Droplet Tracking for Personalised Cancer Treatment

Weronika Ormaniec
ETH Zürich
wormaniec@ethz.ch

Michael Vollenweider
ETH Zürich
michavol@ethz.ch

Richard Danis
ETH Zürich
rdanis@ethz.ch

Sven Gutjahr
ETH Zürich
sgutjahr@ethz.ch

Tiago Hungerland
ETH Zürich
thungerland@ethz.ch

Abstract

Microfluidic droplet assays enable single cell studies that are pioneering methods for personalised Medicine in cancer treatments. However, these time-resolved droplet assays currently suffer from the limitation that droplet tracking over time is often not implemented, due to the massive number of droplets involved in immunoassays, leading to the loss of valuable data points. If successfully designed, such tracking algorithms would provide a new and progressive tool for the development of personalised Medicine, specifically personalised cancer treatments. In this report we propose a first such algorithm which is data-driven but unlike other data-driven approaches, is not strongly based on labelled data. Instead we design a pipeline which connects the necessary building blocks for an optimal transport based droplet tracking. The pipeline we present consists of a pre-processing step, a feature extraction step to obtain a positional, visual, and number of cells embedding for each droplet, and finally a matching step which uses optimal transport to match droplets across temporally separated image frames. To perform reliable visual feature extraction we propose a heuristic approach to automatically label some of the data ourselves. Using these labels, we successfully train a contrastive learning model. We evaluate our pipeline heuristically on real data and quantitatively on simulated data. First, we provide a visual tool to display predicted trajectories with their respective certainty levels. In particular, high uncertainties are visually confirmed when looking at the tracked droplets. Second, we discuss visual embedding results and investigate which visual features the model is capturing. Third, we illustrate the importance of calibrating our model. And finally, we provide results of the overall tracking of our pipeline applied to 20×10^3 droplets across 4 time frames and confirm a precision of above 80% for small, medium, and large movement experiments.

1 Introduction

Droplet-based microfluidics is a core aspect of many advances in modern Medicine and in particular act as a central tool for the development of personalised Medicine. One example, which this report themes, is the use of droplet microfluidics for the development of personalised cancer treatments. To develop such personalised treatment methods, very large-scale experiments are required since the effectiveness of a variety of potential drugs must be tested on different tumour types across many patients. The reason for this is two-fold: First, there is a considerable level of genomic heterogeneity across different patients which possibly renders a drug effective for some patients but ineffective for others [1]. Second, there also exists genomic heterogeneity across tumour cells within the same tumour [2]. The scale of droplet experiments, to address this challenge, can thus be in the order of

Data Science Lab 2023 Final Report.

10^6 droplets per experiment and sheds light on the computational challenges that arise in the analysis of such experiments. In particular, the analysis of such experiments requires the tracking of droplets under a microscope over time. To obtain a usable microscopic resolution, there is a trade-off between the temporal resolution of the images and the number of droplets that can be investigated in a single experiment. The high stakes of developing personalised cancer treatment rapidly make the large-scale analysis of droplet experiments a priority, thus worsening the temporal resolution of experiments. The temporal resolution refers to the time between two consecutive images of droplets under the microscope and the lower this resolution is, the longer are the time intervals over which droplets must be tracked. Tracking droplets through time requires matching one droplet to its positive match from one frame (a microscopic recording at one time point) to the next consecutive frame, thus completing the droplet's trajectory through space and time. When looked at under the microscope, a droplet provides two types of information, namely its spatial (pixel) position in the image as well as visual information, i.e. how the droplet looks. The droplets used to investigate the effects of cancer drugs are produced as large immunoassays consisting of water-in-oil droplets encapsulating a tumour cell as well as paramagnetic nanoparticles. These are coated in chemicals which are bound by target antibodies which the cells secrete upon effective treatment ([3]). Since the number of nanoparticles encapsulated by a droplet is variable during the production process, the so-called beadline which forms by applying a magnetic field across the immunoassay differs visually across droplets. Moreover, the number of cells encapsulated during the generation process may also differ. Both these visual aspects are computationally relevant information when designing a tracking algorithm. Following this motivation, the key challenges to be tackled by a successful droplet tracking algorithm can be summarised as follows:

- **Scale of the problem:** The number of droplets in a single experiment can be in the order of 10^6 droplets. This means that the algorithm must be able to scale to a large number of objects, while using a feasible amount of computational resources with respect to the computation time and memory.
- **Temporal resolution:** The temporal resolution of the microscope is low with images being taken in 30-minute intervals. This means that the algorithm must be able to track droplets over long time intervals.
- **Labelled data:** To label droplet trajectories, we must first successfully track droplets. This means that the algorithm must be able to track droplets without the use of labelled data.

2 Data

The challenge presented to us involves a data set of seven videos, each consisting of 8 or 9 frames (time points) with each frame possessing dimensions of approximately 4500x4500 pixels. It is essential to highlight that, for real-world applications, the images are of a larger scale, and the number of frames per video could exceed the current parameters. The temporal interval between consecutive frames is about 30 minutes. However, it is imperative to note that this temporal resolution may also change. The videos were acquired and provided to us by our challenge giver.

In Figure 1 we can see all frames of one video. It is clearly visible that this does not look like a continuous video but rather like individual consecutive but temporally separated frames. Accordingly, we will treat the video as individual frames. Additionally, in Figure 1 one can nicely see about $8'500$ droplets. There can be up to 10^6 droplets on experimental slides which scientists in the lab of our challenge giver generally use. One of the challenges for our tracking algorithm is that the number of droplets is not fixed and can change slightly across different frames. This is what we will later be referring to as the unbalancedness of the problem. This is due to two reasons: The boundaries of the experimental slides are not captured by the camera and our proposed algorithm will cut the image for one frame into smaller pieces to reduce the computational complexity. Moreover, in Figure 1 it is visible that droplets move across the slide significantly. The origin of droplet movement is unclear but we assume it is caused by the microscope itself and possibly by external factors in the surrounding environment (e.g., researchers walking past the microscope leading to oscillations of the setup). The large number of droplets and their unpredictable movement renders manual tracking intractable.

The camera used in the lab can capture up to 5 different channels: DAPI, FITC, TRITC, Cy5 and a bright field (BF) channel. For this project, we only use the DAPI and BF channels. The DAPI channel is used to detect the fluorescence of the experimentally stained DNA inside cells, which allows for

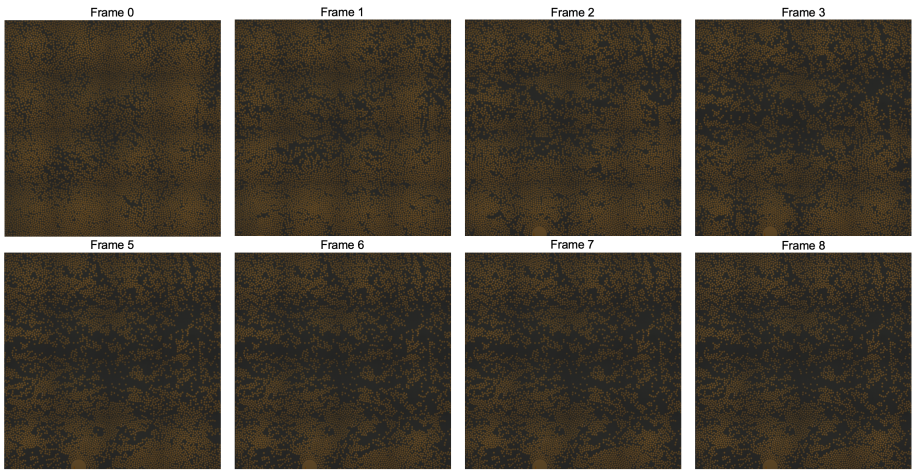


Figure 1: All frames of one video. Large movement.

checking the presence or absence of cells inside droplets. The BF channel contains information about the visual features of the droplets themselves. We don't use the other channels, since these are supposed to change and vary over time.

We identified different kinds of movements. Firstly, in Figure 1 we can recognise a repulsive behaviour, meaning that the droplets diffuse away from a divergence point. In other videos, we observe an attractive behaviour whereby the droplets contract towards a point in space. Due to the rigid nature of microfluidic droplets, there is no overlap of droplets. We also observe that there are droplets which can fuse even though this happens rarely. Finally, what is more frequently observed is one massive droplet,probably formed by many droplets fusing in the initial setup of the immunoassay, which moves through the assay and thereby causes large movement of smaller adjacent droplets. Since we do not have any labelled data to validate our algorithm, we created simulations that aim to mimic the observed movement patterns. This makes it possible to quantitative the performance of our method. Without simulated ground-truth data, we would be limited to qualitative visual investigation of predicted trajectories and time-consuming manual labelling by experts. This is further discussed in section 4 extensively.

