiDE), BMC B11, Lund University

**Email**: [arthur.boffelli\_castro@med.lu.se](mailto:arthur.boffelli_castro@med.lu.se)

## Abstract

# Introduction

Breast cancer constitutes one of the major global health concerns today, and now accounts for the most commonly diagnosed cancer in the world (Wilkinson and Gathani, 2022). After decades of research, major advances have been made in breast cancer treatment. Wang and Wu (2023) concluded that surgery-based local and systemic treatments remain the standard care approach for early breast cancer. Chemotherapy-based systemic treatments is typically used for metastatic breast cancer while surgery is only used for early invasive breast cancer in selected patients. Most new cases will have a 10-year survival rate of approximately 85% or higher, if treated with surgery only (Sopik *et al.,* 2023). The emergence of targeted therapy and immunotherapy has also helped to improve the outcomes of treatments in early and metastatic breast cancer. However, the success of these therapies is still limited due to tumor recurrence. Tumors use a variety of molecular mechanisms to enhance resistance to therapy. One of such mechanisms is when cancer stem cells (CSCs) activate DNA damage repair pathways, thereby instituting drug resistance to DNA-damaging chemotherapy (Nikitaki *et al.,* 2023).

Another mechanism that is commonly ignored is where the cells use a polyploid state, otherwise known as poly‐aneuploid cancer cells (PACCs) to resist chemotherapy and radiotherapy (Carroll *et al.,* 2024). Pienta and colleagues (2020) demonstrated that poly-aneuploid cancer cells (PACCs) play a pivotal role as potent reservoirs of heritable variation that facilitates rapid evolution, speciation, and adaptive response to environmental changes in cancer cells. Of paramount importance for patient prognosis and survival, these PACCs enable evolutionary rescue, development of therapy resistance, and metastatic progression. By definition, poly‐aneuploid cancer cells are cancer cells that have multiplied their aneuploid genome, and become larger, often with a misshapen large nucleus. These cells have been considered unimportant as they are believed to be destined to mitotic collapse (Pienta *et al.,* 2020). However, in recent studies, PACCs are observed in vitro to increase in numbers during therapy, remain in a prolonged senescent-like state, and eventually divide and produce progeny (Carroll *et al.,* 2024). To better understand how the presence of PACCs in primary cancer tissue with a poorer outcome correlates with a higher risk of tumor recurrence in breast cancer patients, these tissues must be studied pre-treatment and post-treatment. However, the large amount of Tissue Micro-Array (TMA) images obtained from breast cancer patients makes a manual analysis time-consuming and the detection of PACCs subjective, since it largely depends on pathologists’ interpretation of large nuclei.

StarDist developed a nuclei detection pipeline and provides pretrained deep learning models that effectively detect and segment nuclei in fluorescent and brightfield (Hematoxylin and Eosin) stained tissue images (Schmidt *et al.,* 2018). StarDist is a deep learning-based nuclei and cell detection and segmentation programme for 2D and 3D microscopy images, and can be used for cell/nuclei segmentation in densely-packed tissues. However, the pretrained StarDist models only work best for H&E (Hematoxylin and Eosin) and fluorescent-stained images. While the StarDist H&E model works when used on DAB (3,3’-diamobenzidine)-stained images, it is limited in accuracy and precision, and misses out a significant number of nuclei. DAB stains are often used in cancer research because it is highly specific for identifying the tumor compartment in tissues, which is characterized by the presence of the Epithelial Cell Adhesion Molecule (EpCAM) biomarker (Patriarca *et al.,* 2012). This project, therefore, focused on training a new model for nuclei detection in breast cancer TMA images which have been stained with EpCAM-DAB using the StarDist deep-learning package in Python, and then adapt an existing automatic PACCs detection pipeline to employ the new model for PACC analysis in breast cancer tissues.

# Materials and methods

## Preliminary Data Exploration and Preparation

The data used in this project is brightfield tissue micro-array (TMA) images stained using EpCAM-DAB and obtained from 1190 pre-treatment breast cancer patients. These images were preliminarily examined by pathologists and confirmed to contain some poly-aneuploid cancer cells. The data comprise a total of 20 high-resolution TMA images, with each microarray containing between 90-130 different tissues cores embedded in it at defined array coordinates. The data used for training the model was extracted from these cores. But first, we split each TMA into its constituent tissue cores using QuPath (Bankhead *et al*., 2017), an open-source software for digital pathology image analysis. QuPath (v0.5.1) requires Groovy scripting to perform most analysis, some of which are provided within the software. Using only 11 TMA images, this yielded a total of 1,391 tissues cores, including 144 blank cores (due to their positions within the microarray). The blank cores were excluded from further analysis.

