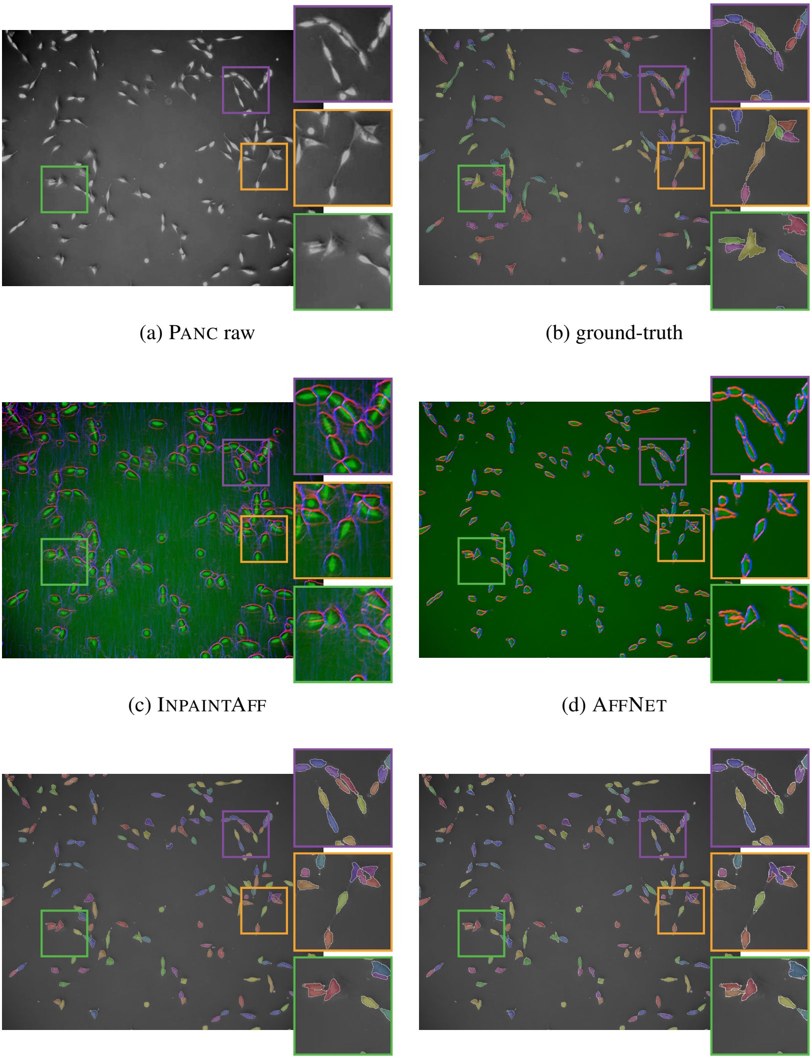
A Instance Segmentation from Foreground Prediction



(a) Panc raw

(b) ground-truth

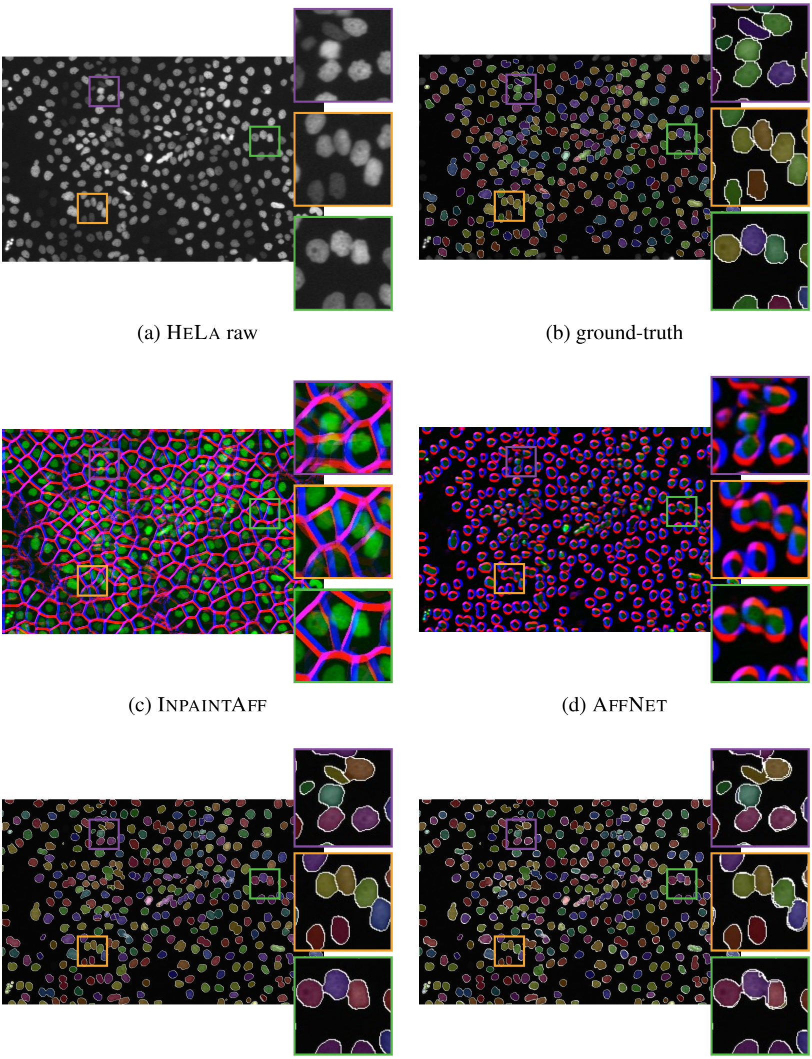
(c) InpaintAff

(d) AffNet

(e) InpaintAff + FgNet50 segmentation

(f) AffNet + FgNet50 segmentation

Figure 5: Sample test images of Panc. Affinities (middle column) are shown as blue/red for x-/y-direction, respectively.



