Comparative study of stochastic filtering and attention-based approaches for intracellular dynamics estimation

Piyush Mishra | पीयूष मिश्र | ପୀୟୂଷ ମିଶ୍ର

Bioimage Mining Group,
Mathematics Institute of Marseille (I2M),
Turing Centre for Living Systems (CENTURI),
Aix-Marseille University

August 23, 2023

Supervisor: Dr. Philippe Roudot

Acknowledgements

I am immensely grateful to my supervisor, Dr. Philippe Roudot, for his guidance, expertise, and continuous encouragement. His insightful feedback and constructive suggestions have played a pivotal role in shaping this research.

I would like to express my deepest gratitude and appreciation to my family, Dr. Paritosh Mishra, Smruti Mishra and Ishani Mishra. Their unwavering support and encouragement have been the bedrock of my academic journey. They have always believed in me and provided me with the guidance and motivation to pursue my educational aspirations. Their sacrifices and endless encouragement have been instrumental in my achievements, and I am truly grateful for their presence in my life.

I would also like to extend my thanks to Jules Vanaret and Laura Neschen for helping me with the development of this project, and to Germán Dario Camargo Ortega for the essential resources to understand the biological context.

I acknowledge the financial support and the structure provided by the Computational and Mathematical Biology master programme via the Turing Centre for Living Systems and its directors, Dr. Laurence Röder and Dr. Florence Hubert.

Finally, I am deeply thankful to the support of my friends – Sandeep Patel, for helping me with Git, Steven Carmignani, Aurèlien Bouillet Sanchez, Thomas David, Nikita Jain, Amisha Jain, Debasish Sahoo and Nainika Gaur for listening to my rants.

Abstract

Fluorescence microscopy is a critical tool for investigating complex biological processes at the cellular level. However, accurately estimating fluorescence signals in noisy and dense environments poses a formidable challenge. In these experiments we focus on developing self-supervised strategies for stochastic filtering in fluorescence microscopy while addressing the inherent complexity and NP-hard nature of the problem.

Stochastic filtering in fluorescence microscopy is deemed NP-hard due to the computational complexity associated with finding the optimal solution. The problem involves estimating the underlying fluorescence signals based on noisy observations and incorporating the stochastic nature of the fluorescence process. This estimation process typically requires considering multiple interacting particles and their potential trajectories, making it computationally demanding. Existing strategies for stochastic filtering in fluorescence microscopy often suffer from issues such as high computational complexity, which restrict their applicability to real-time or large-scale microscopy data analysis. The NP-hardness of the problem further emphasises the need for innovative and efficient methods.

To tackle these challenges, we explore the potential of transformers, a type of neural network architecture, for stochastic filtering in fluorescence microscopy. Transformers have demonstrated exceptional capabilities in capturing complex patterns and dependencies, making them promising candidates for addressing the computational demands of stochastic filtering. By leveraging their ability to model the full context of the noisy and dense fluorescence microscopy environment, transformers offer the potential for more accurate and efficient signal estimation. This analysis reveals the distinct learning mechanisms of these methods and explores the regimes where they excel or face limitations. Understanding the computational characteristics and comparative performance of these approaches provides valuable insights for selecting appropriate methods based on specific requirements.

Contents

Chapter 1

Understanding Dense and Noisy Environments

Intracellular dynamics plays a fundamental role in cell biology and biomedical research, aimed at understanding various cellular processes and their underlying mechanisms. It involves the analysis and monitoring of particles, such as organelles, vesicles, and molecular complexes, within living cells. Accurate tracking of intracellular particles is crucial for elucidating cellular dynamics, protein trafficking [1], cell signalling pathways [2], and other essential cellular functions.

The ability to accurately track intracellular particles in dense and noisy environments has far-reaching implications for various biological applications. It is not only useful for the measurement of biophysical quantities of interest (process number, lifetime, frequency, etc) but also enables the data-driven modelling of complex systems at large scale [3]. For example, robust cell division requires the coordinated positioning and polymerisation of thousands of microtubules to capture chromosomes [4]. Cell migration requires actin assembly, adhesion dynamics formation and signal fluctuation [5] which necessitates tracking several particles. One of the principal reasons why none of these processes are understood fully comes from the complex nature of their dynamics combined with their very large number. Besides, there are many other cases where particle tracking can help us better understand their dynamics. Tracking mitochondria can provide insights into their fusion and fission processes [6], which are critical for maintaining cellular energy balance [7] and regulating apoptosis. Investigating the behaviour of other organelles, such as lysosomes [8] or peroxisomes [9], can also shed light on cellular processes associated with various diseases. Cell migration plays a crucial role in various physiological and pathological processes, including embryonic development [10], wound healing [11], and cancer metastasis [12]. Tracking intracellular particles can facilitate the study of cell migration by providing information on cell shape changes [13], lamellipodia formation [14], and the movement of signalling molecules involved in migration pathways [15]. Accurate tracking in dense and noisy environments is essential for uncovering the complex mechanisms governing cell movement. Development of computational methods to track particles in dense and noisy environments is, thus, essential to automate discovery.

1.1 Fluorescence Microscopy

Fluorescence microscopy [16] is an advanced optical imaging technique widely employed in the field of molecular biology and biomedical research. It utilises the phenomenon of fluorescence, wherein certain molecules, known as fluorophores, exhibit the ability to absorb photons of specific wavelengths and subsequently emit photons of a longer wavelength upon excitation.

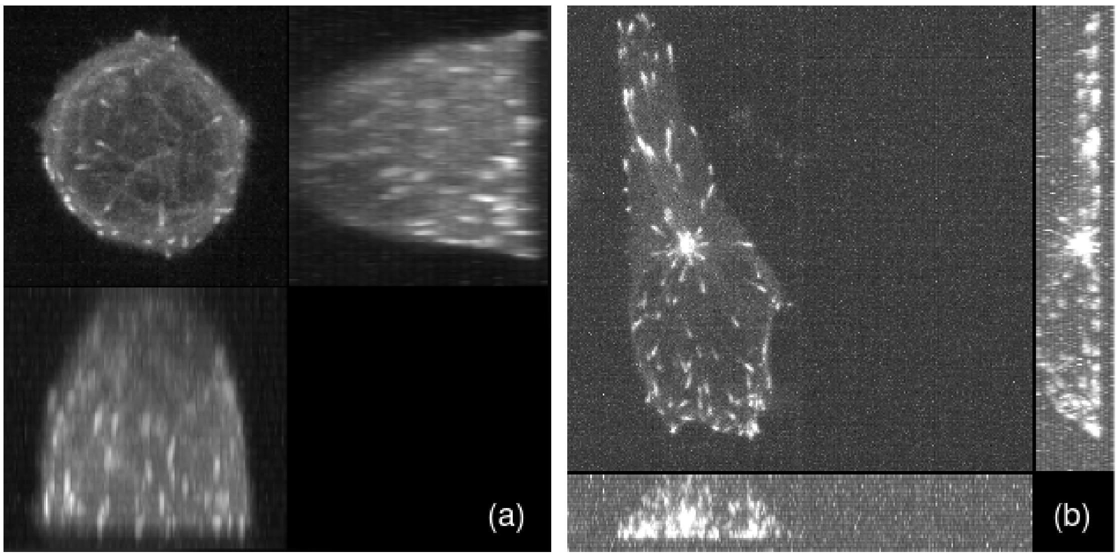

Figure 1.1: Challenges in measuring physiological processes in the context of microtubule dynamics in haematopoietic stem cells; (a) in early haematopoiesis, since the fluorophore expression is very low, it is hard to capture their dynamics, (b) in later stages of haematopoiesis, we are able to better capture the details but the biological research in this regime is limited.

The principle of molecular labelling lies at the core of fluorescence microscopy. This technique enables the visualisation and localisation of specific molecules or structures within biological samples with exceptional precision and sensitivity. To achieve this, target molecules of interest are selectively labelled or tagged with fluorophores that possess distinctive spectral characteristics.

The process of molecular labeling typically involves the conjugation of fluorophores to specific molecular targets, such as proteins, nucleic acids, or small molecules. These fluorophore-conjugated probes or antibodies, referred to as molecular probes, are designed to bind specifically to their intended target, thereby enabling the selective identification and visualisation of the labelled molecules within the sample. Upon illumination with an appropriate excitation light source, the fluorophores attached to the labelled molecules absorb photons of the excitation wavelength. This absorption results in the elevation of the fluorophores' electronic energy states from their ground state to higher energy levels. Following excitation, the fluorophores rapidly undergo a non-radiative relaxation process, returning to their ground state through vibrational and thermal energy dissipation. Alternatively, they may transition to higher energy states with subsequent non-radiative decay. However, the most significant pathway involves the radiative decay or fluorescence emission, wherein the fluorophores release photons with longer wavelengths than those absorbed during excitation. To capture these emitted photons, specialised optical filters are employed in fluorescence microscopy. These filters effectively discriminate against the excitation light and allow only the fluorescence emission to reach the detector, thereby enabling the acquisition of high-contrast images. The fluorescence signals captured by the detector are then translated into a visual representation, often in the form of coloured images, where each color corresponds to a specific fluorophore or molecular probe used for labelling. By employing multiple fluorophores with distinct emission spectra, it becomes possible to simultaneously visualise different molecular targets within the sample, facilitating the study of complex biological processes and spatial relationships.

A main challenge in fluorescence microscopy is the possibility of defective or incomplete expression of the fluorophores within the labelled molecules (see Fig. 1.1). This can occur due to various factors such as improper labelling techniques, suboptimal conjugation chemistry, or inefficient delivery of the labelled probes into the target cells or tissues. In some cases, even with proper labelling procedures, the natural expression levels of the target molecules may be inherently low. This limited abundance of the labelled molecules results in weak fluorescence signals that are challenging to detect above the background noise. Consequently, this can reduce the sensitivity and signal-to-noise ratio of the fluorescence microscopy images, making it difficult to distinguish the specific signal from the background fluorescence. Further, fluorophores used in fluorescence microscopy are prone to photobleaching, which refers to the irreversible loss of fluorescence signal over time due to the photochemical degradation of the fluorophores. Prolonged exposure to excitation light leads to cumulative photodamage and reduced fluorescence intensity, ultimately limiting the duration of imaging experiments.