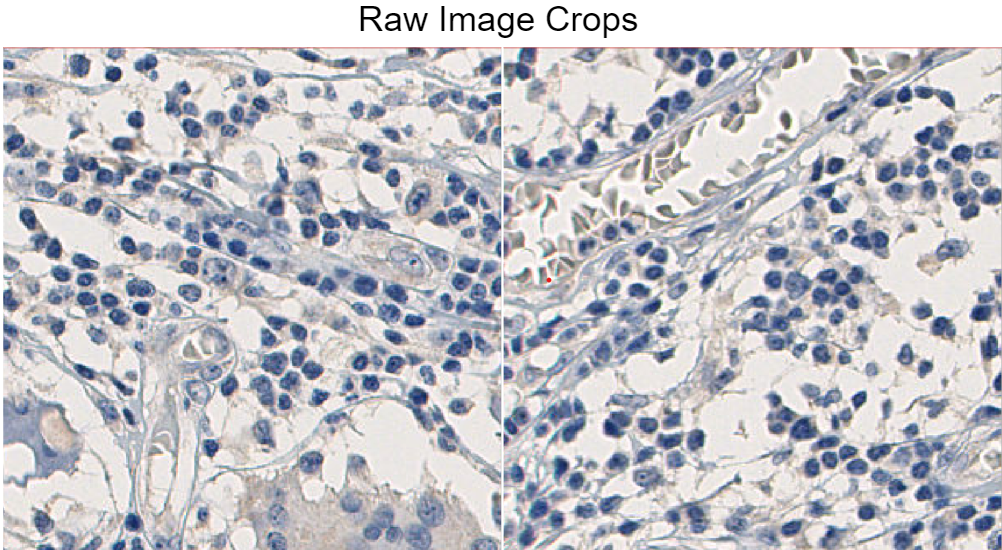
### Creating Crops/Training Data

After de-arraying the tissue cores, another Groovy script is used to automate the random selection of a specific number (20 in this case) of the tissue cores from the dataset, also within QuPath. This randomization is done to ensure that the images selected for training the model are diverse and representative of the entire dataset as much as possible, while the automation makes it easy to handle the large dataset. With 20 tissue cores randomly selected form all the de-arrayed TMA images from multiple patients, crops of these cores were further extracted and annotated as ground truth labels for training using the StarDist deep-learning package in Python. These crops are smaller sections or sub-regions of the cores, which are typical images that the model will be used on. The aim of training the model is to use it to automatically detect, classify, and segment nuclei of cancer cells (including PACCs) in breast cancer tissue microarray images. So, supervised learning was used, whereby the model is trained to make accurate predictions using labelled data, otherwise known as ground truths, with a defined target (Bankhead, 2022).

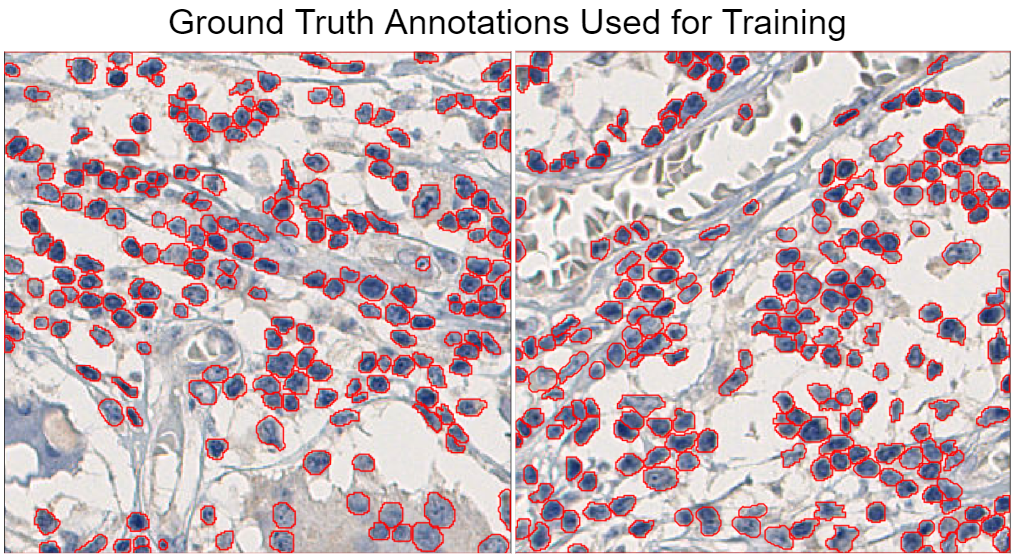
The crops were generated using QuPath. Each crop captures different morphologies and regions/time-points in each tissue core. For each core, 3 crops measuring 300 x 300 Pixels squared in dimension were created. The minimum recommend dimension for StarDist is 128 X 128 Pixels squared. The idea is for each core to contain enough cells/nuclei for annotation.

### Ground Truth Annotation

The ground truth labels were created by manually annotating the nuclei in all the crops as the regions of interest (ROI). A Groovy script is used to export the annotations as masks, which together with the unlabeled/raw images were used for training the nuclei detection model. StarDist recommends at least 10 crops for training. In this case, 51 crops (6 test images, 7 validation images, and 38 training images) are used to provide sufficient training and test data.

