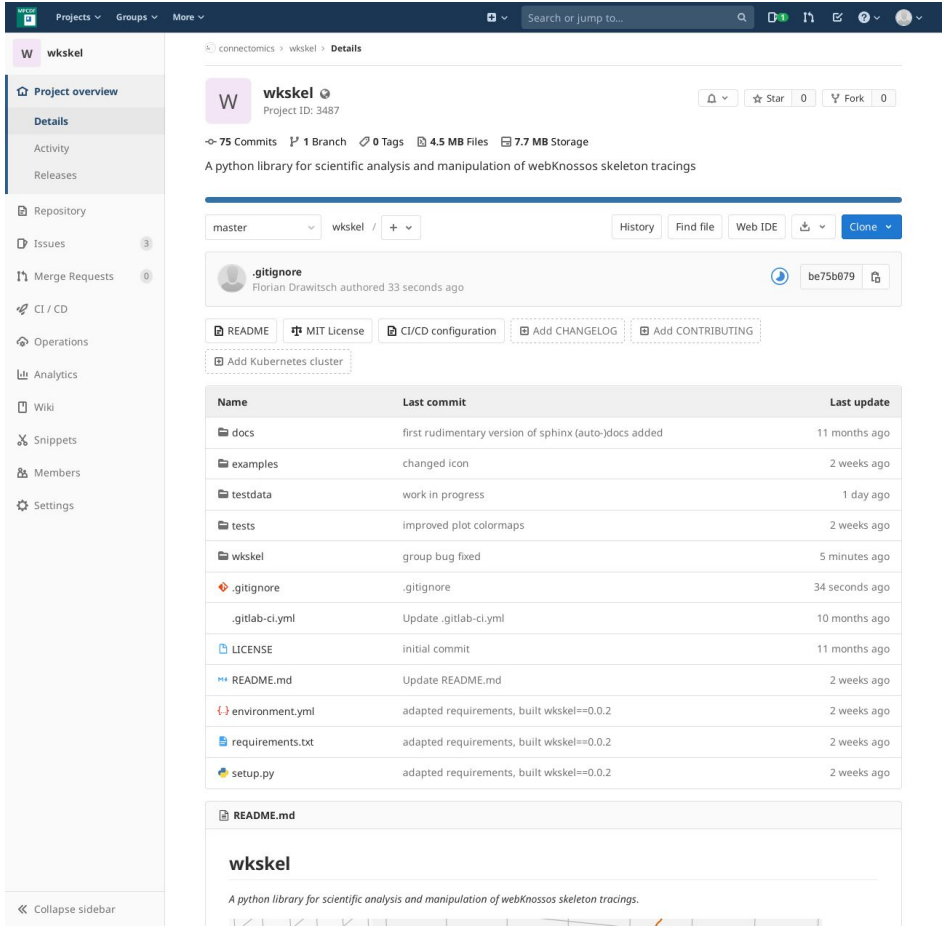
Autoencoders for mSEM artifact detection

Call for contributions: wkskel



wkskel: python library for skeleton manipulation

- Developed on the side from scratch
- Most important functionality is there
- Porting full functionality & performance of matlab skeleton class is a lot of work
- Please help by using, improving & documenting it
- Two contribution strategies possible:
  - Direct commits to master: Please write test for your method (repo has gitlab ci/cd set up)
  - Pull requests

How to install it:
$ pip install wkskel
How to use it:
Jupyter notebook tutorial:
https://gitlab.mpcdf.mpg.de/connectomics/wkskel:examples

Autoencoders for mSEM artifact detection

mSEM artifacts: general

- mSEM artifacts are a problem for automated downstream analysis -> hinder classification / segmentation / agglomeration
- mSEM artifacts come in various forms and are difficult to detect reliably using humans or hand-designed algorithms
- Variety of artifacts and large data size
  ➔ Problematic situation for classical supervised approaches (e.g. U-net)
- Training a *naive* U-net or similar (>E6 params) in a supervised fashion would require HUGE amount of labeled training data

mSEM artifacts: data situation

- Huge amount of unlabeled data (even in mag-8-8-1 ~1TB)
- Much less labeled data (> 500 training patches ~ 10MB):
  - Lablog Summary: https://mhlablog.net/2020/02/05/training-data-for-artefact-detection-in-msem-data/
  - Circular Artifact: https://webknossos.brain.mpg.de/annotations/Explorational/5e387a00010000501b1a41bd#25238,20546,3546,0,0.513,1
  - Horse shoe Artifact: https://webknossos.brain.mpg.de/annotations/Explorational/5e387a00010000501b1a41bd#19895,15530,3732,0,0.513,12
  - Scratch Artifact: https://webknossos.brain.mpg.de/annotations/Explorational/5e387a00010000501b1a41bd#20088,16049,3925,0,0.513,17

Supervised CNN vs. unsupervised autoencoder (AE)

CNN

$f: X \rightarrow Y$

$y = f'(x) + \varepsilon$

$\text{Min}_p \text{Loss}(f'_p(x) - y)$

```
class CNN:
  def predict(x):
    y_hat = f(x)
    return y_hat
```

AE

$e: X \rightarrow L$

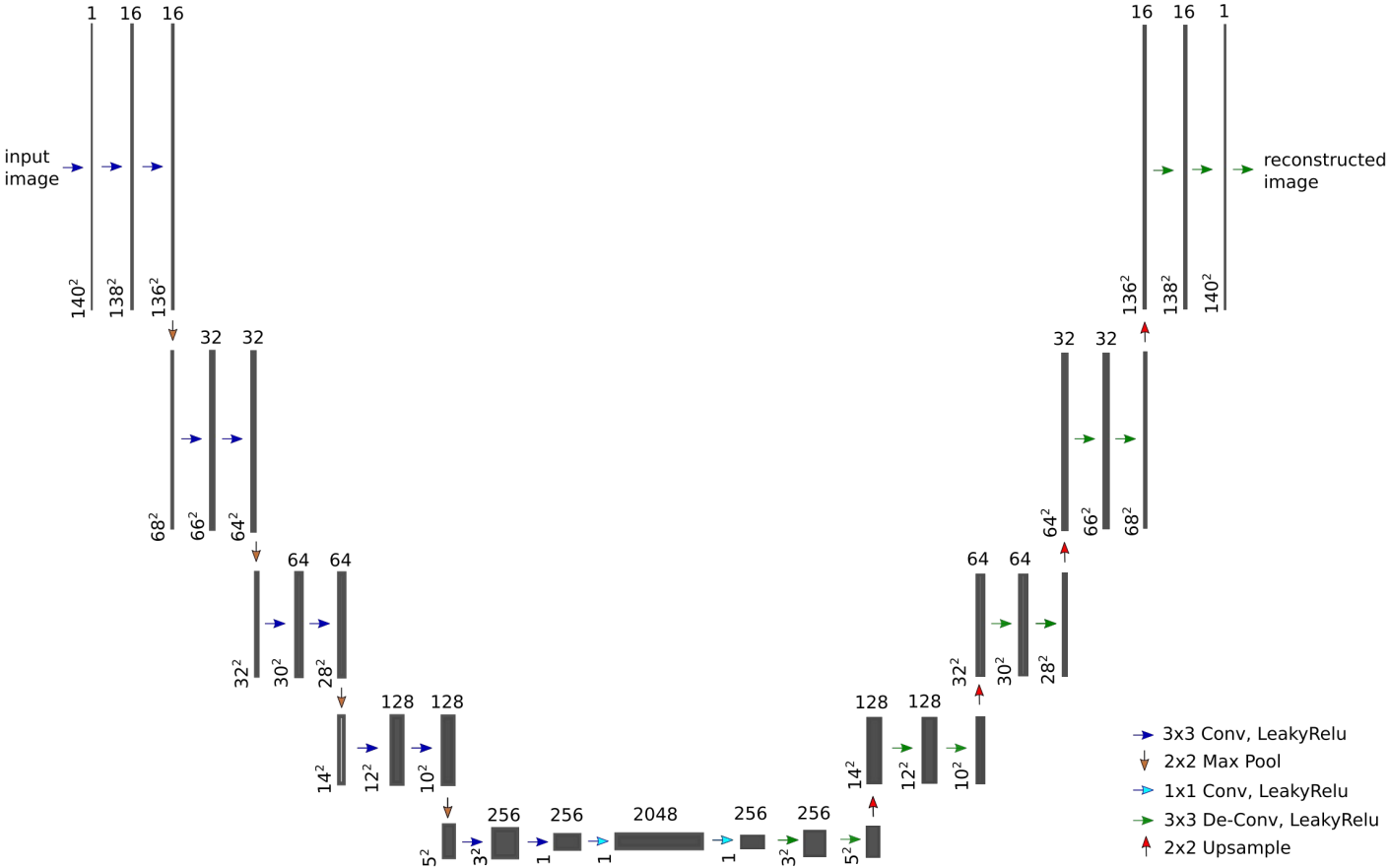
$d: L \rightarrow X$

$x = d'(e'(x)) + \varepsilon$

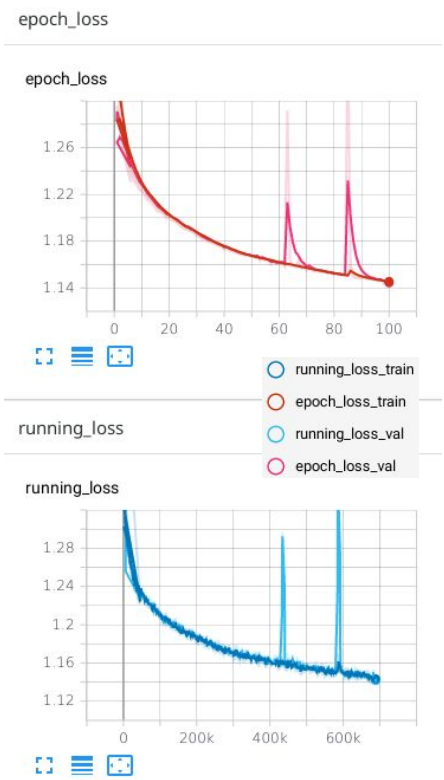
$\text{Min}_p \text{Loss}(d'_p(e'_p(x)) - x)$

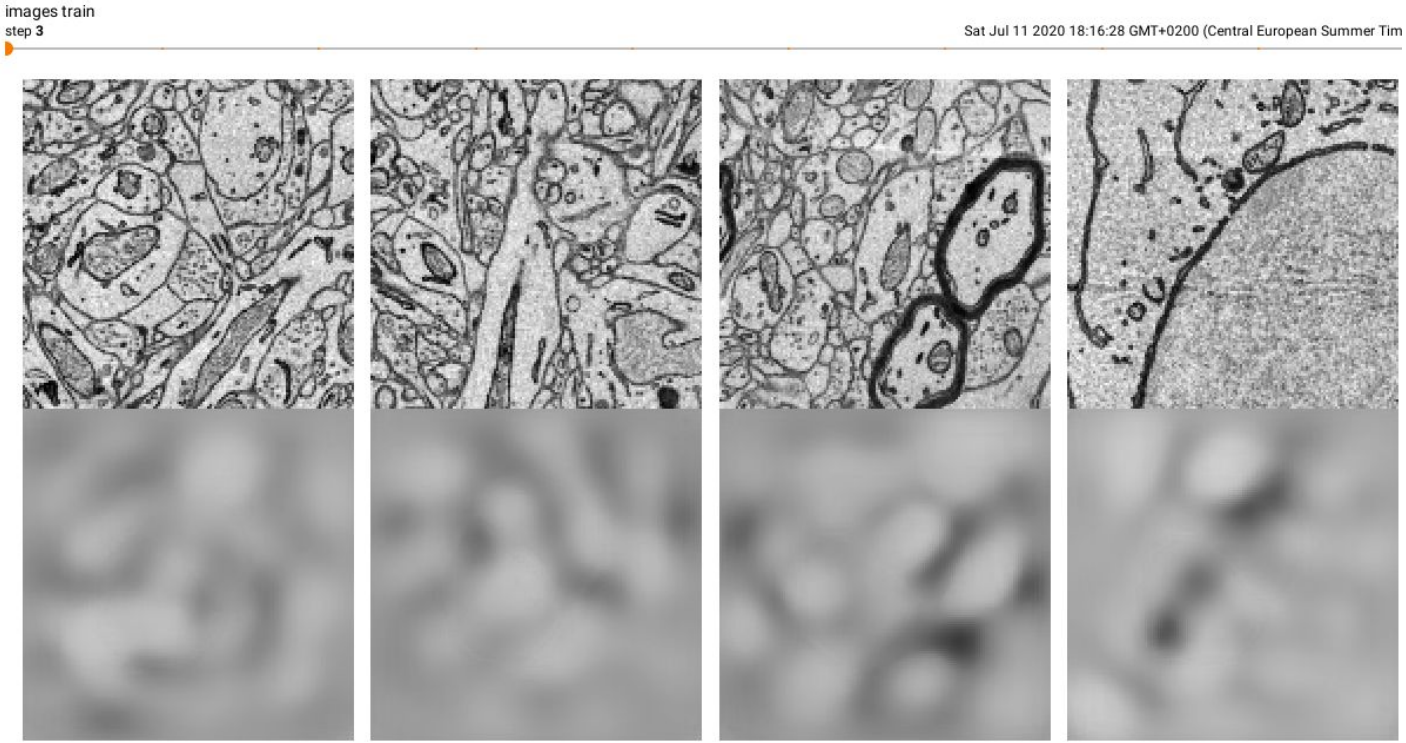
```
class AE:
  def predict(x)
      x_hat = d(e(x))
      return x_hat
```

