UNIVERSITY OF CALIFORNIA SAN DIEGO

DREEM: A Deep Learning System for Tracking Biological Agents at Any Spatiotemporal Scale

A Thesis submitted in partial satisfaction of the requirements
for the degree Master of Science

in

Data Science

by

Aaditya Prasad

Committee in charge:

Professor Uri Manor, Chair
Professor Mikio Aoi
Professor Gal Mishne

2024

The Thesis of Aaditya Prasad is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2024

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to Drs. Talmo Pereira and Uri Manor for their invaluable mentorship throughout the course of both my undergraduate and master's studies. Their patience and generosity has made me into the researcher I am today. Furthermore, I want to extend my thanks to Drs. Mikio Aoi and Gal Mishne for serving on my thesis committee. I would also like to thank Arlo Sheridan for collaborating with me on the development of DREEM, without whom this thesis project would not have been possible. Finally, I would like to thank Vincent Tu for his initial work on this project as well as Cara Schiavon, Theo Couris, Shayan Afshar and the Talmo Lab phenotyping team for the collection and proofreading of the data used to train our models.

Chapter 3, in part is currently being prepared for submission for publication of the material. Prasad, Aaditya; Sheridan, Arlo; Tu, Vincent, Schiavon, Cara; Afshar, Shayan; Couris, Theo; Nono Nathaniel; Intal, Marcus; Abbara, Zaher; Evans, Kaela; Manor, Uri; Pereira, D. Talmo. The thesis author was the primary investigator and author of this material.

Chapter 4, in part is currently being prepared for submission for publication of the material. Prasad, Aaditya; Sheridan, Arlo; Tu, Vincent, Schiavon, Cara; Afshar, Shayan; Couris, Theo; Nono Nathaniel; Intal, Marcus; Abbara, Zaher; Evans, Kaela; Manor, Uri; Pereira, D. Talmo. The thesis author was the primary investigator and author of this material.

Chapter 5, in part is currently being prepared for submission for publication of the material. Prasad, Aaditya; Sheridan, Arlo; Tu, Vincent, Schiavon, Cara; Afshar, Shayan; Couris, Theo; Nono Nathaniel; Intal, Marcus; Abbara, Zaher; Evans, Kaela; Manor, Uri; Pereira, D. Talmo. The thesis author was the primary investigator and author of this material.

ABSTRACT OF THE THESIS

DREEM: A Deep Learning System for Tracking Biological Agents at Any Spatiotemporal Scale

by

Aaditya Prasad

Master of Science in Data Science

University of California San Diego, 2024

Professor Uri Manor, Chair

Analyzing the dynamics underlying various neural phenotypes is key to understanding the systems, cellular, and subcellular processes that guide them. However, while there exist many robust methods for object detection in biological settings, such as SLEAP (Pereira et al., 2022) and CellPose (Stringer et al., 2021), there lacks a universal approach for linking these detections through time. Specifically, given the complex, variable imaging conditions that most biological videography occurs under, current deep learning approaches which exploit only local information may not be suitable for this setting. This is because biological videos contain edge-

cases that are not typically seen in the applications that most multiple object tracking (MOT) approaches are designed for, such as pedestrian and automotive tracking. Here, we introduce DREEM (**D**REEM **R**econstructs **E**very **E**ntity's **M**otion), a deep learning framework which leverages a transformer-based architecture to directly learn the associations between objects in a large temporal context. We demonstrate that DREEM enables the training of state-of-the-art models for biological MOT. Then, we show that DREEM is sample efficient and can transfer seamlessly across a variety of diverse settings, enabling use in a wide variety of fields. Our code and pretrained models will be released at https://github.com/talmolab/biogtr

INTRODUCTION

Analyzing the kinematics underlying various neural phenotypes is key to understanding the systems, cellular, and subcellular processes that guide them. Interestingly, these kinematics can be studied through the lens of both cause and effect. For example, the nascent field of computational ethology argues that the brain's computations and circuits are highly optimized to support natural behaviors, usually generated by the output of movement (Datta et al., 2019; Pereira et al., 2020) On the other hand the movement of biological structures or impairment thereof can cause a downstream phenotype to emerge (Norregaard et al., 2017). While there is a rich history of qualitatively describing biological motion, the shift towards precise quantification requires the unique ability to trace stochastic biological motion at multiple spatiotemporal scales.

Yet, while there exist many robust methods for object detection in biological settings, such as SLEAP (Pereira et al., 2022) and CellPose (Stringer et al., 2021), there lacks a universal approach for linking these detections through time. Specifically, given the complex, variable imaging conditions that most biological videography occurs under, current deep learning approaches which exploit only local information may not be suitable for this setting. This is because biological videos contain edge-cases that are not typically seen in the applications that most multiple object tracking (MOT) approaches are designed for, such as pedestrian and automotive tracking. For example, the inductive biases of commonly utilized convolutional neural networks (CNNs) are not as well suited for the biological domain due the sparsity of microscopy data as well as the lack of differentiating features in animals and particles.

Here, we introduce DREEM (**D**REEM Reconstructs **E**very **E**ntity's **M**otion), a deep learning framework which leverages a transformer-based architecture to directly learn the associations between objects in a large temporal context (Zhou et al., 2022). We demonstrate that DREEM enables the training of state-of-the-art models for biological MOT. Then, we show that DREEM is sample efficient and can transfer seamlessly across a variety of diverse settings, enabling use in a wide variety of fields.

Chapter 1 Multiple Object Tracking

Here we describe the problem of multiple object tracking, common algorithm designs, and their applications.

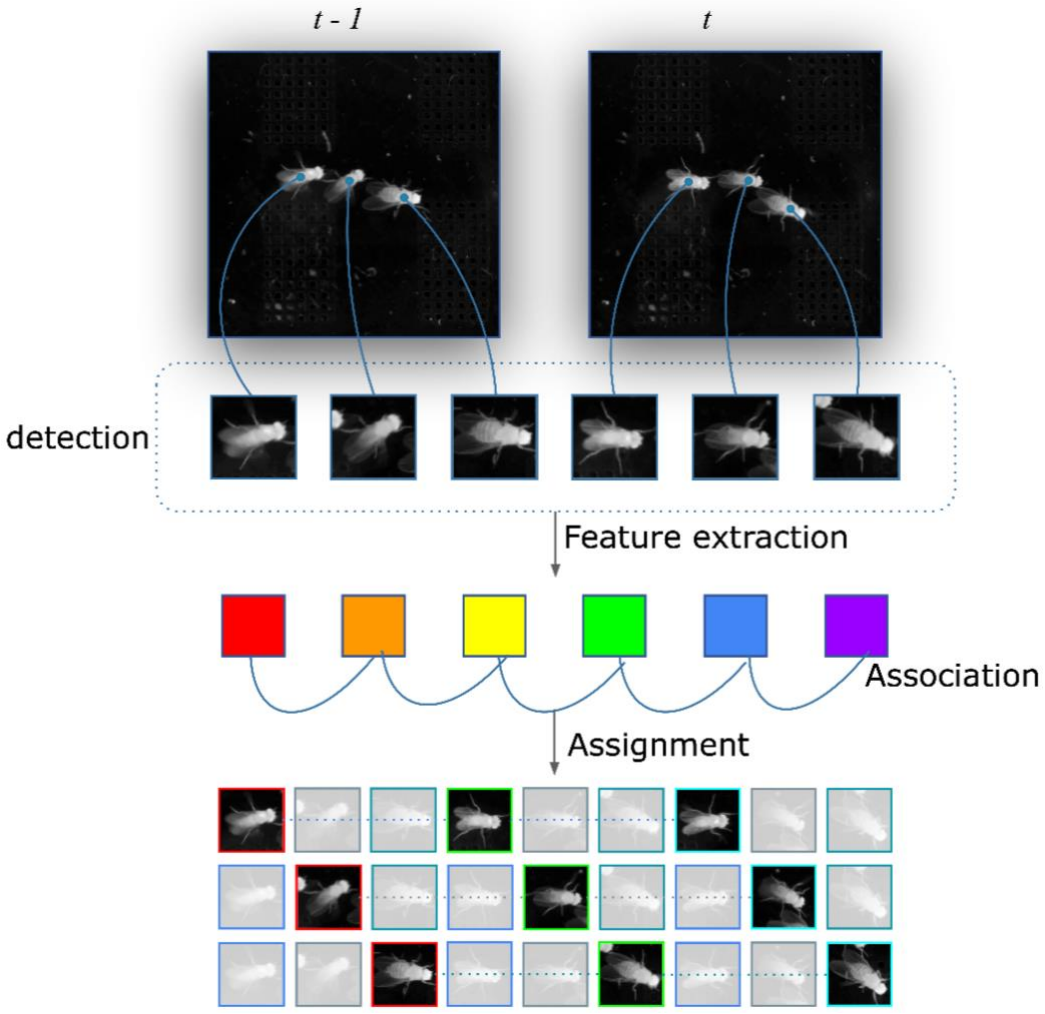

Figure 1.1 Diagram outlining the *tracking-by-detection* framework for MOT.

1 Problem Setup

The problem of Multiple Object Tracking (MOT) consists of following a set of objects of interest throughout an entire video by associating a target ID to a subset of instances through

time (Ciaparrone et al., 2020; Luo et al., 2021). The major issues facing MOT algorithms involve dealing with occlusions during interactions between objects, distinguishing visual appearances, and the initialization (an object entering the frame for the first time) or termination (an object leaving the frame, permanently or not) of tracks (Ciaparrone et al., 2020; Luo et al., 2021). Most MOT algorithms can be classified by whether they use the *Tracking-by-Detection* framework (Ciaparrone et al., 2020; Luo et al., 2021). Furthermore, MOT algorithms can either be *online* (frames are tracked one at a time in sequential order) or *offline* (frames are processed all at once in a batch-wise manner) (Ciaparrone et al., 2020; Luo et al., 2021). These approaches are described below.

