Supplemental Material:
How To Make Your Cell Tracker Say: ”I dunno!”

A. Linear Assignment Formulation

Given two consecutive frames $\mathcal{X}_t$ and $\mathcal{X}_{t'}$, there are three possible associations for the cells in those two frames: A daughter cell $x' \in \mathcal{X}_{t'}$ is linked to a mother cell $x \in \mathcal{X}_t$, a daughter cell remains without a mother or a mother cell remains without a daughter. The latter two cases are relevant *e.g.* if cells enter or leave the region of interest from across the frame border. Technically, we we can consider the cases, were mother or daughter cells are not associated to any other cell as an association to some fallback class $\perp$ and denote the likelihood of a single cell appearing or disappearing as $\mathbb{P}(x'_j \mid \perp)$ and $\mathbb{P}(\perp \mid x_i)$, respectively. Given those probabilities, we denote the joint likelihood of observing detections $\mathcal{X}_{t'}$ given $\mathcal{X}_t$ and some assignment solution A as

$$\mathbb{P}(\mathcal{X}_{t'} \mid \mathcal{X}_t, A) = \prod_{(x_i, x'_j) \in A} \mathbb{P}(x'_j \mid x_i) \prod_{x_i \notin A} \mathbb{P}(\perp \mid x_i) \prod_{x'_j \notin A} \mathbb{P}(x'_j \mid \perp), \tag{22}$$

where any cell not included in the assignment A is either an appearing daughter or disappearing mother. Defining $w_a := \log \mathbb{P}(x'_j \mid \perp)$, $w_d := \log \mathbb{P}(\perp \mid x_i)$ and m, n as the number of appearing and disappearing cells respectively, we obtain the linear assignment formulation from Equation (5) by taking the logarithm of our likelihood from Equation (22).

B. Tracking Cost Functions

The tracking algorithms used in this work are mainly distinguished by the cost function w, which they define. Here, we give the remaining two cost functions, which were not introduced in the main paper, as well as some details on the Transformer-based cost function and elaborate the connections between distance- and activity-based tracking and temperature scaling.

- In distance-based tracking, the cellular features are their positions in the image and the cost function is computed as

$$w_{\text{L2}}(x_i, x'_j) = \frac{\lambda}{2} \| \, x_i - x'_j \, \|_2^2, \tag{23}$$

  which is equivalent to assuming Brownian motion with variance λ^{-1}, *i.e.* $x'_j \sim \mathcal{N}(x_i, \lambda^{-1} I)$.
- The activity-based tracking is similar to the distance-based tracking, but introduces an additional activity value α_i for each mother cell x_i, which scales the variance of the Gaussian likelihood $x'_j \sim \mathcal{N}(x_i, \alpha_i \lambda^{-1} I)$

$$w_{\text{AC}}(x_i, x'_j) = \frac{\lambda}{2\alpha_i} \| \, x_i - x'_j \, \|_2^2. \tag{24}$$

  The activity value is computed from the raw image values within the segmentation mask of x_i. For details, please refer to the original publication [32].
- In the overlap-based tracking, the cellular features are their segmentation masks and the cost function is computed as the negative number of pixels in the overlap, *i.e.*

$$w_{\text{OL}}(x_i, x'_j) = -| \, x_i \cap x'_j \, | \tag{25}$$

- The Transformer-based tracking algorithm *Trackastra* [12] defines a frame-to-frame cost function, but incorporates detections from multiple frames in a sliding window fashion. Given a sliding window of size $\Delta \in \mathbb{N}$, *Trackastra* encodes *shallow* input features $x_i^{(t+\delta)} \in \mathcal{X}_{t+\delta}$ with $\delta = 0, \dots, \Delta$ using two different functions f_θ, g_θ implemented as Transformer neural networks. Given a set of cellular features $\mathcal{X}$, f_θ and g_θ map those features to latent spaces $\mathcal{Y} = f_\theta(\mathcal{X})$ and $\mathcal{Z} = g_\theta(\mathcal{X})$, such that $\mathcal{Y}_i, \mathcal{Z}_i$ are the latent representations corresponding to a particular input feature $\mathcal{X}_i \in \mathcal{X}$. Shallow features used here are position, shape descriptors and image intensities. Frame-to-frame costs are then computed as