Basic multi-res AE for mSEM data

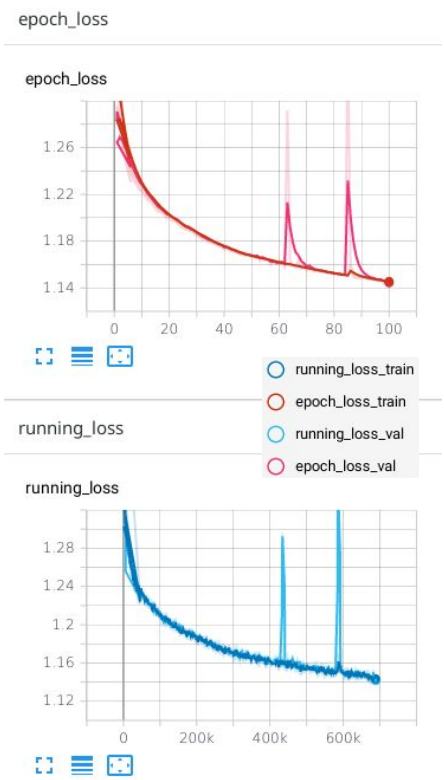


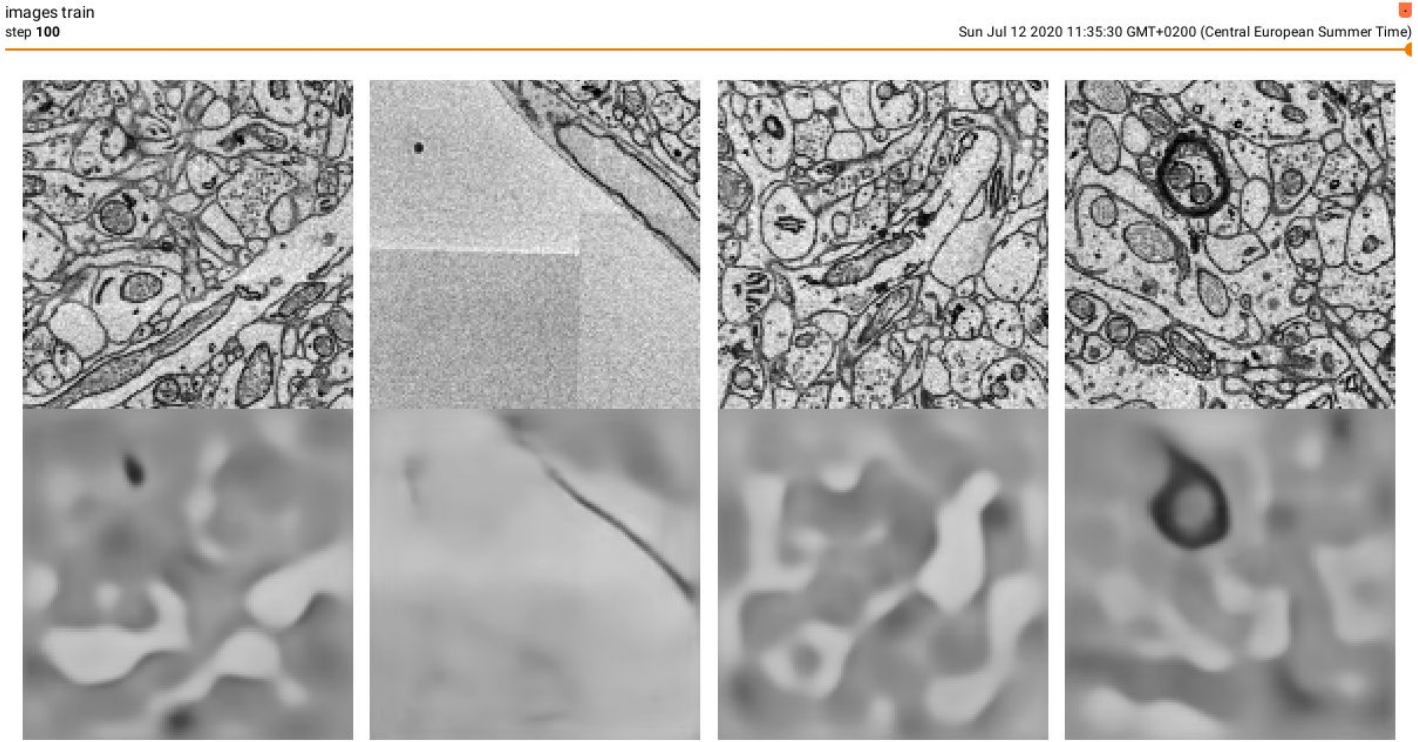
Basic multi-res AE for mSEM data



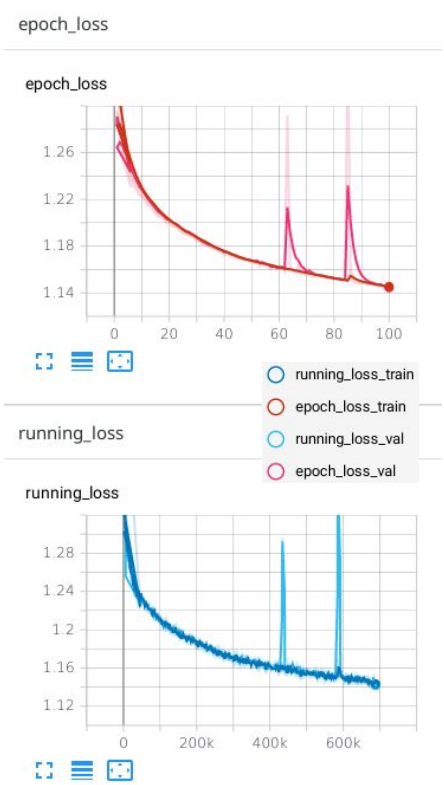


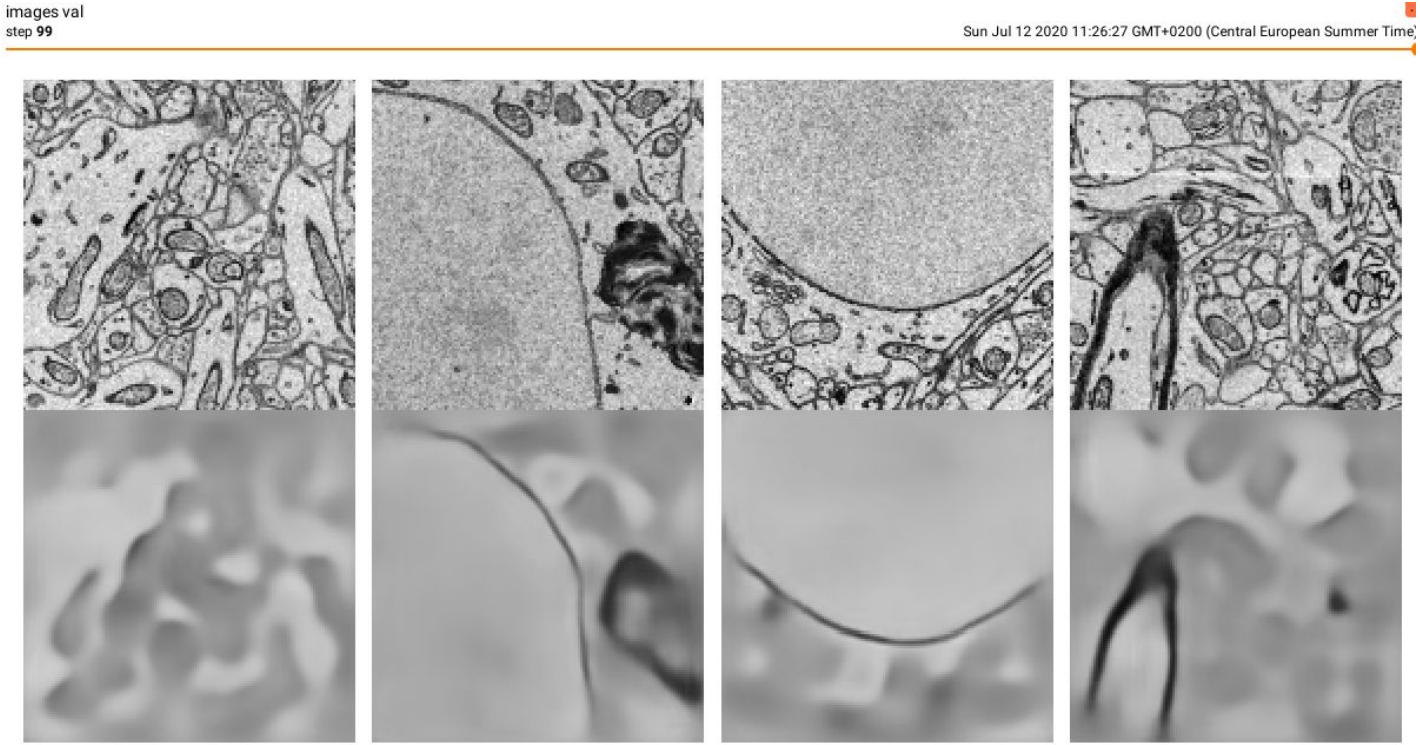
Basic multi-res AE for mSEM data



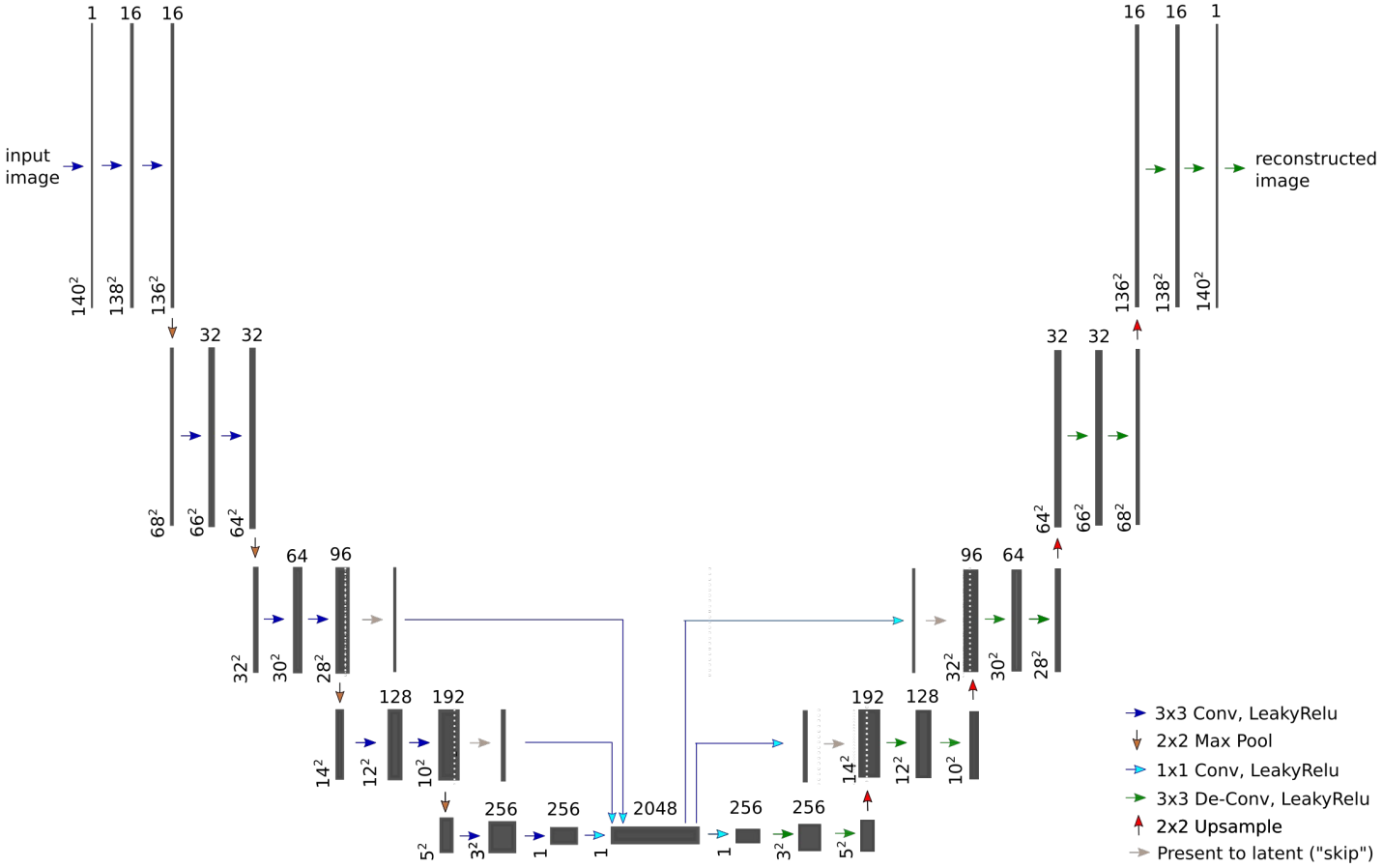


Basic multi-res AE for mSEM data

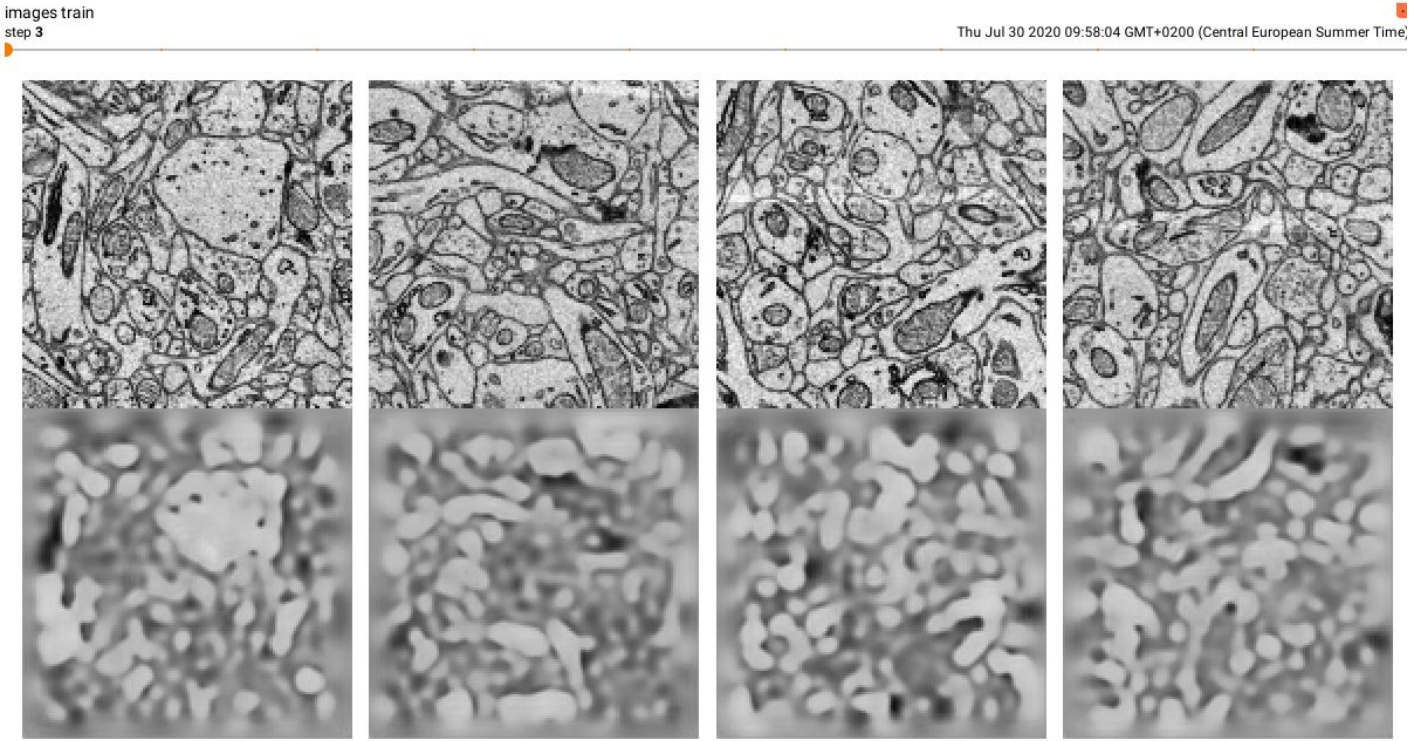




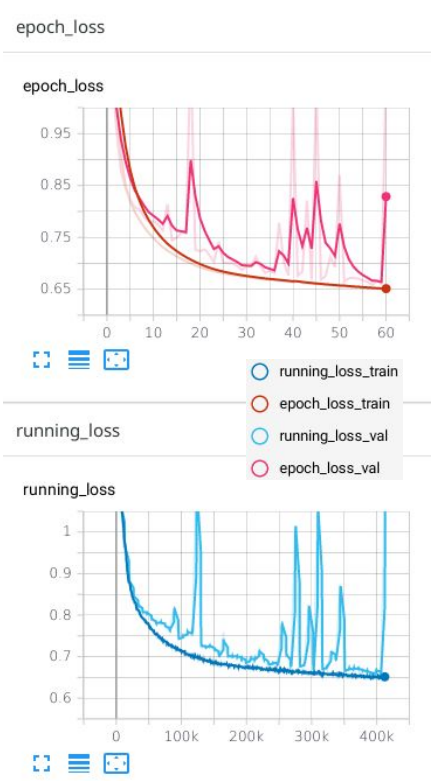
Multi-res “skip” AE for mSEM data

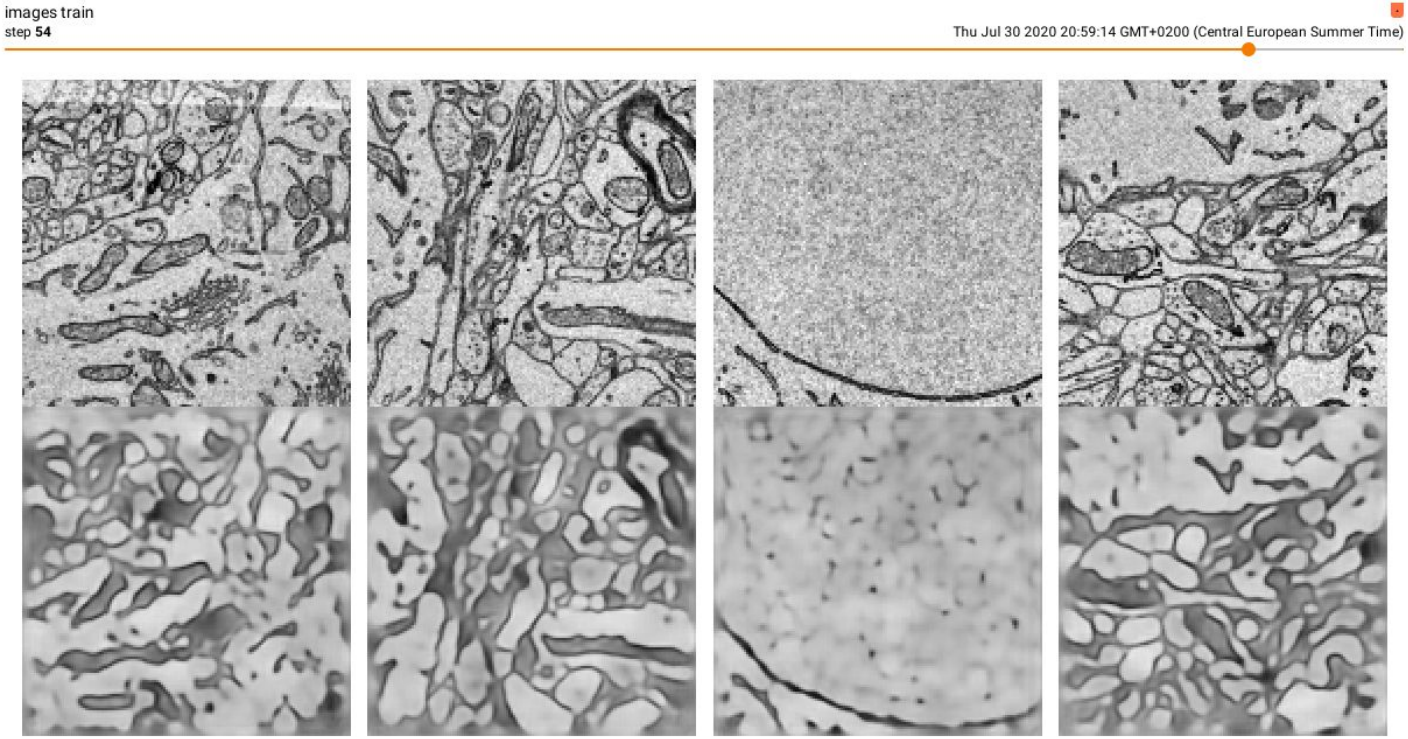