1.2 Case: Molecular Dynamics in Early Haematopoiesis

1.2.1 Haematopoiesis

Haematopoiesis is the process of blood cell formation in the body. It involves the production, differentiation, and maturation of blood cells from haematopoietic stem cells (HSCs) located primarily in the bone marrow. Haematopoiesis ensures the continuous production of various types of blood cells to maintain normal physiological function. During haematopoiesis, HSCs undergo a series of steps to differentiate into different types of blood cells, including red blood cells (erythrocytes), white blood cells (leukocytes), and platelets. This process is tightly regulated by a complex interplay of intrinsic cellular factors and signals from the microenvironment or bone marrow niche.

The differentiation process begins with the commitment of HSCs to specific lineage pathways, such as myeloid or lymphoid lineages. Myeloid lineage gives rise to red blood cells, granulocytes, monocytes, and platelets, while lymphoid lineage produces B and T lymphocytes, which are involved in immune responses. The differentiation and maturation of blood cells involve a series of proliferative, differentiative, and maturation stages, during which specific transcription factors and signaling molecules orchestrate the development of different blood cell lineages. The process also involves complex interactions between haematopoietic cells, the bone marrow microenvironment, and various growth factors and cytokines.

1.2.2 Cytoskeleton dynamics affect nuclear shape in early haematopoiesis

Biedzinski et. al [17] carried out experiments to understand the correlation between the behaviour of microtubules and how it affected the nuclei of the haematopoietic progenitor cells (HPCs). It was found that disrupting microtubule organisation using the drugs nocodazole and taxol led to changes in the shape of the nucleus during haematopoietic differentiation. We schematise these experimentations in Fig. 1.2. Treatment with nocodazole, which depolymerises microtubules, caused the nuclei of haematopoietic progenitor cells (HPCs) to become more elongated and less circular in shape. In contrast, treatment with taxol, which stabilises microtubules, caused the nuclei of HPCs to become more circular and less elongated in shape.

These changes in nuclear shape were attributed to the role of microtubules in regulating the distribution of nuclear pore complexes (NPCs), which are crucial for the transport of molecules in and out of the nucleus[a]. Disrupting microtubule organisation, hence, altered the distribution of NPCs, which in turn affected the shape of the nucleus.

In the aforementioned experiments, the only molecular dynamics that were quantified were those of the NPCs. This is primarily because the speed in which HSCs differentiate is very high, and it is very difficult (as we will develop later in this chapter) to infer the microtubule dynamics from those

[a]It is also established in their study that the organisation of microtubules in HPCs affects the expression patterns of myeloid differentiation. Thus, one can argue that in many cases, geometry affects the genome, which makes it even more important to be able to track particles, from a genomic standpoint.

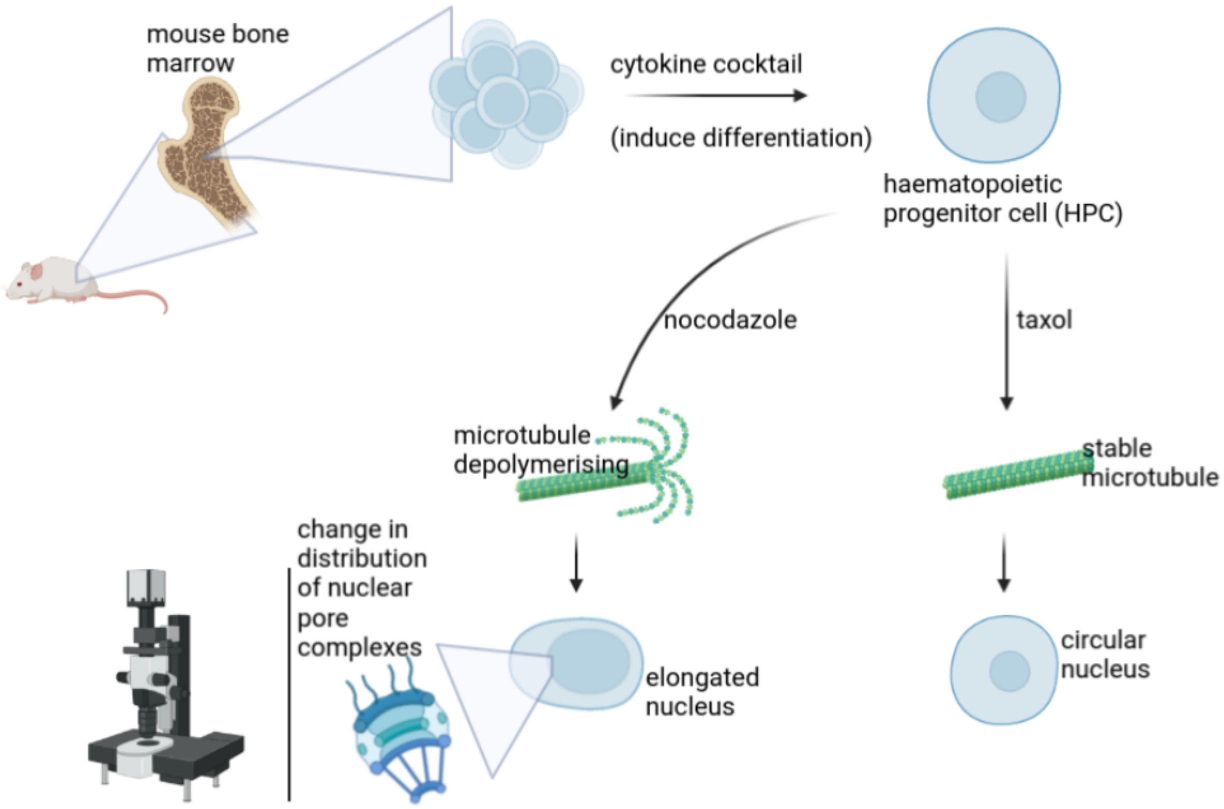

Figure 1.2: Microtubule stability affects the distribution of nuclear pore complexes in mice haematopoietic progenitor cells which further affects the shape of the nucleus.

observations. So, instead, indirect inferences are made to understand the effect of microtubules in differentiation.

1.3 Case: Molecular Dynamics and Neural Stem Cells

1.3.1 Neurogenesis and neural stem cells

Neurogenesis is the process by which new neurons (brain cells) are generated in the brain. It is a fundamental process in brain development, learning, and memory formation. Neural stem cells are specialised cells that give rise to new neurons and other cell types in the brain. Neural stem cells are unique because they have the ability to self-renew, meaning they can divide and produce more stem cells, as well as differentiate into different types of brain cells, including neurons and glial cells (support cells in the brain).

During neurogenesis, neural stem cells undergo a series of complex processes. They first divide, generating either more neural stem cells or progenitor cells. These progenitor cells then undergo further division and differentiation, ultimately developing into fully mature and functional neurons. Neurogenesis is tightly regulated by various factors, including genetic programs, molecular signals, and environmental cues. Disruptions in neurogenesis can have significant implications for brain function and have been associated with neurological disorders, cognitive decline, and mood disorders.

1.3.2 AKNA and its role in neurogenesis

AKNA (At-hook containing transcription factor) is a protein that has been the subject of scientific investigation due to its role in various cellular processes. It is found in the nucleus of cells, where it

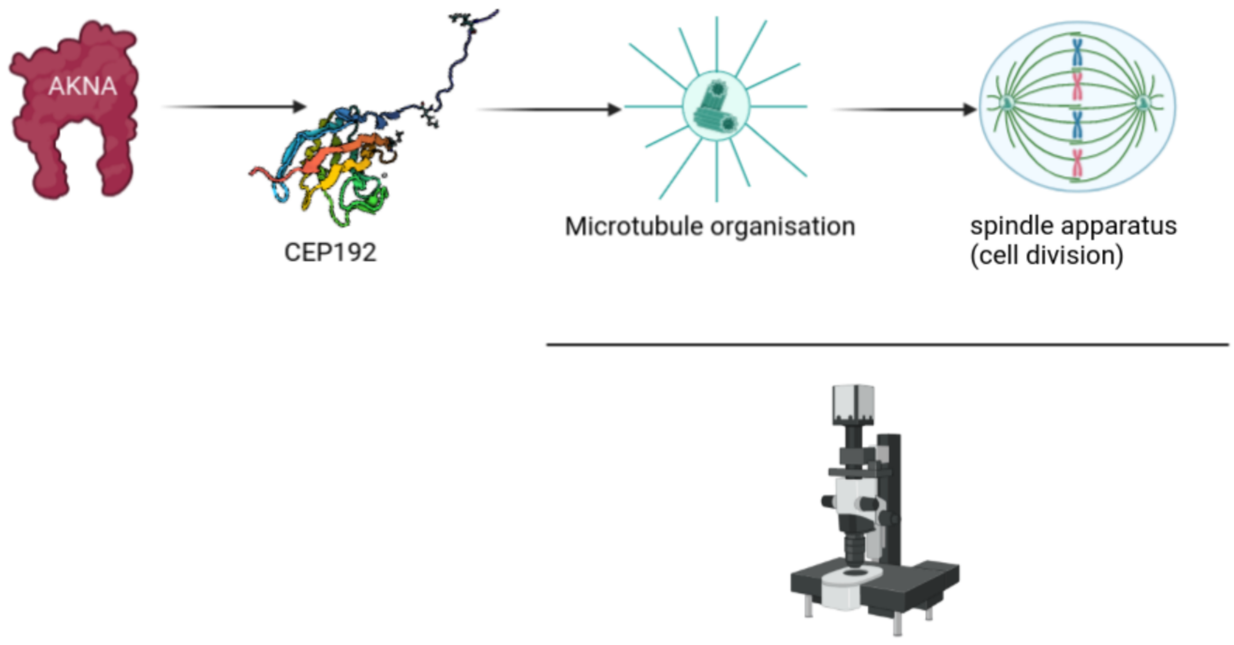

Figure 1.3: AKNA regulates the activity of CEP192, which is involved in microtubule organisation in neural stem cells, thereby facilitating stable cell division.

interacts with DNA and helps regulate gene expression.

One notable aspect of AKNA is its involvement in neurogenesis [18]. Research has shown that AKNA is highly expressed in neural stem cells, which are responsible for generating new neurons. AKNA's presence in these cells suggests that it plays a crucial role in brain development. Specifically, AKNA has been found to be important for organising microtubules within neural stem cells. AKNA helps ensure the proper organisation and functioning of microtubules during neurogenesis (see Fig. 1.3).