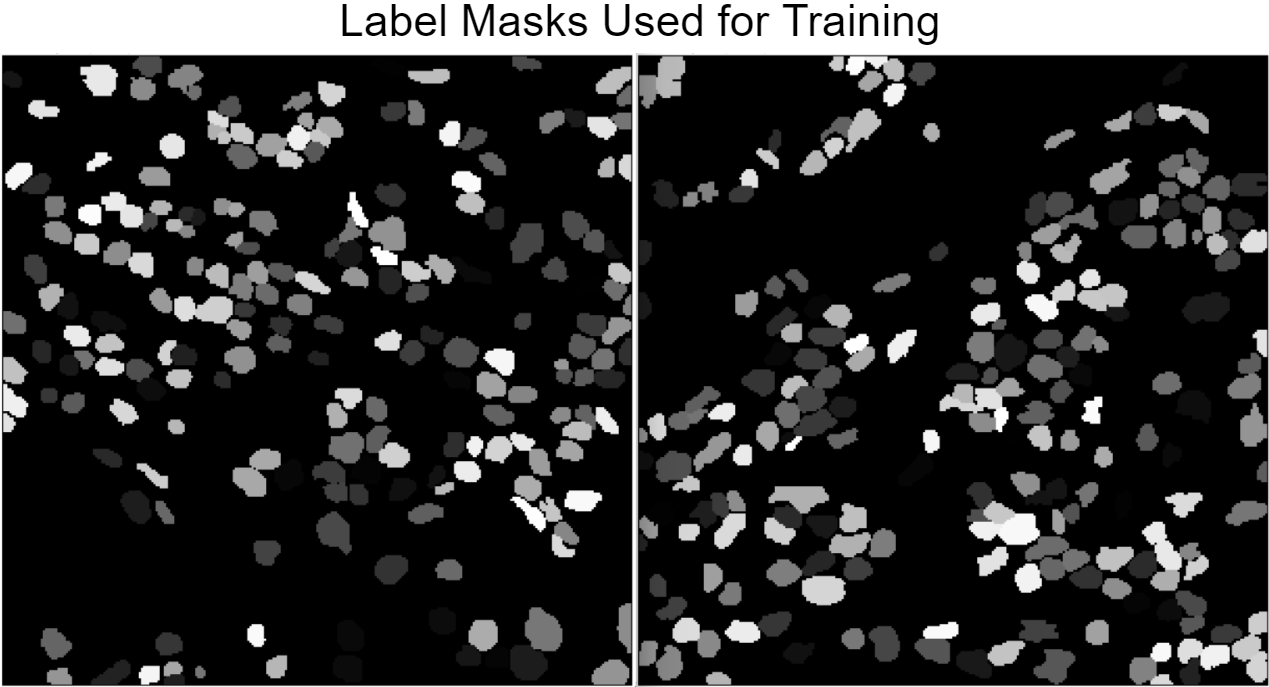


***Fig 1. Raw images of crops (300 X 300 Pixels) obtained from the TMA cores****. Crops were generated using QuPath*



***Fig 2. Annotations (ground truth labels) of crops****. All annotations were created manually using QuPath and exported as masks. The masks are used alongside the actual (unannotated images) for training.*

All crops were obtained from the 20 tissue cores extracted earlier. Like all segmentation networks, the StarDist package uses the actual image and ground truth (masks) to learn a classification task. Since this is supervised machine learning, the label masks or ground truth tells the network the specific class every pixel in the image belongs to. This way, the network learns the exact features of the nuclei in each image based on its corresponding mask, while ignoring any other feature on the images or objects. The images and their corresponding masks were saved in separate folders, but with exact same file names. This is to ensure accurate matching of each image to its corresponding label mask during training.



***Fig 3. Corresponding masks of images used for training the model.***

The final step in the data preparation process is the splitting of the dataset into training and test images. This is a standard procedure in machine learning, and depending on the size of the dataset, the ratio of training-test data can range from 80:20 to 90:10. With the aid of a custom Python script, the dataset containing 51 images and their corresponding masks is split into 6 test and 45 train subsets using an 88:12 ratio. During the training, the StarDist pipeline automatically further splits the training dataset into actual training images and validation images. The network uses the validation data to automatically improve the detection accuracy of the trained model during the learning process.

## Model Training Using StarDist

StarDist (Schmidt *et al.,* 2018) works well for segmenting all kinds of blob-like objects, especially roundish objects like cells and nuclei that have a star-convex shape. It uses a framework trained with a convolutional neural network (CNN) that can predict every pixel in a polygon and segment the cells or nuclei at that position with high accuracy. Schmidt *et al.* (2018) demonstrated that star-convex polygons are an ideal shape representation to accurately localize cell nuclei even in densely-packed tissues with overlapping nuclei. All cell nuclei, including PACCs fall into this category, hence the choice of StarDist for training our model. StarDist supports detection in both 2D and 3D images when trained with 2D and 3D images, respectively. There are other notable deep learning pipelines for nuclei detection and segmentation, such as Cellpose (Stringer *et al.,* 2021), but StarDist outperforms Cellpose on large datasets with densely-packed objects (Kleinberg *et al.,* 2022). StarDist also does a better job at separating overlapping nuclei and outperforms Cellpose with a low signal-to-background ratio of intensity. However, Cellpose appears to be a great fit for 3D cell segmentation (Kleinberg *et al.,* 2022).