2.1 Preprocessing

Before feeding the data into our tracking algorithm, we must preprocess it in two different ways. The first is to ensure good droplet detection and the the second is for engendering effective extraction of visual features. Last year, another Data Science Lab team implemented many of the necessary preprocessing steps which we also decided to use for our method.

2.1.1 Droplet Detection

We detect the droplets in the BF channel. For this, we first apply rank equalisation, to have more contrast in the image and then a Gaussian blur filter to remove noise from the images. The detection of droplets in the image is performed using the Hough Circles algorithm, from the OpenCV library ([4]). The algorithm is based on a three-parameter description of a circle. It will first detect edges in the input image and then, given a specific predefined radius, is looking for the circle centers. In our case each droplet is a circle. We look for radii of different sizes, to capture all droplets. Then all found droplet parameters (centre and radius) are refined using the RANSAC (Random sample consensus) algorithm [5].

2.1.2 Visual Embeddings

For the preprocessing of the raw data for visual embeddings, we process the BF and DAPI channel differently: For the BF channel, we first re-scale the intensity to $[0, 1]$ and then apply median blurring

and rank equalisation. Then we set all values which are below the 50%-percentile of the intensity values to zero. For the DAPI channel we set all values which are below the 80%-percentile of the intensity to zero and then apply the same steps as in the BF channel. In Figure 2 the effect of the different ways of preprocessing are compared.

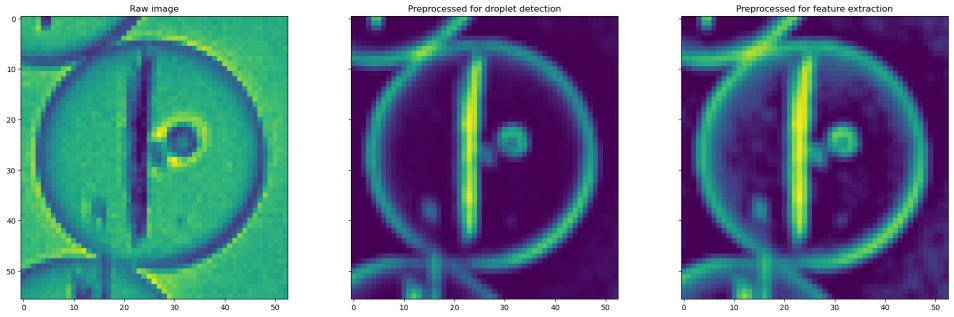


Figure 2: **Left:** raw droplet. **Middle:** preprocessed for droplet detection. **Right:** preprocessed for feature extraction.

2.2 Cell Detection

After preprocessing the data for droplet detection, we can also use it for counting the number of cells in each droplet. We cut out each droplet in the preprocessed DAPI channel to detect whether cells are present in the given droplet or not. To find a cell, we first use the NMS (non-maximum suppression) algorithm to get an intial guess of where cells could be in a droplet. Then, using a threshold, we check whether the persistency and intensity at the proposed position are high enough to be classified as a cell and count the number of cells per droplet.

3 Approach

3.1 Optimal Transport

Given the data and its scale, we decided to look beyond multi-object tracking (MOT) algorithms which frequently find applications when the time resolution across images is very high. Since this is not the case for us, we instead decided to take a more theoretical, optimisation-based approach by investigating the applicability of the theory of optimal transport (OT) to the problem at hand. The provided background is strongly inspired by [6].

In simple terms, OT addresses the question of how to best transport a given source distribution to a target distribution. We will refer to these distributions as point clouds following [7]. We refer to the source distribution as the source point cloud and the target distribution as the target point cloud. In the case of droplet tracking these point clouds are discrete.

To solve the OT problem, a cost matrix $C(\mathbf{x}, \mathbf{y}) \in \mathbb{R}_+^{n \times m}$ is defined. Its entries are given by the cost function $c(\mathbf{x}_i, \mathbf{y}_j)$ which measures the cost between any pair of points $\mathbf{x}_i, \forall i \in \{1, \dots, n\}$ and $\mathbf{y}_j, \forall j \in \{1, \dots, m\}$, where n and m are the number of source and target points respectively. Note that this is the discrete formulation of the OT problem. The cost function is general and determines the problem's geometry. For the case of droplet tracking we define the cost function as

$$c(\mathbf{x}_i, \mathbf{y}_j) = \alpha L_{pos}(\mathbf{x}_i, \mathbf{y}_j) + \beta L_{vis}(\mathbf{x}_i, \mathbf{y}_j) + \gamma L_{cells}(\mathbf{x}_i, \mathbf{y}_j), \tag{1}$$

where L_{pos} refers to the positional loss, L_{vis} refers to the visual embedding loss and L_{cells} refers to the loss related to the number of cells in each droplet. We scale the position embeddings, such that the 95-quantile of the positions is equivalent with the 95-quantile of the visual embeddings, so that α and β can of same size and we can interpolate between them. Specifically, each loss is defined as

follows:

$$L_{pos}(\mathbf{x}_i, \mathbf{y}_j) \coloneqq \left\|\mathbf{x}_i^{pos} - \mathbf{y}_j^{pos}\right\|_2 \tag{2}$$

$$L_{cells}(\mathbf{x}_i, \mathbf{y}_j) \coloneqq \left|\mathbf{x}_i^{cells} - \mathbf{y}_j^{cells}\right| \tag{3}$$

$$L_{vis}(\mathbf{x}_i, \mathbf{y}_j) \coloneqq \begin{cases} \left\|\mathbf{x}_i^{vis} - \mathbf{y}_j^{vis}\right\|_2 \\ 1 - \cos(\mathbf{x}_i^{vis}, \mathbf{y}_j^{vis}) \end{cases} \tag{4}$$

The parameters α, β, γ are tunable hyperparameters which determine the relative importance of each loss. The next ingredient for solving the OT problem is known as the transport matrix $T(\mathbf{x}, \mathbf{y}) \in \mathbb{R}_+^{n \times m}$ which will capture the optimal transport plan between the source and target point clouds. We can think of each point of the source and target cloud having a certain mass and thus being represented by a certain mass vector. For the source points, the mass is encoded in $\mathbf{a}$ and for the target points it is encoded in $\mathbf{b}$. These quantities are crucial for the so-called balancedness of the OT problem as in general, following transport, the mass of the source points should have been transported to the target points without loss. Transport matrices that fulfil this property are defined as

$$U(\mathbf{a}, \mathbf{b}) \coloneqq \{T \in \mathbb{R}_+^{n \times m} \mid T\mathbf{1}_m = \mathbf{a}, T^T\mathbf{1}_n = \mathbf{b}\} \tag{5}$$

and the most primitive OT problem is then to solve the objective

$$\min_{T \in U(\mathbf{a}, \mathbf{b})} \langle T, C \rangle. \tag{6}$$

To ensure that this problem computationally feasible we regularise the objective (even though in certain cases it may be computationally feasible without regularisation) and since we are in the unbalanced case we relax the balancedness constraints yielding the objective

$$\min_{T \in \mathbb{R}_+^{n \times m}} \langle T, C \rangle + \rho_{\mathbf{a}} KL(T\mathbf{1}_m || \mathbf{a}) + \rho_{\mathbf{b}} KL(T^T\mathbf{1}_n || \mathbf{b}) - \epsilon E(T), \tag{7}$$

where $E(T)$ is the entropy-regulariser of T. The parameters $\rho_{\mathbf{a}}$ and $\rho_{\mathbf{b}}$ allow for some slack in the balancedness constraint originally imposed on the transport matrix by $U(\mathbf{a}, \mathbf{b})$. The regularisation parameter ϵ controls the regularisation strength and will influence how "willing" a given source point is to share its mass with multiple points of the target cloud rather than just giving all its mass to a single point. The OT problem in 7 is known as the fully unbalanced entropic regularised OT problem and can be solved using a slightly modified version of the commonly used Sinkhorn algorithm [8].