2 Tracking-by-Detection

The tracking-by-detection framework is the major way in which MOT algorithms are posed. It is made up of four main steps:

1. Localization: Determining the spatial position of every object to be tracked.
2. Feature Extraction: Computing a set of statistical properties of each object that can be used for distinguishing trajectories.
3. Association: Estimating the probability that a set of objects belong to the same trajectory
4. Assignment: Mapping a set of detections to a set of trajectories such that no two detections in the same frame can have the same trajectory.

Localization (Detection)

The localization or detection step consists of finding all objects of interest within the set of frames that will be tracked. These detections can be represented as centroids, bounding boxes, poses or segmentations. While MOT algorithms are sensitive to the performance of

the detection algorithms (i.e., false positives and false negatives can lead to erroneous results), most current detection methods are quite robust and are not the cause of most issues [6].

Feature Extraction

Feature extraction is the process of summarizing the statistical properties of each detection to be used in the association step. Most MOT algorithms use some combination of a visual and motion model (Ciaparrone et al., 2020; Luo et al., 2021). The visual model computes features that describe an object according to some local, regional or statistical measure. Some examples are optical flow (local), histogram of oriented gradients (regional), and normalized cross correlation (statistical) (Luo et al., 2021). These features are then fused through an application of some operation $f: \mathbb{R}^{n \times d} \rightarrow \mathbb{R}^{n}$ such as concatenation, addition, or multiplication.

Association

Association can be described as computing $P(t_x = i \mid f)$ where t_x is the track id of instance x, i is the track index, and f, is the features provided. Usually, this distribution is estimated by computing a pairwise distance matrix which compares all detections in a certain set of frames. For instance, one could measure the correlation or cosine similarity between each pair of vectors to get the association matrix.

Assignment

The final step of the tracking-by-detection framework involves assigning each detection to a trajectory. This is also where offline and online trackers usually differ.

Online Tracking

In online tracking, the detections in only one frame are assigned. This involves first assigning all instances in the first track to a separate detection. Then, usually in a pairwise fashion, the detections in frame *t* are associated with the trajectories from frame *t-1*. Finally, a linear assignment algorithm such as the Hungarian Algorithm(Kuhn, 1955) is used to assign each instance to a trajectory in a one-to-one fashion.

Offline Tracking

In offline tracking, rather than only assign one frame at a time, the detections from multiple frames are compared and assigned. This usually needs a more global assignment algorithm such as Integer Linear Programming (Malin-Mayor et al., 2023) or *k*-shortest paths (Berclaz et al., 2011). Offline algorithms are usually more robust due to the ability to use global information to fix past mistakes but are computationally expensive to optimize.

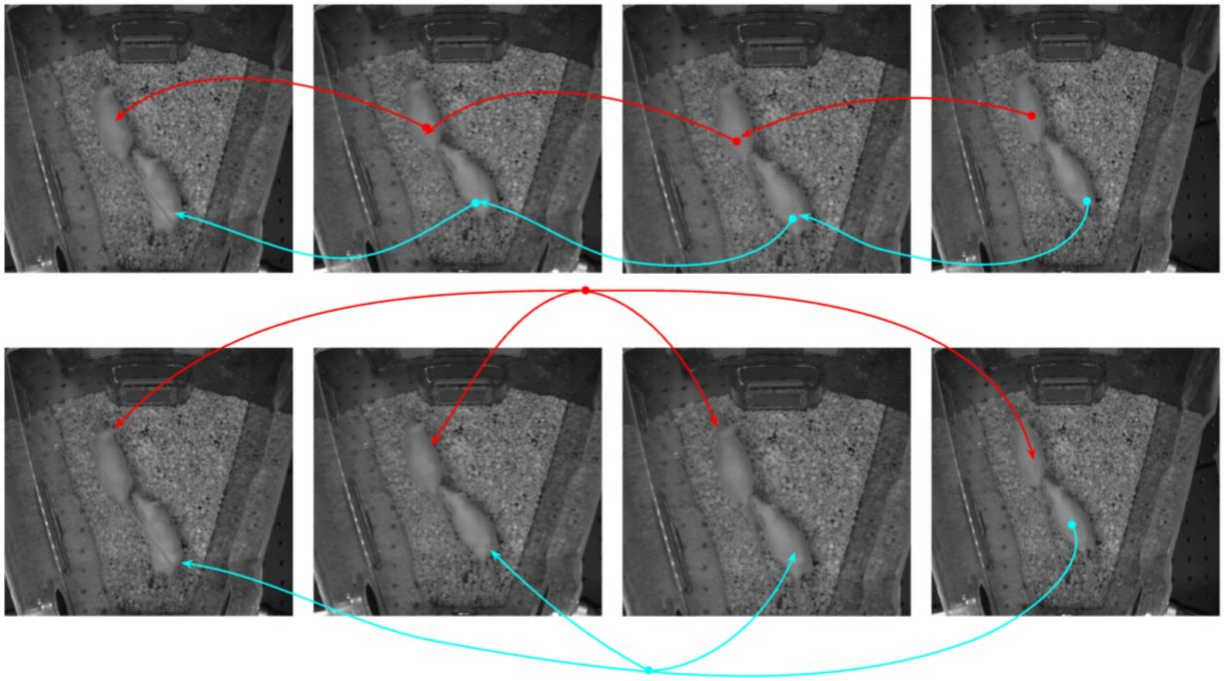

Figure 1.2 Local (online, *top*) vs Global (offline, *bottom*) tracking.

3 Applications

Traditionally, multiple object tracking has applications in pedestrian tracking, self-driving cars, robotics, and action recognition (Ciaparrone et al., 2020). Within biological contexts, action recognition is also of interest (also known as behavioral classification or clustering). Furthermore, there is an active interest in quantifying and analyzing the trajectories of biological agents (Datta et al., 2019; Norregaard et al., 2017; Pereira et al., 2020). For instance, systems neuroscientists may want to correlate the trajectory of an animal during behaviors such as navigation or courtship with recordings from a certain region of the brain to disentangle how that circuit's computations contribute to the behaviors expressed (Datta et al., 2019; Pereira et al., 2020). At the same time, the study of how an animal's behavior changes as a disease progresses could lead to new biomarkers for early intervention or avenues for intervention to reduce the symptoms of the disease. Finally, studying how the motion of

subcellular components contribute to the health of the cell may lead to treatments that restore function to a cell by recovering the movement of its organelles (Norregaard et al., 2017).

Chapter 2 Literature Review

In this section, we review the current state-of-the-art approaches to multiple object tracking in classical and biological settings, with a particular emphasis on deep learning-centric approaches. Classical approaches to multiple object tracking involve extracting hand-crafted statistical features, computing a similarity function, then assigning trajectories based on those similarities. However, with the huge success of deep learning for computer vision tasks, most modern approaches develop methods that use artificial neural networks in at least one of the steps described above. Because CNNs and transformer-based architectures form the foundation for many state-of-the-art approaches, we briefly describe them here.

1 Convolutional Neural Networks

The convolutional neural network was first proposed by (Lecun et al., 1998) in the 1990s before reaching the mainstream in 2012 when (Krizhevsky et al., 2012) demonstrated state-of-the-art classification performance on the ImageNet (Deng et al., 2009) dataset with the AlexNet architecture. The CNN is composed of a sequence of convolutional operations which perform element-wise multiplication between filters (also called kernels) and patches of an input (usually an image) to output feature maps. Through training, these networks learn useful hierarchical features for completing the task such as edges or textures. These layers are usually followed by an activation layer such as ReLU to introduce non-linearities into the network. To compress the input and make the networks more efficient (Li et al., 2022), a pooling layer is then applied to

summarize a neighborhood of each feature map through operations like taking the max or average.

Additionally, to overcome the vanishing gradient problem and enable extremely deep CNN architectures, (He et al., 2015) proposed the residual block which introduces skip connections to the standard convolutional block. Skip connections allow gradients to skip layers, if necessary, in order to flow through the entire network. Specifically, the residual block learns the difference between input and output (the residual mapping) by passing the input through an identity path and operations on the input through the residual path.

Finally, after a suitable amount of convolution and pooling operations, the feature maps are often flattened into a single vector, called an embedding, which can then be used downstream for a multitude of tasks. For instance, it can be used as the input to a multi-layer perceptron for classification or regression. In the context of multiple object tracking, the embedding serves as a set of learned features that can be used for computing similarities between instances for association and assignment. For a more rigorous discussion of CNN architectures see (Li et al., 2022)

2 Convolutional Neural Networks in MOT

Here we cover some state-of-the-art applications of CNNs for multiple object tracking. Within the MOT pipeline, CNNs are often the standard approach for localizing objects and computing features (Ciaparrone et al., 2020) These deep features are then compared downstream with a distance metric such as cosine similarity and tracking is handled with classical linear assignment algorithms such as Hungarian Matching (Kuhn, 1955).

CNN based Detectors

For bounding-box regression-based approaches, two popular approaches are the R-CNN (**R**egions with CNN Features) (Girshick et al., 2014) and the YOLO (**Y**ou **O**nly **L**ook **O**nce) (Redmon et al., 2016) models.

The basic idea behind R-CNN (Girshick et al., 2014) is to search for a subset of patches within an image called region proposals using a selective search algorithm. Then, these patches are fed into a CNN for feature extraction, and a bounding box/class ID is predicted. Further work (Girshick, 2015) proposes speeding up the model through working in the feature space by first generating convolutional feature maps then searching for region proposals. Finally, current state of the art models replace the selective search algorithm with a CNN that predicts proposal regions (Ren et al., 2016). Additionally, the mask r-cnn (He et al., 2018) extends the r-cnn to segmentation by adding a mask prediction branch to the model.