### Programmes/Packages Installation

All annotations and predictions were run in QuPath (v0.5.1) (Bankhead *et al*., 2017). The Model training script is run solely in a Google Colab Notebook (0.0.1a2) as individual code blocks. But it can also be run in Jupyter Notebook and other IDEs or initiated from the terminal. Colab works well for machine learning, provides access to GPUs, has a simple interactive interface with TensorFlow (2.15.0), which is a major dependency for training a StarDist model, pre-installed. The training data (raw images and labelled masks) are imported into the Colab Notebook, where the StarDist model training pipeline is executed in Python (3.10.12) (Schmidt *et al*., 2018; Weigert *et al*., 2020; Weigert & Schmidt, 2022). Before running the script, StarDist (0.9.1) and TensorFlow (if using Colab, the latest TensorFlow comes pre-installed) was installed. The software Git (2.43.2) was used for the version control and GitHub for public sharing of all scripts and processes used in this project. The Notebook containing the entire training script code is freely [available on GitHub](https://github.com/Elijah-Ugoh/Model-Training-For-Nuclei-Segmentation/blob/master/Model_Training_Script.md). The code was stepwise manner, starting from installing StarDist. StarDist requires TensorFlow, a versatile machine learning (ML) library. The library is free and open-source, and typically used to build and deploy ML models. It also includes TensorBoard, which provides interactive visualisation while the model is being trained, allowing us to track and visualise metrics, such as epoch loss, the model graph (ops and layers), histograms of weights, biases as they change over time.

### Data Augmentation and Model Optimisation

StarDist also employs data augmentation to artificially create more training data by transforming existing images and masks into different geometrical, yet plausible forms. The augmenter function applies random rotations, flips, and intensity changes to the images, which are typically sensible for (2D) microscopy images, thereby making the trained model able to detect varying instances or forms of the original image. With the validation images, the model is constantly tested for accuracy and optimized during training. This helps improve the model's performance and accuracy on unseen data. Finally, the optimized threshold values were saved to disk so that they are automatically loaded with the model weights. This ensures consistent and reliable performance of the model across different unseen datasets and applications.

### Conversion to ONNX and Prediction

The trained model is exported as a zip file from Colab and saved locally. Because of compatibility troubles when trying to run the trained StarDist model in QuPath or other image processing software like ImageJ/Fiji (Schindelin *et al.,* 2012), the StarDist model requires post-processing to produce an instance cell/nuclei segmentation. This problem is typical when TensorFlow 2.0 and later is used in the training. To resolve the issue, the trained model is loaded in a Python environment with TensorFlow 1.x (we used v1.15.5) installed, and then exported it again. Ideally, this will work in Fiji. To use the trained model in QuPath, we further converted it to ONNX (Open Neural Network Exchange) format. This conversion allows for interoperability between different deep learning frameworks, in this case, enabling the model trained in TensorFlow to be run in other frameworks (that support ONNX) with minimal effort. To achieve this, tf2onnx (v1.16.1) was installed in the same Python environment, and then used to convert the trained model into ONNX format, which was later easily loaded into QuPath (using a StarDist extension for QuPath).

The model was included in an existing pipeline for PACC detection. The pipeline uses a Groovy script executed in QuPath for segmentation of the nuclei, classification of detections, and detection of PACC objects. Predictions were run on data (images) that the model didn’t see during trained. Data used are actual tissue microarray cores that contain cancer cells, including PACCs. With the measurement results obtained in detections, further analysis, such as core mean cell size for PACC size calculation can be performed in R.