Multi-res “skip” AE for mSEM data

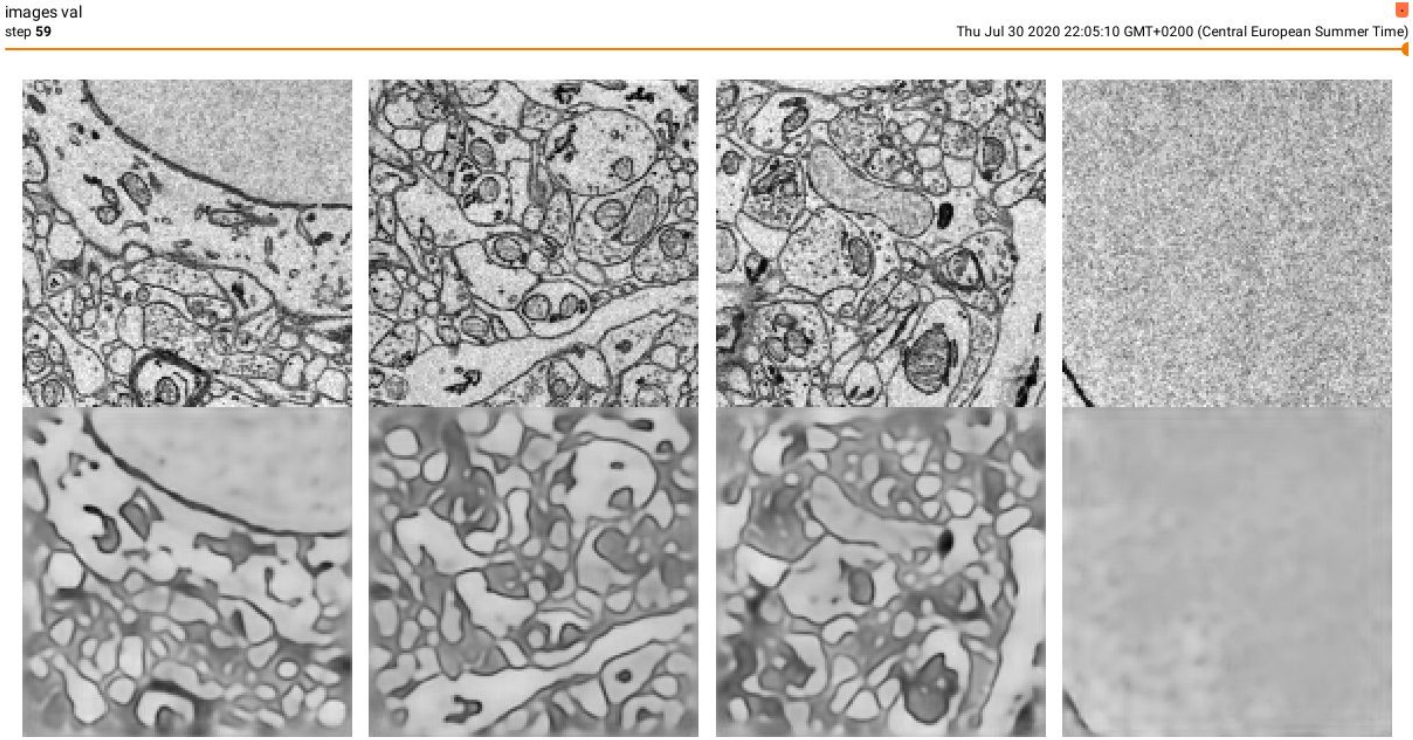


Multi-res “skip” AE for mSEM data

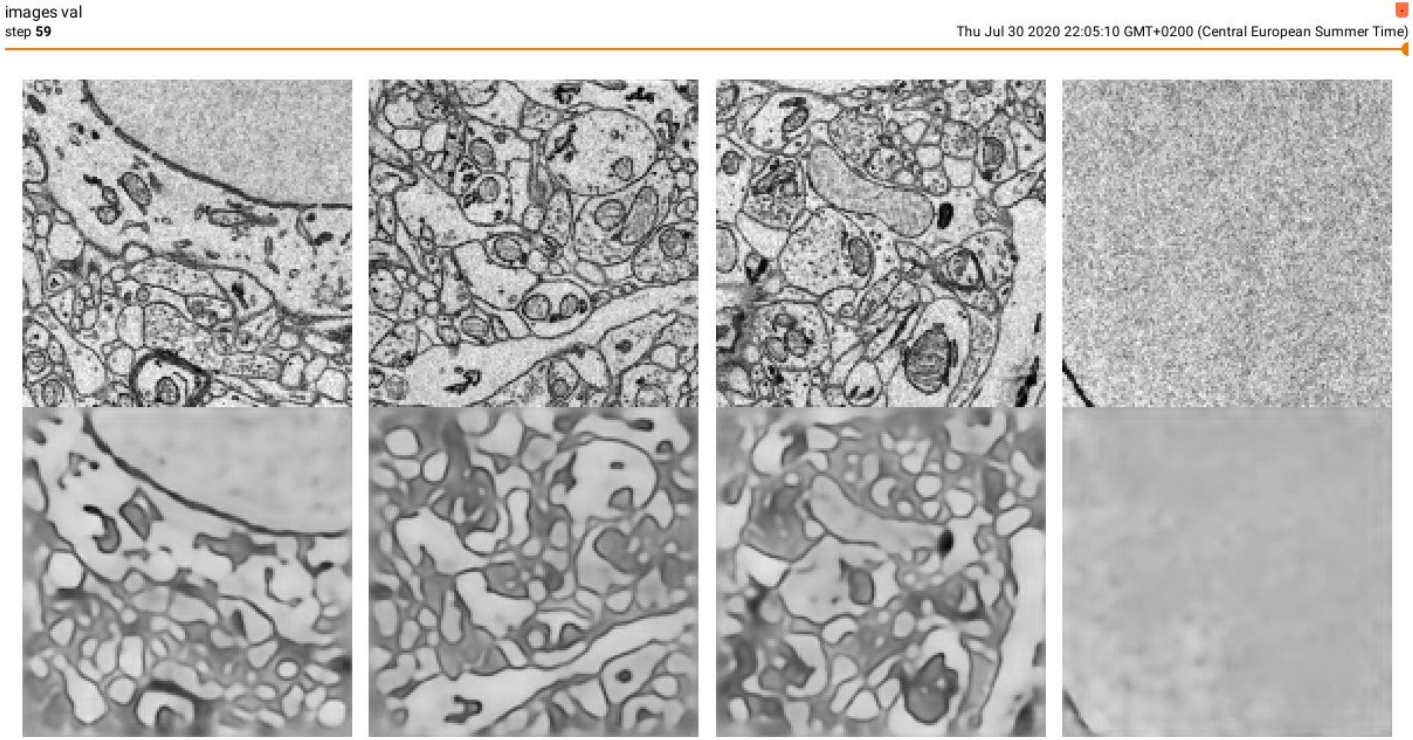




Multi-res “skip” AE for mSEM data



Multi-res “skip” AE for mSEM data



Unsupervised AE vs. semi-supervised encoder + classifier

AE

$e: X \rightarrow L$

$d: L \rightarrow X$

$x = d'(e'(x)) + \varepsilon$

$\text{Min}_p \text{Loss}(d'_p(e'_p(x)) - x)$

```
class AE:
  def predict(x)
      x_hat = d(e(x))
      return x_hat
```