On the other hand, the YOLO model (Redmon et al., 2016) moves away from region-based algorithms by splitting the image into a grid and predicting a class probability along with bounding box coordinates. Regions with a high class probability are used for detections. YOLO models use a single convolutional network but struggle with smaller objects due to spatial constraints.

While YOLO and R-CNN models can be used for other localization tasks such as segmentation, another common approach for both pose-estimation and segmentation is the use of U-Net (Ronneberger et al., 2015) based architectures. The U-Net consists of a down-sampling branch which compresses the input, usually to a single feature vector, along with an up-sampling branch which brings the image back to its original size.

However, unlike auto-encoders which reconstruct the input, UNets instead predict a probability map where each pixel is the probability that it belongs to a certain class (i.e. background vs foreground or key-point etc).

CNNs for feature extraction

CNNs are the most used deep learning model for feature extraction due to their natural inductive biases for learning useful low-level spatial features, and ability to compress images into lower dimensions. The standard way CNNs are used for feature extraction is by removing the fully-connected layers that follow the final pooling layer of a classification model. This is referred to as a "backbone" as the one-dimensional feature vector it outputs can be used for a variety of tasks. Common backbones in the computer vision community are Residual Networks (He et al., 2015), EfficientNets (Tan and Le, 2020), and ConvNeXts (Liu et al., 2022) for a range of factors including performance, model size, inference speed etc. Since many of these networks have been pretrained for classification, they may already have good feature detectors for tracking. Thus, the pretrained weights are simply fine-tuned in a process known as transfer learning. Another common approach is the use of "Siamese" (Chicco, 2021) networks which feed two different detections into two separate CNN branches and are trained jointly to learn features from each detection that indicate whether they are the same instance.

3 The Transformer Architecture

Originally proposed by (Vaswani et al., 2023) in 2017 for natural language processing (NLP), the Transformer architecture has achieved state-of-the-art across multiple machine learning domains ranging from computer vision, audio-processing, and robotics to protein

folding and drug design (Lin et al., 2022). Although the transformer was designed for NLP, we will describe it in terms of general sequence modeling.

The input to the transformer module is a set of d dimensional vectors one for each element in the sequence (i.e. a $b \times n \times d$ tensor where b is the batch size, n is the number of elements in the sequence and d is the dimensionality of the embedding vector). Then, a set of query, key, and value vectors are computed by multiplying the input embeddings by a set of learned weight matrices. When the query, key, and value embeddings are identical, then it is called *self*-attention but when the queries are different from the keys and values (which are the same), it is known as *cross*-attention. Each query is then compared to each value through the attention mechanism, which is the dot product between the two scaled by the dimensionality of the embedding. This "attention-score" indicates the weight each input should have. Next, the attention score is multiplied by the value to get the attention-weighted representation of the sequence.

Because the self-attention mechanism is permutation invariant (changing the position of each element in the sequence doesn't affect the output), a positional embedding is added to the input embedding before the attention score is computed to inject information about the absolute location of the element in the sequence. Traditionally, these positional encodings are a fixed sinusoidal mapping, but in modern approaches the encodings are learned. Finally, the self-attention mechanism is applied several times in parallel (multi-head attention). Because the attention weights are learned, each head learns a different dependency between the different elements in the sequence. A set of multi-head attention layers followed by a projection-head and layer normalization is called a transformer block. Thus, a full transformer network is composed of a stack of transformer layers that encode the contextual relationships between each element of

the sequence (the encoder), and when necessary, a decoder which transforms the encoder output into a set of prediction probabilities used to solve the desired task. For a complete review of the history and current improvements on transformer architectures see (Lin et al., 2022).

4 The Transformer Architecture in MOT

Here, we cover some of the state-of-the-art approaches for MOT that utilize the transformer architecture. Although most tracking-by-detection approaches using deep learning use a CNN-based detector, such as YOLO or Mask-RCNN, we would like to highlight the DETR (Carion et al., 2020) framework as it forms the basis and inspiration for many of the other transformer-based MOT algorithms discussed below.

DETR

The **De**tection **Tr**ansformer is a general framework for processing intra-image relationships, developed specifically for the object-detection task. DETR is composed of a CNN backbone and an *n*-layer transformer encoder-decoder architecture. The CNN backbone extracts features for each position in an image and outputs a $d \times h \times w$ tensor which is flattened into $d \times h \cdot w$ to be used as input into the transformer where d is the dimensionality of the embedding vectors and h, w are the height and width of the image respectively. The transformer encoder-decoder then globally processes all positions to compute the bounding boxes for each detection and optionally a class identity.

The Trackformer (Carion et al., 2020) architecture introduces a *tracking-by*-attention framework by essentially adding another DETR architecture to the original for the temporal step. That is, in an online fashion, trackformer uses a CNN backbone to encode each detection output

from DETR into a feature vector which is then queried against the tracks from the previous frame to extend the track. On the other hand, the **M**ultiple **O**bject **Tr**acking Transformer (MOTR) (Zeng et al., 2022) modifies DETR by replacing the object queries with track queries which are updated each frame at a time. Thus, MOTR uses a single CNN-transformer pair instead of two sets. Interestingly, TransMOT (Chu et al., 2021), a similar approach to MOTR, attempts to learn the association and assignment part through modeling the MOT problem as a sparse spatiotemporal graph.

Some recent works try to move away from the tracking by detection framework into tracking points – thereby eliminating dependency on the detection step. Point tracking approaches such as CoTracker (Karaev et al., 2023) and TAPIR (Doersch et al., 2023) attempt to locate a set of single points at every frame. They do this by querying the local features in a neighborhood around a set of points throughout time against each other. Although these methods are efficient and are not affected by upstream steps, they lose out on global spatial context which hurts their ability to resolve occlusions.

5 Current approaches for multiple object tracking in biological contexts

In this section we shift attention towards approaches that are specifically designed for biological contexts. We divide this section into microscopy tracking and animal tracking.

Microscopy Tracking

Within the context of microscopy, tracking is usually divided into cell-tracking (Maška et al., 2023), organelle tracking (Norregaard et al., 2017), and particle tracking (Chenouard et al., 2014). Particle tracking, the smallest spatial setting, involves tracking single molecules such as motor proteins. Unlike the other two settings, particle tracking

usually does not have to deal with cases such as divisions and fusions. However, these particles' trajectory often follow a Brownian process and contain limited visual features (Chenouard et al., 2014). Organelle tracking is similar to particle tracking but focuses on larger objects such as mitochondria and lysosomes. In this setting, objects can often fuse and divide but move in much more stereotyped patterns such as along a neurite (Norregaard et al., 2017). Finally in cell tracking, we track whole cells, which divide but never fuse (Maška et al., 2023).

Tracking approaches can either handle divisions through lineage tracing (Malin-Mayor et al., 2023) (i.e. keeping track of parent history) or treat each division as a new object appearance. Because the microscopy community has interest in fine-grained morphological features, the detection representation is classically posed as segmentation. There are many high-accuracy segmentation approaches such as CellPose (Stringer et al., 2021), Weka (Arganda-Carreras et al., 2017), and StarDist (Schmidt et al., 2018; Weigert et al., 2020) which can be used for the detection step. Standard approaches to microscopy tracking usually involve heuristic based methods. For example, TrackMate (Ershov et al., 2022) uses deep learning approaches for the detection step, but then uses an overlap tracker for assignment. TrackMate also allows users to include certain rules such as minimum track lengths, maximum distances, and more to filter tracklet proposals. On the other hand, some approaches use a global graph based optimization such as Integer Linear (Malin-Mayor et al., 2023) Programming and Dynamic Programming (Maška et al., 2023). Finally, many recent approaches use deep learning in a similar fashion as above. Interestingly, as of 2023, the results of the CellTracking Challenge showed "no significant difference in performance between machine-learning and non-machine

learning approaches" (Maška et al., 2023). Thus, there is significant interest in developing a robust deep learning-based tracking method.

Animal Tracking

There has been a surge of interest in studying the behavior of animals in naturalistic settings. This involves tracking multiple free-moving animals in both lab cage and open field settings. While microscopists like to use segmentation representations for the analysis of (sub)cellular morphology, neuroethologists are more interested in interactions between animal body parts. Thus, objects are represented as poses detected through softwares such as SLEAP (Pereira et al., 2022) and DeepLabCut (Mathis et al., 2018). While these approaches include tracking algorithms, they rely on heuristics based on appearance and motion. On the other hand, idTracker (Romero-Ferrero et al., 2019) is an approach that uses deep learning to detect frames of interest where difficult cases such as occlusions occur. Then, a second network compares the crossover events to current tracks to resolve identities. However, there is not a standard end-to-end deep learning-based tracking method for animal tracking specifically, let alone both animal and microscopy tracking.