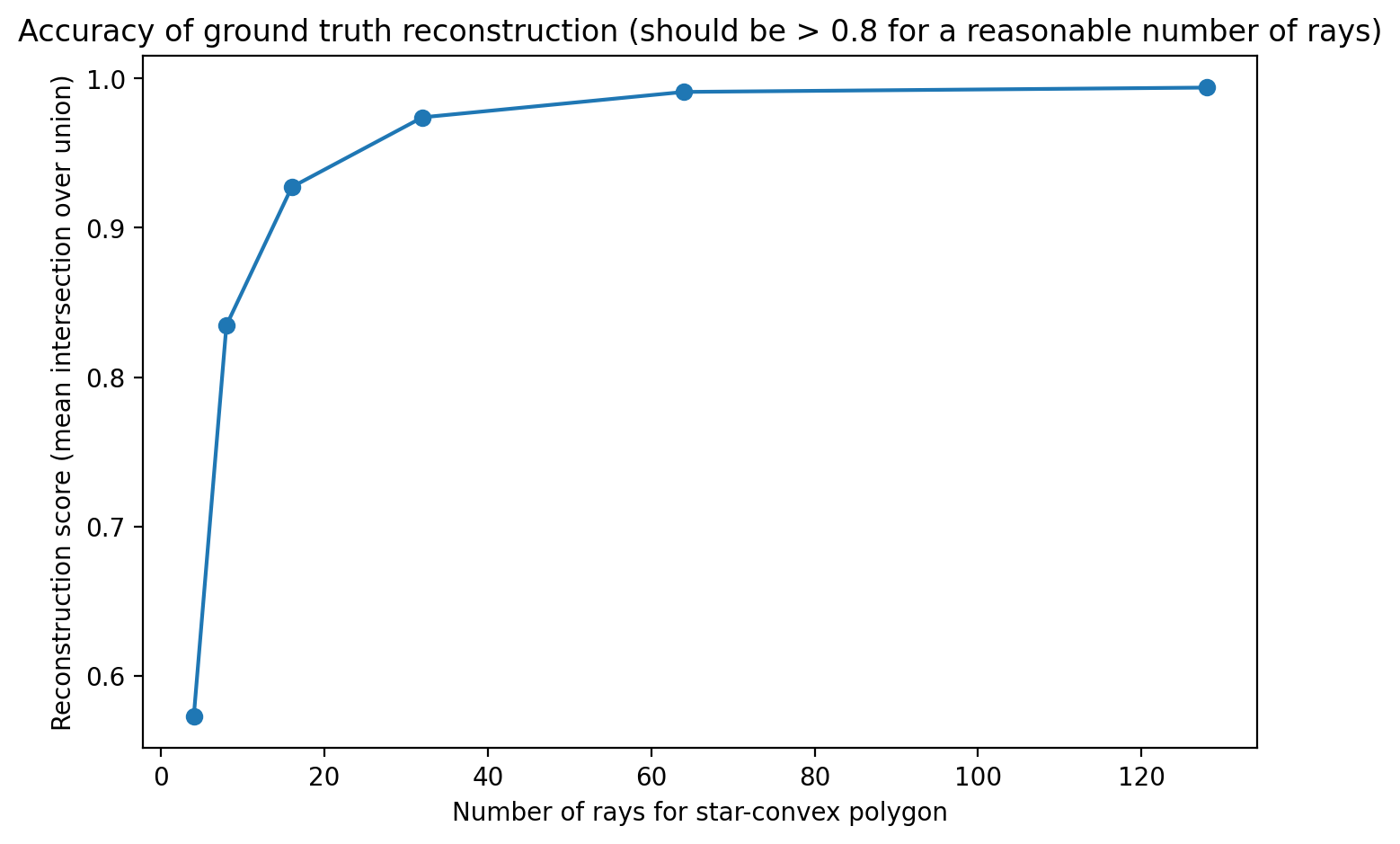
# Results

Nuclei predictions were executed in both Fiji and QuPath. However, Fiji wasn’t great for accessing the accuracy of the prediction of the model in larger slide images or tissue cores. Unlike QuPath, it returns the predictions as a separate output (label image only), the ROI (region of interest) Manager, or both. QuPath shows all of these in one simple, easy-to-manipulate interface that’s ideal for both small and larger images. QuPath also proved to be most reliable for accessing the precision and accuracy of nuclei detection and segmentation in our use case.