Encoder + classifier

$e: X \rightarrow L$

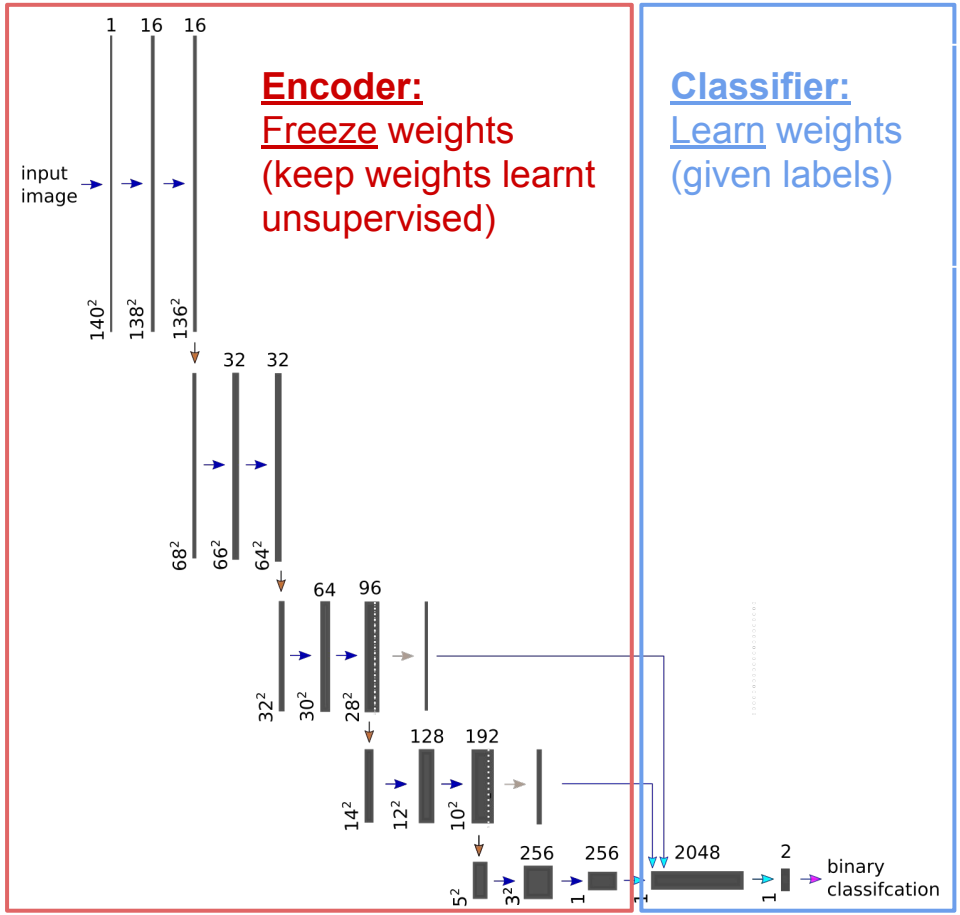
$c: L \rightarrow Y$

$y = c'(e'(x)) + \varepsilon$

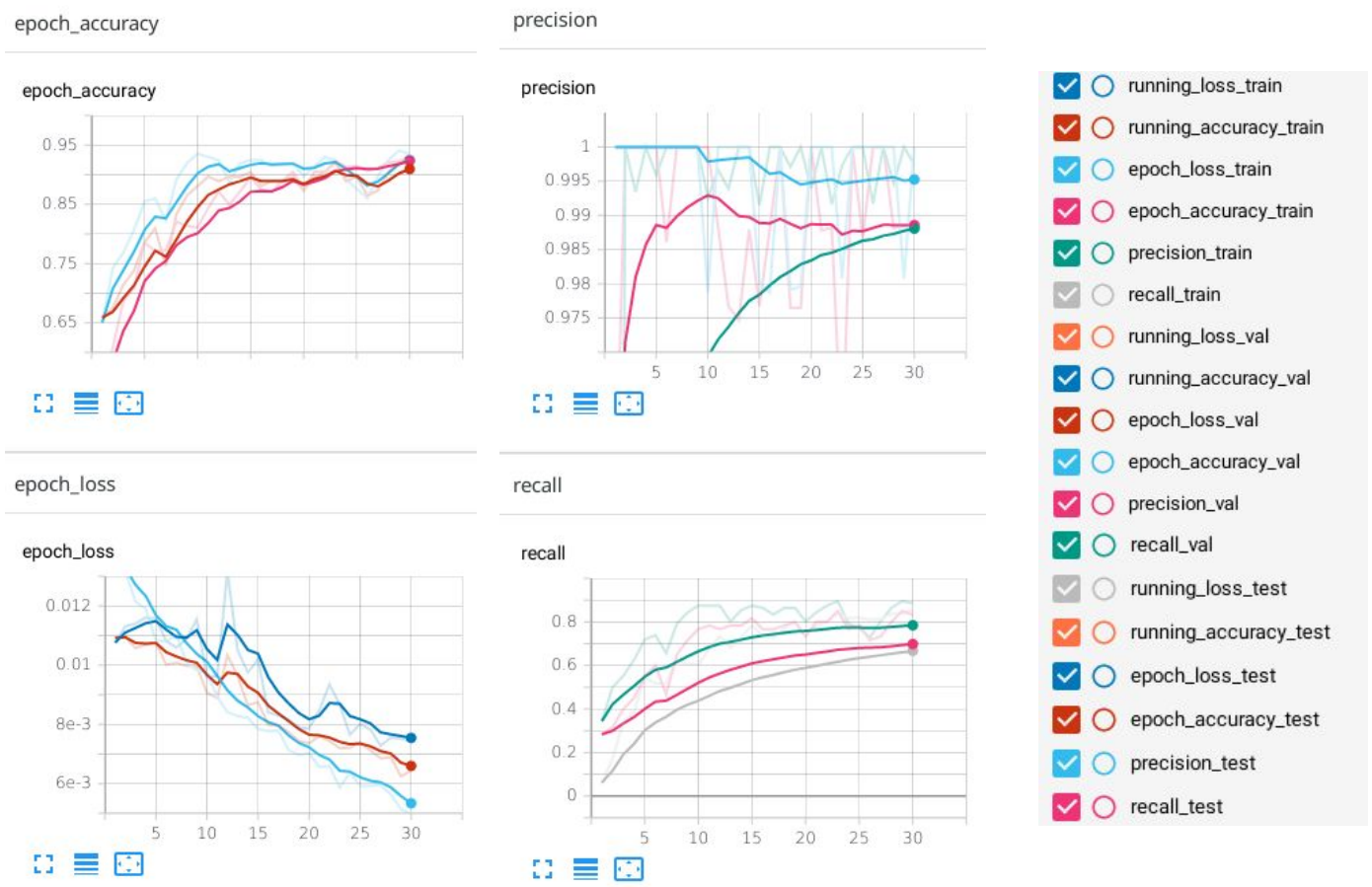
$\text{Min}_p \text{Loss}(c'_p(e'(x)) - x)$

```
class EncoderClassifier:
  def predict(x)
      y_hat = c(e(x))
      return y_hat
```

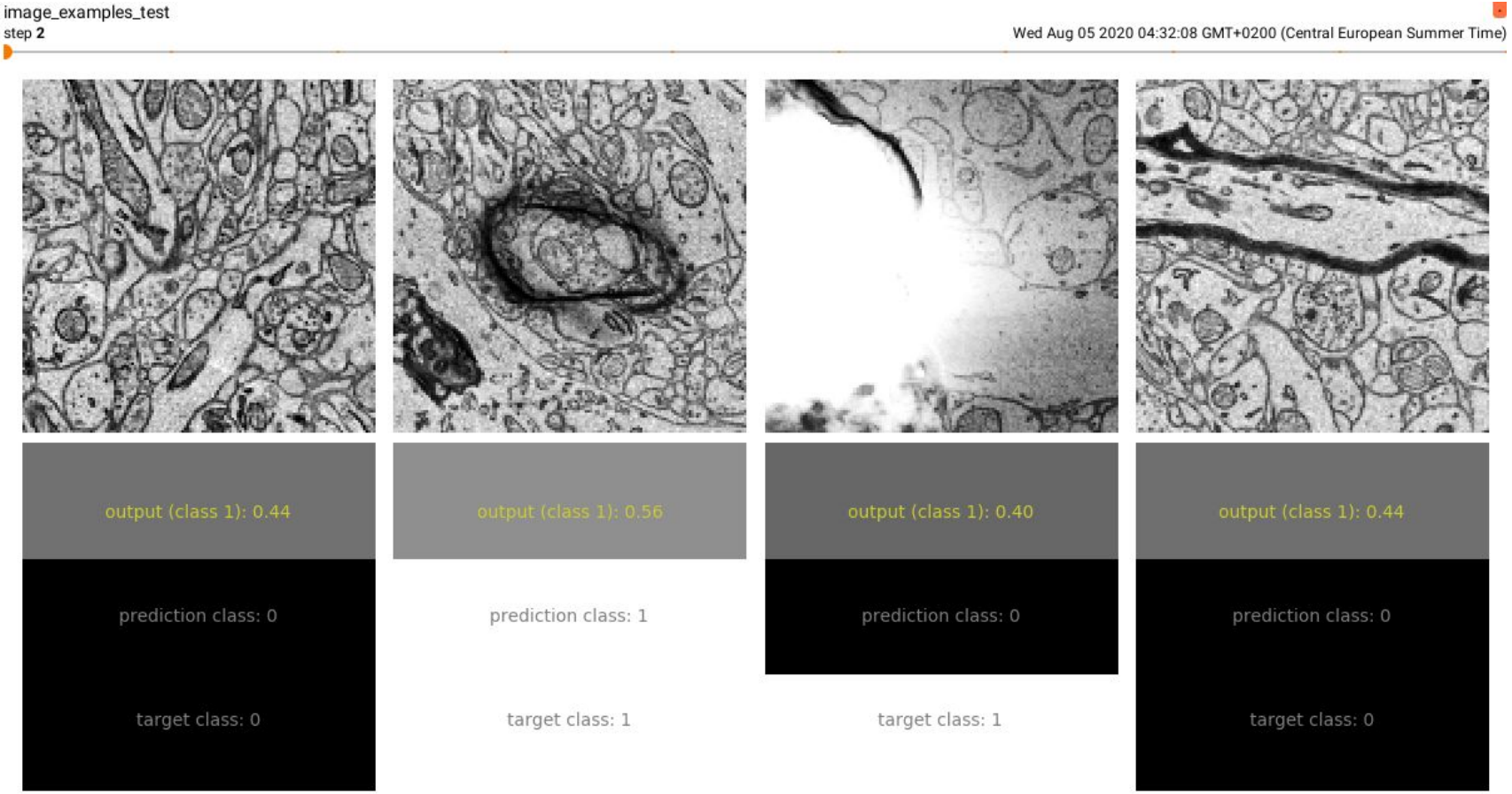
Multi-res “skip” encoder + binary classifier for mSEM data



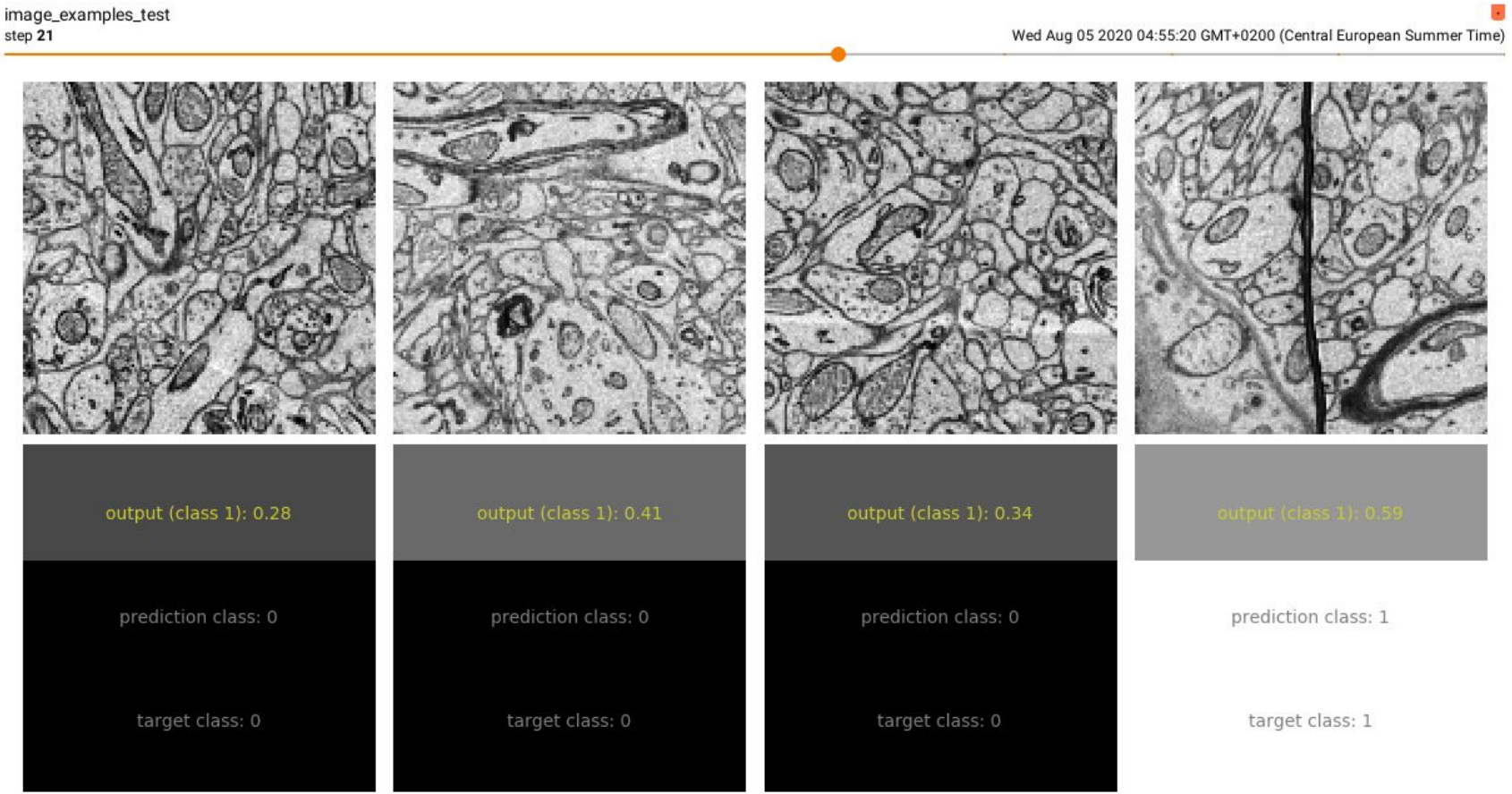
Multi-res “skip” encoder + binary classifier for mSEM data



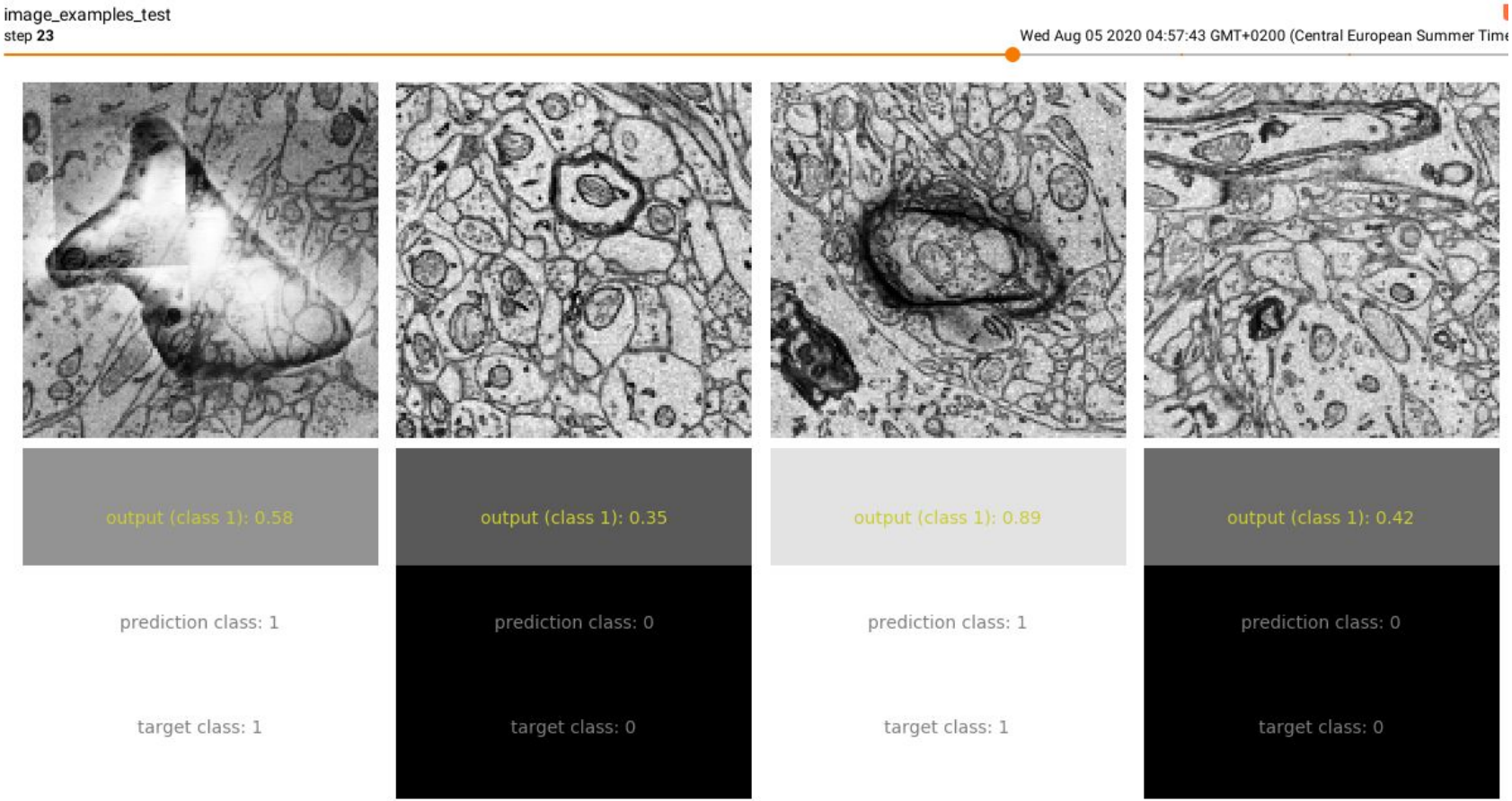
Multi-res “skip” encoder + binary classifier for mSEM data