Chapter 3 Methodology

1 Datasets

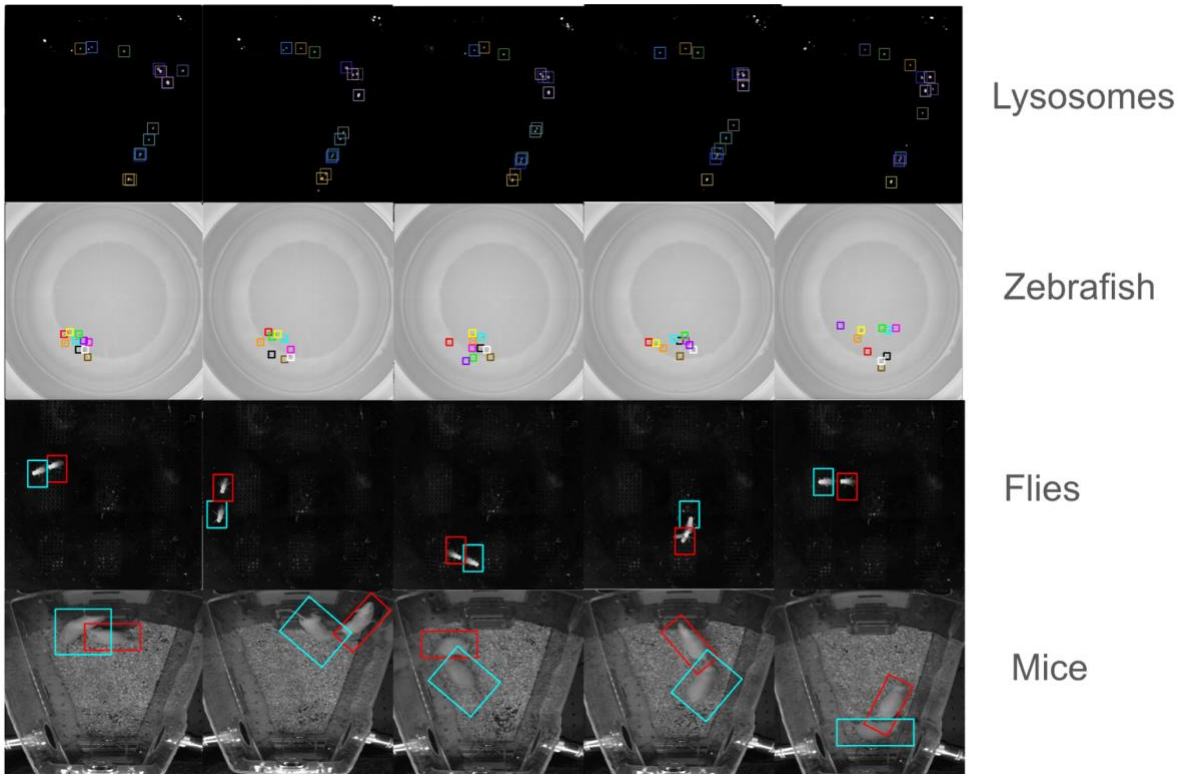

Figure 3.1 Sample Frames from each dataset.

For testing DREEM on the animal tracking dataset we train and test on a diversity of animal types, environmental conditions, and population sizes. Sample frames from each dataset can be viewed in Figure 2.1. Namely, we evaluate on tracking zebrafish, flies, and mice to cover the range of animal sizes and visual features.

For zebrafish, we use the publicly provided dataset curated by (Romero-Ferrero et al., 2019). This data consists of 4 videos of 10 zebrafish swimming around for around 40 thousand frames recorded at 30 frames/second. Next, we rely on the *flies13* dataset collected for (Pereira et al., 2022). This dataset contains a collection of clips of two flies interacting, a single 10 thousand

frame clip of 4 flies in a group setting, and a 20 thousand frame clip of 8 flies together, all recorded at 150 frames per second.

Finally, we utilize 3 different datasets each containing two mice with different combinations of fur colors: white, black, and agouti (brown). We first track the *mice (home cage)* dataset used in (Pereira et al., 2022). This dataset contains 40 videos with 2400 frames, recorded at 40 frames per second. The *mice (btc)* dataset is a 24-hour continuous homecage monitoring setup recorded by the Salk Behavioral Testing core for the study of behavioral biomarkers for ALS disease progression. Finally, the *SLAP M74* dataset is a 3D pose estimation dataset curated for testing multi-view pose estimation and view synthesis. These mice are also in home-cage setups with increasing levels of complexity. Here, we focus solely on tracking the top view of the mice.

Towards the microscopic length scale, we focus on two major datasets. The first is an Airyscan confocal imaging dataset recording lysosomal dynamics in both wildtype and Charcot-Marie-Tooth phenotype neurites. This data was first tracked with (Ershov et al., 2022) and then proofread in the SLEAP GUI to generate gold-standard ground-truth data. For the second dataset, we utilize the publicly available 2d+t cell-tracking challenge data (Maška et al., 2023). While we train on all the data that could fit in a single GPU's memory, we only test on the Mouse muscle stem cells in hydrogel microwells and the HeLa cells on a flat glass.

2 Data Preprocessing

For all datasets, we first augment each video using a random selection of the following augmentations: random rotations between 0-45 degrees, gaussian/motion blurs, and random brightness/contrast adjustment. Each augmentation was selected with a probability of 0.3 and

applied to each frame independently. Next, we crop a bounding box centered around each instance and pad the bounding box by 5 pixels. For animals, we first try to use a user-defined anchor (e.g., the thorax for flies) as the bounding box centroid, but if that is unavailable then we default to the centroid computed for the predicted pose skeleton. However, for the microscopy images we always use the centroid of the mask or trackmate output. The bounding box size is determined by the size of the instance and can range from 30 to 400 pixels. Finally, we chunk up each video separately into smaller clips as input to the model.

3 Architecture

We base DREEM on an approach known as Global Tracking Transformers (Zhou et al., 2022). Unlike in traditional MOT methods, because there are already many robust tools for object detection in biological settings, we assume detections are fixed and instead learn a mapping that associates a set of all detected instances in a video to a set of trajectories through time. This allows us to use a light-weight model consisting of only a CNN-based feature extractor and a transformer encoder-decoder architecture. Thus, the input into the feature-extractor is a set of crops centered around each instance detected throughout the video. From the feature extractor, we get an $N \times D$ set of feature vectors where N is the number of instances in the clip and D is the dimensionality of the feature vector space (Fig. 3.2a). We then send this output into a single layer transformer encoder-decoder block which outputs an $N \times N$ association matrix $\mathbf{A}$, where $\mathbf{A}_{i,j}$ represents the probability that instance i belongs to the same trajectory as instance j (Fig 3.2b).

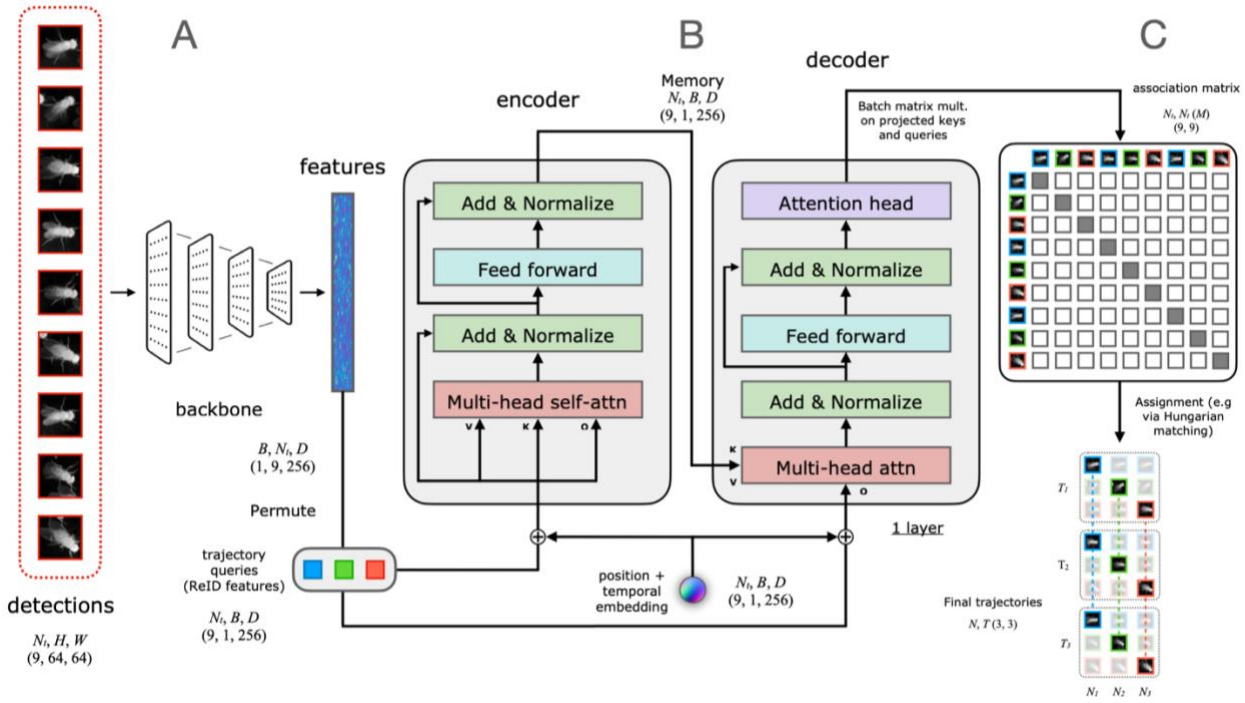

Figure 3.2 Architecture Diagram.

From left to right: a) First all detected instances in *T* frames are cropped around an anchor point and fed into a CNN-based feature extractor. b) The output of this feature extractor (reid features) is then passed into a transformer encoder-decoder with 1 encoder and 1 decoder layer which learns the association between every instance. c) This association matrix is then used as weights for an assignment algorithm which outputs trajectories for each instance.

4 Training

We train all models for 50 epochs, where each batch is a clip containing 32 consecutive frames with at least one detection in the window. While the frames are always consecutive, during training we always shuffle the clips to prevent overfitting. However, during validation order is preserved. Usually, we train on all clips but when the dataset size becomes too large or for sample efficiency experiments, we may randomly sample a subset of clips to train on. We utilized the same hyperparameters for all training runs after conducting a standard grid search. Specifically, we train a network made up of a resnet18 (He et al., 2015) backbone and a single linear layer to project the features into a 128-dimensional vector space, and a transformer

encoder-decoder. We use a single layer and a dropout rate of 0.1 for both the encoder and the decoder. We utilize learned positional and temporal embeddings rather than the standard sinusoidal implementation described in (Vaswani et al., 2023). Finally, we optimize a cross-entropy loss between the assigned trajectories and ground-truth ones as detailed by (Zhou et al., 2022) using the Adam optimizer (Kingma and Ba, 2017) with a learning rate of 10^{-4} and no weight decay. We decay the learning rate by a factor of 0.5 every 5 epochs if the loss fails to improve by at least 10^{-3}. While we always monitor the validation loss for saving model checkpoints, we also save models corresponding to the lowest number of identity switches during validation. Finally, we monitor for at least a validation loss decrease of at least 0.1 every 10 epochs for early stopping.