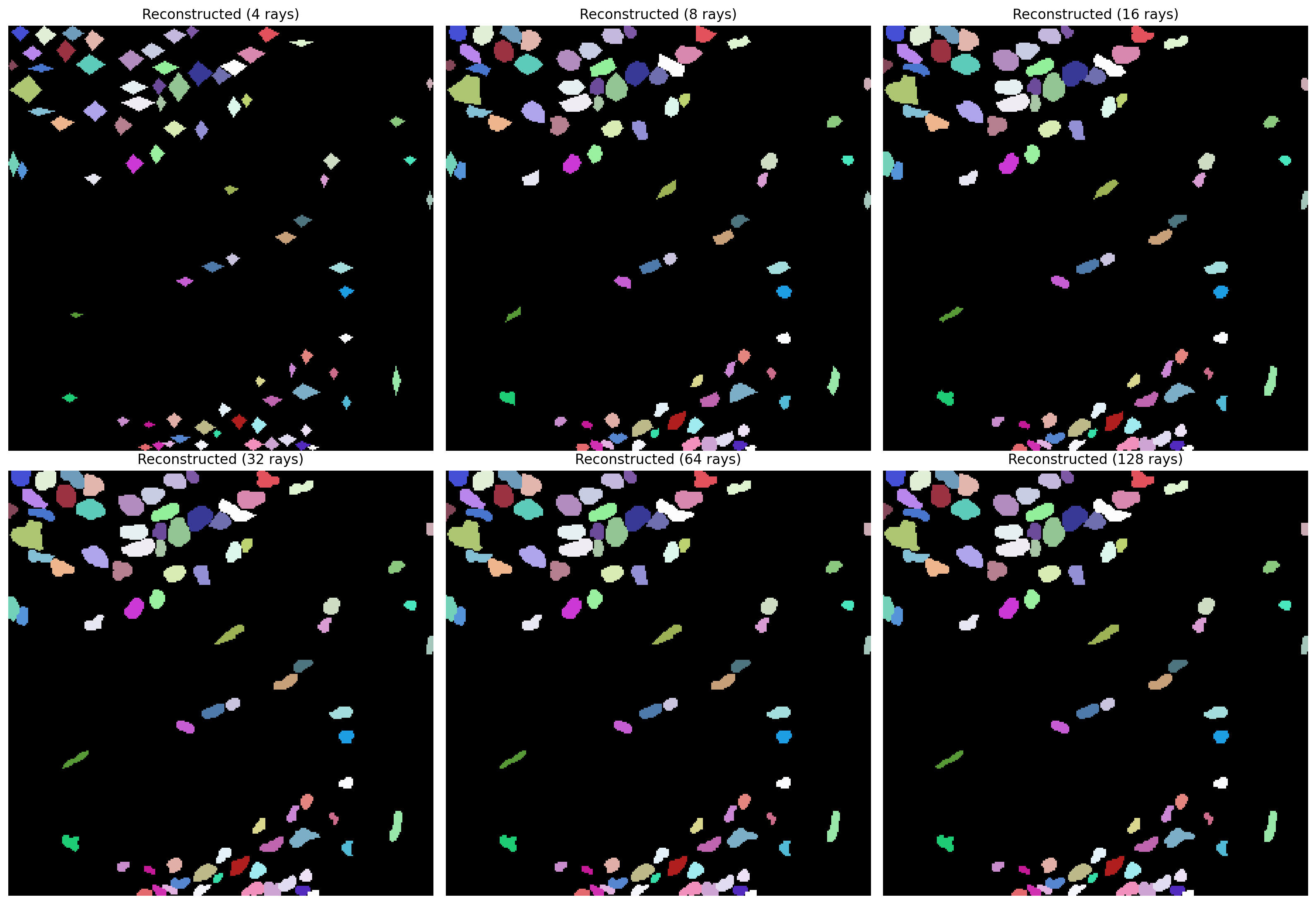
The model was trained twice, with subsequently higher number of training data. This was done to ascertain the minimum amount of training data required to obtain optimum prediction from the model and to avoid overfitting/overtraining (Charilaou & Battat, 2022). With only 10 annotated crops, the prediction accuracy was relatively high compared to the pre-trained H&E model provided by StarDist, but not sufficiently accurate for reliable downstream analysis. With 28 more annotations, the epoch loss during training was lower and the prediction accuracy was significantly better.

## Data Fitness for StarDist

An important step before the training is to check if the data loaded into the model training pipeline is actually ideal for deep learning using the StarDist package. StarDist works best for detecting roundish objects (Schmidt *et al.,* 2018), as it uses star-convex polygons to approximate the shapes of cell nuclei in microscopy images. A fundamental principle used by StarDist is to train the model to predict the distances to an object’s (nucleus) boundary along a fixed set of rays and object probabilities. So, we evaluated the performance of the training data with the StarDist model using different numbers of rays by reconstructing the ground truth labels and calculating the mean Intersection over Union (IoU) scores for each configuration or reconstruction. IOU is a model performance metric that we used to evaluate the accuracy of both the ground truth annotations and segmentations the model will produce based on those annotations (Rahman & Wang, 2016). The code for this analysis is included in the StarDist [data preparation script on GitHub](https://github.com/stardist/stardist/blob/main/examples/2D/1_data.ipynb). The results (Fig.4) showed that our data is fit for training with StarDist.



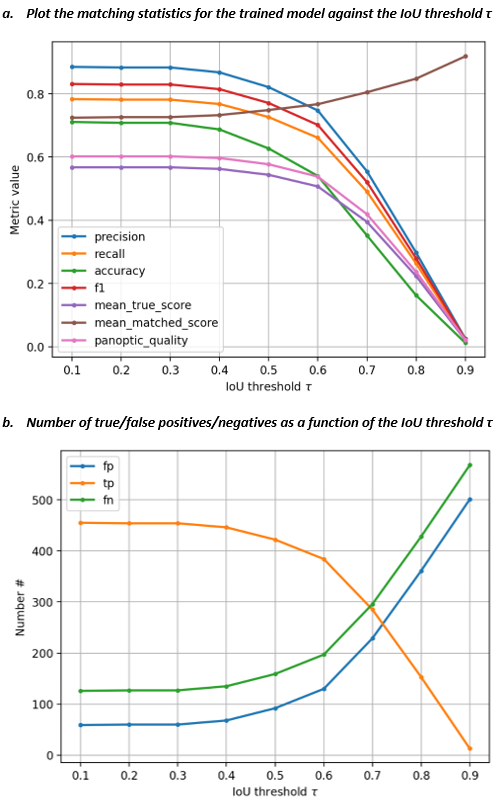
***Fig 4. Plot of reconstruction score (Intersection over Union).*** *This plot shows the relationship between different number of rays (4, 8, 16, 32, 64, and 128) used in the StarDist model and the reconstruction score, which represents the mean Intersection over Union (IoU) between the original labels and the reconstructed labels. The IoU quantifies the overlap between the predicted bounding box and the ground truth bounding box or annotated region in the training data (ROI). Here, the accuracy of the ground truth reconstruction is greater than 0.8 for at least 16 rays in a star-convex polygon.*