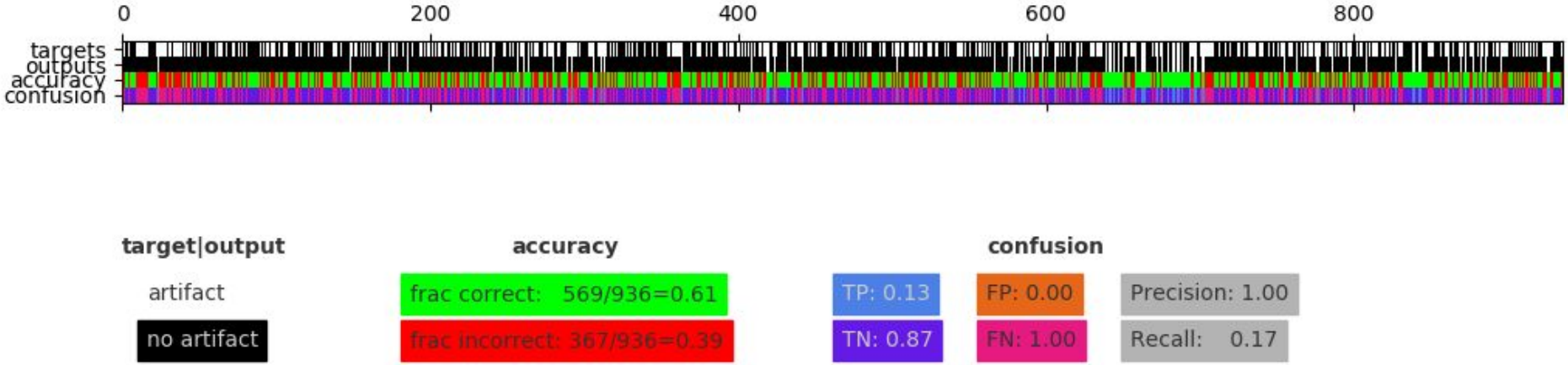
Multi-res “skip” encoder + binary classifier for mSEM data



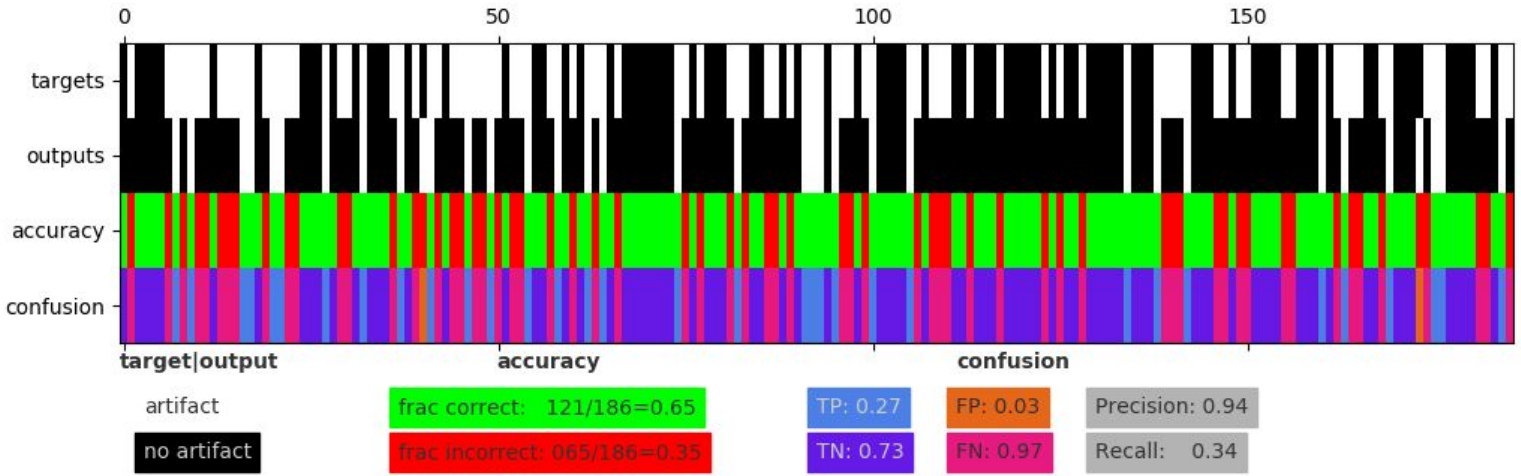
Multi-res “skip” encoder + binary classifier for mSEM data



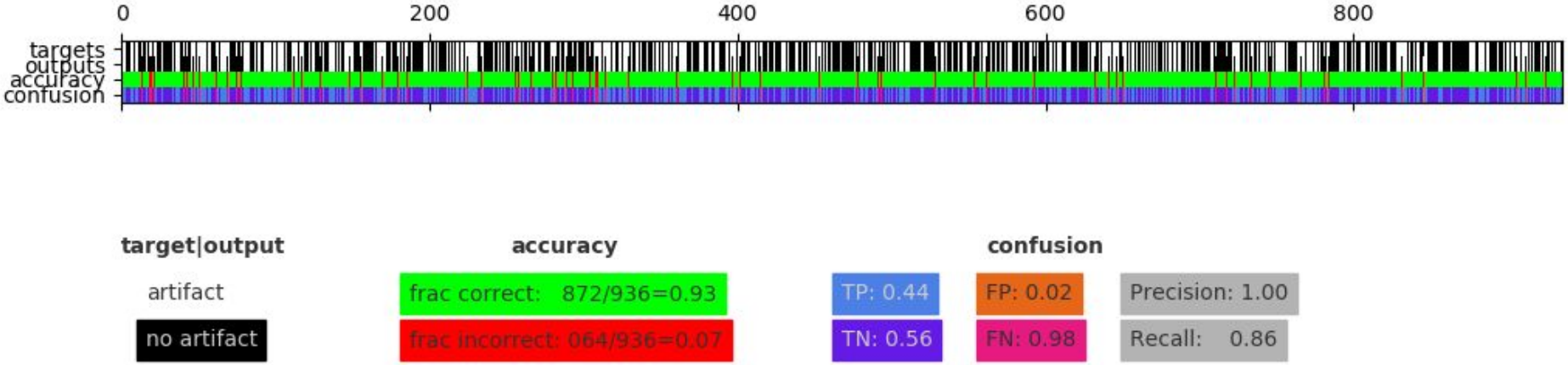
Multi-res “skip” encoder + binary classifier for mSEM data



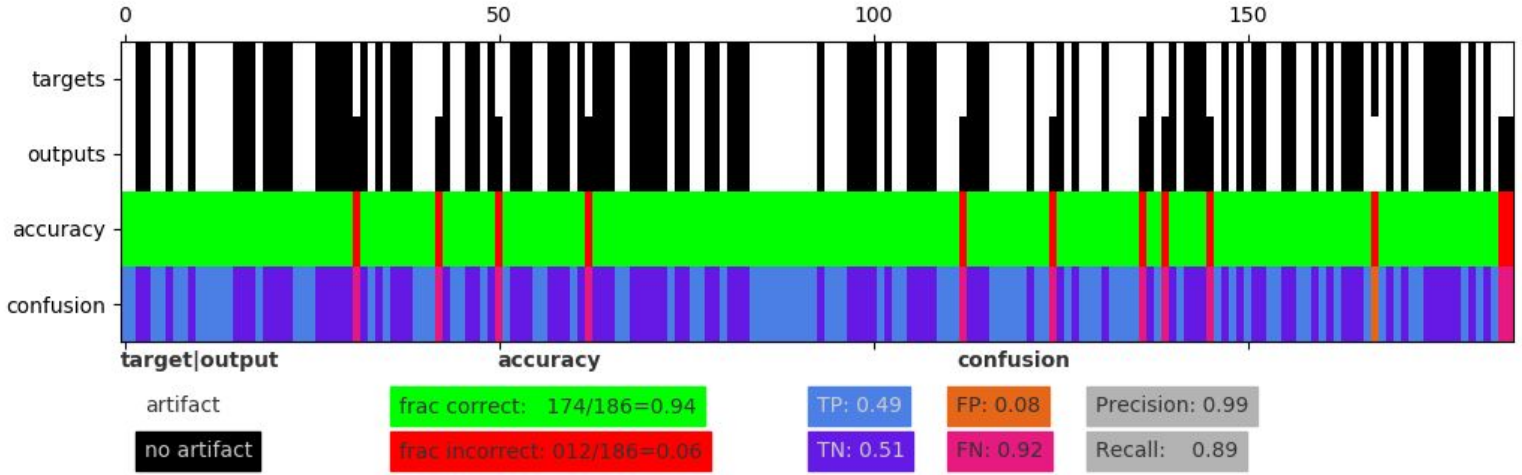
Multi-res “skip” encoder + binary classifier for mSEM data



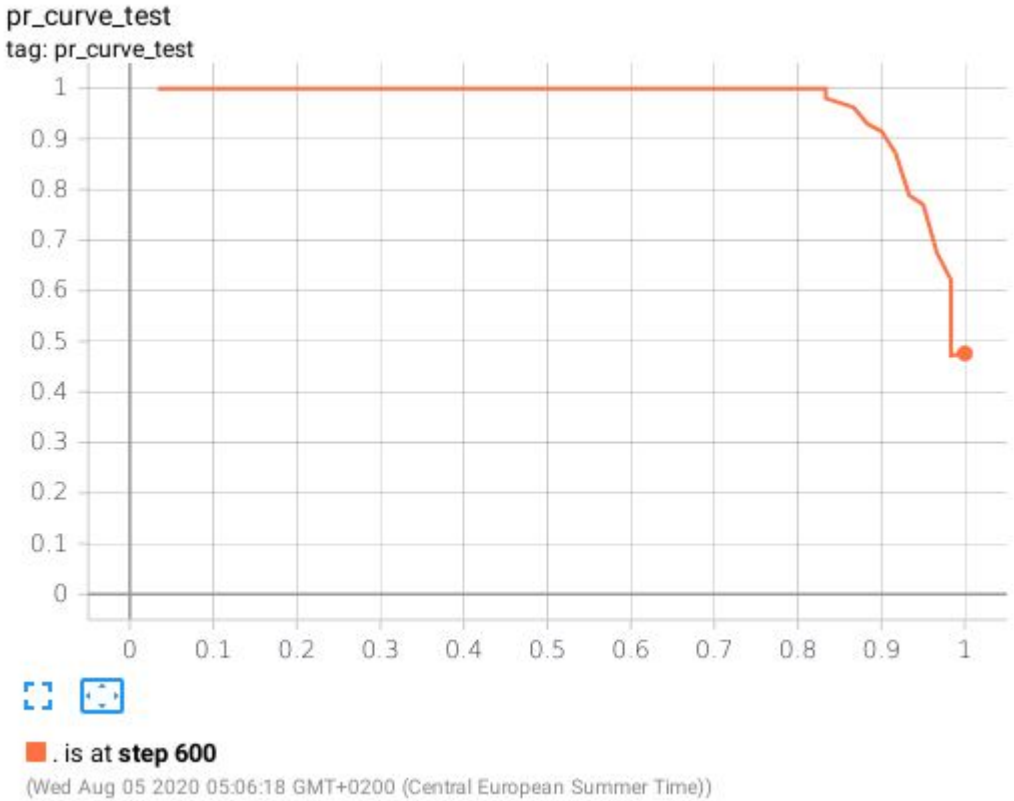
Multi-res “skip” encoder + binary classifier for mSEM data