5 Post-processing

During inference, we apply a set of post-processing reweighting methods to the association matrix based on spatiotemporal distances. Specifically, we multiply each entry by $0.9^{\Delta t}$ where Δt is the relative temporal distance in frames between instance *i, j*. Then, we weight the entry by the intersection-over-union between instance *i, j*. Finally, we set a user defined confidence score of 0.1 for assigning tracks. If the association between an instance and a track is less than the confidence score, and there are still tracks to be created (based on the max tracks defined by the user), a new track is created.

6 Sliding Inference

Finally, during inference, this association matrix is used in an assignment algorithm such as Hungarian Matching (Kuhn, 1955) to output trajectory assignments for each instance.

Specifically, we take a sliding window approach. In the initial frame, each instance is initialized as a track and added to a distinct queue per track. Then, we compute the association score between the instances in a single frame and the trajectories in the queue. We retain the last 8 frames worth of instances for each track to query against. If a trajectory fails to appear within a user-defined window (e.g., 32 frames) then that track is terminated and removed from the queue.

Acknowledgements

Chapter 3, in part is currently being prepared for submission for publication of the material. Prasad, Aaditya; Sheridan, Arlo; Tu, Vincent, Schiavon, Cara; Afshar, Shayan; Couris, Theo; Nono Nathaniel; Intal, Marcus; Abbara, Zaher; Evans, Kaela; Manor, Uri; Pereira, D. Talmo. The thesis author was the primary investigator and author of this material.

Chapter 4 Results

1 Baseline Results and Comparisons to State of the Art

We demonstrate the performance of DREEM as a function of identity switches per frame on a wide variety of datasets including flies and mice (Pereira et al., 2022), zebrafish (Romero-Ferrero et al., 2019), and lysosomes (Fig. 4.1). We train a model on each of these datasets separately and then evaluate on a held-out set. We see across the board that DREEM can produce trajectories with minimal identity switches, indicating that it can learn a motion model for a wide variety of applications.

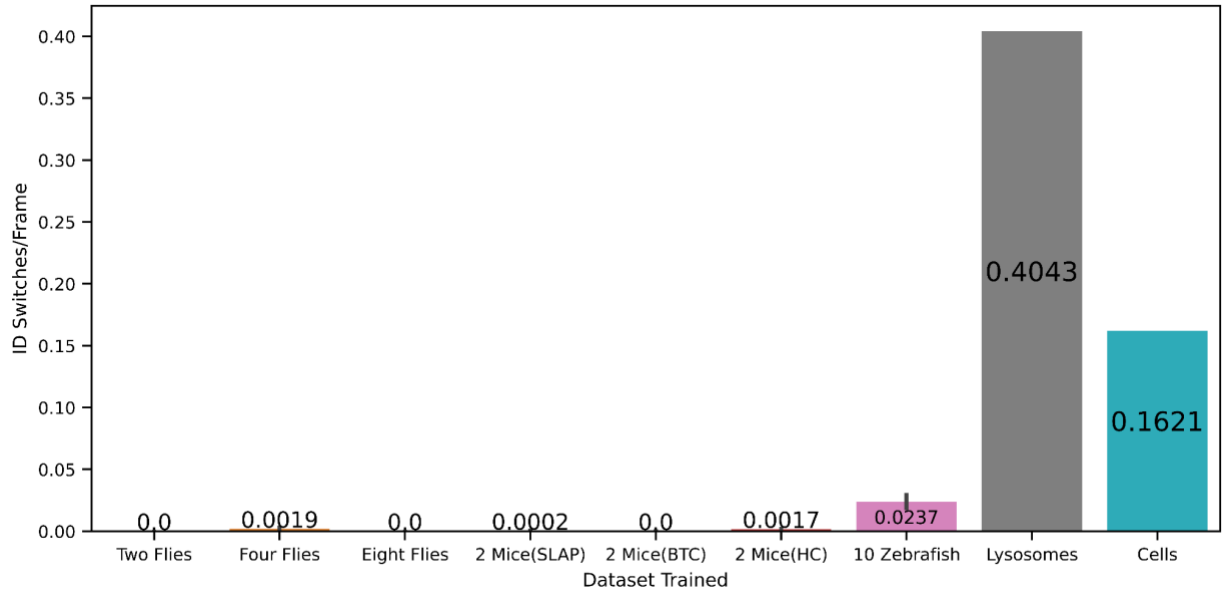

Figure 4.1 DREEM's performance on a variety of test datasets.

DREEM also out-performs well relative to state-of-the-art methods in the field. Specifically, we compare against the baseline tracker in SLEAP, which uses a flow-shift-based tracking approach (Pereira et al., 2022), in the mice, zebrafish and lysosome tasks. We find that

DREEM consistently outperforms SLEAP with upwards of a 90% reduction in identity switches, especially in harder cases such as the lysosomes and zebrafish (Fig. 4.2).

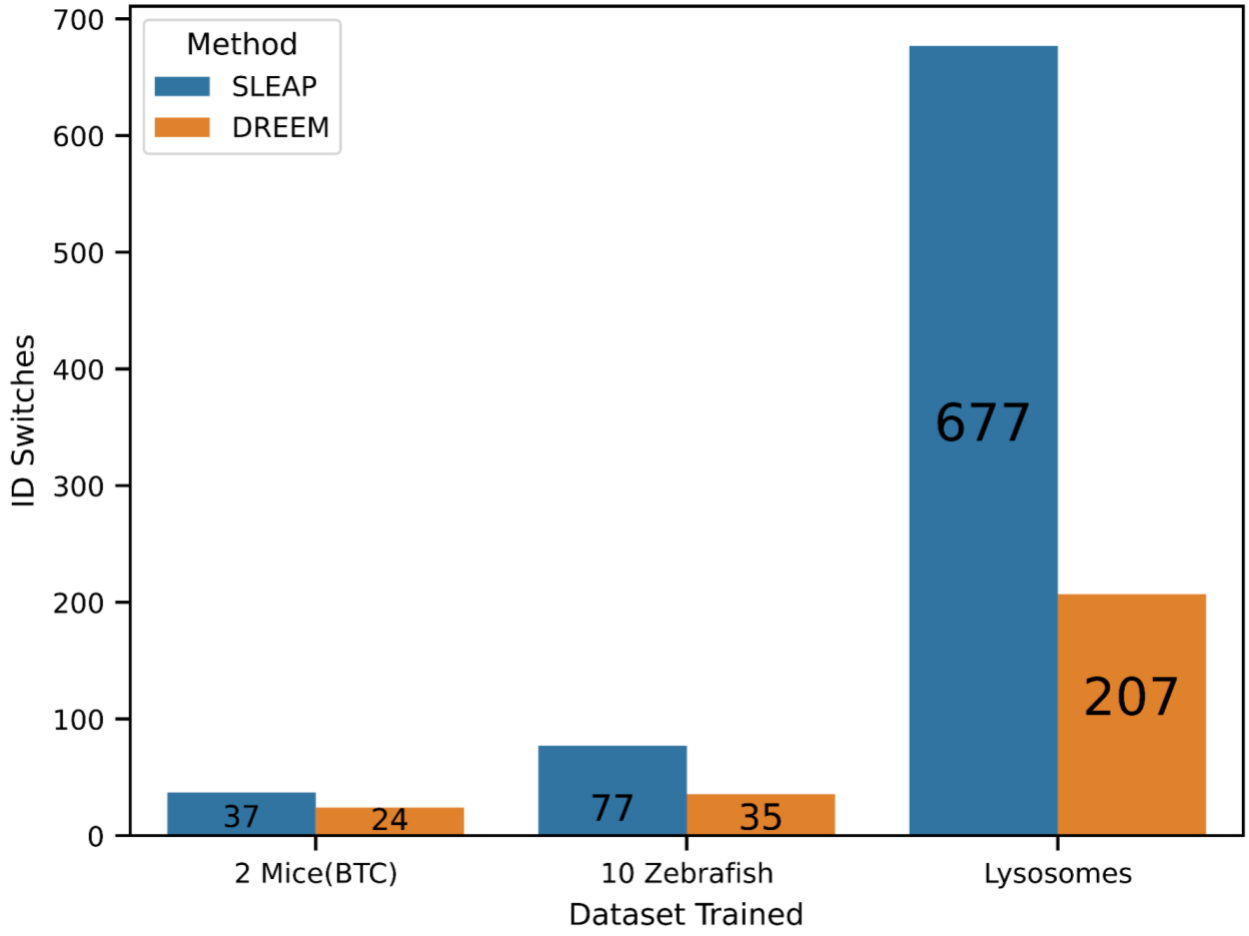

Figure 4.2 Comparisons between DREEM and SLEAP's optical flow tracker.

2 Sample Efficiency

Although DREEM works extremely well for biological multiple object tracking, we needed to know how practical the model would be for use in a general audience. Specifically, if DREEM needs an infeasible amount of data to train on to get good performance, it may not be suitable for many researchers who may be in a data-limited regime or for whom it may be infeasible to ground-truth data. Thus, we conducted a sample efficiency experiment where we trained networks on a random subset of the dataset with n_i = {1, 2, 5, 10, 25, 50, 100, 200, 500,

N} clips of length T = 32 frames, where N is the number of chunks in the full dataset. We see that after n_i = 100 clips (3200 frames), DREEM tends to not benefit from additional training samples (Fig. 4.3). We hypothesize that this is because DREEM mainly needs to see a representative sample of possible movements the objects can make to learn a good association mapping, and many chunks may have movements the model has already seen and thus are unnecessary.