$$w(x_i, x'_j) = \frac{1}{2} \|\mathcal{Y}_i - \mathcal{Z}'_j\|_2^2 - \frac{1}{2} \left(\|\mathcal{Y}_i\|_2^2 - \|\mathcal{Z}'_j\|_2^2 \right) = -\mathcal{Y}_i^\top \mathcal{Z}'_j, \tag{26}$$

  where $\mathcal{Y}_i, \mathcal{Z}'_j$ are the latent representations of x_i, x'_j from consecutive frames, respectively, and $\mathcal{Y}, \mathcal{Z}$ were computed on the union of all cellular features $\mathcal{X} = \bigcup_{\delta=0}^{\Delta} \mathcal{X}_{t+\delta}$ from the sliding window.

B.1. Brownian Motion & Temperature Scaling

To show that temperature scaling effectively scales the variance of the Brownian motion assumed by the distance- and activity-based tracking algorithm, we consider Equation (20), plug in Equation (15) and absorb the temperature into a scaled cost function:

$$\begin{aligned} P_{i|j\tau} &= \frac{P_{ij}^{\tau}}{\sum_k P_{kj}^{\tau}} && (27) \\ &= \frac{\exp\{-\tau w(x_i, x'_j)\}}{\sum_{x_k \in \mathcal{X}_t} \exp\{-\tau w(x_k, x'_j)\}} && (28) \\ &= \frac{\exp\{-\tilde{w}(x_i, x'_j)\}}{\sum_{x_k \in \mathcal{X}_t} \exp\{-\tilde{w}(x_k, x'_j)\}}, && (29) \end{aligned}$$

where we define $\tilde{w}(x_i, x'_j) := \tau w(x_i, x'_k)$. Now setting $w = w_{\text{L2}}$, we see that $\tilde{w}(x_i, x'_j) = \frac{\tau\lambda}{2} \| x_i - x'_j \|_2^2$, which is equivalent to assuming $x'_j \sim \mathcal{N}(x_i, (\tau\lambda)^{-1} I)$. Similarly, for $w = w_{\text{AC}}$, we get that $\tilde{w}(x_i, x'_j) = \frac{\tau\lambda}{2\alpha_i} \| x_i - x'_j \|_2^2$, which is equivalent to assuming $x'_j \sim \mathcal{N}(x_i, \alpha_i(\tau\lambda)^{-1} I)$. Finally, if we set $w = \| f_\theta(x_i) - f_\theta(x_j) \|_p^p$, where $\| \cdot \|_p$ is the p-norm and f_θ some function projecting our cell features into some latent space, then this is equivalent to assuming $f_\theta(x'_j) \sim \mathcal{GN}(f_\theta(x_i), \tau^{-1} I, p)$, where $\mathcal{GN}(\mu, \alpha, \beta)$ is the generalized normal distribution with location μ, scale α and shape β.

C. Parental Softmax

A practical issue that might arise from the DBMC approach is caused by appearing cells, that *e.g.* cross the image border. In this case, the true probability for any detection x_i from frame t to be the mother of the appearing cell x'_j should be zero. However, the plain softmax distribution from Equation (15) cannot capture this case, as the denominator would become 0, if the tracking algorithm were to correctly yield edge-wise probabilities $\exp\{w(x_i, x'_j)\} = 0$. Thus, Gallusser and Weigert [12] introduce the *parental softmax*

$$\mathbb{P}((x_i, x'_j) \in A \,|\, x'_j, \mathcal{X}_t, \mathcal{X}_{t'}) = \frac{\exp\{w(x_i, x'_j)\}}{1 + \sum_{x \in \mathcal{X}_t} \exp\{w(x, x'_j)\}}. \tag{30}$$