Multi-res “skip” encoder + binary classifier for mSEM data



Multi-res “skip” encoder + binary classifier for mSEM data

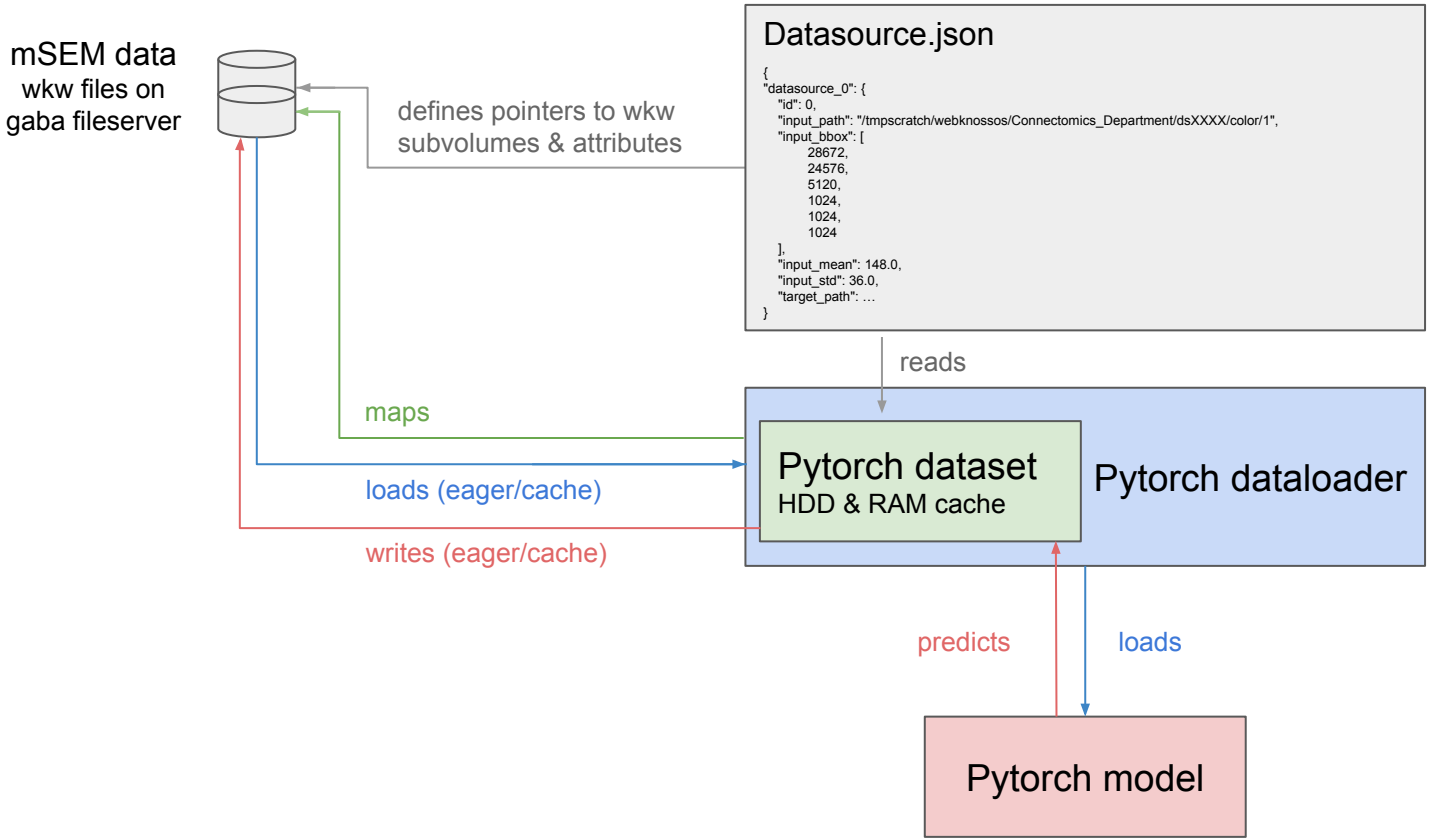


Improving the model

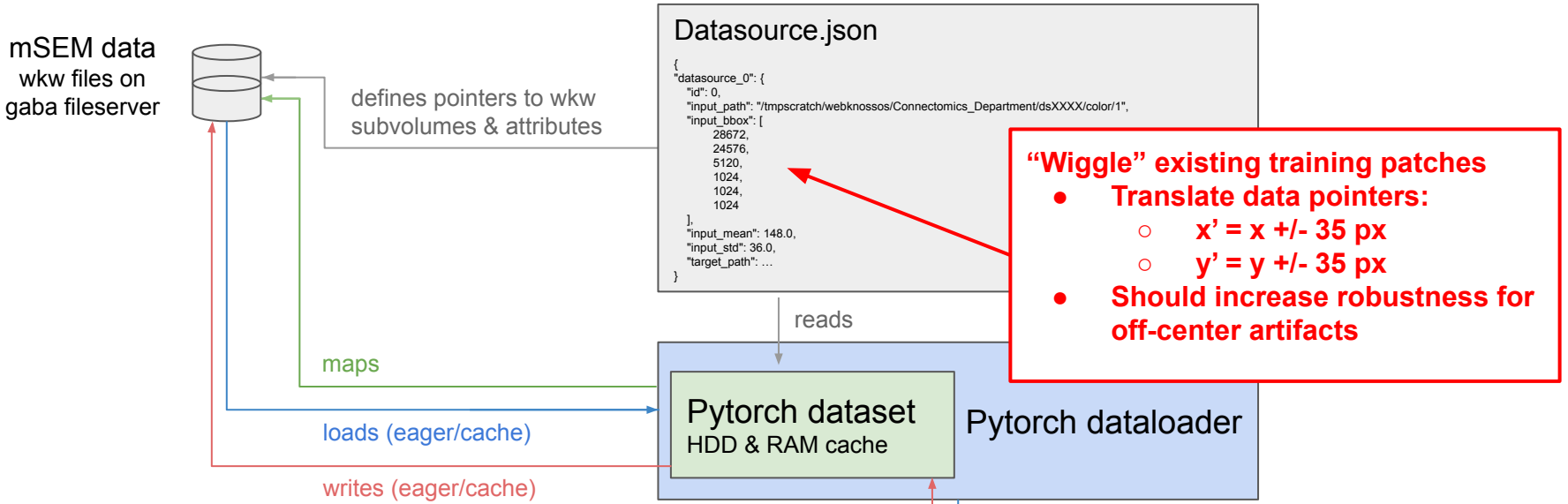
How to get pr curve up?

- Increase amount of training data
- Increase variation in training data / augmentation
- Increase complexity of classifier network

Interlude: Data architecture



Improving the model: Increase train data amount & variation



Dataset

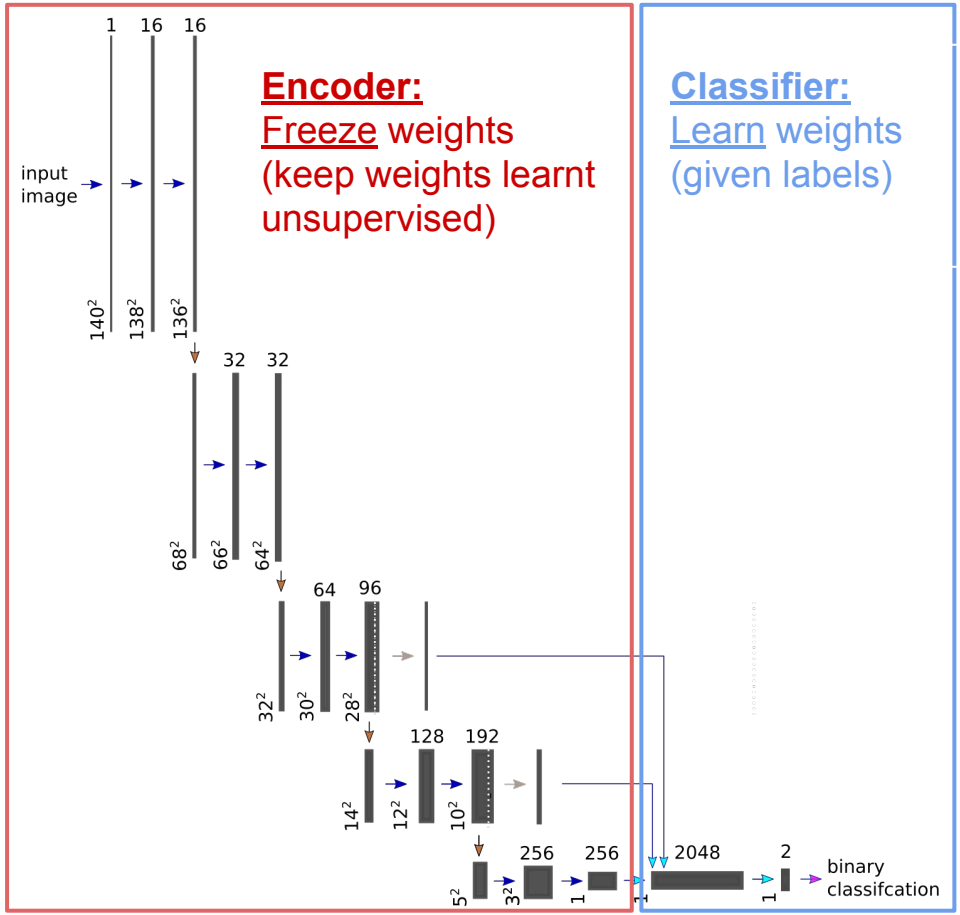
Total: 8562 (140x140x1) labelled samples

- Train: 7276 (2545 artifact, 4731 no artifact)
- Validation: 1284 (454 artifact, 829 no artifact)