3.2 Visual Embeddings

As some of the images exhibit large movement of droplets between frames, we hypothesise that solving the optimal transport problem on positional data alone is not sufficient to track the droplets reliably. Upon closer observation of single droplets, we found they have visual features which only change slightly between frames; for example, the number of cells as well as the thickness and number of magnetic beadlines. This consistency of visual information is illustrated in Figure 3. Hence, we want to incorporate this additional information in our tracking algorithm for which we embed the droplets in a common latent space for our loss as defined in 4. This is done using a convolutional neural network (CNN) with a contrastive learning approach to extract the visual features.

3.2.1 Training Data

For the training of our CNN we use images in which there is visibly little droplet movement. We can extract droplets which we believe to be tracked correctly by only considering detected droplets whose position in the next frame is within a small radius of the position in the current frame. We validated this approach by manually checking the tracking results for a subset of the droplets. Finally, we extract 30k droplet patches of pixel size 40 by 40 for training and another 30k droplet patches from other images for validation.

3.2.2 CNN - Encoder

As the droplet patches are of size 40 by 40 with limited visual features, we use a small model called EfficientNet-B1 [9] to avoid overfitting. The model features 6 million parameters and is pre-trained on ImageNet [10]. We use the output of the last convolutional layer as our visual embedding. The default output size of the EfficientNet-B1 is 1280, which we reduce to 20 by using a fully connected layer.

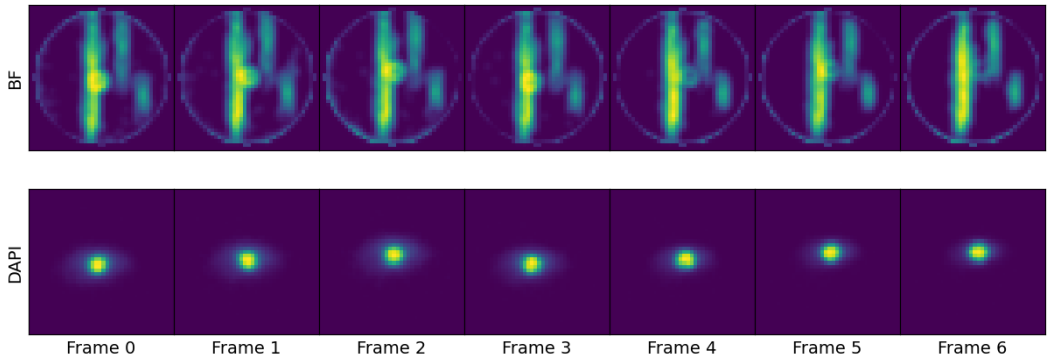


Figure 3: Droplet patch across 7 frames.

3.2.3 Contrastive Learning

In order to learn the low dimensional visual embeddings, we use the contrastive InfoNCE loss [11], which is defined as follows:

$$\mathcal{L}_{\text{InfoNCE}} = -\log \frac{\exp(z_i^T z_j/\tau)}{\sum_{k=1}^{N} \mathbb{1}_{[k\neq i]} \exp(z_i^T z_k/\tau)}. \tag{8}$$

Here z_i and z_j denote the visual embeddings of droplets i and j, N is the batch size and τ is a temperature parameter controlling the entropy of the resulting distribution. The InfoNCE loss encourages the visual embeddings of two droplets from the same trajectory to be close together and the visual embeddings of two droplets from different trajectories to be far apart as illustrated in Figure 4.

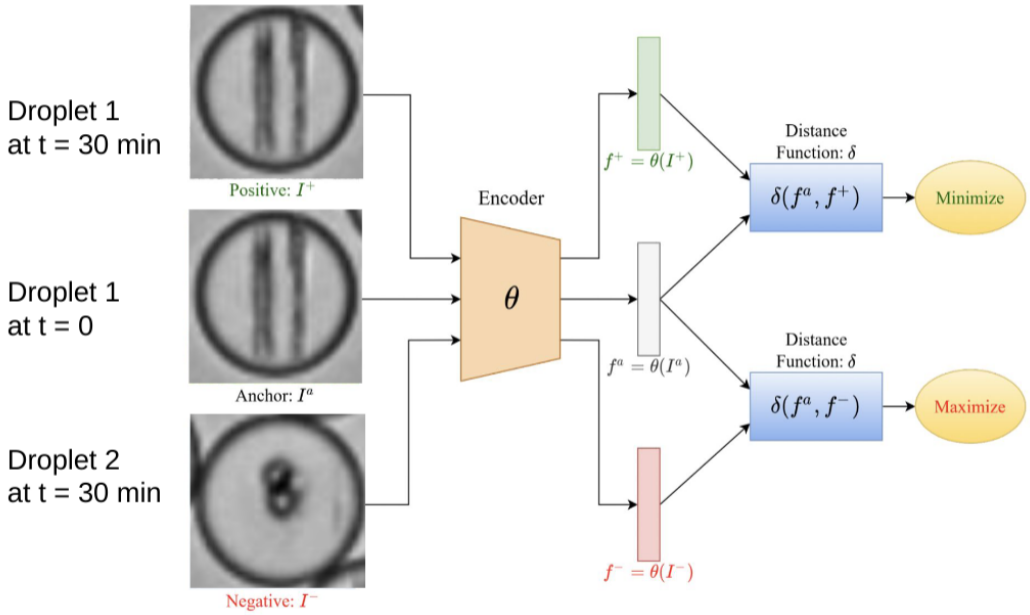


Figure 4: Contrastive learning of droplet representations. Adapted from [12]

.

We use the Adam optimiser ([13]) with a learning rate of 10^{-3} and a batch size of 128. Performing a grid search we found that an embedding size of 20 yields the best trade-off between accuracy and computational resources. As a higher embedding size only yielded marginal accuracy improvements, but significantly increases the space needed for storage.

3.2.4 Visual Embedding Metrics

Choosing the model hyperparameters. To gauge the quality of the visual embeddings, we mainly use the accuracy of matched droplets across two frames. We compare each droplet with its 128 nearest neighbours in the next frame and consider the droplet to be correctly matched if the distance to its positive match is minimal amongst all distances computed. Further, we consider two different scenarios. In the first we consider droplets which contain cells only. For the second we to use all droplets. With this approach, we get accuracies of 0.94 and 0.91 on the validation dataset for the two scenarios respectively.

Metrics for evaluating the model. For evaluating the embeddings we consider two metrics: accuracy and area under the ROC curve. We report the obtained values of the metrics in Section 5.

- **Top 1 and top 5 accuracy.** We assume we have a dataset of matched droplets. For a given droplet embedding we compute its k-nearest neighbors amongst embeddings of all the droplets from the consecutive frame. If the matched droplet embedding is present in the k-nearest neighbors we consider it a positive match. The accuracy is the ratio of positive matches to the number of considered droplets.
- **Area under the ROC curve.** To obtain the ROC curve, we compute the distances between embeddings of correctly matched droplets (positive samples) and between randomly matched droplets (negative samples). Then we concatenate the two lists of distances and sort them in increasing order. For a given k we look at the first k elements and call the fraction of the ones coming from positive samples a true-positive rate (TPR) and the fraction of the ones coming from the negative samples a false-positive rate (FPR). By varying k we obtain paired values of TPRs, and FPRs which we plot as a ROC curve. Finally, we compute the area under the curve.

3.3 Trajectory Creation

Solving the OT problem between two given frames provides us with the OT matrix $T \in \mathbb{R}_{+}^{n \times m}$, where n and m are the numbers of droplets in the source and target frame, respectively. An entry $T_{i,j}$ indicates how much mass should be transported from the i^{th} droplet in the source to the j^{th} in the target. Since the OT problem is unbalanced it is crucial to keep track of global droplet IDs and how they correspond to the row and column indices of the OT matrix. Let d_i^s denote the given global ID of the i^{th} droplet in the source. Now we also assign this ID to the droplet in the target frame to which most mass is being transported:

$$d_{\tau(i)}^t \leftarrow d_i^s \quad \textit{for} \quad \tau(i) \coloneqq \arg \max_{k \in \{1,\dots,m\}} T_{i,k}. \tag{9}$$

The corresponding matching score is $s_i = T_{i,\tau(i)}$, which quantifies the transported mass. By applying this to all droplets and frames, we can predict the trajectories of all droplets present in the first frame. It needs to be noted, that the matching rule 9 allows for multiple source IDs to be matched to the same global target IDs. Such multiplicity may indicate that a droplet went missing in between two frames or that the OT solution is simply inaccurate. Similarly, not all target droplets will be assigned a global ID, which may represent appearing droplets. In principle, it is possible to also record such emerging trajectories. However, this makes the downstream analysis considerably more complicated.