***Fig 5. Reconstruction of an example image from the training dataset using various number of rays.*** *The plots above show that the minimum reasonable number of arrays that yields a high mean IoU score between the original labels and the reconstructed labels is 16 rays. 32, 64, and 128 produce similar results, but are more accurate. Reconstructions with 4 and 8 rays are the least accurate.*

## Model Accuracy as a Function of IoU Thresholds

After training and optimizing the model, it was further evaluated against the validation dataset at various IoU thresholds to determine its performance in nuclei segmentation.



***Fig 5. Evaluating the accuracy of the trained StarDist 2D model.*** *(a) Matching statistics is computed for the validation data at various IoU thresholds (ranging from 0.1 to 0.9) and plot it as a function of the IoU threshold. (b) The numbers of true/false positives/negatives are plotted as a function of the IoU threshold. The plots provide insights into how well the model will perform under different evaluation criteria.*

The matching statistics (Fig. 5) include critical metrics like the precision of detections, mean true score, accuracy of prediction, F1 score, panoptic quality, mean matched score, number of predictions, false positives, true positives, and false negative predictions of the model. In the second subplot (b), the lines representing the number of false positives (fp), true positives (tp), and false negatives (fn) intersect at the IoU threshold value of 0.7 along the x-axis. This intersection point indicates that, at an IoU threshold of 0.7:

* The number of fp is equal to the number of tp, meaning that the model is making as many correct positive predictions as it is making incorrect positive predictions.
* The number of fn is also equal to the number of tp, indicating that the model is missing the same number of true positive instances as it is correctly identifying.

The closer the IoU score is to 1 (conventionally ranges from 0 to 1), the better the prediction. In practice, an IoU score of 0.5 is often used as a threshold for good object detection. An IoU score of 0.5 or higher indicates that the predicted bounding box sufficiently captures the object of interest, even if it might not perfectly align with the ground truth. Since the IoU score is above 0.5, the precision and detection accuracy of the model considered acceptable. This is consistent with the model precision (0.82) and accuracy (0.63) at an IoU of 0.7.

Table 1: Breakdown of Model Matching Statistics and Values at IoU threshold (τ) of 0.7

|  |  |  |
| --- | --- | --- |
| **Statistic** | **Meaning** | **Value** |
| **Threshold (τ)** | The threshold value used for determining true positives, false positives, and false negatives. In this case, τ=0.5 means that any IoU value greater than or equal to 0.5 is considered a positive match |  |
| **True Positives (tp)** | The number of correctly detected objects or instances by the model, where both the predicted and ground truth objects have sufficient overlap (IoU ≥ threshold) |  |
| **False Positives (FP)** | The number of objects or instances predicted by the model that do not have sufficient overlap (IoU < threshold) with any ground truth objects |  |
| **False Negatives (fn)** | The number of ground truth objects or instances that were not detected by the model |  |
| **Precision** | Measures the proportion of correctly predicted positive instances out of all instances predicted as positive by the model. |  |
| **Recall** | Also known as sensitivity or true positive rate, it measures the proportion of correctly predicted positive instances out of all true positive instances |  |
| **Accuracy** | Measures the overall correctness of the predictions made by the model |  |
| **F1 Score** | The harmonic mean of precision and recall, providing a balance between the two metrics |  |
| **n\_true** | The total number of ground truth objects or instances |  |
| **n\_pred** | The total number of objects or instances predicted by the model |  |
| **Mean True Score** | The mean IoU score computed for all ground truth instances |  |
| **Mean Matched Score** | The mean IoU score computed for all matched (correctly detected) instances |  |
| **Panoptic Quality** | A metric that combines both segmentation and detection accuracy, providing a unified measure of model performance |  |