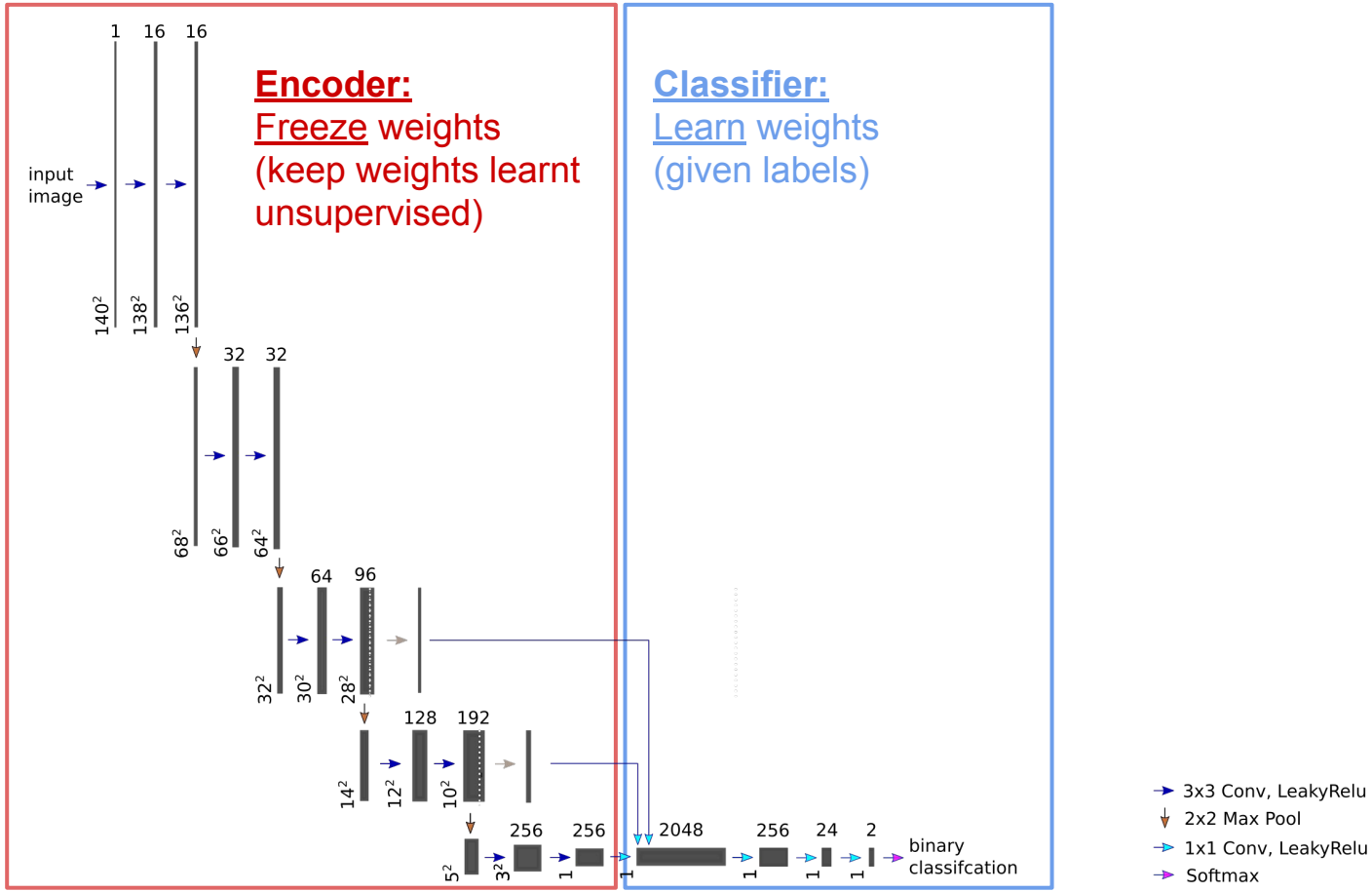
-> counts include wiggles

-> counts do not take into account random augmentations

Improving the model: previous classifier complexity

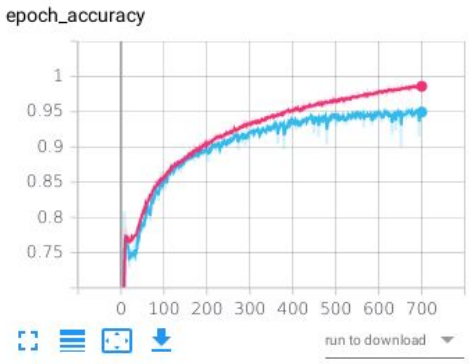


Improving the model: increased classifier complexity

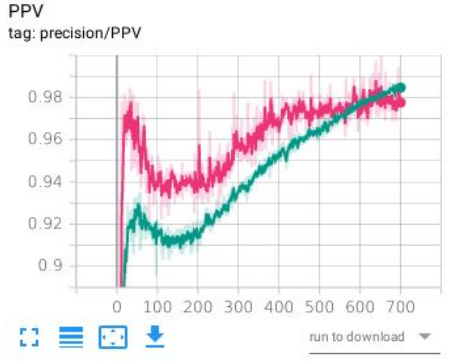


Improving the model: training performance

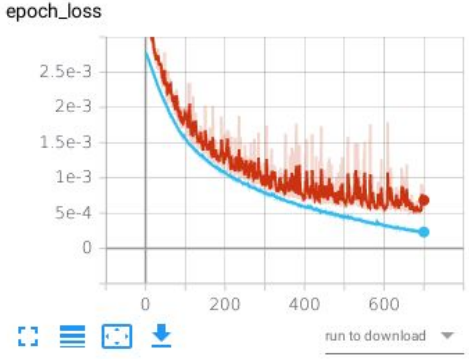
epoch_accuracy



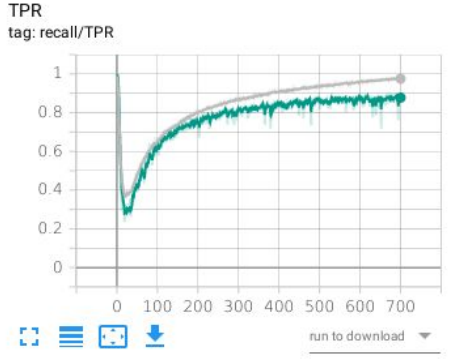
precision

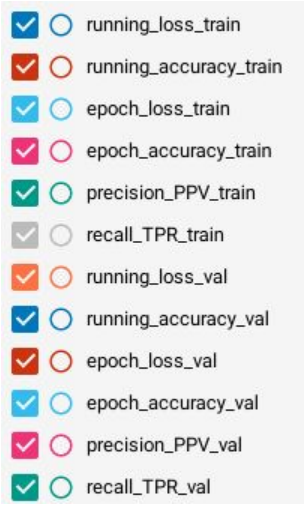


epoch_loss

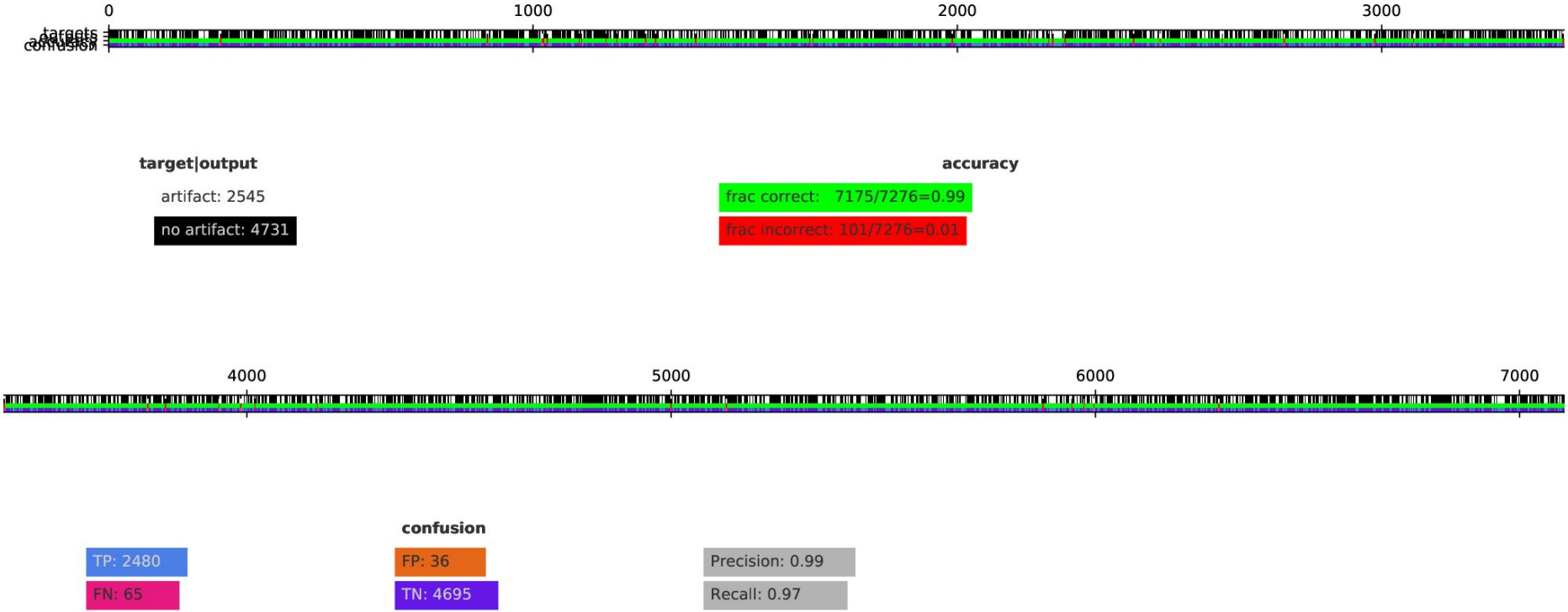


recall

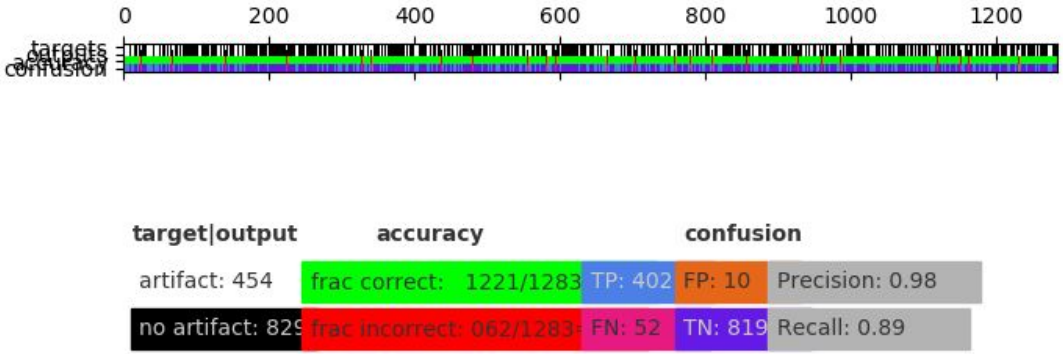




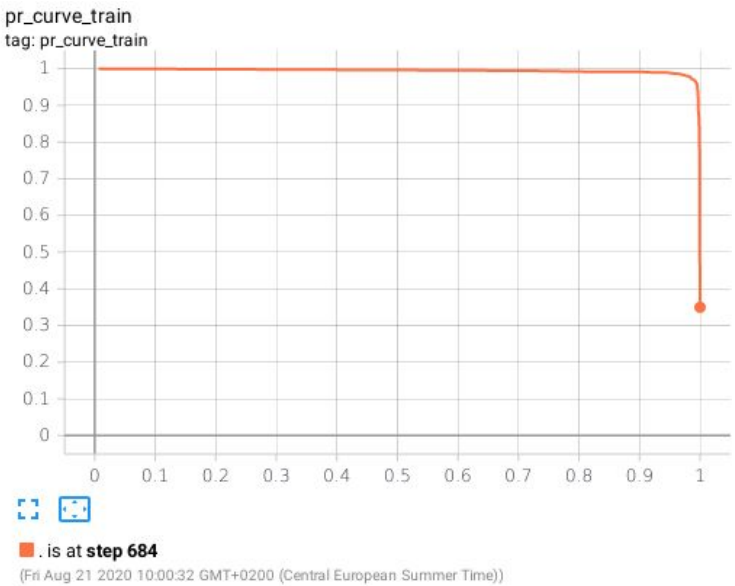
Improving the model: training performance (train set)

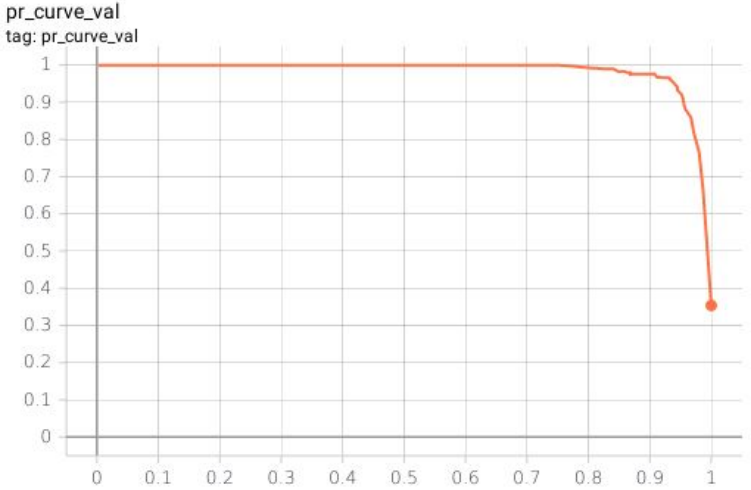


Improving the model: training performance (val set)



Improving the model: precision-recall curves





Inference on wk data

- Model input shape = (140, 140, 1), output_shape = (1, 1, 1)
  - predictions are sparse floats [0,1]
- Sparse floats can be written to compressed wkw files (20 mb instead of 4 gb), rendering in webknossos works
- Computational cost for predicting artifacts for a 1024^3 raw data cube with stride=(35,35,1)
  - cpu: ~ 60 min @ gabag; 16 cores
  - gpu: ~ 5 min @ gabag; tesla v100
- Mag8-8-1 of Meike's scMS109_1to7199 is comprised of ~ 3500 cubes:
  - cpu: 3500 * 60/60 * 16 = 65,000 core hours
  - gpu: 3500 * 5/60 = 290 gpu hours

Inference on wk data: examples

The good

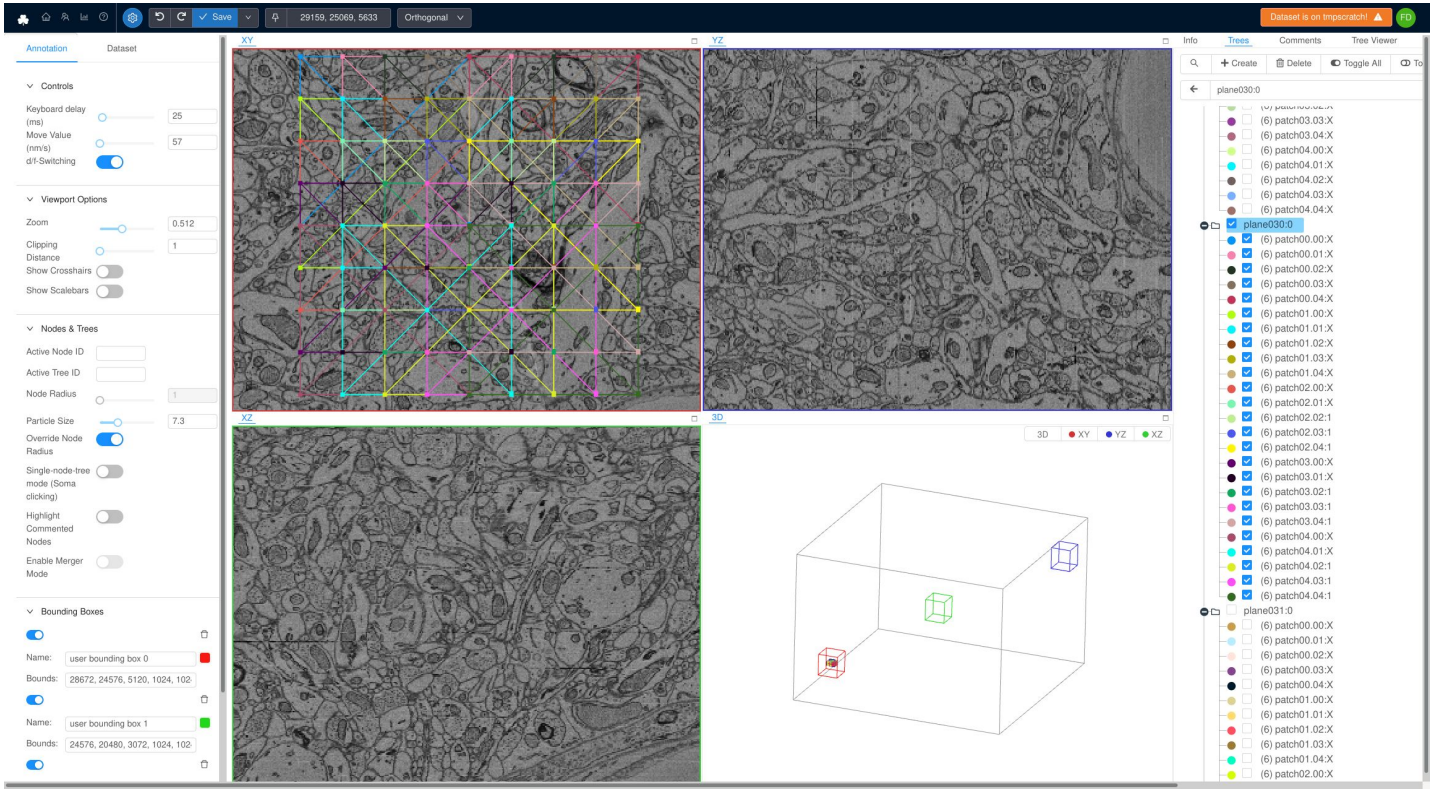
- Circular artifact
  https://webknossos.brain.mpg.de/datasets/Connectomics_Department/2018-11-13_scMS109_1to7199_v01_l4_06_24_fixed_mag8_artifact_pred/view#20952,15727,1336,0,0,467
- Fold
  https://webknossos.brain.mpg.de/datasets/Connectomics_Department/2018-11-13_scMS109_1to7199_v01_l4_06_24_fixed_mag8_artifact_pred/view#20725,15822,1362,0,0,467
- Horse shoe
  https://webknossos.brain.mpg.de/datasets/Connectomics_Department/2018-11-13_scMS109_1to7199_v01_l4_06_24_fixed_mag8_artifact_pred/view#20686,15980,1380,0,0,319

The bad & ugly

- Myelin (bad)
  https://webknossos.brain.mpg.de/datasets/Connectomics_Department/2018-11-13_scMS109_1to7199_v01_l4_06_24_fixed_mag8_artifact_pred/view#20795,15782,1389,0,0,263
- Blood vessel (bad)
  Not covered by current predictions but known from previous experience

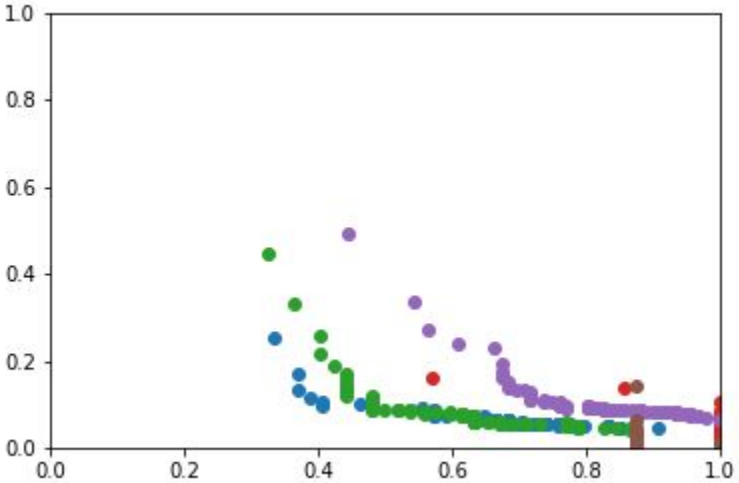
Inference on wk data: test data evaluation

- 3 distributed test boxes, each (10x10x2) um^3
- Used skeleton tracing for classification: Each prediction patch indicated by tree, all patch trees of one plane are part of one tree group
- -> comment hierarchy labels patches



Inference on wk data: test data evaluation

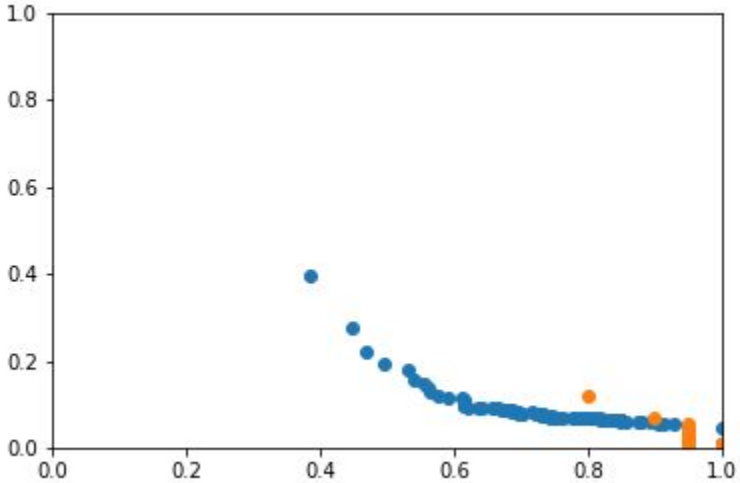
Pred test boxes separate



	condition_positive	condition_negative
predicted_positive	145	1784
predicted_negative	53	2293

Precision = tp/(tp+fp) = 145/(145+1784) = 0.10
Recall = tp/(tp+fn) = 145/(145+53) = 0.73