Experiments studying AKNA have demonstrated that its deficiency can lead to disorganised microtubules in neural stem cells. This disruption in microtubule organisation, in turn, impairs neurogenesis and can lead to defects in brain development and function. Conversely, overexpression of AKNA enhances microtubule organisation and promotes neurogenesis. This suggests that AKNA plays a crucial role in orchestrating the dynamic processes of neural stem cell division and migration, which are essential for proper brain development. These conclusions could only be reached by tracking microtubules.

1.4 Tracking Particles is Hard

The concern in contexts like the ones discussed above is that the observations of fluorescence microscopy are very noisy, and these environments are biologically very dense, i.e., there are many particles in a confined space whose trajectories most certainly cross or coincide with each other. This is computationally challenging because it involves associating the observations (i.e., the particle positions) over time with individual particles in the scene. In particular, the challenge arises when particles are closely spaced and/or have highly variable or complex motion patterns, which can lead to ambiguity in associating observations with individual particles. Formally, this is a data association problem – given a set of observations over time and a set of possible particle trajectories, the task is to determine the association between the observations and the true particle trajectories. This is typically done using probabilistic models, such as Bayesian filtering or maximum likelihood estimation. We will discuss theories of Bayesian filtering techniques in Ch. 2.

The complexity of the problem depends on the number of particles, the dimensionality of the problem, the density of particles, the variability of particle motion, and the amount of noise in the observations. In general, the problem becomes computationally intractable as the number of particles and the dimensionality of the problem increase. In particular, the problem of particle tracking in dense and noisy environments is known to be NP-hard[b] because it can be reduced to the problem of finding the maximum clique[c] in a graph [19] [20]. Specifically, each particle trajectory can be represented as a node in a graph, and edges are added between nodes if the corresponding trajectories overlap in time and space.

The problem at hand involves solving an inverse problem to reconstruct trajectories based on measurements that contain noise. In this scenario, we have a collection of particles whose motion parameters we need to estimate. These motion parameters describe the behaviour of each particle, such as its position, velocity, or acceleration. Additionally, we need to determine the association between the measurements and the trajectories [21], i.e., which measurements correspond to which particles at each time step. However, due to the presence of noise in the measurements and the lack of direct knowledge about the motion parameters and associations, the problem becomes challenging. To complicate matters further, the number of possible associations between measurements and trajectories grows exponentially as the number of particles and time steps increase. This exponential growth leads to an extremely large number of hypotheses to consider, making it computationally infeasible to exhaustively evaluate all possibilities. Consequently, finding the optimal solution becomes a computationally complex and NP-hard problem.

[b]In computer science, an NP-hard problem is a problem that cannot be solved in polynomial time for all cases.

[c]In graph theory, a clique is a group of vertices (or nodes) in a graph such that every vertex in the group is connected to every other vertex in the group. Finding a maximum clique in a graph means finding the largest possible group of vertices that are all connected to each other.

Chapter 2

The Bayes-ics of Particle Tracking

We established in the previous chapter that measuring intracellular dynamics is an important tool to understanding physiological processes. Yet, we also showed that an exhaustive analysis of all of the trajectory set was not computationally feasible. In this chapter we discuss conventional approaches used to provide a sub-optimal solution to our problem in the stochastic filtering framework. We know that this is particularly challenging when there are uncertainties, such as sensor noise or external disturbances, or in most cases, both. We will explore, on a theoretical and superficial level, the problems that these kinds of experimental errors pose and the mathematical basis of the ways in which we can tackle them.

2.1 Dynamic Estimation of a Single Molecular Process with Stochastic Filtering

In order to introduce notations and the stochastic filtering concepts, we present in this section the trivial case were a single particle is present in the field of view. Consider this particle to be at state $\mathbf{x}_t$ at a time instant t.

$$\mathbf{x}_t = \begin{pmatrix} x_t \\ y_t \\ dx_t \\ dy_t \end{pmatrix}$$

where x_t and y_t are components of positions in the x- and y- axes respectively, and dx_t and dy_t are the components of velocity in these axes respectively. It might be tempting to believe that if the current state, $\mathbf{x}_t$, and the dynamic model are known to us, it is fairly easy to "calculate" the next state. This is not true because measurements largely come with a factor of error. This error depends on many parameters – calibration, signal to noise ratio, etc (measurement noise). Further, a target motion might not be aligned with what is expected of the motion (process noise). Further, to estimate the current state, one would need to remember all historical measurements thereby leading to a requirement of large memory. One would, moreover have to recalculate repeatedly to update the estimated value after every new measurement. A more practical approach is to keep only the previous estimate and update that. Kalman filtering can thus be summarised into two steps:

1. Estimate the current state based on two things – measurement and the previous prediction.
2. Using the model of prediction, predict the next state based on the current state estimate.

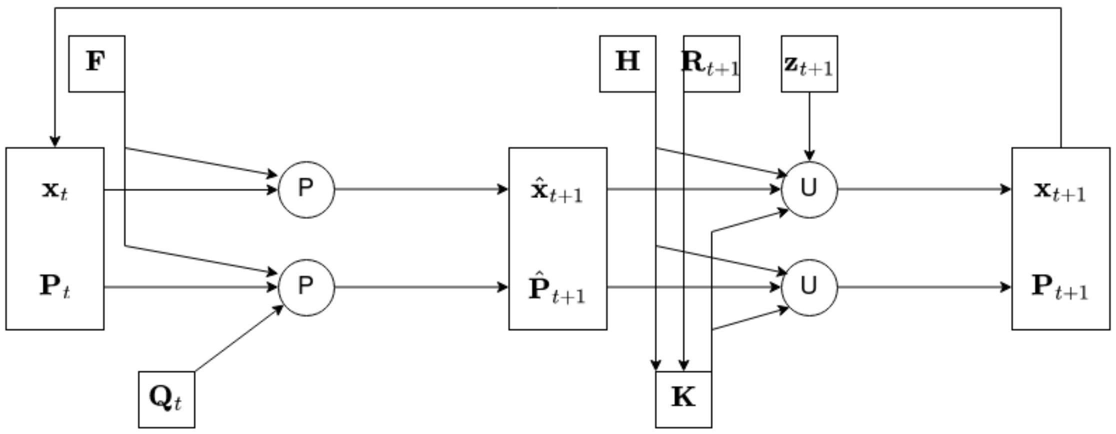

Figure 2.1: A schematic representation of the Kalman filtering technique (Note: `P` refers to the prediction mechanism and `U` refers to the updation mechanism)

Mathematically, if we have a state transition model $\mathbf{F} \in \mathbb{R}^{4\times4}$ (which would govern the motion of the particle, say, Brownian or directed), an observation model $\mathbf{H} \in \mathbb{R}^{4\times4}$ which would map the true state space (the ground truth) to the observed space (the measurements from fluorescence microscopy), then we can write the filtering equations as below:

$$\mathbf{x}_{t+1} = \mathbf{F}\mathbf{x}_t + \boldsymbol{\mu}_{t+1} \tag{2.1}$$

$$\mathbf{z}_{t+1} = \mathbf{H}\mathbf{x}_{t+1} + \boldsymbol{\omega}_{t+1} \tag{2.2}$$

$\mathbf{z}_t$ represents the time t measurement of the true state $\mathbf{x}_t$.

Here, $\boldsymbol{\mu}_t \in \mathbf{M}$ represents the process noise as discussed above. It is drawn from a multi-variate normal distribution with zero mean, $\mathcal{N}$, and its covariance $\mathbf{Q}_t : \boldsymbol{\mu}_t \sim \mathcal{N}(0, \mathbf{Q}_t)$. Similarly $\boldsymbol{\omega}_t \in \boldsymbol{\Omega}$ is the observed noise, which is assumed to be a zero mean normal white noise with covariance $\mathbf{R}_t : \boldsymbol{\omega}_t \sim \mathcal{N}(0, \mathbf{Q}_t)$.

Note that besides the state of the filter being represented by the *a posteriori* state estimate $\mathbf{x}_t$, another component of its representation is given by its *a posteriori* covariance matrix estimate $\mathbf{P}_t$.

So, when the state filter is predicted, both of these representations are predicted.

Predicted *a priori* state estimate:

$$\hat{\mathbf{x}}_{t+1} = \mathbf{F}\mathbf{x}_t \tag{2.3}$$

Predicted *a priori* covariance estimate[a]:

$$\hat{\mathbf{P}}_{t+1} = \mathbf{F}\mathbf{P}_t\mathbf{F}^T + \mathbf{Q}_t \tag{2.4}$$

Now that the predictions are out of the way, one ought to update the filters. In order to update, a term called innovation[b] is introduced.

Measured innovation:

$$\tilde{\mathbf{y}}_{t+1} = \mathbf{z}_{t+1} - \mathbf{H}\hat{\mathbf{x}}_{t+1} \tag{2.5}$$

[a]If we multiply the distribution $\mathbf{x}_t$ with the matrix $\mathbf{F}$, then its covariance gives the identity $cov(\mathbf{F}\mathbf{x}_t) = \mathbf{F}.cov(\mathbf{x}_t).\mathbf{F}^T$. With only this transformation, everything should work fine if the state evolves based on its own properties. Everything should still be fine if the state evolves based on certain external forces as long as those forces are known. To account for the uncertainty of those external forces, the additive term $\mathbf{Q}_t$ is incorporated.

[b]Innovation can be defined as the amount of $\mathbf{z}_{t+1}$ not explained by the current prediction

Innovation covariance:

$$\mathbf{S}_{t+1} = \mathbf{H}\hat{\mathbf{P}}_{t+1}\mathbf{H}^T + \mathbf{R}_{t+1} \tag{2.6}$$

Using the innovation covariance $\mathbf{S}_{t+1}$, an optimal Kalman gain $\mathbf{K}_{t+1}$ is calculated:

$$\mathbf{K}_{t+1} = \hat{\mathbf{P}}_{t+1}\mathbf{H}^T\mathbf{S}_{t+1}^{-1} \tag{2.7}$$

Now, we have everything to update the filter:

$$\mathbf{x}_{t+1} = \hat{\mathbf{x}}_{t+1} + \mathbf{K}_{t+1}\tilde{\mathbf{y}}_{t+1} \tag{2.8}$$

$$\mathbf{P}_{t+1} = (\mathbf{I} - \mathbf{K}_{t+1}\mathbf{H})\hat{\mathbf{x}}_{t+1} \tag{2.9}$$

where $\mathbf{I}$ is the identity matrix.