Ideally, a score s_i would also indicate some confidence level in the proposed match. Linearly scaling all entries in the OT matrix onto the range $[0, 1]$ before applying 9, may already provide an adequate approximation for such confidence levels. Let the resulting scores be denoted by $\tilde{s}_i$. However, it is not clear whether the resulting values can be interpreted as probabilities and provide a well-calibrated model. In the next section, we discuss the idea of model calibration and propose a calibration method.

3.4 Model Calibration

Calibration curves, also known as reliability diagrams, provide a visual assessment of the calibration of probabilistic predictions from a binary classifier. These curves plot the predicted probability against the positive label frequency, offering insights into how well the model's predictions align with actual outcomes (see examples in section 5). We employ isotonic regression to map the match scores $\{\tilde{s}_i\}_{i=1}^n$ to calibrated probabilities $\{p_i\}_{i=1}^n$, where the idea is to align the predicted scores

with the observed frequencies of positive outcomes. Given a set of predicted scores $\{\tilde{s}_i\}_{i=1}^n$ and the corresponding observed binary ground truth $\{y_i\}_{i=1}^n$, isotonic regression finds a monotonic, non-decreasing $f : \mathbb{R} \to [0, 1]$ which minimizes the following objective:

$$\sum_{i=0}^{n}(y_i - f(\tilde{s}_i))^2. \tag{10}$$

The calibrated probabilities are then given by $p_i = f(\tilde{s}_i)$ for $i = 1, ..., n$. The monotonicity constraint ensures that the recalibrated probabilities maintain their order, preserving the relative ranking of samples. In scikit-learn [14], the isotonic regression solver is based on the Pool-Adjacent-Violators Algorithm (PAVA) [15]. In the context of our pipeline, we train the isotonic regression model on simulated data and then apply it when performing tracking on other simulated data and real data.

3.5 Pipeline implementation

Our final data processing pipeline, presented in Figure 5, is implemented fully in Python. The pipeline can be run to track the droplets in real images or simulated data. Its core components, such as the visual embedding creation, computation of OT matrices and creation of trajectories are shared between these two modes. This ensures the consistency of results on simulated and real datasets. To configure our pipeline we use Hydra [16] - a framework for hierarchical parameter specification through .yaml files, which simplifies experimentation and makes the pipeline easy to use. One of the key features of the pipeline is caching. We made sure to store computationally heavy intermediate results thus allowing for easy recovery of the progress in case of a crash.

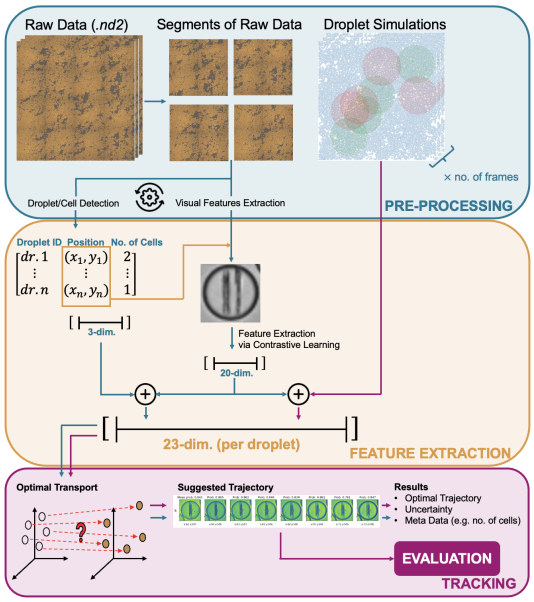


Figure 5: Simplified schema of the complete data processing pipeline.

3.5.1 Processing real data

The pipeline accepts .nd2 files as input. Depending on the specified setting, the image can be processed as a whole or in smaller segments. Firstly, the image segments are preprocessed for droplet detection and visual embedding computation. Then the droplets and the cells inside them are detected and droplet patches are cut out of an appropriate preprocessed image. Next, a pre-trained embedding model is loaded and used to produce the visual embedding for each patch. Droplet location, the visual embedding and the number of cells in the droplet are then scaled and concatenated. Finally, this representation of droplets is fed into the optimal transport, which produces transport matrices. These are later postprocessed to get the trajectories and their confidence estimations. The results are saved as a .csv file.

In order to visually evaluate the performance of our method on real datasets we integrated two visualizers into our pipeline: the visualizer of trajectory curves implemented by the DSL team from last year and the matched droplet visualizer shared with us by our challenge giver. The resulting visualizations can be seen in Section 5.

3.6 Processing simulated data

The pipeline for processing simulated data accepts as input a .csv file with droplet locations obtained as described in Section 4, .npy file with matched droplet patches and a .csv with the number of cells per droplets. Instead of providing the last two files, one can obtain them using the pipeline itself. This, however, requires running our code on the real low-movement image first, to obtain some high-confidence trajectories.

The pipeline matches provided location data with the patches and the number of cells, keeping track of the droplet ID across frames. The next steps up to obtaining calibrated trajectories are the same as while running the pipeline on the real data. Since the pipeline keeps track of the droplet ID across frames, at the end of the run, metrics for the trajectories, described in Section 4, are computed and reported.

4 Experimental Setup

4.1 Simulation of Data

By applying our method to real data with very little movement, we were able to predict a set of trajectories for which we are confident that the majority thereof are true positives. This confidence stems from visual observations using the visualizer tool and filtering based on the assigned uncertainty from the algorithm (at least some uncertainty threshold), distance from the image segment border (at least 20 pixels) and amount of movement (at most 5 pixels). We then combine the matched patches with simulated position data to imitate the real data with significant movement. This allows us to quantitatively evaluate our method on various kinds of movement, while still relying on visual embeddings.

4.1.1 Implementation

Ideally, the simulated positions should resemble different types of movement found in the real data. As briefly discussed in Section 2, by investigating several batches of microscopy images, it appears that most movement occurs in either of two ways. Sometimes there are large droplets of an area of at least 10-100 times the one of a regular droplet which move across the plate with varying speed. Other times, gaps emerge in different areas and grow over time. These two can also be observed in combination in some experiments. We implemented a simulation which can approximately imitate these modes of movement. It randomly initializes a fixed number of droplets, represented as circles with configurable radii, in a rectangular area. Further, it randomly initializes several attractive and repulsive force areas shaped like circles. If a droplet is contained within such an area it is attracted or repulsed from the circle centre with a strength proportional to its distance from the centre. Lastly, a very large droplet is added. All entities can be configured to move with some intrinsic velocity. Further, it is possible to add random noise to the positions of regular droplets. Every step of the simulation then updates the positions of all entities according to the attractive and/or repulsive forces and a simple elastic collision rule for collisions among droplets. For the evaluation part, we used segments of four frames of simulations with 6k and 20k droplets, each with small, medium and large movement. The degree of movement is determined by varying the strength of certain interactions and toggling the existence of force areas and large droplets. The source code is written in C++ and the generated positional data can be found on the GitHub page. Figure 6 shows four frames of a simulation with 20k droplets, medium movement, a few large attraction areas (marked in green), several small rejection areas (marked in red) and a medium-sized large droplet.

4.2 Evaluation Metrics

The advantage of having simulated ground-truth data is that we can assess the quality of tracking quantitatively without expert labelling. This allows for tuning of OT hyper-parameters and choosing a

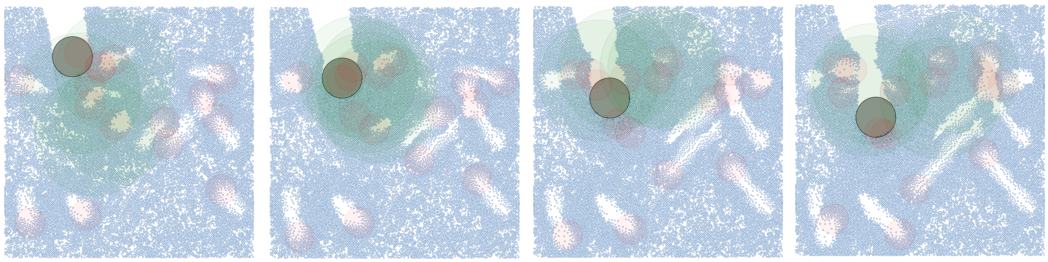


Figure 6: Simulation of positions of 20k droplets. Medium movement.

suitable contrastive learning model. For every predicted matching of droplet i and some other droplet between a source and target frame, let $p_i \in [0, 1]$ be the assigned confidence level and $y_i \in \{0, 1\}$ indicate whether the match is a true or false positive. Then several metrics can be computed using the set of tuples $\{(p_i, y_i)\}_{i=1}^{n}$, where n is the number of droplets in the source frame. We can also compute any of these across all frames to obtain total metrics describing the performance on whole trajectories.