by adding a constant to the denominator. From a statistical perspective, this might seem like an issue, as the parental softmax distribution does not sum up to 1 anymore. In fact, however, this constant summand represents an unnormalized probability for the implicit class $\perp$ representing the lack of an adequate mother. The normalized probability of this choice can be computed as

$$\mathbb{P}((\perp, x'_j) \in A \,|\, x'_j, \mathcal{X}_t, \mathcal{X}_{t'}) = 1 - \sum_{x \in \mathcal{X}_t} \mathbb{P}((x, x'_j) \in A \,|\, x'_j, \mathcal{X}_t, \mathcal{X}_{t'}). \tag{31}$$

D. Bayesian Tracking Transformer

For the Bayesian Transformer-based tracking we trained *Trackastra* [12] using MC Dropout [11] at varying dropout probabilities $p \in \{0.5, 0.25, 0.125\}$. In Figure 8, we present the ECE and sparsification as before in Section 4.1 & 4.2, but across the tested dropout rates.

Our results indicate no clear trend, favoring neither lower nor higher dropout probabilities. For the ECE, especially FP 🟩 and FP+A 🟩 show only slight variance in performance depending on the dropout probability. For sparsification, some variance is noticeable, though only the FP 🟩 method using entropy-based uncertainty estimation indicates a trend favoring higher dropout probabilities. For the temperature scaled versions (FP+TS 🟧 & FP+A+TS 🟧) ECE seems to be improved by a lower dropout probability, however an inverted trend is visible for sparsification, where high dropout probabilities achieve higher accuracy improvements.

Given the recent advances in Bayesian deep learning, applying more sophisticated techniques like Laplace approximations [24, 30] or variational inference [35] might further improve the performance of our Bayesian neural perturbation method.

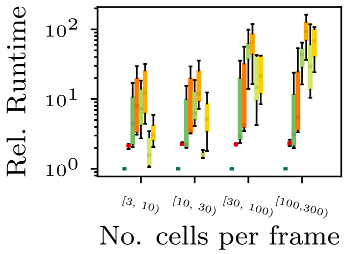

Figure 7. Relative runtime of our different methods (*cf*. Table 1) compared to the SM method as a function of the average number of cells per frame. The SM method has virtually the same costs as vanilla uncertainty-unaware tracking. Legend is the same as for Figure 10.

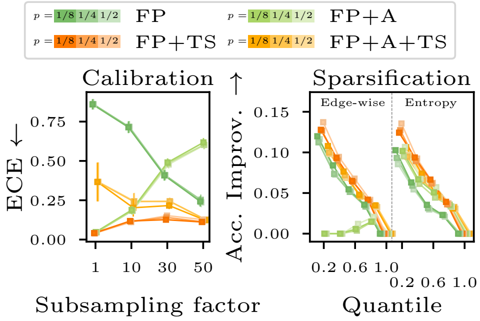

Figure 8. Calibration and sparsification as discussed in Section 4.1 and 4.2 shown for the Transformer-based tracking using Monte Carlo dropout at varying dropout rates.

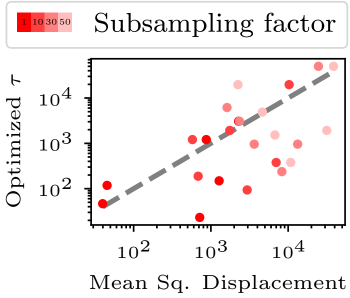

Figure 9. Optimized temperature τ of our SM+TS 🟥 method plotted against the empirical mean squared displacement of cells based on the ground truth tracking for each. The dashed line depicts 1-to-1 correspondence.