2.2 Dynamic Estimation of Multiple Molecular Processes in Clutter

Increasing the frame rate in fluorescence microscopy improves the temporal resolution of the imaging process – we can obtain a more detailed representation of the dynamic events occurring within the sample. With a higher frame rate, there are more data points available for tracking, allowing for more precise determination of particle trajectories and velocities. This is particularly beneficial when studying fast-moving particles or particles undergoing rapid changes in their movements. Biological systems often exhibit stochastic behaviour or intermittent events that occur sporadically. By increasing the frame rate, the likelihood of capturing these rare events within the imaging time window can be increased. This would provide valuable insights into the underlying system dynamics and can lead to a deeper understanding of biological processes.

While increasing the frame rate offers these advantages, it is important to understand the challenges that arise with higher frame rates, specifically in terms of clutter. As the frame rate increases, so does the frequency of image acquisition. This can result in an increased amount of noise and unwanted signals, leading to higher clutter levels in the images. Understanding and managing this clutter is crucial for accurate particle tracking and reliable analysis of fluorescence microscopy data.

2.2.1 Problem Formulation

Consider a set of N fluorescence-labelled particles $\mathbf{X} = \{\mathbf{X}^i\}_{i=0:N}$ where $\mathbf{X}^i = \{\mathbf{x}^i_{t_0}, ..., \mathbf{x}^i_{t_0+l}\} \in \mathbb{R}^{2\times l}$ is a sequence of positions describing an unknown dynamic process, $t_0 \in \mathbb{R}$ denoting its time of birth and $l \in \mathbb{R}$ denoting its lifetime. Further consider that the measurements of the set $\mathbf{X}$ acquired from the microscopy in a duration of $t = T$ time steps is denoted as $\mathbf{Z} = \{\mathbf{Z}_t\}_{t=0:T}$.

$\mathbf{Z}$ is the union of measurements that come from the particles of interest in $\mathbf{X}$, that we denote by $\mathbf{Z}_p$ and the measurements that come from clutter (false positives), that we denote by $\mathbf{Z}_c$[c]. Therefore, $\mathbf{Z} = \{\mathbf{Z}_p \cup \mathbf{Z}_c\}$. As discussed in Ch. 1, the optimal reconstruction from $\mathbf{X}$ to $\mathbf{Z}$ is NP-hard – i.e., it requires the evaluation of each possible combination in $\mathbf{Z}$ to optimise $p(\mathbf{X}|\mathbf{Z})$. The following Bayesian filtering equation provides an iterative approach to approximate this problem:

[c]This problem of clutter arises because in order to understand particle behaviour, a naive solution would be to increase the frequency of acquisition, but this increase in acquisition leads to a high rate of false positives that can be modelled from a Poisson point process.

$$p(\mathbf{X}_t|\mathbf{Z}_{1:t}) = p(\mathbf{Z}_t|\mathbf{X}_t)\int p(\mathbf{X}_t|\mathbf{X}_{t-1})p(\mathbf{X}_{t-1}|\mathbf{Z}_{1:t-1})d\mathbf{X} \tag{2.10}$$

In order for this equation to hold true, we need to account for certain assumptions. This approach considers that $\mathbf{X}_t$ follows a Markovian assumption with known transition probability $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ and an *a priori* $p(\mathbf{X}_0)$. We also have to assume that $\mathbf{Z}_t$ only depends on $\mathbf{X}_t$. These hypotheses are summarised in the state-space model in eq. 2.1 and eq. 2.2.

Note that the Bayesian filtering view is a generalised approach to the Kalman filtering view. While this iterative formulation provides a framework to assess the probability $p(\mathbf{X}_t|\mathbf{Z}_{1:t})$, finding its optimum still requires the filtering of all possible combinations in $\mathbf{Z}$. This inevitably calls for ad-hoc heuristics to prune the tree of possible associations and reduce the number of sequences to be evaluated concurrently. Generally, this is carried out through repeated cycles of state prediction $p(\mathbf{X}_t|\mathbf{X}_{t-1})$, finding a reduction of the possible number of associations between states and measurements through the computation of $p(\mathbf{Z}_t|\mathbf{X}_t)$, and state update $p(\mathbf{X}_t|\mathbf{Z}_t)$. We will develop more on this later in this chapter, when we discuss multiple hypothesis tracking.

2.2.2 Hypothesis Pruning

By now, we have understood that increasing the frame rate leads to a higher rate of image acquisition, providing more temporal information and improving the ability to capture fast dynamic processes. However, a higher frame rate also introduces challenges, such as increased clutter in the images. Clutter refers to the presence of unwanted or irrelevant signals, noise, or particles that can interfere with accurate particle tracking. This increased clutter in high frame rate microscopy generates a larger number of hypotheses for particle tracking algorithms. Hypotheses represent potential associations between observed measurements and true target tracks. As the clutter level rises, the number of false alarms and spurious measurements also increases. These false alarms can lead to the generation of numerous incorrect hypotheses, making the particle tracking task computationally complex and challenging. This changes the probability to associate the measurements to the predictions to now account for all the hypotheses available at a specific time step.

$$p(\mathbf{Z}_t|\mathbf{X}_t) = \sum_{\boldsymbol{\eta}_i^t \in \mathbf{H}'_t} p(\mathbf{Z}_t|\mathbf{X}_t, \boldsymbol{\eta}_i^t)p(\boldsymbol{\eta}_i^t|\mathbf{X}_t) \tag{2.11}$$

where $\boldsymbol{\eta}_i^t$ is the best association between the hypotheses and the tracks, and $\mathbf{H}'_t = \{\boldsymbol{\eta}_i^t\} \subset \mathbf{H}_t$ is the set of all of these best associations. Note that this is also true for when there is no clutter, but when there are multiple particles.

This, then makes the iterative Bayesian formulation described in eq. 2.10 as follows:

$$p(\mathbf{X}_t|\mathbf{Z}_{1:t}) = \sum_{\boldsymbol{\eta}_i^t \in \mathbf{H}'_t} p(\mathbf{Z}_t|\mathbf{X}_t, \boldsymbol{\eta}_i^t)p(\boldsymbol{\eta}_i^t|\mathbf{X}_t)\int p(\mathbf{X}_t|\mathbf{X}_{t-1})p(\mathbf{X}_{t-1}|\mathbf{Z}_{1:t-1})d\mathbf{X} \tag{2.12}$$

To address this problem, an effective strategy for hypothesis pruning is required. Hypothesis pruning involves selecting the most likely and reliable hypotheses while discarding or disregarding the incorrect or unlikely ones. The goal is to reduce the computational burden associated with a large number of hypotheses and to improve the accuracy and efficiency of particle tracking algorithms.

In the context of this study, at least for the traditional Bayesian approaches, we define this best association in terms of the least Mahalanobis distance[d]. But the question remains, "is there a better strategy for selecting the best hypotheses out of all hypotheses?" This is a question that we will come back to in Ch. 4 when we test the limits of computations of multiple particle tracking algorithms.

2.3 A Simple Pruning Strategy for Multiple Particle Tracking

The theories described in the previous section only start to become a little useful when there is a huge number of particles in a dense context and when there is asynchronous appearance and disappearance of particles. The resulting likelihood helps in deciding when the object starts and ends while remaining scalable. Because of the heterogeneity of motion that can be seen in intracellular transport, a single Kalman filter cannot account for it. Unpredictable changes in a particle's observable velocity cannot be handled by a single constant speed model.

Importantly, this approach guarantees scalalability but does not solve the problem of the exponential number of hypotheses to be filtered. The most elementary approach to solve this challenge relies on the selection of the most likely hypothesis at each time step. For example, considering the likelihood of associating the i^{th} trajectory to the j^{th} measurement at time t:

$$c_{i,j} = p(\mathbf{z}_t^j|\mathbf{x}_t^i) \tag{2.13}$$

We can get the most likely set of assignment $\{a_{i,j}\}$ by solving the linear assignment problem:

$$argmax_{a_{i,j}} \sum_i \sum_j c_{i,j} a_{i,j} \tag{2.14}$$

under the constraint that each trajectory can only be associated to a single measurement:

$$\sum_j a_{i,j} = \sum_i a_{i,j} = 1 \tag{2.15}$$

As such, finding the optimal set of trajectories relies on the level of ambiguity in the measurement that maximises the likelihood $p(\mathbf{z}_t^j|\mathbf{x}_t^i)$ where $\mathbf{x}_t^i$ is the predicted state with probability $p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)$. If a model is not able to predict the motion from one frame to the next, then the method cannot "connect the dots".

2.3.1 Intuitive understanding of multiple hypothesis tracking

At the heart of MHT is the maintenance and update of a set of hypotheses. Each hypothesis represents a possible assignment of measurements to tracks, aiming to accurately associate measurements with the corresponding tracks. First, we start with a set of existing tracks that represent the estimated state of the objects being tracked. Measurements from the current time step are received and processed to extract relevant information about object positions, velocities, or other descriptors. Next, for each track, a set of potential new hypotheses is generated by associating the measurements with the track. Multiple

[d]Mahalanobis distance takes into account the covariance structure of the data, which makes it suitable for datasets with correlations and unequal variances across dimensions. It provides a more accurate measure of similarity or dissimilarity between points in a multidimensional space by considering the relative importance of each dimension. By incorporating the covariance matrix, Mahalanobis distance provides a way to normalise distances and give more weight to dimensions with lower variability.

hypotheses are created to account for uncertainties and potential false measurements. These hypotheses are then evaluated and pruned based on their plausibility and compatibility with the measurements and existing tracks. The remaining hypotheses are updated using Bayesian filters such as the Kalman filter. These filters refine the state estimates by predicting the future object states and correcting them based on the new measurements.

Track management is an essential aspect of MHT. It involves updating track states, associating new measurements, and handling track initiation, termination, or merging. The surviving hypotheses from multiple tracks are combined to create a global hypothesis that provides the most likely assignment of measurements to tracks. This step involves evaluating the consistency and compatibility of the hypotheses across different tracks and resolving potential conflicts or ambiguities.

Finally, a decision is made based on the global hypothesis to determine the most likely object tracks and their states. This decision-making process can involve selecting the best hypothesis, associating tracks with unique object identities, or handling situations where objects appear, disappear, or interact. The MHT algorithm iterates through these steps over time, allowing it to track multiple objects robustly in a dynamic environment. It effectively handles uncertainties, occlusions, and clutter by maintaining and updating a set of hypotheses, enabling it to recover from tracking failures or challenging scenarios.