4.2.1 Precision

A simple way to evaluate the performance of our tracking approach is to assess the precision given as:

$$\text{Precision} = \frac{1}{n} \sum_{i=1}^{n} y_i. \tag{11}$$

Further, one can compute the precision within the top k choices of the tracking method. Let $r_i = \text{rank}(i)$ be the rank of the i^{th} droplet ordered in descending order with respect to p_i (i.e., highest to lowest probability). Then the Precision@k is given as:

$$\text{Precision@k} = \frac{1}{k} \sum_{r_i=1}^{k} y_i \tag{12}$$

4.2.2 Area Under the Precision-Recall Curve (AUPRC)

AUPRC focuses on the trade-off between precision and recall across different classification thresholds, making it particularly useful in imbalanced class scenarios. First, we compute precision and recall for different threshold values using the scikit-learn library in Python [14]. Any predicted match with a confidence above a certain threshold value is considered a positive (match), all others are interpreted as negatives (no match). The resulting curve and the corresponding area under the curve help assess the model's performance in terms of discrimination.

4.2.3 Brier Score

For our method to be practically useful, users need to be able to trust the confidence levels p_i provided by the algorithm. As outlined in Subsection 3.3, we can investigate calibration plots to check whether the predicted probabilities align with true outcomes on average. Further, one can also compute the Brier score:

$$BS = \frac{1}{n} \sum_{i=1}^{n} (y_i - p_i)^2. \tag{13}$$

It needs to be noted that while the Brier score does assess the degree of calibration, from [17] we know, that it also quantifies discriminative power, and randomness in the data. Hence, the score cannot be used to solely assess calibration. Again, we use the scikit-learn library for its implementation.

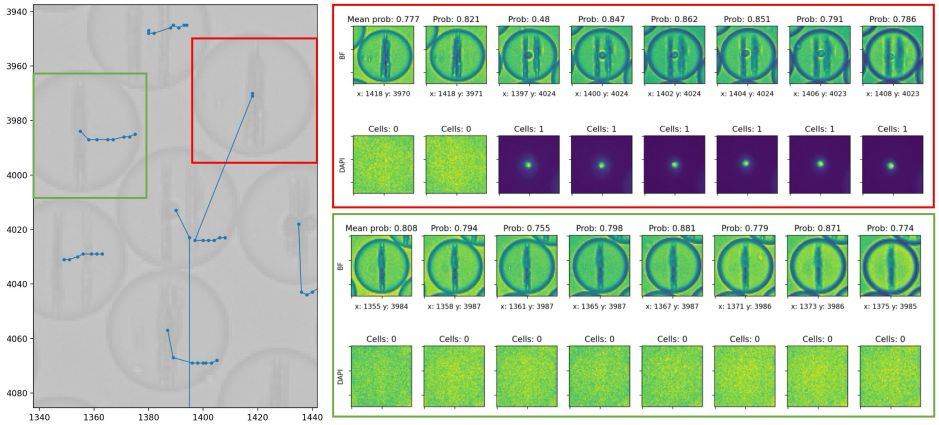


Figure 7: **Left:** Crop of one region taken from a small movement dataset. Blue dots and lines indicate trajectories over all eight time frames. In the background in grayscale the actual image can be seen. **Right:** BF and DAPI channel trajectories of two droplets from the left. Optimal transport transition probabilities are indicated above BF channel where probability refers to the transition from the previous droplet to the droplet over which the probability of interest is indicated. The mean probability of trajectory is included above the first time frame. Information about the number of cells is included above the DAPI channel at each time frame.

5 Results

5.1 Line and Visual Trajectories

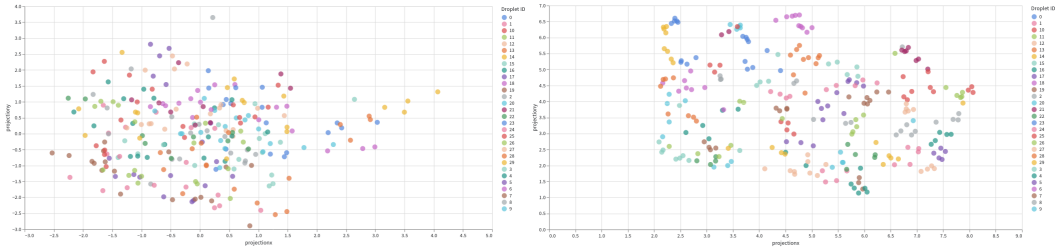
In Figure 7 we show an interesting section from a small movement dataset. In the left image, the line trajectories, indicated by blue dots connected with blue lines, represent the predicted trajectories across all time frames for the small movement dataset. In the background, we see the underlying grayscale image during the first time frame of the dataset. We observe that some of the line trajectories are smooth across all time frames whereas others show clear and abrupt jumps.

The right image illustrates the predicted trajectories for the top-right and top-left droplet trajectories, marked in red and green respectively. For a given trajectory we illustrate the cropped droplet patches at each time frame in the bright-field (BF) and DAPI channels. Further, we provide the computed transition probabilities for each transition, where the value above a patch corresponds to the transition from the *previous* frame to the *current* one. The value above the first frame is the mean of all transition probabilities of the trajectory. We observe high probabilities of approx. 80% for both droplets except from the second to third frame for the red droplet where we report a probability of just 48%. This is also the transition where the number of cells jumps from 0 to 1 which is clearly visible in the DAPI channel.

Combining the information from both images allows us to understand what is happening. We can see that the jump in the red droplet's line trajectory corresponds to the transition from the second to the third frame. This aligns nicely with the low transition probability of 48% meaning the OT output is uncertain. From the BF channel, it is visible that the tracking is inaccurate since we are no longer observing the same droplet. Aside from the purely visual information in the BF channel this is in particular confirmed by looking at the number of cells which, if tracking and cell detection is correct, remains constant across time frames for the same droplet. Since the number of cells increases from 0 to 1, we identify an erroneous tracking. For the green droplet, we observe a smooth line tracking and thus do not expect erroneous tracking. This is confirmed through the BF and DAPI channel images in which the number of cells remains at 0. Moreover, the transition probabilities are high across all time frames further backing up this result.

5.2 Visual embeddings of droplets

In this section, we present visualizations of the embedding space and metrics for two selected embedding models. Model 1 was trained only on patches of droplets that contained cells, while



(a) Droplet embeddings reduced with PCA. (b) Droplet embeddings reduced with UMAP.

Figure 8: Embeddings of random 30 droplets across 9 frames, coloured by their droplet ID.

Model 2 was trained on droplets with and without cells. A detailed description of the metrics can be found in Section 3.

In Table 1 we report the performance of the model on the validation set (used also to select the best model) and on a completely separate test set, coming from a different experiment. The test set was not accessed during the fitting and choosing of the model. As the exact values of the metric depend on the sample size, we report each metric twice for the test set - once for a subsample of the test set matching the size of the validation set (1848 droplets across 9 frames) and once for the whole test set (13584 droplets across 9 frames). To obtain the results from Table 1 we compare the embeddings of droplets coming from consecutive frames and average the results across frame transitions. We performed the experiment with metrics based on both Euclidean and cosine distance. The results are comparable for both distances, hence we report only the metrics based on the Euclidean distance.