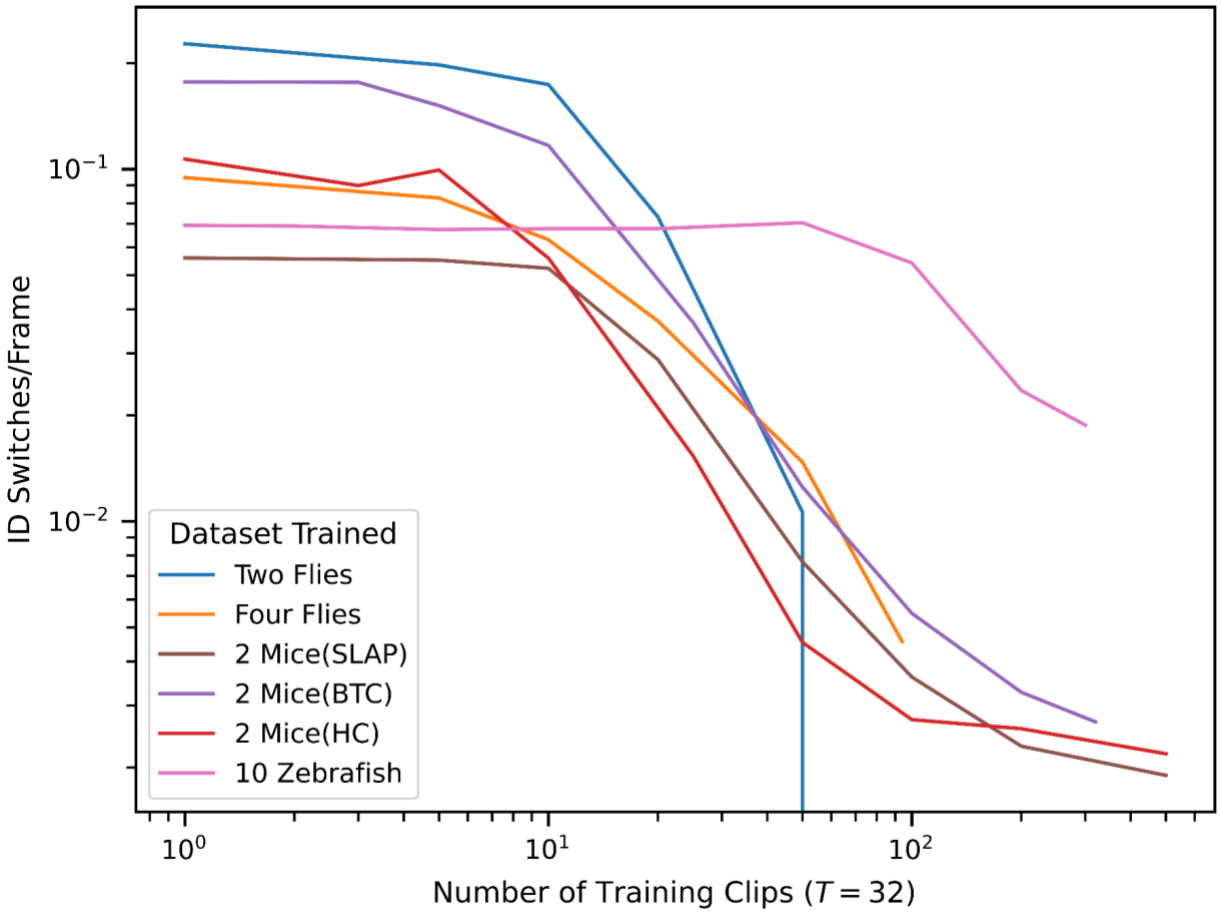

Figure 4.3 DREEM's accuracy as a function of training set size.

3 Out-of-Distribution (OOD) Generalization

We next investigated how well DREEM could generalize to out-of-distribution scenarios such as varying environmental conditions as well as number of trajectories.

Number of Trajectories.

If we need to train on the exact same number of instances as we run inference on, then this method may not be suitable, especially in cases like microscopy where we may not know how many objects we are tracking beforehand. We find that when trained on different train and test dataset sizes, DREEM generalizes well to different trajectory counts. (Fig. 4.4)

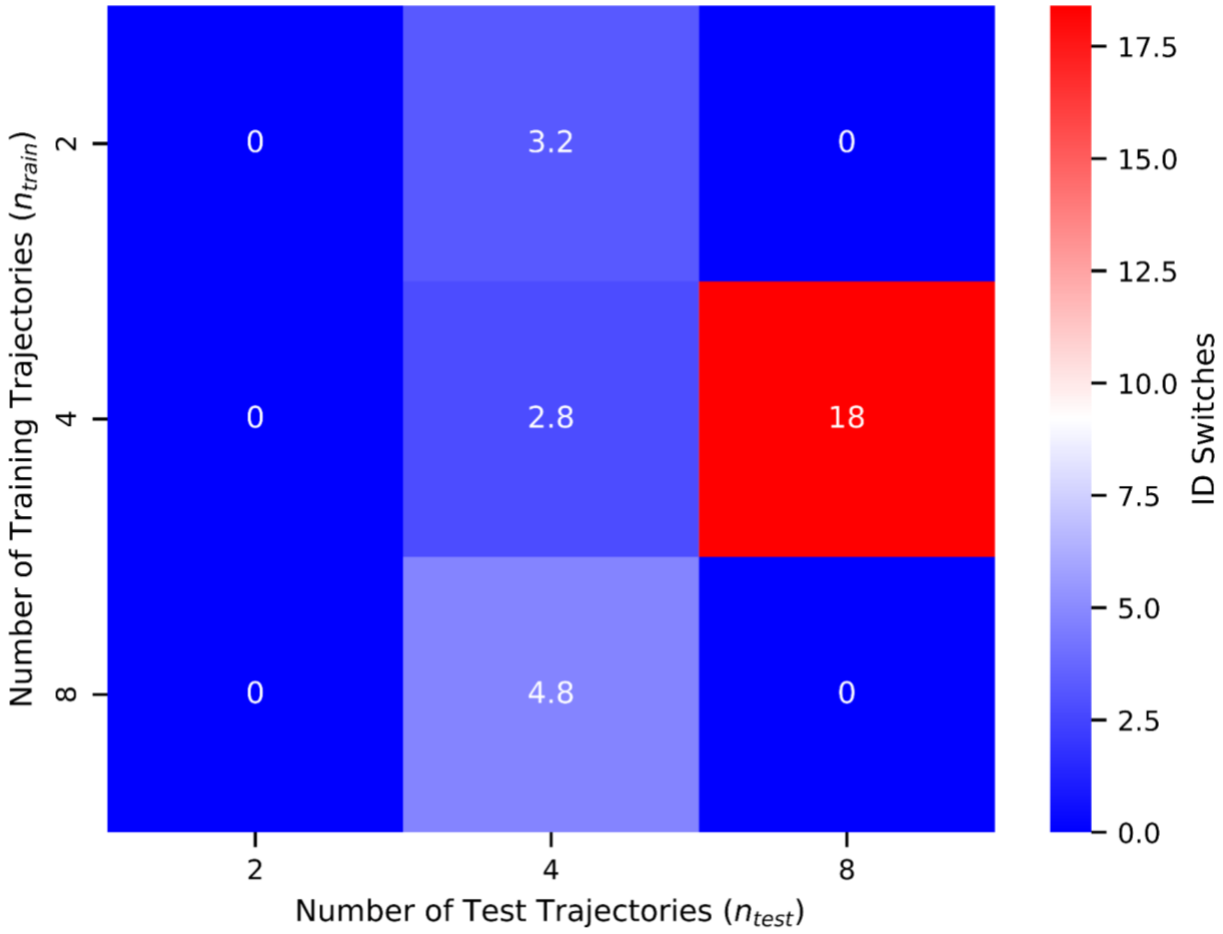

Figure 4.4. DREEM's OOD accuracy with respect to number of trajectories.

Environmental Conditions.

We also demonstrate that DREEM can generalize to different experimental settings by training on videos of mice in various home-cage set ups and running inference on out-of-

distribution ones (Fig 4.5). We find that fine-tuning on a small subsample of data in both out-of-distribution settings can recover full performance. This opens up the possibility of labs sharing pretrained models with each other thus reducing the need to generate more ground truth data. It may even be possible to train a single large "foundation model" for tracking a specific biological agent.