2.4 Discussion

We described the theory and a simple implementation of a stochastic filtering approach-tailored multiple particle tracking. Stochastic filtering enables the modelling of the randomness of biodynamics and captors, combined with a temporally greedy approach to hypothesis pruning, to limit the computational cost While several recent studies have focused on "connecting the dots" by improving motion prediction through deep learning approaches [22]-[23], they all rely on simulations that must readily reproduce the biophysical process of interest. An obvious alternative to improving the predictive power of the dynamic model without adding *a priori* information is to increase the frame rate and reduce the apparent motion. However, the limited number of photons emitted per fluorophores imposes a strong reduction in signal-to-noise ratio (we talk about this phenomenon on a superficial level in the next chapter). The detection of all targets thus comes with many false positives, i.e. clutter. This creates many ambiguities in the association between state and measurements.

However, recent developments in the field of stochastic filtering suggest that measurement coming from clutter can be precisely modelled and identified through two types of random finite sets (RFS), specifically a Poisson point process (PPP) for the clutter and multiple Bernoulli mixture model (MBM). This joint model, called PMBM for short, has been chosen because both the distributions are conjugate priors[e].

[e]In Bayesian probability theory, if the posterior distribution is in the same probability distribution family as the prior probability distribution, the prior and posterior are then called conjugate distributions, and the prior is called a conjugate prior for the likelihood function. This means that a PMBM distribution remains a PMBM distribution after prediction and update through the stochastic filtering process as seen in eq. 2.10.

Chapter 3

The Transformer: Theory and Methods

Deep learning has revolutionised numerous fields by enabling machines to learn and understand complex patterns from vast amounts of data. Among the various deep learning architectures, transformers have emerged as a powerful paradigm for processing sequential data. In this chapter, we begin with an introduction to the fundamental concepts of deep learning. We explore the key ideas behind neural networks, their training process, and the impressive capabilities they offer. Understanding these concepts is crucial for grasping the subsequent discussion on transformers. We uncover the underlying mechanisms that make transformers unique and allow them to capture global dependencies within sequences as we unveil the inner workings of transformers.

We have now well established that tracking particles accurately in cluttered environments poses significant challenges due to occlusions, overlapping objects, and noise. Fortunately, transformers offer a promising solution for tackling this problem. By leveraging their bidirectional processing capabilities and attention mechanisms, transformers can effectively extract relevant features from sequential data, leading to more robust and accurate particle tracking, even in cluttered scenarios. We explore the latest research and developments in this area, showcasing how transformers have pushed the boundaries of particle tracking performance.

3.1 Main Terminologies – Deep Learning

- **Feed-forward neural network**: A feed-forward neural network is a type of computer model inspired by the human brain. It is designed to process information in a specific way, moving in one direction from the input to the output. Imagine you have a bunch of interconnected dots, called neurons. Each neuron takes information from the previous neurons, does some calculations (see activation function), and passes the result to the next neurons.

- **Activation function**: Activation functions introduce non-linearity into neural networks, allowing them to model complex relationships. They determine the output of a neuron based on its input.

- **Weights, biases, training and learning**: Weights and biases are parameters that determine how information flows between neurons. Weights control the strength of connections, while biases introduce flexibility. During training, these parameters are adjusted to improve the network's ability to make accurate predictions. Deep learning models are trained using a process called backpropagation. During training, the model is presented with labelled examples, also known as a training dataset. The model learns to make accurate predictions by adjusting the weights and biases associated with its connections based on the errors (see loss function) it makes.

- **Backpropagation**: Backpropagation is an algorithm used to train neural networks by computing the gradient of the loss function with respect to each weight and bias in the network. The gradient provides information about the direction and magnitude of change needed to minimise the loss function. By using the gradient to update the weights and biases of the network, the network can learn to make better predictions on the training data. The backpropagation algorithm works by propagating the loss from the output layer of the network back to the input layer. The gradient of the loss function with respect to each weight and bias is computed using the chain rule of calculus. Once the gradient of the loss function with respect to each weight and bias has been computed, the weights and biases can be updated using an optimisation algorithm.
- **Loss function**: Loss functions measure the difference between the predicted output and the true output. They guide the learning process by providing a quantitative measure of how well the model is performing.
- **Optimisation**: Optimisation algorithms are used to update the model's weights and biases iteratively. These algorithms aim to find the optimal set of parameters that minimise the loss function and improve the model's predictive performance.
- **Learning rate**: The learning rate is a hyperparameter used in training neural networks that determines the step size taken by the optimisation algorithm during backpropagation. The learning rate controls the speed at which the weights and biases of the network are updated, and it is a critical parameter that can have a significant impact on the performance of the network: if the learning rate is too small, the network may take a long time to converge to a good solution, while if the learning rate is too large, the network may overshoot the optimal solution and diverge.
- **Epoch**: An epoch refers to a complete pass through the entire training dataset during the training process of a neural network. During an epoch, the training algorithm updates the model's weights based on the gradient of the loss function computed for each example in the dataset. In practice, deep learning models are typically trained over multiple epochs to improve their accuracy and convergence. The number of epochs required to train a model depends on various factors such as the complexity of the model, the size of the dataset, and the learning rate of the algorithm.
- **Overfitting**: Overfitting occurs when a model performs well on the training data but fails to generalise to new, unseen data.

3.2 Superficial Understanding of the Transformer

The transformer is a type of neural network architecture that was introduced in 2017 by Vaswani et al [24]. It was designed specifically for natural language processing tasks, such as language translation and language modelling, and it has since become one of the most popular and effective neural network architectures in this field.

The transformer architecture (Fig. 3.1) is composed of an encoder and a decoder, which are connected by a self-attention mechanism. The self-attention mechanism allows the network to attend to different parts of the input sequence when making predictions, which enables it to model long-range dependencies. The transformer encoder processes input sequences, capturing contextual information, while the transformer decoder generates output sequences, utilising the encoded information for generating accurate predictions or translations.

The encoder takes an input sequence of tokens, such as words or characters, or in the case of these experiments, measurements at each time frame (or time step), and produces a sequence of hidden representations that capture the semantic meaning of the input. Each layer in the encoder contains two sublayers: a multi-head self-attention layer and a feed-forward neural network layer. The multi-head

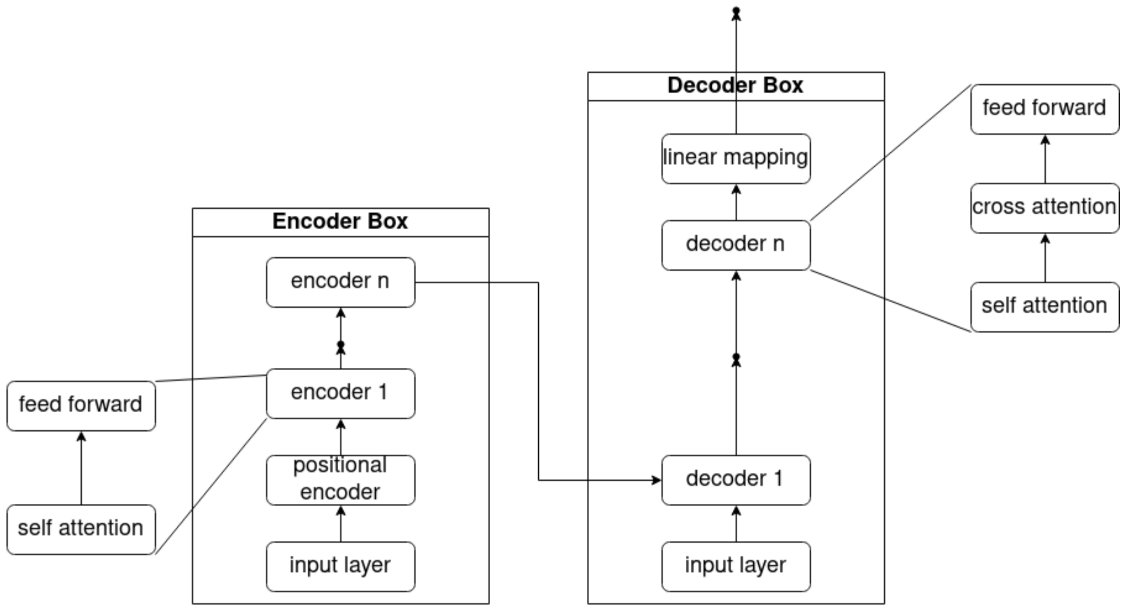

Figure 3.1: A vanilla transformer mechanism

self-attention layer allows the network to attend to different parts of the input sequence simultaneously, while the feed-forward neural network layer applies a non-linear transformation to the hidden representations. We will discuss how attention works in detail in the next section.

The decoder takes the sequence of hidden representations produced by the encoder and generates an output sequence of tokens. Like the encoder, each layer in the decoder contains two sublayers: a multi-head self-attention layer and a feedforward neural network layer. In addition, the decoder also includes a third sublayer, called cross-attention, that performs attention over the encoder's output sequence, allowing it to selectively attend to parts of the input sequence that are relevant for generating the output.

One of the key advantages of the transformer architecture is its ability to process input sequences in parallel, which makes it highly scalable and efficient. This has enabled researchers to train very large models on massive amounts of data, resulting in state-of-the-art performance on a wide range of natural language processing tasks. Henceforth, this architecture will be referred to as the vanilla transformer. Any further modifications during the course of development of this architecture will be reasoned as appropriately named.

3.3 Zooming In: Attention is all you need!

3.3.1 Self-Attention

Attention ([24], Fig. 3.2) is a mechanism used in neural networks to selectively focus on different parts of the input when making predictions. In the context of the transformer architecture, attention is used to weigh the importance of different input elements when computing the hidden representations for each layer. In the context of the problem at hand, these input elements can be thought of as the states of the particle (its x- and y- coordinates).