Metric	Model 1 (validation)	Model 1 (test subsample)	Model 1 (test)	Model 2 (validation)	Model 2 (test subsample)	Model 2 (test)
TOP 1 ACC	0.952 (0.013)	0.992 (0.002)	0.983 (0.003)	0.963 (0.011)	0.993 (0.001)	0.985 (0.002)
TOP 5 ACC	0.981 (0.008)	0.999 (0.001)	0.996 (0.001)	0.986 (0.006)	0.999 (0.000)	0.997 (0.001)
AUROC	0.988 (0.001)	0.990 (0.000)	0.990 (0.000)	0.988 (0.001)	0.990 (0.000)	0.990 (0.000)

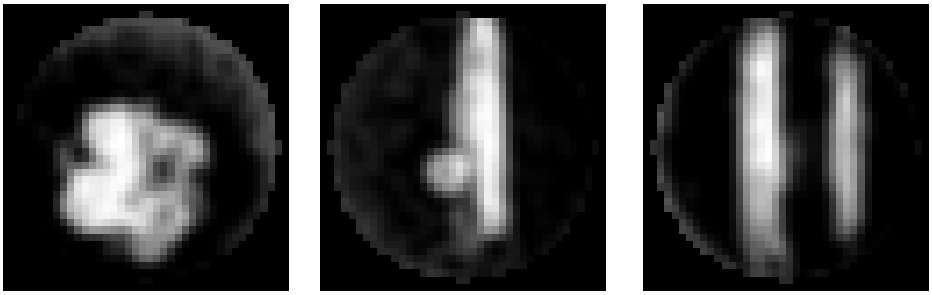
Table 1: Top 1 and top 5 accuracy as well as area under the ROC curve (mean and its standard error across 9 frames). Number of droplets in the *validation* and *test subsample* datasets: 1848. Number of droplets in the test dataset: 13584. Metrics are based on Euclidean distance.

Both models produce meaningful embeddings and hence allow for matching of droplets based on distance in the embedding space. The model trained on both droplets with and without cells slightly outperforms the one trained on images of droplets with cells only. Comparable performance on the validation and test set suggests that the models should generalise across experiments. We cannot be sure however that they will generalize across different movement types, as both datasets come from images with low (validation) and very low (test) movement. The supervised nature of our experiments and the lack of matched data for different types of movement prohibits us from evaluating the models in a broader setting. Finally, we notice that increasing the sample size impacts the accuracy of matching based on distance only marginally, although the task becomes much more difficult. This suggests that the embeddings are a useful part of the system even for big images and that OT-based matching should perform well for costs prioritising matching in the visual embedding space.

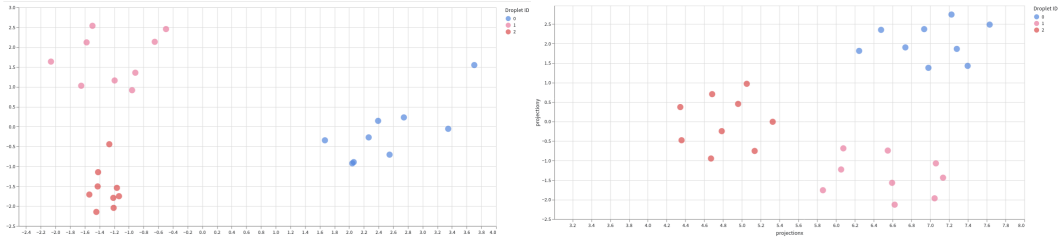
We now move on to discussing the visualizations of the embeddings generated using Model 2. Embeddings generated using Model 1 behave very similarly, hence we omit them from this analysis. To asses both the global and local structure of the embedding space, we reduce the dimensionality using both, PCA and UMAP (nearest neighbors=15, minimal distance=0.1, spread=10).

In Figure 8 we present the visualization of the embeddings of random 30 droplets across 9 frames colored by their droplet ID. For both dimensionality reduction techniques, but especially for UMAP, we can see that the same-coloured points, representing the same droplet, tend to form groups, which is an expected and desirable property. In Figure 9 we repeat the same visualization for 3 selected droplets with distinctive looks and observe that they form well-separated clusters.

One concern of ours was that the visual embeddings could pick up some image artifacts related to location. All our data used to train and evaluate the embeddings are low movement. Hence, we could

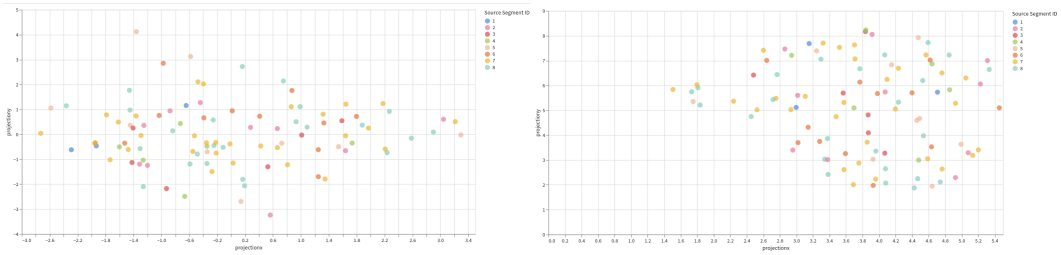


(a) Blue. (b) Pink. (c) Red.



(d) Droplet embeddings reduced with PCA. (e) Droplet embeddings reduced with UMAP.

Figure 9: Embeddings of selected 3, differently looking droplets across 9 frames, coloured by their droplet ID together with the original droplet images.



(a) Droplet embeddings reduced with PCA. (b) Droplet embeddings reduced with UMAP.

Figure 10: Embeddings of 100 random droplets from a single frame, coloured by the image segment.

have potentially mistaken the exploitation of such local artifacts for good performance. To be sure that this is not the case, we visualize the influence of the original droplet location on the embedding space. We split a single frame of the original image into segments and coloured droplets based on the segment they are located in. In Figure 10 we observe that the embeddings of droplets coming from different segments are well mixed for both dimensionality reduction algorithms. Moreover, if we compare the UMAP visualizations from Figure 8 and Figure 10, we see that embeddings coming from data assigned to the same segment do not form similar structures to the ones coming from the same droplet across frames.

We finish the discussion about the embedding by sharing with the reader a link to an interactive embedding visualization, where by hovering with the mouse cursor over the point one can see an underlying droplet image. From this visualization, we see that droplets with similar magnetic bids (their number and width) tend to be located close to each other in the reduced embedding space.

5.3 Model Calibration

Figures 11, 12 and 13 show calibration plots for the tracking of around 8k droplets for small, medium and large movement. The title indicates the type of movement and the frame range on which we applied our evaluation pipeline. For all Figures, on the left one can find the calibration plot for which we used the raw scaled OT matrix entries as confidence levels for the match predictions. On the right, one can see the calibration plots for the same trajectories on the same type of movement but

with calibrated probabilities. Here we calibrate by training the isotonic regression model discussed in Section 3.3 on the results for medium movement across frames F_0 to F_3 and applying it to the respective match scores. For the calibration plot in Figure 12, to avoid testing on training data, different frames of the simulation and different image patches were used to generate the results. The dotted line indicates a perfectly calibrated model, the area marked in grey shows the distribution of predicted confidence scores and the Brier score across all frames is given in the grey box (lower values are desirable). Differently coloured lines correspond to the predictions for transitions between different pairs of frames.

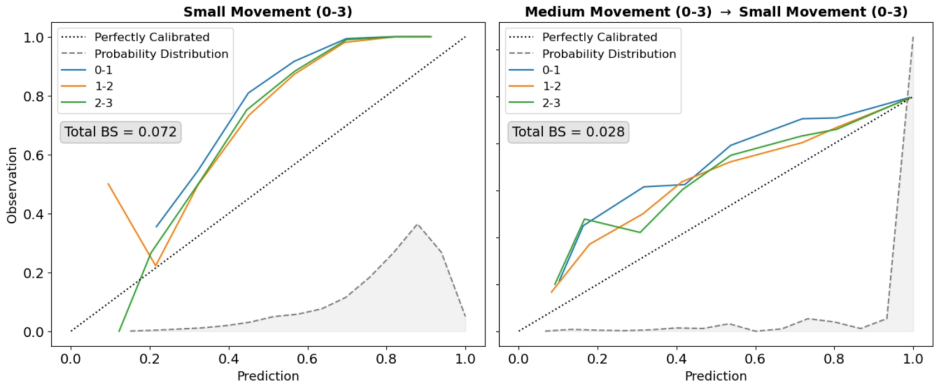


Figure 11: Model calibration for small movement, trained on medium movement.

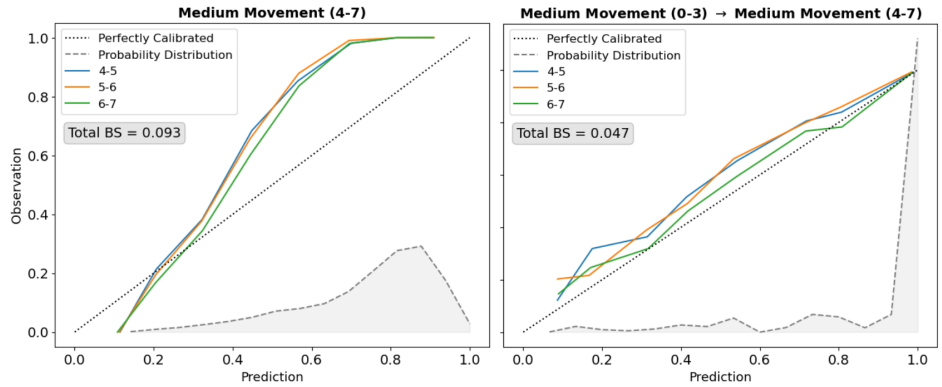


Figure 12: Model calibration for medium movement, trained on medium movement.