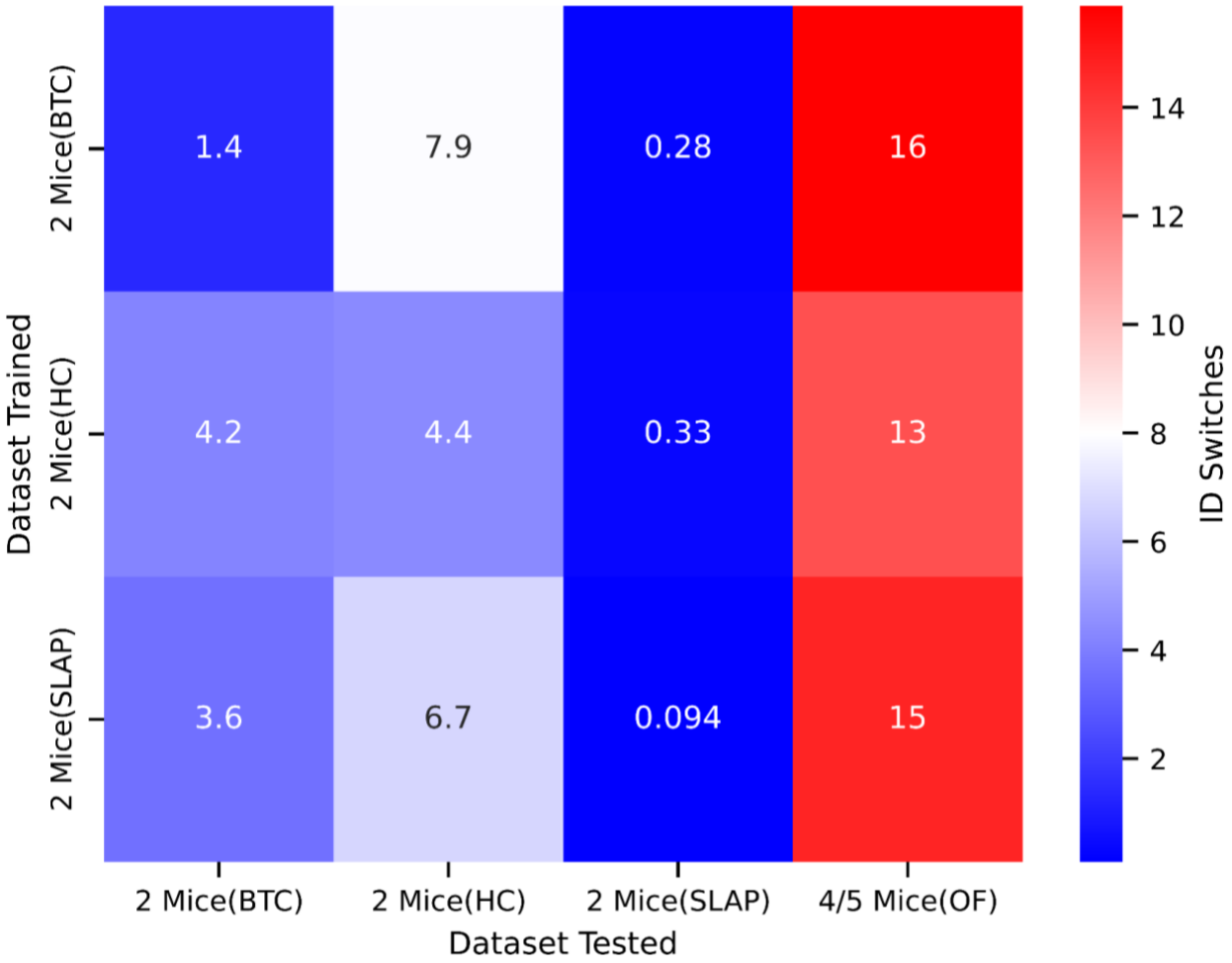

Figure 4.5 DREEM's OOD accuracy with respect to environmental conditions.

Acknowledgements

Chapter 4, in part is currently being prepared for submission for publication of the

material. Prasad, Aaditya; Sheridan, Arlo; Tu, Vincent, Schiavon, Cara; Afshar, Shayan; Couris, Theo; Nono Nathaniel; Intal, Marcus; Abbara, Zaher; Evans, Kaela; Manor, Uri; Pereira, D. Talmo. The thesis author was the primary investigator and author of this material.

Chapter 5 Discussion

Here, we introduced DREEM (**D**REEM Reconstructs **E**very **E**ntity's **M**otion), a deep learning framework which leverages a transformer-based architecture to directly learn the associations between objects in a large temporal context (Zhou et al., 2022). DREEM learns this global association between all instances in a clip by first extracting the visual features from crops centered around each instance. A single layer encoder-decoder model is then applied in this feature space to output an $n \times n$ association matrix. Finally, this association matrix is used as the weights for an assignment algorithm such as Hungarian matching (Kuhn, 1955). We demonstrated that DREEM enables the training of state-of-the-art models for biological MOT. Finally, we showed that DREEM is sample efficient and can transfer seamlessly across a variety of diverse settings, enabling use in a wide variety of fields.

1 Limitations

One potential trade-off of DREEM's architecture structure is that the computational complexity scales with the number of total detections throughout the video. Furthermore, as DREEM relies on visual features for tracking, performance may suffer in conditions where the visual features are identical (e.g. mice with the same-colored fur) or the resolution is low.

2 Future Directions

Improved Feature Engineering

Further work could investigate developing a better motion model that reduces the reliance of the model on deep visual features. For instance, optical flow is a method for

approximating the statistics underlying the motion of objects between frames. One could imagine the optical flow of a video as input for DREEM by either stacking the image and its optical flow along the channel dimension and using a single CNN encoder to extract features (weight sharing) or by leveraging a separate encoder for the image and optical flows separately (Luo et al., 2021). Other features that may provide more information for solving the association task could take inspiration from the computer vision community's long history of developing hand-crafted features such as Histogram of Oriented Gradients (HoG) and Scale Invariant Feature Transforms (SIFT) (Luo et al., 2021).

Experimenting with Relative Positional Encodings

Furthermore, since the advent of the transformer model, much work has been done exploring the use of relative positional encodings to improve performance. While traditional positional encodings inject information about the absolute position of the embedding with respect to the rest of the sequence (in our case spatial and temporal), relative positional encodings (Huang et al., 2018; Shaw et al., 2018) provide information about the distances between tokens in the sequence. The relative positions of objects should provide useful information for the task of object tracking specifically (i.e. objects close together spatially in adjacent frames should have a higher probability than those further away of being associated while objects temporally far from each other should have a lower probability than those close together). Thus, using relative positional embeddings could alleviate the need for post-processing by letting the model learn relationships between distances specifically. In fact, rotary positional embeddings (RoPEs) (Su et al., 2023) provide absolute and relative positional embeddings simultaneously by encoding the distances in rotations of the embedding vectors.

Self-supervised and Auxiliary Learning

Instead of learning to extract features from multimodal inputs, another approach that could be taken is learning more robust features from a single visual input through transfer, auxiliary, or self-supervised learning tasks. Although the features of natural images are not conducive to fine-tuning on biological videos, the intermediate representations of the biologically-tuned detector model (e.g. from the downsample block of a segmentation or pose UNet) could be fed downstream to the model and tracking could be learnt jointly with detection in a parallel branch. Alternatively, although originally designed for 3D neuronal segmentation, the auxiliary local shape descriptor task (Sheridan et al., 2023) of predicting a 13-dimensional vector describing the 3D local shape could be extended to the temporal dimension instead. Additionally, it may be possible to pretrain the model using a pretext task such as time-arrow prediction (Gallusser et al., 2023) (TAP) or a student-teacher bootstrapping algorithm (Doersch et al., 2024).

Global Assignment-based Tracking

Finally, DREEM's ability to output solely an association matrix could enable the use of more globally optimal algorithms such as Integer Linear Programming (Malin-Mayor et al., 2023) or *k*-Shortest Paths Algorithms (Berclaz et al., 2011) Another interesting line of direction would be to investigate the use of DREEM's encoder embeddings as input into a Graph Neural Network that learns to solve the assignment problem directly (Brasó and Leal-Taixé, 2020). In fact, it may be possible to parameterize the entire architecture as a Graph Transformer (Dwivedi and Bresson, 2021). Overall,

while DREEM is not a single model that can be used on a wide variety of settings, it is a single framework that can be leveraged to link trajectories across multiple domains.

Acknowledgements

Chapter 5, in part is currently being prepared for submission for publication of the material. Prasad, Aaditya; Sheridan, Arlo; Tu, Vincent, Schiavon, Cara; Afshar, Shayan; Couris, Theo; Nono Nathaniel; Intal, Marcus; Abbara, Zaher; Evans, Kaela; Manor, Uri; Pereira, D. Talmo. The thesis author was the primary investigator and author of this material.

REFERENCES

Arganda-Carreras, I., Kaynig, V., Rueden, C., Eliceiri, K.W., Schindelin, J., Cardona, A., Sebastian Seung, H., 2017. Trainable Weka Segmentation: a machine learning tool for microscopy pixel classification. Bioinforma. Oxf. Engl. 33, 2424–2426. https://doi.org/10.1093/bioinformatics/btx180

Berclaz, J., Fleuret, F., Turetken, E., Fua, P., 2011. Multiple Object Tracking Using K-Shortest Paths Optimization. IEEE Trans. Pattern Anal. Mach. Intell. 33, 1806–1819. https://doi.org/10.1109/TPAMI.2011.21

Brasó, G., Leal-Taixé, L., 2020. Learning a Neural Solver for Multiple Object Tracking. https://doi.org/10.48550/arXiv.1912.07515

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S., 2020. End-to-End Object Detection with Transformers. https://doi.org/10.48550/arXiv.2005.12872

Chenouard, N., Smal, I., de Chaumont, F., Maška, M., Sbalzarini, I.F., Gong, Y., Cardinale, J., Carthel, C., Coraluppi, S., Winter, M., Cohen, A.R., Godinez, W.J., Rohr, K., Kalaidzidis, Y., Liang, L., Duncan, J., Shen, H., Xu, Y., Magnusson, K.E.G., Jaldén, J., Blau, H.M., Paul-Gilloteaux, P., Roudot, P., Kervrann, C., Waharte, F., Tinevez, J.-Y., Shorte, S.L., Willemse, J., Celler, K., van Wezel, G.P., Dan, H.-W., Tsai, Y.-S., de Solórzano, C.O., Olivo-Marin, J.-C., Meijering, E., 2014. Objective comparison of particle tracking methods. Nat. Methods 11, 281–289. https://doi.org/10.1038/nmeth.2808

Chicco, D., 2021. Siamese Neural Networks: An Overview, in: Cartwright, H. (Ed.), Artificial Neural Networks, Methods in Molecular Biology. Springer US, New York, NY, pp. 73–94. https://doi.org/10.1007/978-1-0716-0826-5_3

Chu, P., Wang, J., You, Q., Ling, H., Liu, Z., 2021. TransMOT: Spatial-Temporal Graph Transformer for Multiple Object Tracking. https://doi.org/10.48550/arXiv.2104.00194

Ciaparrone, G., Sánchez, F.L., Tabik, S., Troiano, L., Tagliaferri, R., Herrera, F., 2020. Deep Learning in Video Multi-Object Tracking: A Survey. Neurocomputing 381, 61–88. https://doi.org/10.1016/j.neucom.2019.11.023