## Detection and Segmentation Results on Whole TMA Cores

# References

1. Wilkinson, L., & Gathani, T. (2022). Understanding breast cancer as a global health concern. The British journal of radiology, 95(1130), 20211033. <https://doi.org/10.1259/bjr.20211033>.
2. Wang, J., & Wu, S. G. (2023). Breast Cancer: An Overview of Current Therapeutic Strategies, Challenge, and Perspectives. Breast cancer (Dove Medical Press), 15, 721–730 <https://doi.org/10.2147/BCTT.S432526>.
3. Sopik, V., Lim, D., Sun, P., & Narod, S. A. (2023). Prognosis after Local Recurrence in Patients with Early-Stage Breast Cancer Treated without Chemotherapy. Current oncology (Toronto, Ont.), 30(4), 3829–3844. <https://doi.org/10.3390/curroncol30040290> .
4. Nikitaki Z, Michalopoulos I, Georgakilas AG. Molecular inhibitors of DNA repair: searching for the ultimate tumor killing weapon. Future Med Chem. 2015 Aug;7(12):1543-58. doi: 10.4155/fmc.15.95. Epub 2015 Aug 26. PMID: 26306465.
5. Carroll, Christopher & Manaprasertsak, Auraya & Castro, Arthur & van den Bos, Hilda & Spierings, Diana & Wardenaar, René & Bukkuri, Anuraag & Engström, Niklas & Baratchart, Etienne & Yang, Minjun & Biloglav, Andrea & Cornwallis, Charlie & Johansson, Bertil & Hagerling, Catharina & Arsenian-Henriksson, Marie & Paulsson, Kajsa & Amend, Sarah & Mohlin, Sofie & Foijer, Floris & Hammarlund, Emma. (2024). Drug-resilient Cancer Cell Phenotype Is Acquired via Polyploidization Associated with Early Stress Response Coupled to HIF2α Transcriptional Regulation. Cancer research communications. 4. 10.1158/2767-9764.CRC-23-0396.
6. Pienta, K. J., Hammarlund, E. U., Axelrod, R., Brown, J. S., & Amend, S. R. (2020). Poly-aneuploid cancer cells promote evolvability, generating lethal cancer. Evolutionary applications, 13(7), 1626–1634. <https://doi.org/10.1111/eva.12929>.
7. Schmidt, U., Weigert, M., Broaddus, C., & Myers, G. (2018). Cell Detection with Star-Convex Polygons. In A. Frangi, J. Schnabel, C. Davatzikos, C. Alberola-López, & G. Fichtinger (Eds.), Medical Image Computing and Computer Assisted Intervention – MICCAI 2018 (pp. 11071). Lecture Notes in Computer Science. Springer, Cham. <https://doi.org/10.1007/978-3-030-00934-2_30>.
8. Patriarca, C., Macchi, R. M., Marschner, A. K., & Mellstedt, H. (2012). Epithelial cell adhesion molecule expression (CD326) in cancer: a short review. Cancer treatment reviews, 38(1), 68–75. <https://doi.org/10.1016/j.ctrv.2011.04.002>.
9. Bankhead, P., Loughrey, M. B., Fernández, J. A., Dombrowski, Y., McArt, D. G., Dunne, P. D., McQuaid, S., Gray, R. T., Murray, L. J., Coleman, H. G., James, J. A., Salto-Tellez, M., & Hamilton, P. W. (2017). QuPath: Open source software for digital pathology image analysis. Scientific reports, 7(1), 16878. <https://doi.org/10.1038/s41598-017-17204-5>.
10. Peter Bankhead. (2022). Developing image analysis methods for digital pathology. The Journal of Pathology, 257(4), 391-402. <https://doi.org/10.1002/path.5921>
11. Stringer, C., Wang, T., Michaelos, M., & Pachitariu, M. (2021). Cellpose: a generalist algorithm for cellular segmentation. Nature methods, 18(1), 100–106. <https://doi.org/10.1038/s41592-020-01018-x>.
12. Kleinberg, G., Wang, S., Comellas, E., Monaghan, J. R., & Shefelbine, S. J. (2022). Usability of deep learning pipelines for 3D nuclei identification with Stardist and Cellpose. Cells & development, 172, 203806. <https://doi.org/10.1016/j.cdev.2022.203806>.
13. Weigert, Martin & Schmidt, Uwe & Haase, Robert & Sugawara, Ko & Myers, Gene. (2020). Star-convex Polyhedra for 3D Object Detection and Segmentation in Microscopy. 3655-3662. 10.1109/WACV45572.2020.9093435.
14. Weigert, Martin & Schmidt, Uwe. (2022). Nuclei segmentation and classification in histopathology images with StarDist for the CoNIC Challenge 2022.
15. Schindelin, Johannes & Arganda-Carreras, Ignacio & Frise, Erwin & Kaynig, Verena & Longair, Mark & Pietzsch, Tobias & Preibisch, Stephan & Rueden, Curtis & Saalfeld, Stephan & Schmid, Benjamin & Tinevez, Jean-Yves & White, Daniel & Hartenstein, Volker & Eliceiri, Kevin & Tomancak, Pavel & Cardona, Albert. (2012). Fiji: An Open-Source Platform for Biological-Image Analysis. Nature methods. 9. 676-82. 10.1038/nmeth.2019.
16. Charilaou, P., & Battat, R. (2022). Machine learning models and over-fitting considerations. World journal of gastroenterology, 28(5), 605–607. <https://doi.org/10.3748/wjg.v28.i5.605>.
17. Rahman, M.A. & Wang, Yang. (2016). Optimizing Intersection-Over-Union in Deep Neural Networks for Image Segmentation. In: Bebis, G., et al. Advances in Visual Computing. ISVC 2016. Lecture Notes in Computer Science(), vol 10072. Springer, Cham. <https://doi.org/10.1007/978-3-319-50835-1_22>