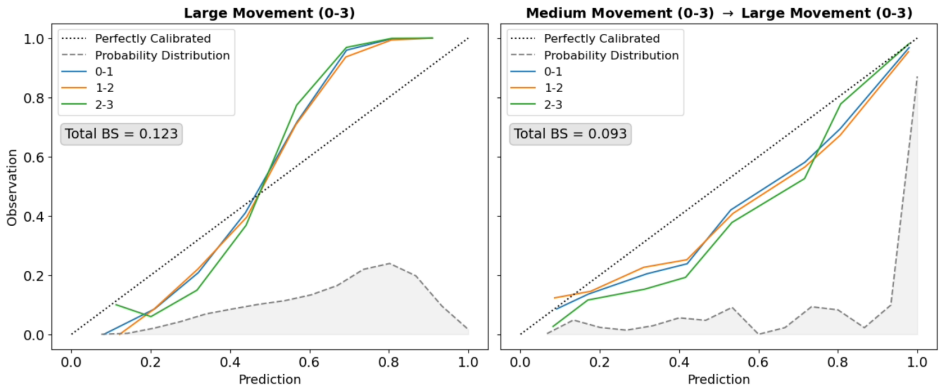


Figure 13: Model calibration for large movement, trained on medium movement.

5.3.1 Brier Scores

When comparing the plots on the left side across all Figures, we observe that more movement leads to larger Brier scores, indicating worse calibration, lower discriminative power and/or more randomness in the data. This confirms that the problem becomes increasingly hard to solve for larger degrees of movement. The decrease in the Brier score via calibration ($0.072 \rightarrow 0.028$ for small movement, $0.093 \rightarrow 0.047$ for medium movement and $0.123 \rightarrow 0.093$ for large movement) indicates that isotonic regression can improve model predictions significantly, even when the model is trained in a different setting.

5.3.2 Calibration Plots

For small and medium movement, the calibration graphs for the uncalibrated results consistently lie above the dotted line indicating perfect calibration. This implies that for such settings the scaled OT matrix entries, when interpreted as confidence levels, are generally not confident enough. In other words, our pipeline predicts lower probabilities for match proposals than necessary. For large movement, this is also the case for predicted probabilities greater or equal to 0.5. For predictions with probabilities below that threshold, however, the pipeline is overconfident. Calibration via isotonic regression leads to two major changes. Firstly, the probability distribution is skewed to higher numbers after calibration. Secondly, the graphs are significantly closer to the dotted line than without calibration. Further, it seems that when training the regression model on medium movement, the calibrated probabilities are slightly underconfident for small movement and overconfident for large movement. These results indicate that when the simulated data on which the isotonic regression is trained is moderately similar to the real data, calibration may significantly improve the interpretation of the confidence levels provided by our pipeline. This would make our tool more trustworthy and as a consequence more useful for its users. The true effectiveness remains to be evaluated on real ground-truth data, however.

5.4 Parameter search for simulated data

In order to find optimal parameters for our pipeline we have run an extensive parameter search. Due to the task being computationally heavy, we firstly run a finer search on the simulated data with 6000 droplets across 4 frames and after selecting promising subset of parameters we rerun the experiment on data with 20000 droplets across 4 frames and repeat all the experiments for small, medium and large movement simulation.. The parameters we consider are:

- $\alpha \in \{0, 0.1, 0.3, 0.5, 1\}$ which represents the contribution of the location loss to the cost function (as described in Section 3). Furthermore, we set $\beta = 1 - \alpha$ and $\gamma = 0$ in order to reduce the search space and because we consider interplay between location and visual loss to be especially important to investigate.
- $\rho_a = \rho_b \in \{0.99, 0.999\}$ which should be adapted to the unbalancedness of the data.
- $\epsilon_{rel} \in \{5 \cdot 10^{-3}, 5 \cdot 10^{-4}, 5 \cdot 10^{-5}\}$ which specifies the regularising ϵ in the OT as ϵ_{rel} times the mean entry of the cost matrix.
- Visual embedding loss which can be either Euclidean or cosine distance.
- Visual embedding model which can either be Model 1 trained only on droplets with cells or Model 2 trained on both droplets with and without cells, similar as in Section 5.2.

We observed that irrespective of other parameters the choice of ϵ_{rel} either does not significantly influence the metrics or a choice of $\epsilon_{rel} = 5 \cdot 10^{-3}$ gives the best result. Similarly, we observed that the cost based on Euclidean distance consistently performs better than the cosine distance and that the choice of $\rho_a = \rho_b = 0.999$ is optimal for small and medium movement data while $\rho_a = \rho_b = 0.99$ is better for large movement. Hence, we plot the metrics for runs with these selected parameters.

Our simulation has 20k droplets in each frame, but we cut the boundary to introduce unbalancedness. This gives us about 16k droplets with a little bit of unbalancedness. We present the precision for all droplets and the precision@10k. From the plots in Figure 14, we clearly see that bigger values of α work better for smaller movement, for medium and large movement taking an alpha of about 0.5 performs best. This shows two things: For small movement just using the positions is sufficient for the optimal transport to work and as soon as there is more movement we have to include visual

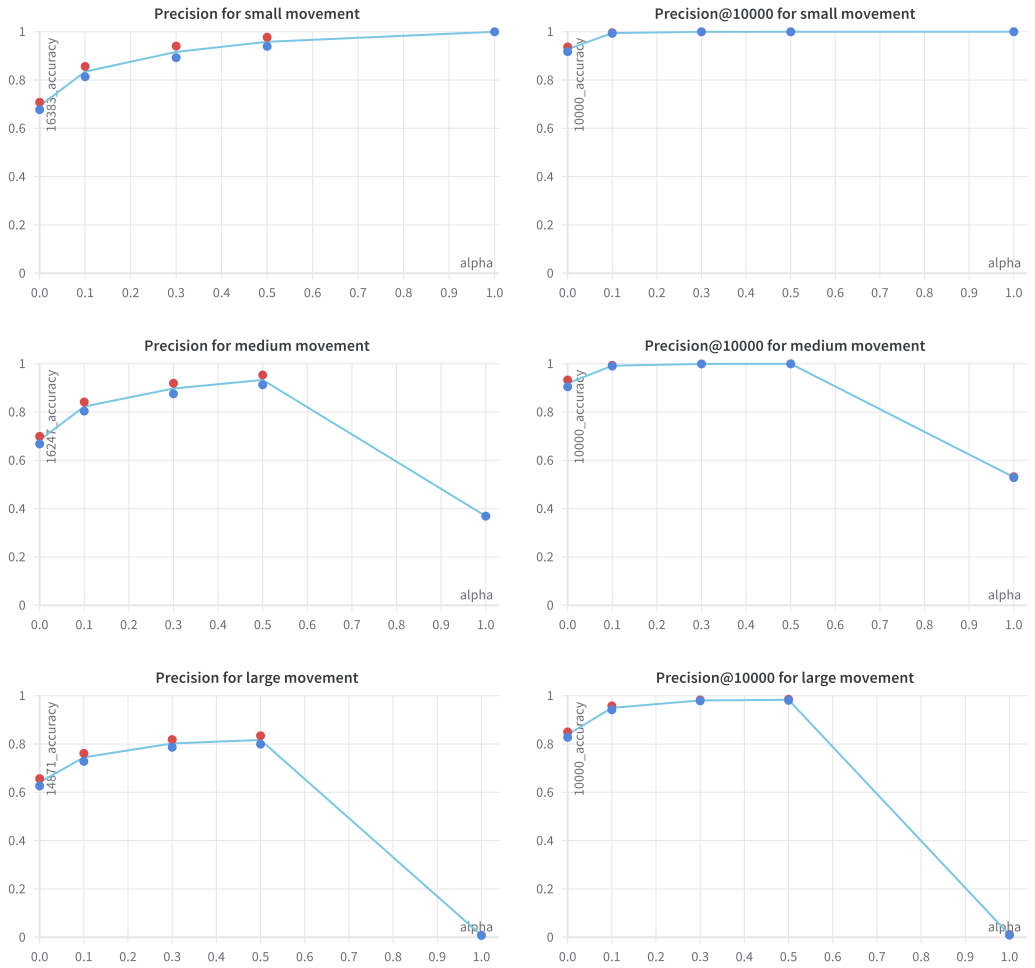


Figure 14: Precision score for small, medium and large movement simulations as a function of α. Red and blue dots represent the runs with Model 1 and Model 2 used for the visual embeddings respectively. Other parameters: $\rho_a = \rho_b = 0.999$ for small and medium movement and $\rho_a = \rho_b = 0.99$ for large movement, $\epsilon_{rel} = 0.005$. Embedding space cost defined through Euclidean distance. Corresponding AUPRC scores for the best performing parameter sets for small, medium and large movement were 1.000, 0.998, 0.987 respectively.