Datta, S.R., Anderson, D.J., Branson, K., Perona, P., Leifer, A., 2019. Computational Neuroethology: A Call to Action. Neuron 104, 11–24. https://doi.org/10.1016/j.neuron.2019.09.038

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L., 2009. ImageNet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern

Recognition. Presented at the 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

Doersch, C., Yang, Y., Gokay, D., Luc, P., Koppula, S., Gupta, A., Heyward, J., Goroshin, R., Carreira, J., Zisserman, A., 2024. BootsTAP: Bootstrapped Training for Tracking-Any-Point. https://doi.org/10.48550/arXiv.2402.00847

Doersch, C., Yang, Y., Vecerik, M., Gokay, D., Gupta, A., Aytar, Y., Carreira, J., Zisserman, A., 2023. TAPIR: Tracking Any Point with per-frame Initialization and temporal Refinement. https://doi.org/10.48550/arXiv.2306.08637

Dwivedi, V.P., Bresson, X., 2021. A Generalization of Transformer Networks to Graphs. https://doi.org/10.48550/arXiv.2012.09699

Ershov, D., Phan, M.-S., Pylvänäinen, J.W., Rigaud, S.U., Le Blanc, L., Charles-Orszag, A., Conway, J.R.W., Laine, R.F., Roy, N.H., Bonazzi, D., Duménil, G., Jacquemet, G., Tinevez, J.-Y., 2022. TrackMate 7: integrating state-of-the-art segmentation algorithms into tracking pipelines. Nat. Methods 19, 829–832. https://doi.org/10.1038/s41592-022-01507-1

Gallusser, B., Stieber, M., Weigert, M., 2023. Self-supervised Dense Representation Learning for Live-Cell Microscopy with Time Arrow Prediction, in: Greenspan, H., Madabhushi, A., Mousavi, P., Salcudean, S., Duncan, J., Syeda-Mahmood, T., Taylor, R. (Eds.), Medical Image Computing and Computer Assisted Intervention – MICCAI 2023, Lecture Notes in Computer Science. Springer Nature Switzerland, Cham, pp. 537–547. https://doi.org/10.1007/978-3-031-43993-3_52

Girshick, R., 2015. Fast R-CNN. https://doi.org/10.48550/arXiv.1504.08083

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. https://doi.org/10.48550/arXiv.1311.2524

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2018. Mask R-CNN. https://doi.org/10.48550/arXiv.1703.06870

He, K., Zhang, X., Ren, S., Sun, J., 2015. Deep Residual Learning for Image Recognition. https://doi.org/10.48550/arXiv.1512.03385

Huang, C.-Z.A., Vaswani, A., Uszkoreit, J., Shazeer, N., Simon, I., Hawthorne, C., Dai, A.M., Hoffman, M.D., Dinculescu, M., Eck, D., 2018. Music Transformer. https://doi.org/10.48550/arXiv.1809.04281

Karaev, N., Rocco, I., Graham, B., Neverova, N., Vedaldi, A., Rupprecht, C., 2023. CoTracker: It is Better to Track Together. https://doi.org/10.48550/arXiv.2307.07635

Kingma, D.P., Ba, J., 2017. Adam: A Method for Stochastic Optimization. https://doi.org/10.48550/arXiv.1412.6980

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. ImageNet Classification with Deep Convolutional Neural Networks, in: Advances in Neural Information Processing Systems. Curran Associates, Inc.

Kuhn, H.W., 1955. The Hungarian method for the assignment problem. Nav. Res. Logist. Q. 2, 83–97. https://doi.org/10.1002/nav.3800020109

Lecun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86, 2278–2324. https://doi.org/10.1109/5.726791

Li, Z., Liu, F., Yang, W., Peng, S., Zhou, J., 2022. A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects. IEEE Trans. Neural Netw. Learn. Syst. 33, 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827

Lin, T., Wang, Y., Liu, X., Qiu, X., 2022. A survey of transformers. AI Open 3, 111–132. https://doi.org/10.1016/j.aiopen.2022.10.001

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., Xie, S., 2022. A ConvNet for the 2020s. https://doi.org/10.48550/arXiv.2201.03545

Luo, W., Xing, J., Milan, A., Zhang, X., Liu, W., Kim, T.-K., 2021. Multiple Object Tracking: A Literature Review. Artif. Intell. 293, 103448. https://doi.org/10.1016/j.artint.2020.103448

Malin-Mayor, C., Hirsch, P., Guignard, L., McDole, K., Wan, Y., Lemon, W.C., Kainmueller, D., Keller, P.J., Preibisch, S., Funke, J., 2023. Automated reconstruction of whole-embryo cell lineages by learning from sparse annotations. Nat. Biotechnol. 41, 44–49. https://doi.org/10.1038/s41587-022-01427-7

Maška, M., Ulman, V., Delgado-Rodriguez, P., Gómez-de-Mariscal, E., Nečasová, T., Guerrero Peña, F.A., Ren, T.I., Meyerowitz, E.M., Scherr, T., Löffler, K., Mikut, R., Guo, T., Wang, Y., Allebach, J.P., Bao, R., Al-Shakarji, N.M., Rahmon, G., Toubal, I.E., Palaniappan, K., Lux, F., Matula, P., Sugawara, K., Magnusson, K.E.G., Aho, L., Cohen, A.R., Arbelle, A., Ben-Haim, T., Raviv, T.R., Isensee, F., Jäger, P.F., Maier-Hein, K.H., Zhu, Y., Ederra, C., Urbiola, A., Meijering, E., Cunha, A., Muñoz-Barrutia, A., Kozubek, M., Ortiz-de-Solórzano, C., 2023. The Cell Tracking Challenge: 10 years of objective benchmarking. Nat. Methods 20, 1010–1020. https://doi.org/10.1038/s41592-023-01879-y

Mathis, A., Mamidanna, P., Cury, K.M., Abe, T., Murthy, V.N., Mathis, M.W., Bethge, M., 2018. DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. Nat. Neurosci. 21, 1281–1289. https://doi.org/10.1038/s41593-018-0209-y

Norregaard, K., Metzler, R., Ritter, C.M., Berg-Sørensen, K., Oddershede, L.B., 2017. Manipulation and Motion of Organelles and Single Molecules in Living Cells. Chem. Rev. 117, 4342–4375. https://doi.org/10.1021/acs.chemrev.6b00638

Pereira, T.D., Shaevitz, J.W., Murthy, M., 2020. Quantifying behavior to understand the brain. Nat. Neurosci. 23, 1537–1549. https://doi.org/10.1038/s41593-020-00734-z

Pereira, T.D., Tabris, N., Matsliah, A., Turner, D.M., Li, J., Ravindranath, S., Papadoyannis, E.S., Normand, E., Deutsch, D.S., Wang, Z.Y., McKenzie-Smith, G.C., Mitelut, C.C., Castro, M.D., D'Uva, J., Kislin, M., Sanes, D.H., Kocher, S.D., Wang, S.S.-H., Falkner, A.L., Shaevitz, J.W., Murthy, M., 2022. SLEAP: A deep learning system for multi-animal pose tracking. Nat. Methods 19, 486–495. https://doi.org/10.1038/s41592-022-01426-1

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You Only Look Once: Unified, Real-Time Object Detection. https://doi.org/10.48550/arXiv.1506.02640

Ren, S., He, K., Girshick, R., Sun, J., 2016. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. https://doi.org/10.48550/arXiv.1506.01497

Romero-Ferrero, F., Bergomi, M.G., Hinz, R.C., Heras, F.J.H., de Polavieja, G.G., 2019. idtracker.ai: tracking all individuals in small or large collectives of unmarked animals. Nat. Methods 16, 179–182. https://doi.org/10.1038/s41592-018-0295-5

Ronneberger, O., Fischer, P., Brox, T., 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. https://doi.org/10.48550/arXiv.1505.04597

Schmidt, U., Weigert, M., Broaddus, C., Myers, G., 2018. Cell Detection with Star-convex Polygons. pp. 265–273.

Shaw, P., Uszkoreit, J., Vaswani, A., 2018. Self-Attention with Relative Position Representations. https://doi.org/10.48550/arXiv.1803.02155

Sheridan, A., Nguyen, T.M., Deb, D., Lee, W.-C.A., Saalfeld, S., Turaga, S.C., Manor, U., Funke, J., 2023. Local shape descriptors for neuron segmentation. Nat. Methods 20, 295–303. https://doi.org/10.1038/s41592-022-01711-z

Stringer, C., Wang, T., Michaelos, M., Pachitariu, M., 2021. Cellpose: a generalist algorithm for cellular segmentation. Nat. Methods 18, 100–106. https://doi.org/10.1038/s41592-020-01018-x

Su, J., Lu, Y., Pan, S., Murtadha, A., Wen, B., Liu, Y., 2023. RoFormer: Enhanced Transformer with Rotary Position Embedding. https://doi.org/10.48550/arXiv.2104.09864

Tan, M., Le, Q.V., 2020. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. https://doi.org/10.48550/arXiv.1905.11946

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2023. Attention Is All You Need. https://doi.org/10.48550/arXiv.1706.03762

Weigert, M., Schmidt, U., Haase, R., Sugawara, K., Myers, G., 2020. Star-convex Polyhedra for 3D Object Detection and Segmentation in Microscopy, in: 2020 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 3655–3662. https://doi.org/10.1109/WACV45572.2020.9093435

Zeng, F., Dong, B., Zhang, Y., Wang, T., Zhang, X., Wei, Y., 2022. MOTR: End-to-End Multiple-Object Tracking with Transformer. https://doi.org/10.48550/arXiv.2105.03247

Zhou, X., Yin, T., Koltun, V., Krähenbühl, P., 2022. Global Tracking Transformers. https://doi.org/10.48550/arXiv.2203.13250