The self-attention mechanism in the transformer architecture is called scaled dot-product attention. Recall from the previous section that this is the mechanism present in both the transformer encoder as well as the transformer decoder. In this mechanism, each input sequence element is associated with three vectors: a query vector, a key vector, and a value vector. The query vector is used to compare

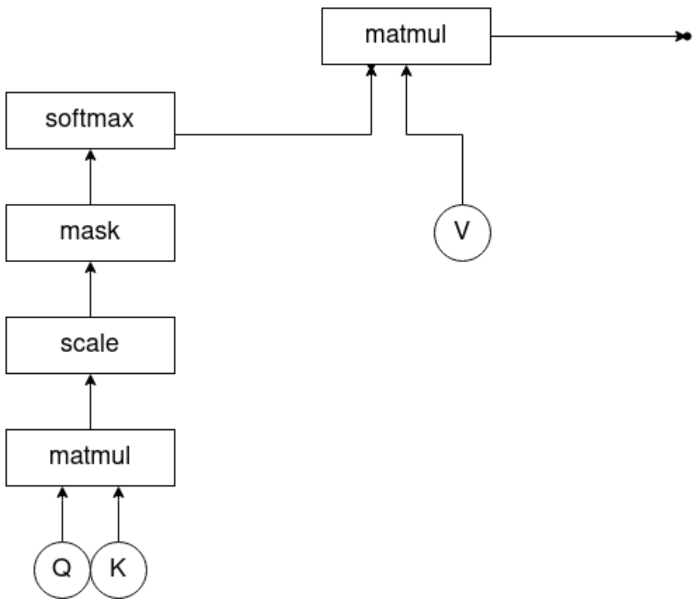

Figure 3.2: A standard attention mechanism, it takes in the vectors Q and K, carries out the consequent dot product and softmax before combining the V vector to get the attention of the entire sequence. Note: `matmul` refers to a shorthand notation of the operation of the dot-product of two matrices, `scale` normalises this dot-product with the dimensionality $\sqrt{d_k}$, and `mask` prevents certain states from being attended.

the importance of each state to other states, while the key vector and value vector are used to represent the "meaning" of each state[a].

$$attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{3.1}$$

To compute the self-attention for each element, the dot product of the query vector Q and the key vector K is computed, and then the result is divided by the square root of the dimensionality of the key vectors. This computation is then passed through a softmax[b] function to obtain a set of attention weights, which represent the relative importance of each token in the input sequence. The attention weights are then used to compute a weighted sum of the value vectors, which gives the final representation for each token. This weighted sum represents the contribution of each token to the overall meaning of the input sequence.

In the transformer architecture, the self-attention mechanism[c] is performed multiple times in each layer of the encoder and decoder. This allows the network to attend to different parts of the input sequence at each layer and capture complex dependencies between the elements of the sequence.

[a]This is, arguably, very abstract since it has its analogy in linguistics where one could imagine each state to be a word of the text. Thus, attention seeks to learn the language by understanding the importance of each word.

[b]A softmax function is a mathematical function that takes a vector of numbers as input and normalises the vector to produce a probability distribution. The output of a softmax function is a vector of the same dimensionality as the input vector, with each element in the output vector representing the probability of the corresponding element in the input vector. – $softmax(x) = \frac{e^x}{\sum_{x_i \in X} e^{x_i}}$

[c]In general discourse, the jargon used to refer to this process is "to attend", i.e., for example, the model attends to different parts of the input sequence.

3.3.2 Cross-Attention

We saw that in self-attention, which was used within a single sequence, each element attended to all other elements within the same sequence. This allowed the model to capture relationships and dependencies between different elements of the sequence, enabling the understanding of long-range dependencies and context.

In contrast, cross-attention involves two sets of inputs: a source sequence and a target sequence. The source sequence typically contains information that the model uses to attend to specific parts of the target sequence. Each element in the target sequence generates a query, while each element in the source sequence generates key-value pairs. The query from the target sequence attends to the key-value pairs from the source sequence, producing attention weights that indicate the relevance or importance of each source element to the target element. The attention weights are then used to compute a weighted sum of the source values, resulting in a contextual representation for each target element.

The key distinction between self-attention and cross-attention lies in the scope of dependencies captured. In self-attention, all elements within the same sequence can attend to each other, allowing for rich context modelling. Cross-attention, on the other hand, captures dependencies between elements of two different sequences, allowing the model to selectively attend to relevant parts of the source sequence while processing the target sequence. This mechanism is particularly useful in tasks where the understanding of relationships between input and output sequences is crucial, such as machine translation. By attending to specific parts of the source sequence, the model can incorporate relevant context and generate accurate translations.

3.3.3 Attention Head

The input is transformed into multiple parallel representations through linear projections, resulting in different sets of queries, keys, and values. These sets are often referred to as heads. Each head operates independently and learns to attend to different aspects or patterns within the input. This parallel processing allows the model to capture diverse and complementary information, enabling it to model complex relationships and capture different types of dependencies more effectively.

After computing attention weights for each head, the results are usually combined or concatenated and passed through another linear transformation to produce the final output. This mechanism allows the model to leverage multiple perspectives and focus on different aspects of the input, enhancing its ability to capture and represent information accurately. The term multi-head attention is used to reflect the use of multiple attention heads in parallel, providing the model with enhanced representational capacity and capturing a broader range of contextual information.

3.4 Prediction Mechanism and Teacher Forcing

Before we dive deep into the aspect of development, a last piece of theory that we need to understand is teacher forcing. During training, the decoder input is typically provided as the ground truth output sequence shifted by one time step. This allows the model to learn to predict each element of the output sequence conditioned on the previous elements.

However, during inference, the ground truth output sequence is not available, so the model needs to generate the output sequence one element at a time based on its previous predictions. This is done by using the previous predicted element as the input to the decoder for generating the next element:

$$p(\hat{\mathbf{x}}_t, \hat{\mathbf{x}}_{t-1}, ..., \hat{\mathbf{x}}_1) = \prod p(\hat{\mathbf{x}}_t | \hat{\mathbf{x}}_{t-1}) p(\mathbf{z}_0) \tag{3.2}$$

where $\mathbf{z}_0$ is the first measured state at time step $t = 0$ and $\hat{\mathbf{x}} = \{\hat{\mathbf{x}}_i\}_{i \in 1:t}$ is the sequence of predictions. As a simple analogy, think of it as the auto-complete feature in smartphones – you give it the first word ($\mathbf{z_0}$), it gives you suggestions for the next words as you go ($\hat{\mathbf{x}}$) based on the language that it has already learnt. This is exactly the same mechanism of sequence generation as that of ChatGPT.

In machine learning discourse, this approach is commonly known as teacher forcing [25] during training, where the decoder input is the ground truth output sequence shifted by one time step. During inference, the same approach is used, but the previous predicted output element is used as the decoder input instead of the ground truth output.

The basic intuition behind teacher forcing is to use the true output sequence from the training set as input to the decoder, rather than using the decoder's own output from the previous time step. In other words, at each time step during training, the decoder receives the correct output element from the training data as input, rather than using its own predicted output from the previous time step. For example, let's say we have a sequence-to-sequence model that is trained to translate sentences from English to French. During training, the model is given pairs of English and French sentences as input and output, and the goal is to learn to generate the correct French sentence given an English sentence as input. In the teacher forcing approach, at each time step during training, the decoder receives the correct French word from the training data as input, rather than using its own predicted French word from the previous time step. This allows the model to learn to generate accurate translations by comparing its predicted output to the true output at each time step.

Teacher forcing helps to stabilise the training process by providing more accurate input to the decoder. It helps to prevent the model from getting stuck in a loop where it could generate incorrect output and then use that output as input to generate further incorrect output. Moreover, it can help to improve the quality of the generated output, since the model is being trained on accurate, ground truth data.

3.5 Development

3.5.1 Data Generation

Let $X(t)$ denote the geometric Brownian motion process at time t. Given the following parameters:

- X_0: The initial value of the process.
- μ: The process noise, representing the standard deviation of the random fluctuations.
- T: The total time period for which the process is generated.
- N: The number of time steps or increments used to discretise the time period.
- δ: The drift coefficient.

We calculate the time increment dt using $dt = \frac{T}{N}$. Then, we create an array of time points t using $t = [0, \frac{T}{N}, \frac{2T}{N}, ..., T]$.

Next, we generate an array W of independent standard normal random variables of length N. Using the cumulative sum of W multiplied by the square root of the time increment, $\sqrt{dt}$[d]. We, thus, obtain a realisation of the standard Brownian motion process:

[d]By multiplying W_i with $\sqrt{dt}$, the resulting values are scaled appropriately to match the desired time increment. This scaling is necessary because the standard Brownian motion process has the property that the variance of the increment over a time interval Δt is proportional to Δt. By multiplying the standard normal random variables by $\sqrt{dt}$, the resulting increments are adjusted to have the desired variance Δt.

$$B(t) = \sum_{i=1}^{N} W_i \sqrt{dt} \tag{3.3}$$

Finally, the geometric Brownian motion process $X(t)$ is obtained as:

$$X(t) = X_0 e^{(\delta - \frac{1}{2}\mu^2)t + \mu B(t)} \tag{3.4}$$

So, $X(t)$ gives us one sequence of Brownian motion. Henceforth, data that contains only this information will be referred to as false-alarm-free data (and in some contexts, the ground truth). This can be understood as the movement of one particle and hence, also an environment with one track and no noise. To simulate an increase in the process noise, we increase μ as discussed above.

To simulate an increase in the measurement noise, we incorporate an additive term $\omega \sim \mathcal{N}(0, \sigma^2)$ to the above equation, where we call σ the scale of this noise.

$$Z_p(t) = X_0 e^{(\delta - \frac{1}{2}\mu^2)t + \mu B(t)} + \omega \tag{3.5}$$

Henceforth this information Z_p will be referred to as the particle measurements. The set of all measurements that we get from the formulation of $Z_p(t)$ is referred to as $\mathbf{Z}_p$, as we had discussed in Ch. 2.

As a final step, we add clutter ($\mathbf{Z}_c$) to this information. Recall from Ch. 2 that this clutter is simulated from a Poisson point process. To model clutter using a Poisson point process, we use the following steps:

1. Determine the region or space of interest where clutter is to be modeled.
2. Choose an appropriate intensity parameter λ that represents the clutter density. This parameter determines the average number of clutter points in a given area or volume.
3. Generate random points according to the Poisson point process. The number of points to generate can be determined by sampling from a Poisson distribution with parameter λ, yielding the count of clutter points.
4. Distribute the generated clutter points randomly within the region of interest.

3.5.2 The Best Vanilla Transformer

The idea in general is to develop a mechanism that would be able to map from $\mathbf{Z}$ to $\mathbf{X}$, and by learning these mappings, will subsequently be able to predict $\hat{\mathbf{X}}$. As a starting point for testing, we work with the false-alarm-free data (as described in the previous section).