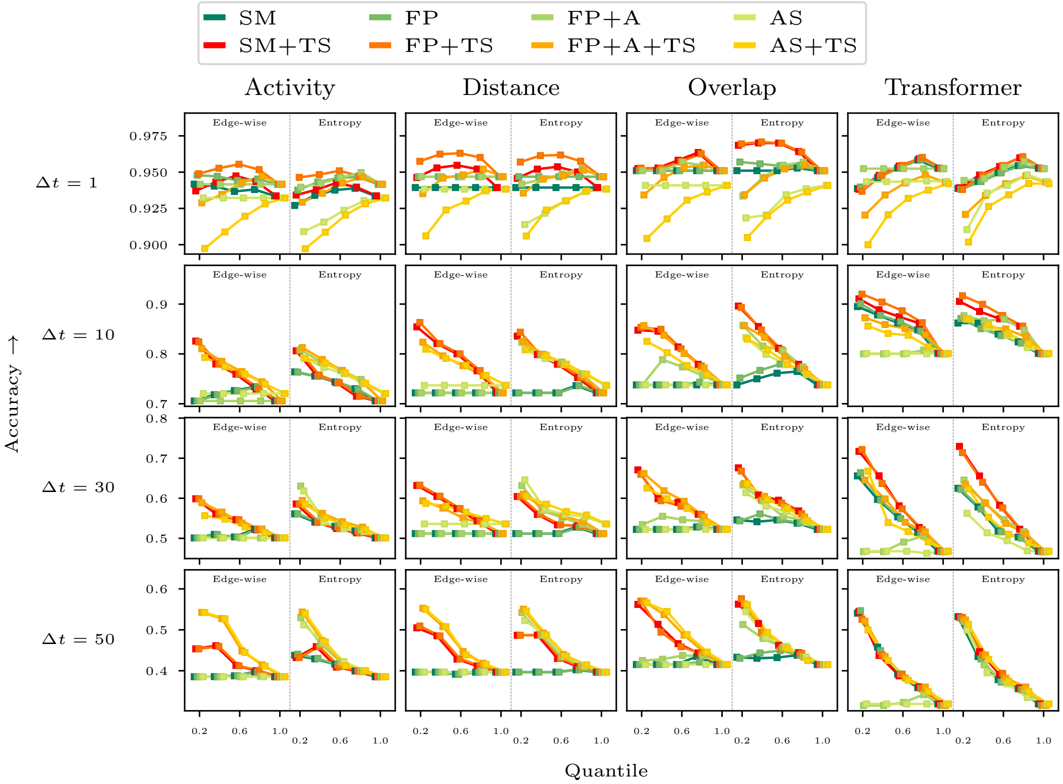

Figure 10. Mean accuracy of tracking predictions per temporal resolution, sparsified at varying thresholds, which were computed as quantiles over the respective sparsification criterion, *i.e*. either the *edge-wise* probability or the daughter-wise *entropy*.