Pred test boxes combined

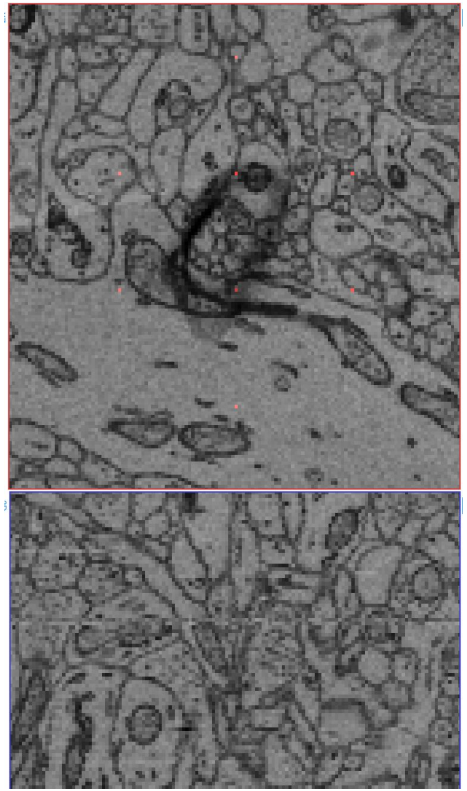


	condition_positive	condition_negative
predicted_positive	12	381
predicted_negative	3	3732

Precision = tp/(tp+fp) = 12/(12+381) = 0.03
Recall = tp/(tp+fn) = 12/(12+3) = 0.80

Inference on wk data: test data evaluation

- What we know about artifacts:
- Even small artifact spread (focally, xy-plane) over multiple prediction patches
- However, artifacts don't extend over multiple planes (axially). Instead, positive predictions extending over multiple planes might correspond to blood vessels, myelin etc.
- -> apply this geometrical prior to sparse predictions



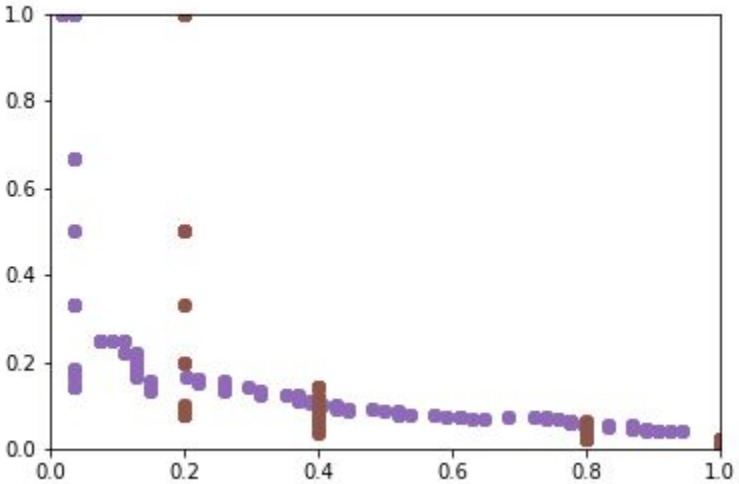
```
pred_filt = pred * kernel

kernel = array([
        [
            [-0.025, -0.05 , -0.025],
            [-0.05 ,   -0.2  , -0.05 ],
            [-0.025, -0.05 , -0.025]
        ],[
            [0.125,  0.25 ,  0.125],
            [ 0.25,    0.5  ,  0.25 ],
            [0.125,  0.25 ,  0.125]
        ],[
            [-0.025, -0.05 , -0.025],
            [-0.05 ,   -0.2  , -0.05 ],
            [-0.025, -0.05 , -0.025]
        ]])
```

- Problem: smears out bimodal probability distribution
- Instead: Morphological operations?

Inference on wk data: test data evaluation

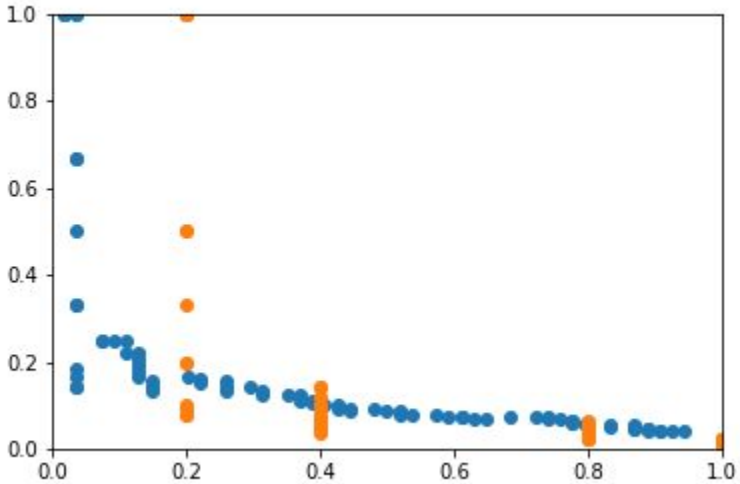
Pred filtered test boxes separate



	condition_positive	condition_negative
predicted_positive	51	381
predicted_negative	111	3732

Precision = tp/(tp+fp) = 51/(51+381) = 0.12
Recall = tp/(tp+fn) = 51/(51+111) = 0.31

Pred filtered test boxes combined



	condition_positive	condition_negative
predicted_positive	12	381
predicted_negative	3	3732

Precision = tp/(tp+fp) = 12/(12+381) = 0.03
Recall = tp/(tp+fn) = 12/(12+3) = 0.80

Inference on wk data: Todo

-> Explain extreme performance drop

- Validate by running inference on traindata
- If train metrics reproducible using inference code (model weight loading etc.):

<u>Problem could be real</u>

- Rooted in actual encoding + classification performance
  -> re-train network with much more artifact-free data

<u>Problem could be pseudo</u>

- Unexpected cause
  -> strange effect of batch norm
  -> inference bug
  -> metrics bug

Moritz Helmstaedter

Gilles Laurent
Johannes Letzkus

Sylvia Krauss-Fernando

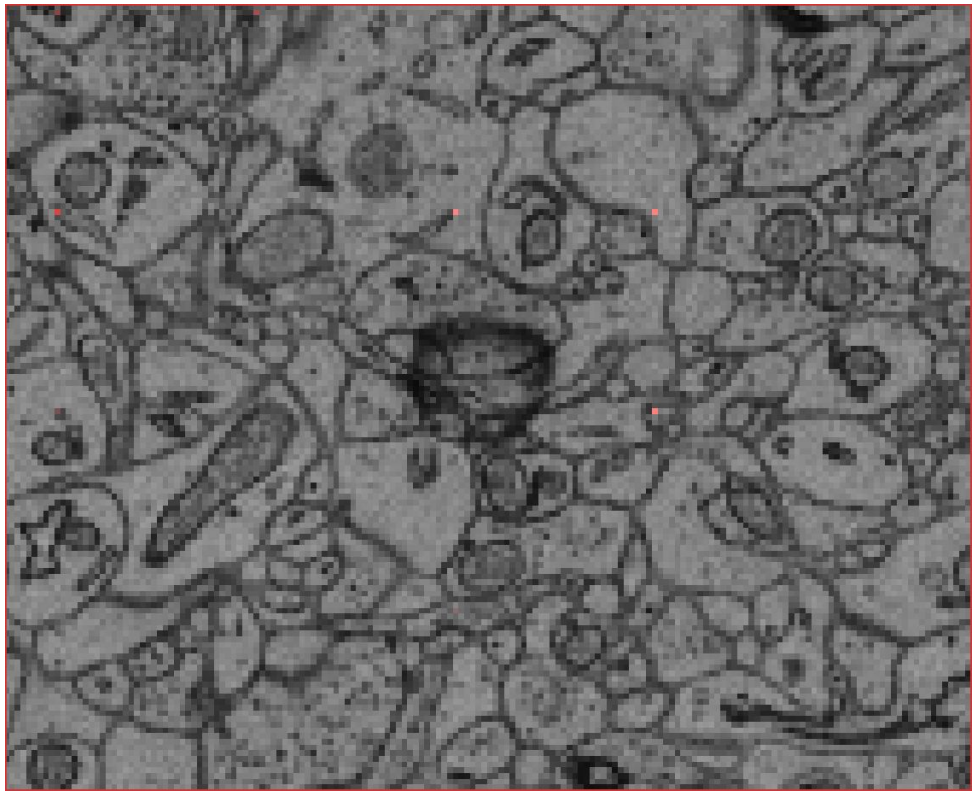
Iris Wolf
Lev Dadashev
Selina Horn
Smaro Soworka

Heiko Wissler

Thanks :)

Ali Karimi
Alessandro Motta
Meike Sievers
Jakob Strahle
Manuel Berning
Benedikt Staffler
Emmanuel Klinger
Martin Schmidt
Yagmur Yener
Sahil Loomba
Anjali Gour
Abdelrahman Khalifa
Helene Schmidt
Vijay Gangadharan
Kun Song
Zhihui Feng
Kevin Boergens
Yunfeng Hua
Philip Laserstein
Philip Bastians

Appendix: Qualitative observations prediction quality



Observation

Prediction peak not centered on artifact

Possible cause

Artifact class not represented robustly enough in traindata (e.g. represented with low n + off-center)

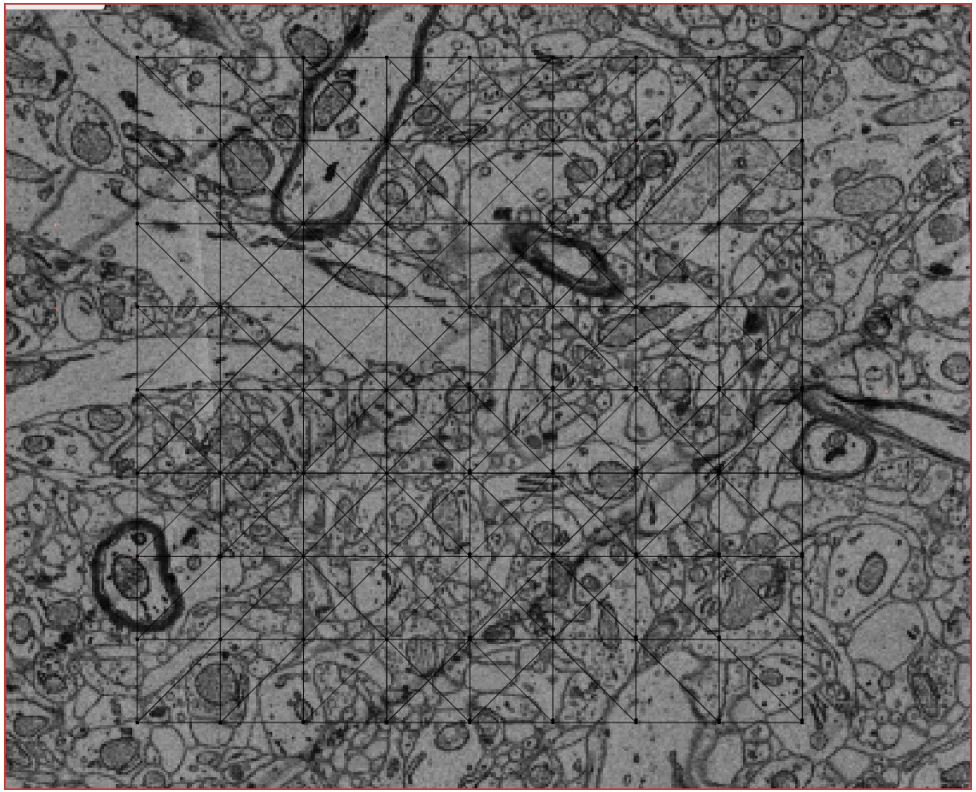
Possible alleviation

Increase availability in traindata

- Add more (e.g. selectively for certain artifact size ranges)
- Augment more (current wiggling with stride 35)

https://webknossos.brain.mpg.de/annotations/Explorational/5f5622a401000010004af1c6#29133,25108,5630,0,0,200,8034

Appendix: Ground truth for raw data fringe cases



Artifact or no artifact?

Equidistant dark stripes

https://webknossos.brain.mpg.de/annotations/Explorational/5f569d4001000050004afa25#20970,15849,1540,0,0,360

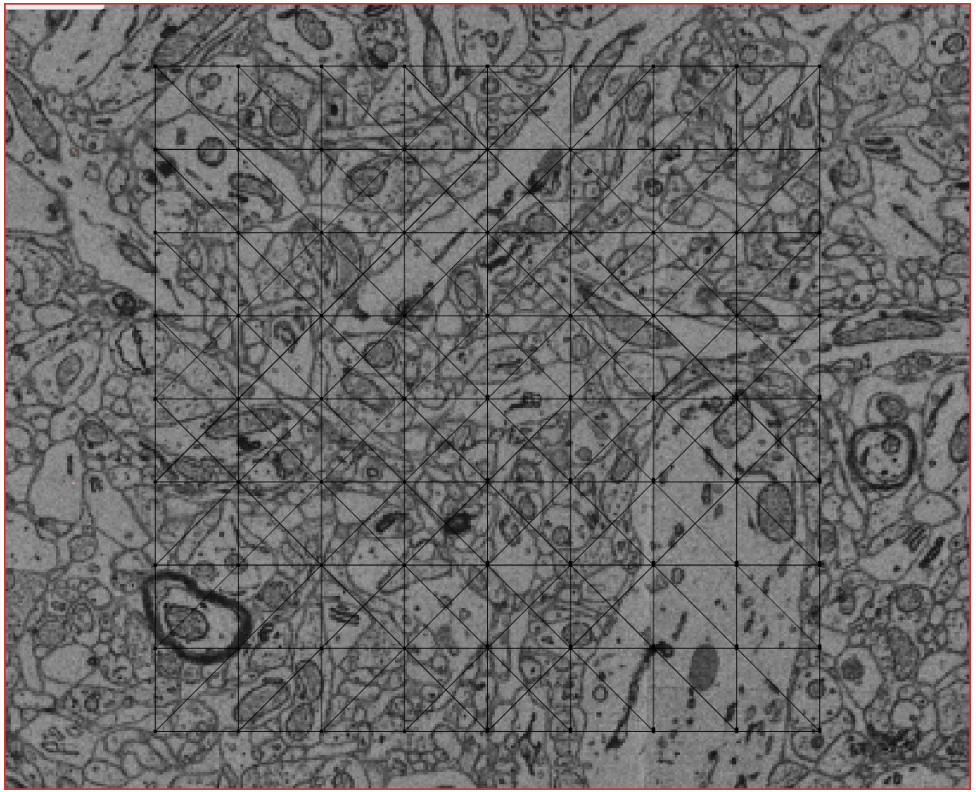
Appendix: Ground truth for raw data fringe cases

Artifact or no artifact?

Brightness variation

https://webknossos.brain.mpg.de/annotations/Explorational/5f569d4001000050004afa25#20970,15849,1540,0,0,360

Appendix: Ground truth for raw data fringe cases



Artifact or no artifact?

Out of focus

https://webknossos.brain.mpg.de/annotations/Explorational/5f569d4001000050004afa25#20970,15849,1552,0,0.360