From Fig. 3.3, it can be seen that a vanilla architecture consisting of 3 encoders and 2 decoders (henceforth `3E2D`) gives the lowest training and validation loss. These results are attained completely through a brute-force approach wherein many possible encoder-decoder combinations were tested. However, the reason why this architecture works the best is because of the nature of the attention mechanism. Each decoder layer attends to the output of the previous decoder layer, as well as the output of the encoder layers. The encoder layers capture the features of the input sequence and generate a sequence of hidden representations, while the decoder layers generate the output sequence based on the hidden representations from the encoder and the previous decoder layers.

So clearly, it is evident that the number of encoder and decoder layers in a transformer network can significantly impact the performance of the model. `3E2D` gives better results because it allows the model to capture more complex features of the input sequence. We can, hence, see that the encoder layers are responsible for extracting high-level features from the input sequence, and increasing the number of encoder layers can provide the network with a more comprehensive understanding of the input sequence – by increasing the dimension of the parameter space used to represent the sequences.

Further, as we see, using fewer decoder layers helps reduce the risk of overfitting and improves generalisation performance. Increasing the number of decoder layers can make the network more prone to overfitting on the training data[e], which is suggested by these preliminary results.

3.5.3 Cyclical Annealing of the Learning Rate

In this section, we do a small detour to talk about this mechanism that helps us converge the loss function in a more stable manner as shown in Fig. 3.4.

Cyclical annealing of the learning rate [26] is a technique used to vary the learning rate during training in order to improve the performance of the network. The basic idea is to increase the learning rate for a certain number of epochs and then decrease it for a certain number of epochs, repeating this cycle throughout training. The motivation behind cyclical annealing is that using a high learning rate can help the network escape local minima, while using a low learning rate can help the network fine-tune its parameters for better performance. By varying the learning rate in a cyclical manner, the network can explore a wider range of parameter settings and potentially find better solutions.

There are several different ways to implement cyclical annealing of the learning rate, but one common approach is to use a triangular learning rate policy. In this approach, the learning rate starts at a minimum value and increases linearly for a certain number of epochs, then decreases linearly back to the minimum value for the same number of epochs. This triangular cycle can be repeated several times throughout training. Another approach to cyclical annealing is to use a cosine learning rate policy, where the learning rate decreases according to a cosine function over the course of training. This approach is based on the idea that a cosine function can provide a smooth decrease in learning rate that is less likely to cause the network to overshoot the optimal solution. In our experiments, a cosine cyclical annealing of the learning rate is used.

When the learning rate is too low, the optimisation process may get stuck in a local minimum, leading to slow convergence of the loss function. On the other hand, when the learning rate is too high, the optimisation process may overshoot the global minimum and oscillate around it, leading to instability in the loss function. By gradually increasing the learning rate during the first few epochs, cyclical annealing allows the optimisation process to explore the parameter space more quickly and find a good starting point for convergence. After the learning rate is increased, it is gradually decreased over the next few epochs, allowing the optimisation process to exploit the information gained during the exploration phase and refine the model parameters. By balancing exploration and exploitation, cyclical annealing helps the model achieve better generalisation performance, i.e., the ability to perform well on new, unseen data.

During training, the gradients of the loss function with respect to the model parameters may become noisy or fluctuate due to factors such as noisy data or high learning rates (as shown in Fig. 3.4).

[e] **The curious case of 1-track `1E1D` and `4E4D`:** Upon closer examination, Fig. 3.3 shows that the validation loss is much lower than the training loss for `1E1D` and `4E4D` in case of 1-track prediction. While this is not impossible, it is rare. One possibility is that there is a mismatch between the training and validation data. For example, the validation data may be easier to learn from than the training data, or the training data may contain some biases or artifacts that the model is not able to exploit.

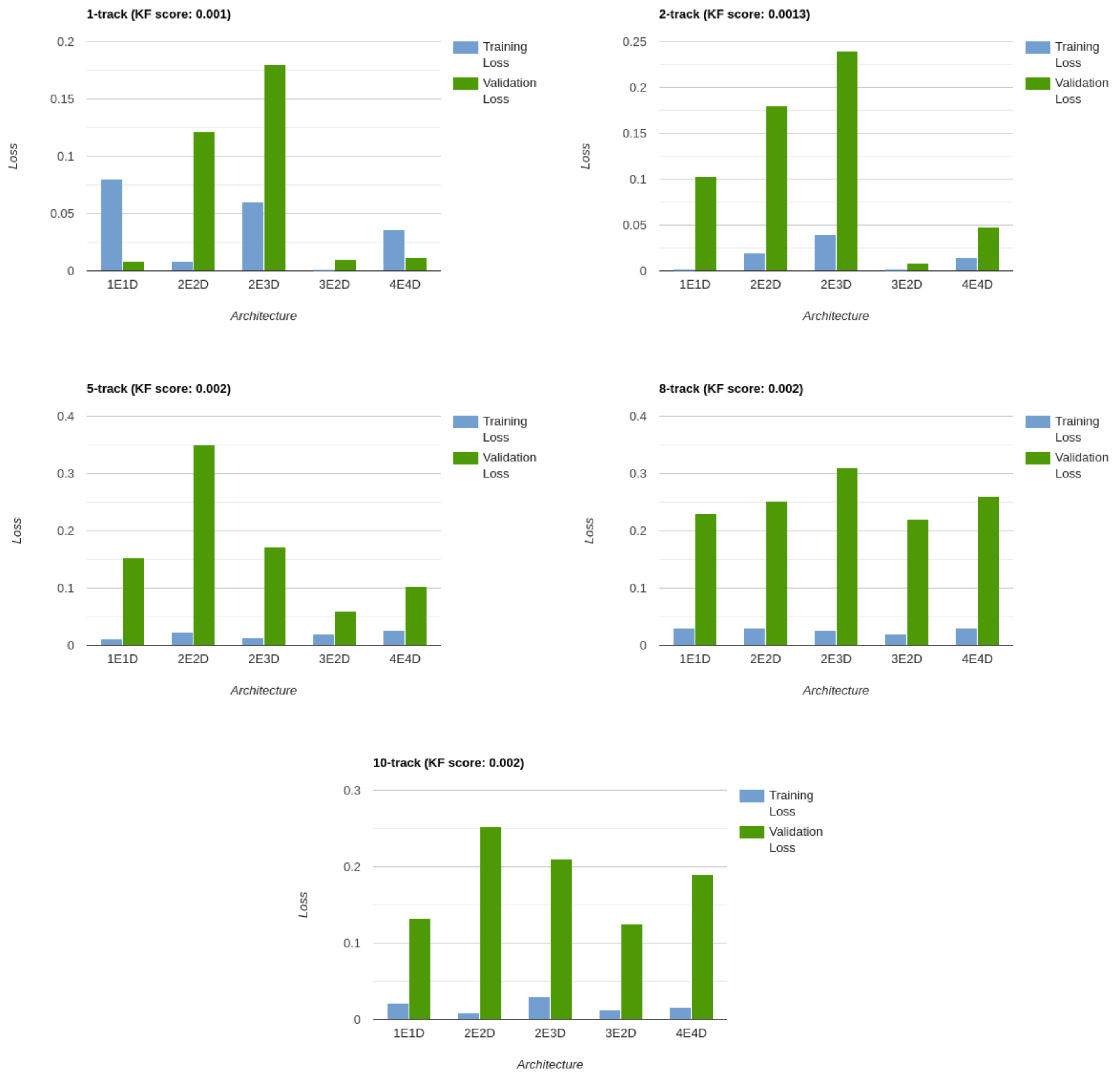

Figure 3.3: Performance: testing for different combinations of encoders and decoders of v1.0 for different number of particles in the system

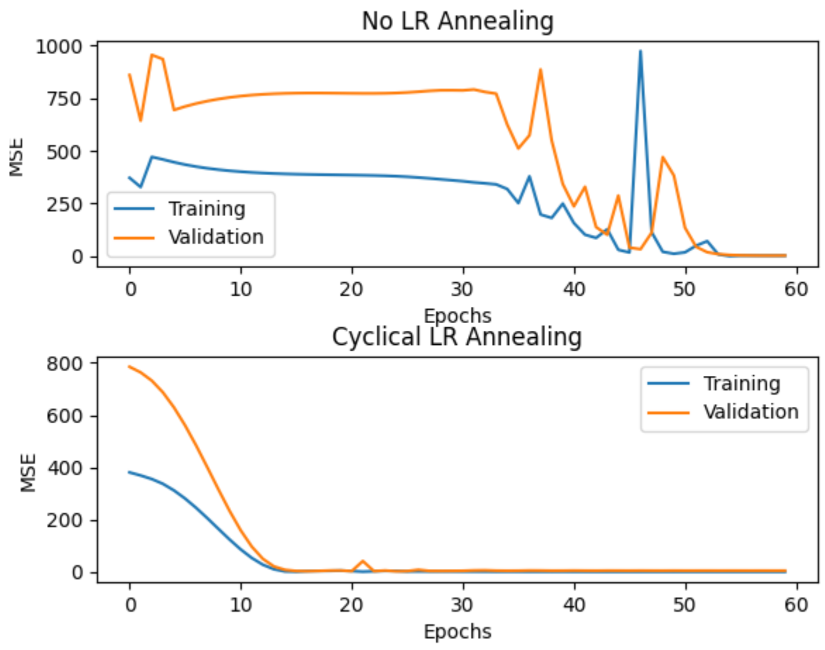

Figure 3.4: In noisier regimes, cyclical annealing of the learning rate can ensure a more stable convergence of the loss as the model learns

By gradually decreasing the learning rate after the exploration phase, cyclical annealing can help in reducing the impact of noisy gradients and achieve a more stable convergence of the loss function.

3.5.4 How does pure attention perform?

At this point, it would be interesting to ask ourselves, "What is attention actually learning?" This question is pertinent to further demystify the powers of attention and unlock it, instead of looking at it as a black box. In this attempt, we decided to have a model whose primary information capturing aspect would be the attention block. The model that comes closest to this type of a mechanism is an encoder-only model (Fig. 3.5). By using an encoder only model, we can exploit, as we discuss later, the implicit bidirectionality of attention.

What is bidirectionality?