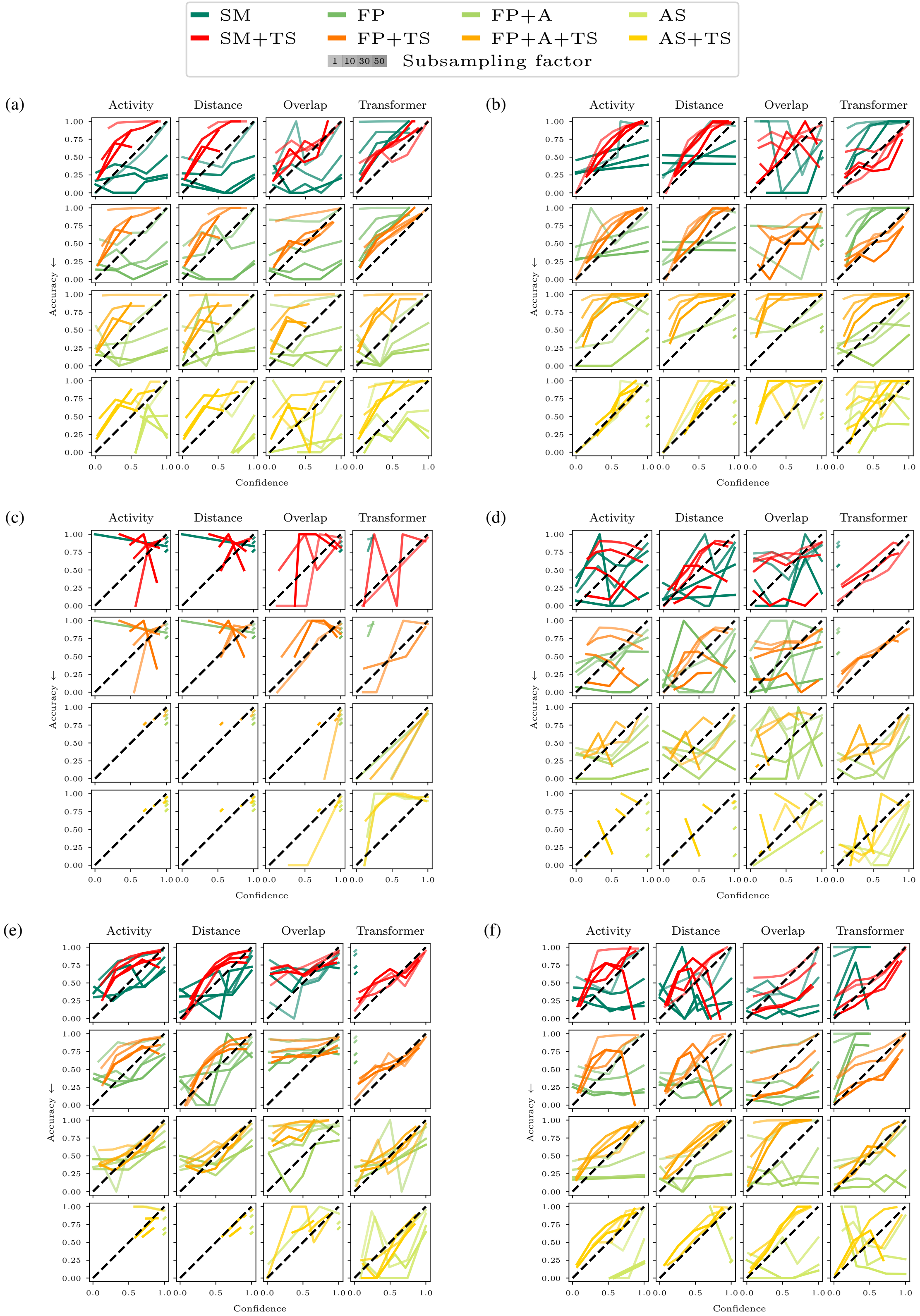

Figure 11. Calibration curves of all tested methods and datasets. Datasets are (a) BF-C2DL-HSC, (b) BF-C2DL-MuSC, (c) DIC-C2DH-HeLa, (d) Fluo-N2DL-HeLa, (e) PhC-C2DL-PSC all taken from the Cell Tracking Challenge [25, 26] and, (f) Tracking-One-in-a-Million dataset [34]. The gray dashed line depicts 1-to-1 correspondence, *i.e.* perfect calibration.

Table 2. Broad-brush overview of pros and cons of our methods for probabilistic cell tracking: Softmax'ing (SM), Feature Perturbation (FP), Feature Perturbation with Assignment (FP+A) and Assignment Sampling (AS). We compare our methods in terms of calibration (*cf*. Figure 5), 'usefulness' of the uncertainty estimates (*cf*. Figure 6), runtime overhead (*cf*. Figure 7) as well as technical considerations.

Method	Pros	Cons
SM 🟩🟥	• Calibrated and useful uncertainty if using TS • Cheap • Almost no modifications required	• Inexpressive uncertainty without TS
FP 🟩🟧	• Calibrated and useful uncertainty if using TS	• Inexpressive uncertainty without TS • Moderately expensive • Requires noise distribution to perturb features
FP+A 🟩🟧	• Useful uncertainty even without TS	• Expensive • Requires noise distribution to perturb features
AS 🟩🟨	• Useful uncertainty even without TS	• Moderately expensive
TS 🟥🟧🟨	• Calibrated uncertainties	• Requires annotated data