# 🧠 Deep Learning Segmentation Pipeline for the Spires-Jones Lab

This pipeline provides a deep learning-based alternative to the traditional thresholding approach used in the Spires-Jones lab. It supports both the **Cellpose CNN** model and the **SAM2 Transformer** model for image segmentation.

The system allows users to either retrain models or load pretrained ones via a command-line interface. This design enables flexible model usage without modifying the core Python code. Segmentation behavior can be adjusted through mode selection and parameter tuning, depending on the object type.

## ⚙️ Key Features

* Compatible with **Cellpose** and **SAM2** segmentation models
* **Flexible run modes**: tune, train, evaluate
* Command-line interface for full pipeline control
* Output includes logs, performance metrics, visualizations, and trained models

## 📜 Example Command

python "cellpose\_retraining.py" \

-mn PSD \

-tts 0.9 \

-mode train\_test\_mixed\_batch \

-input1 "<path\_to\_batch\_B\_input>" \

-gt1 "<path\_to\_batch\_B\_gt>" \

-input2 "<path\_to\_batch\_A\_input>" \

-gt2 "<path\_to\_batch\_A\_gt>" \

-ts 0.3 \

-tunep 1 \

-trainp 1 \

-testp 1 \

-numt 7 \

-nume 599 \

-run\_mode tune

## 🚀 Run Modes

The pipeline supports three main execution modes:

* tune: Runs hyperparameter tuning using the specified tuning split.
* train: Trains the model using the given training data and number of epochs.
* eval: Evaluates a trained model on the test dataset. The test batch can differ from the training batch depending on the selected training mode.

## 📂 Output Files

Each run produces five key outputs to support training transparency and post-run analysis:

1. **Logs**  
   Show current processing steps, training loss, and test loss. Helpful for identifying model convergence.
2. **Parameter Configurations (JSON)**  
   Saves hyperparameters used in the run. Useful for auditing and comparing models.
3. **Results CSVs**  
   Contain performance metrics for individual slices and for overlapping masks (via post-processing).
4. **Visual Comparisons**  
   Side-by-side plots showing original images, predicted masks, and ground truth labels for both high- and low-performing predictions.
5. **Saved Models**  
   Trained models are saved and can be reused for future evaluations without retraining.

## 🎛️ Parameter Overview

| **Parameter** | **Description** |
| --- | --- |
| -mn | Model name (e.g., PSD). Also defines save folder structure. |
| -tts | Train-test split ratio (e.g., 0.8). |
| -mode | Segmentation mode (e.g., train\_test\_mixed\_batch). |
| -input1, -gt1, -input2, -gt2 | Paths to image and ground truth data. |
| -ts | Tuning data split ratio (e.g., 0.3). |
| -tunep | Proportion of tuning data used in tuning (0–1). |
| -trainp | Proportion of training data used in training (0–1). |
| -testp | Proportion of test data used in evaluation (0–1). |
| -numt | Number of tuning trials (e.g., 7). |
| -nume | Number of epochs (e.g., 599). |
| -run\_mode | Mode to execute (tune, train, eval). |

## 🧪 Notes

* Model training and evaluation depend on the balance and quality of input data across batches.
* Use the train\_test\_mixed\_batch mode when combining batches with differing ground truths.