(a) HeLa raw

(b) ground-truth

(c) InpaintAff

(d) AffNet

(e) InpaintAff + FgNet50 segmentation

(f) AffNet + FgNet50 segmentation

Figure 6: Sample test images of HeLa. Affinities (middle column) are shown as blue/red for x-/y-direction, respectively.

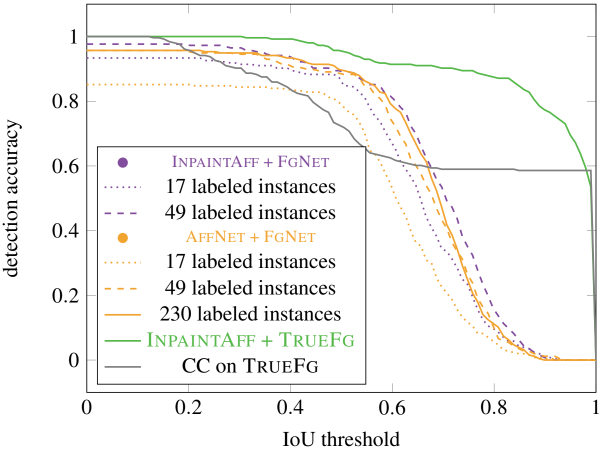


Figure 7: Detection accuracy over different IoU thresholds on PANC. Over a large range of IoU thresholds, INPAINTAFF in combination with a foreground network FGNET trained on 49 labeled instances has a higher detection accuracy then the fully supervised method AFFNET trained on 230 labeled instances.

Model Architectures For the inpainting network underlying INPAINTAFF, we use a downscaled version of the architecture proposed by Liu et al. [19], *i.e.*, a U-NET architecture with a depth of four resulting in five levels with 64, 128, 256, 512, and 512 feature maps, each. We train the network for 1M iterations using the ADAM optimizer and the loss proposed by Liu et al. [19] that is comprised of a perceptual, style, total variation and reconstruction loss.

FGNET is a PIX2PIX network [10, 38] with a depth of six layers, containing 64 initial features maps, trained using ADAM to minimize a binary cross-entropy loss [12].

Since we use the MUTEXWATERSHED to post-process affinity predictions, we use the same training procedure proposed by Wolf et al. [2018] for AFFNET (PIX2PIX architecture). In particular, we also use the Sørensen-Dice coefficient [7, 25] loss and the same affinity neighborhood (12 distances, up to 27 pixels).

B Neighborhood Selection and Inference:

As discussed, the updates of the equation (11) can be limited to N, a set of pixels close to the boundary of M and $\overline{M}$. Formally, let $\mathrm{FOV}(i,d)$ be the set of all pixels closer to pixel i than the max distance d. Then

$$N(M,d) = \Big\{ i \in \Omega \;\Big|\; \mathrm{FOV}(i,d) \cap \overline{M} \neq \emptyset \text{ and} \tag{7}$$

$$\mathrm{FOV}(i,d) \cap M \neq \emptyset \Big\} \tag{8}$$

We find empirically that decreasing d over time aids the regions to converge. In our experiments we use a constant d for the first half of updates and then decrease it linearly. Additionally, we find that smoother boundaries can be achieved by interleaving updates with

$d = 1$ every second iteration and smoothing the reconstruction error over neighboring pixels. For the smoothing we convolve the reconstruction error with a gaussian kernels of $\sigma \in [0.1, 1, 5, 10]$ and add them to the pixelswise reconstruction error.

C Train/Test Split of CTC

Each dataset of the Cell Tracking Challenge contrains two training (labeled *t01* and *t02*) and two test videos. Since our inference method requires a considerable amount of computational resources a direct evaluation on the CTC servers on the official testing data is not possible. Therefore, we split the publicly available data for each dataset into a train and testing dataset.

For the PANC *(PhC-C2DL-PSC)* dataset we train on frame 182 of video *t02*, validate on frame 25 of *t02* and test on frames $[98, 122]$ of video *t01*. This uses all 4 available labeled frames of the dataset.

For the HELA *(Fluo-N2DL-HeLa)* dataset we train on frames $[13, 52]$ of video *t01*, validate on frame 76 of *t01* and test on all (even partially labeled) frames $[23, 35, 36, 67, 75, 78, 79, 87]$ of video *t02*.

The trainings sets with reduced number of instances were generated by first, using a random subset of labeled frames and then cropping the training images spatially. We alternate between halving the image size in x and y-direction, taking away from both sides thus keeping the center constant.

D Inpainting masks

We adapt the inpainting network training procedure of Liu et al. [19], but use a different mask generation method. In every training iteration, we generate new random masks by, first sampling a uniform noise image with shape (d,d), where d is drawn randomly from $\{2, 4, 16, 32\}$. This noise image is bicubically upsampled to the full patch dimension and thresholded at 0.5 to yield a binary patch mask.

E Affinity-Based Segmentation

We derive a segmentation from affinities aff using the MUTEXWATERSHED on a XY-plane neighborhood graph with local attractive edges $[-1, 0], [0, -1]$ and sparse repulsive edges: $[-9, 0], [0, -9], [-9, -9]$, $[9, -9], [-9, -4], [-4, -9], [4, -9], [9, -4], [-27, 0], [0, -27]$. The graph weights for the local attractive edges are equivalent to the affinities and the repulsive edges costs are the α-weighted inverted affinities $\alpha(1 - \text{aff})$.