information to get high scores. Our challenge giver is willing to not have every droplet tracked, but of the ones that are tracked we should be certain. So the intended use of the algorithm is to report only the trajectories with high certainty and thus taking $\alpha = 0.5$ yields almost perfect tracking for 10k most certain droplets, which is more than 50% of the droplets in scope.

6 Conclusion

In this project, we present an algorithm to track droplets captured in an immunoassay in an (almost) unsupervised manner by using the theory of optimal transport. We adapt existing implementations of solving optimal transport problems and Deep Learning algorithms to our specific challenge and combine them into an end-to-end pipeline.

While we are currently only working with droplets, our approach can be generalised easily to various kinds of tracking problems. One crucial step for improving our pipeline is to experimentally produce real ground truth data and investigate how accurate our model is when applied on real data. Moreover,

to be able to understand what the model is doing and basing its predictions off, we must ablate the importance of each component of the OT cost function.

Nonetheless, we are confident that the platform we provide will significantly help experimental researchers with the ultimate goal of developing successful personalised Medicine.

To summarise, we suggested a droplet tracking pipeline which we can validate quantitatively on simulated data as well as qualitatively on real data. Furthermore, we have presented various visualisation tools which can be used by experts to manually confirm whether our predicted trajectories are correct.

Our first next steps, which were not feasible within the scope of the Data Science Lab, are to run experiments on larger volumes of data, to re-train or even adaptively train the visual embedding model which seems to have captured some domain-specific correlations, as well as design more meaningful performance metrics to be able to provide more quantitative results.

Fortunately, we have been offered the opportunity to publish our work and thus will be following the necessary steps in the near future.

7 Contribution

Weronika:

- Implementation of visual embedding evaluation metrics.
- Evaluation of the droplet embeddings created by the DSL team from last year and final evaluation of our droplet embeddings.
- Visualizations of the embedding space.
- Design and implementation of the main pipeline and evaluation pipeline with Mike.
- Implementation of part of the utility functions in the early stages of the embedding model development, together with Richard.
- Implementation of matched data extraction from the results produced by our pipeline and extracting matched data for the final experiments.

Mike:

- Investigation of optimal transport objective and solver with Sven and Tiago.
- Implementation of simulations in C++ for positional ground-truth generation.
- Implementation of trajectory creation and uncertainty measures with Sven.
- Implementation of custom filtering of predicted trajectories.
- Implementation of metrics for the evaluation of our tracking.
- Design and implementation of model calibration via isotonic regression.
- Design and implementation of the main pipeline and evaluation pipeline with Weronika.

Richard:

- Idea for contrastive visual embedding learning.
- Idea of extracting matched droplets by only considering those with very small movement.
- Model selection.
- Implementation of model training and evaluation for contrastive learning model.
- Running experiments on simulations, in particular to tune hyperparameters for the full pipeline.

Sven:

- Combined our approach with the Data Science Lab from last year, i.e. the preprocessing.
- Investigation of optimal transport objective and solver with Mike and Tiago.
- Implementation of trajectory creation and uncertainty measures with Mike.

- Implementation of custom filtering of predicted trajectories.
- Integration of the visualizers to work with our approach and output.

Tiago:

- Implementation of the initial Optimal Transport approach using the *OTT* library of [7].
- Further development of OT solver in team with Sven and Mike.
- Performance of heuristic assessment of functioning of OT using mainly positional data and line trajectories; later with Sven a little visual checking.
- Investigation of quantifiable evaluation metrics.
- Implementation of sliding window to cut large images into smaller sub-images with Sven.
- Communication of team progress at the DSL Check-In and poster presentation.

References

[1] H. Nakamura, Y. Arai, Y. Totoki, T. Shirota, A. Elzawahry, M. Kato, N. Hama, F. Hosoda, T. Urushidate, S. Ohashi, N. Hiraoka, H. Ojima, K. Shimada, T. Okusaka, T. Kosuge, S. Miyagawa, and T. Shibata, "Genomic spectra of biliary tract cancer," *Nature Genetics*, vol. 47, no. 9, pp. 1003–1010, 2015. [Online]. Available: https://doi.org/10.1038/ng.3375

[2] J. J. G. Marin, P. Sanchon-Sanchez, C. Cives-Losada, S. del Carmen, J. M. González-Santiago, M. J. Monte, and R. I. R. Macias, "Novel pharmacological options in the treatment of cholangiocarcinoma: Mechanisms of resistance," *Cancers*, vol. 13, no. 10, 2021. [Online]. Available: https://www.mdpi.com/2072-6694/13/10/2358

[3] Y. Bounab, K. Eyer, S. Dixneuf, M. Rybczynska, C. Chauvel, M. Mistretta, T. Tran, N. Aymerich, G. Chenon, J.-F. Llitjos, F. Venet, G. Monneret, I. A. Gillespie, P. Cortez, V. Moucadel, A. Pachot, A. Troesch, P. Leissner, J. Textoris, J. Bibette, C. Guyard, J. Baudry, A. D. Griffiths, and C. Védrine, "Dynamic single-cell phenotyping of immune cells using the microfluidic platform dropmap," *Nature Protocols*, vol. 15, no. 9, pp. 2920–2955, 2020. [Online]. Available: https://doi.org/10.1038/s41596-020-0354-0

[4] abidrahmank, "Hough circle transform," 2013. [Online]. Available: https://github.com/abidrahmank/OpenCV2-Python-Tutorials/blob/master/source/py_tutorials/py_imgproc/py_houghcircles/py_houghcircles.rst

[5] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, p. 381–395, jun 1981. [Online]. Available: https://doi.org/10.1145/358669.358692

[6] C. Bunne, "Optimal transport in learning, control, and dynamical systems," *ICML*, 2023.

[7] M. Cuturi, L. Meng-Papaxanthos, Y. Tian, C. Bunne, G. Davis, and O. Teboul, "Optimal transport tools (ott): A jax toolbox for all things wasserstein," *arXiv Preprint arXiv:2201.12324*, January 2022.

[8] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Advances in Neural Information Processing Systems*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds., vol. 26. Curran Associates, Inc., 2013. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2013/file/af21d0c97db2e27e13572cbf59eb343d-Paper.pdf

[9] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *CoRR*, vol. abs/1905.11946, 2019. [Online]. Available: http://arxiv.org/abs/1905.11946

[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *CoRR*, vol. abs/1409.0575, 2014. [Online]. Available: http://arxiv.org/abs/1409.0575

[11] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *CoRR*, vol. abs/1807.03748, 2018. [Online]. Available: http://arxiv.org/abs/1807.03748

[12] R. Kundu, "The beginner's guide to contrastive learning," *v7labs*, 2022.

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[15] J. de Leeuw, K. Hornik, and P. Mair, "Isotone optimization in r: Pool-adjacent-violators algorithm (pava) and active set methods," *Journal of Statistical Software*, vol. 32, no. 5, p. 1–24, 2009. [Online]. Available: https://www.jstatsoft.org/index.php/jss/article/view/v032i05

[16] O. Yadan, "Hydra - a framework for elegantly configuring complex applications," Github, 2019. [Online]. Available: https://github.com/facebookresearch/hydra

[17] A. H. Murphy, "A new vector partition of the probability score," *Journal of Applied Meteorology and Climatology*, vol. 12, no. 4, pp. 595 – 600, 1973. [Online]. Available: https://journals.ametsoc.org/view/journals/apme/12/4/1520-0450_1973_012_0595_anvpot_2_0_co_2.xml