In the context of sequences, bidirectionality refers to the ability to process information in both forward and backward directions. Typically, when we talk about sequences, we refer to a series of elements ordered in a specific manner, such as a sentence, DNA sequence, or time series data (in the context of this study). In natural language processing, bidirectional models have been developed to better understand and generate text by considering the surrounding context in both directions. The idea behind bidirectional models is that the meaning of a word in a sequence can depend on both the preceding and succeeding words. By processing the sequence in both directions, the model can capture dependencies and relationships that may be missed in a unidirectional approach.

It is important to keep in mind that while any kind of transformer learns the bidirectionality of sequences because that is what transformers are inherently meant to do, we come the closest to visualise this bidirectional learning through an encoder-only mechanism – encoder-only models learn the entire context in one pass, so visualising their outputs can help us get a grasp on what attention actually learns. On the other hand, using a model with the transformer decoder would help us understand how good the model is at generating this sequence based on what it has learnt (see teacher forcing). One popular example of a bidirectional model is the Bidirectional Encoder Representations from Trans-

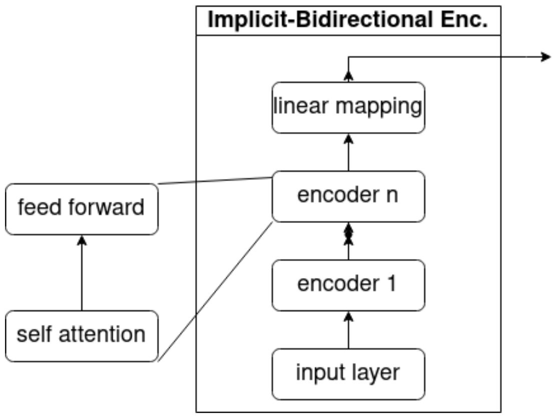

Figure 3.5: The implicit bidirectional encoder

formers (BERT) [27]. BERT takes advantage of bidirectionality by using a transformer architecture to process the sequence both left-to-right and right-to-left simultaneously. This allows the model to generate contextualised representations for each word in the sequence based on its surrounding context in both directions.

Multi-head attention and its implicit bidirectionality

When used as part of a transformer model, attention ends up being used in a bidirectional fashion where the input sequence is first processed by a self-attention layer that allows each position to attend to all other positions in the sequence. This creates a representation that captures bidirectional dependencies and can be used as input to the subsequent layers. In this way, multi-head attention, in combination with other transformer components, can be used to model bidirectional sequences. Since we already have the entire sequence, we can use that to understand the past and the future behaviours.

Fig. 3.6 visualises how this model declutters the noisy environment. While not a perfect measure of judging how well the model behaves, visualisations are a good starting point to get a sense of what is going on. All the plots on the left show what the actual particular motion ought to be. These are referred to as the ground truths, and this is what the model must succeed at reconstructing. On the centre of each plot, an observation of what a realistic light-sheet microscopy output would look like is simulated – it is made up by the data generation strategy discussed above. This renders the observation non-inferrable by simply eyeballing. On the right of each plot, we see what model is able to predict from its input.

It can clearly be inferred visually that as the environment gets denser, the ability for the model to declutter and, hence, efficiently track the particles becomes worse. In the case for the 10-particle system, for example (Fig. 3.6, bottom), the model tries to construct the ground truth but is unable to perfectly match with the reality. The drawbacks are further corroborated and formalised in the next section.

3.5.5 Limitations: SNR and Poisson Rate

Signal-to-Noise Ratio (SNR) is a measure of the strength of a signal relative to the background noise [28]. SNR is typically expressed in decibels (dB), and higher SNR values indicate better signal quality and less noise interference.

$$SNR = 10.log_{10}\frac{P_s}{P_n} \tag{3.6}$$

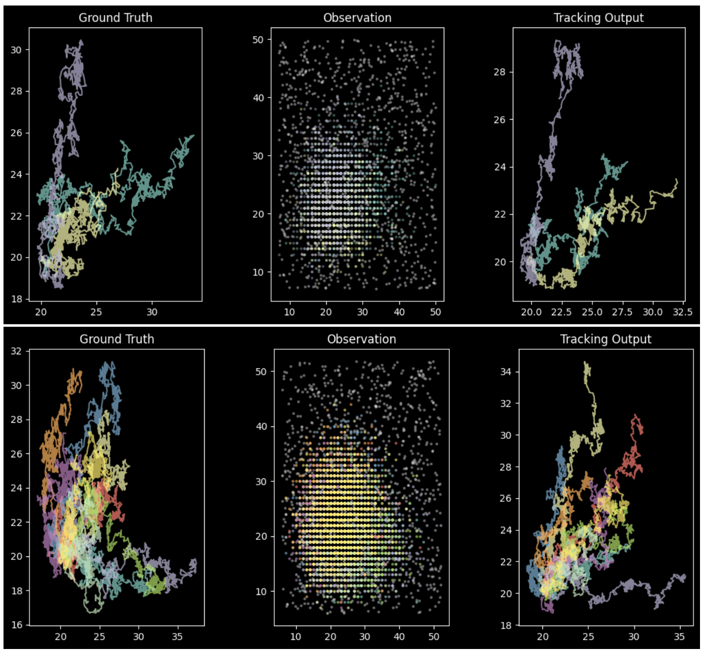

Figure 3.6: Visualisations of the model prediction for a 3-particle system (top) and a 10-particle system (bottom) – each plot shows what the actual movement is like (left), what the model gets as its input (centre) and what the model predicts (right). **We see that coming from a mechanism that was developed for language processing and applying it to particle tracking without changing anything gives results that are not the best but promising.**

where P_s and P_n are the powers of the signal and the noise, respectively. Mathematically, these are given by:

$$P_s = \frac{1}{|\mathbf{S}|} \sum_{i \in \mathbf{S}} i^2 \tag{3.7}$$

$$P_n = \frac{1}{|\mathbf{N}|} \sum_{i \in \mathbf{N}} i^2 \tag{3.8}$$

where $\mathbf{S}$ is the set of elements in the clean signal, $\mathbf{N}$ is the set of elements in the noisy signal, $|\mathbf{S}|$ is the number of elements in $\mathbf{S}$, and similarly, $|\mathbf{N}|$ is the number of elements in $\mathbf{N}$.

The SNR value shows how strong the signal is compared to how strong the noise is. A 20 dB SNR, for instance, indicates that the signal strength is 100 times higher than the noise power. A signal's quality is poor when the signal strength is equal to the noise strength, or an SNR value of 0 dB. In the context of analysing how well the model is decluttering a noisy environment, it is thus, of keen interest to see how SNR changes with noise levels.

As established in the previous chapters (and [29]), the clutter in the observation of fluorescence microscopy can be modelled from a Poisson point process. By studying the SNR versus this Poisson rate[f],

[f]In Fig. 3.7, we see that the shape of the plot is a rectangular hyperbola. This is because the SNR is proportional to the square root of the signal, while the noise is proportional to the square root of the Poisson rate. Consider X and Z to be two signals, X being the clean signal (ground truth) and Z being the measured (noisy) signal. We know that $SNR \propto \frac{X}{\sqrt{N}} = \frac{X}{\sqrt{\lambda}}$, where λ is the Poisson rate (expectancy of N, noise sampled from a Poisson point process). This implies that $SNR^2.\lambda \propto X^2 = C$ where C is some arbitrary positive constant since X is the ground truth. This shows a rectangular hyperbolic relationship between SNR^2 and λ.

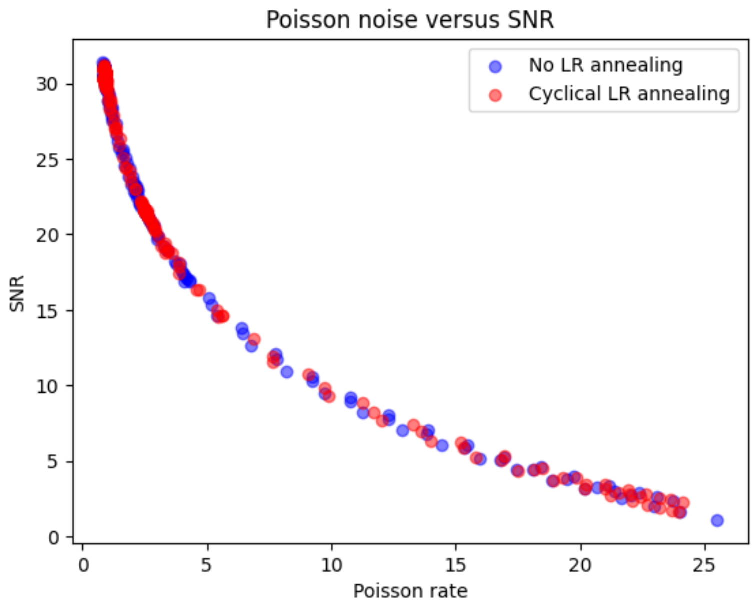

Figure 3.7: Change in SNR on an increase in the Poisson rate

we can gain insights into how the clutter affects the performance of the neural network model, and how different levels of clutter can impact the accuracy and reliability of the decluttering process.

From Fig. 3.7, it can be seen that in noisy cases, the SNR drastically degrades. This explains why, despite a good convergence in the loss function (MSE), we don't have an accurate reconstruction of the ground truth. If the signal or noise are both weak, a low MSE can nevertheless lead to a low SNR. For instance, a low MSE can nevertheless produce a poor SNR if the signal is faint or tainted by noise. In certain scenarios, the model may predict the target variable well but may still find it difficult to separate the signal from the noise, resulting in a poor SNR.

Lack of Robustness to Sampling Changes

This is where it is important to think about the experimental bottlenecks that can come up – during experimentation, we don't necessarily have time-steps in the order of 10^3, there are usually some 100 to 150 time-steps. So, the model should effectively work on sequences with smaller time-steps. In Fig. 3.8, we see, again, that **despite a great convergence in MSE, the tracking output is not compatible with the ground truths**. In other words, while the model is decluttering the noisy environment, it is not decluttering to something that is desirable.

3.5.6 Can a Tracking-Specific Loss Function Help?

In the previous version, we saw that despite the fact that the loss was converging, that convergence in loss did not translate to a desirable filtering. This suggests that it was learning how to minimise its losses but it was not learning how to correctly superimpose what it was predicting to the ground truth. Both of these usually mean the same thing, which is why in this context, this becomes especially interesting as a point for algorithm development. On a fundamental level, this problem means that the way in which the model learns needs to be changed, or in more concrete terms, the loss function, that the model learns to minimise, needs to be changed so that it accounts for the issue discussed above.

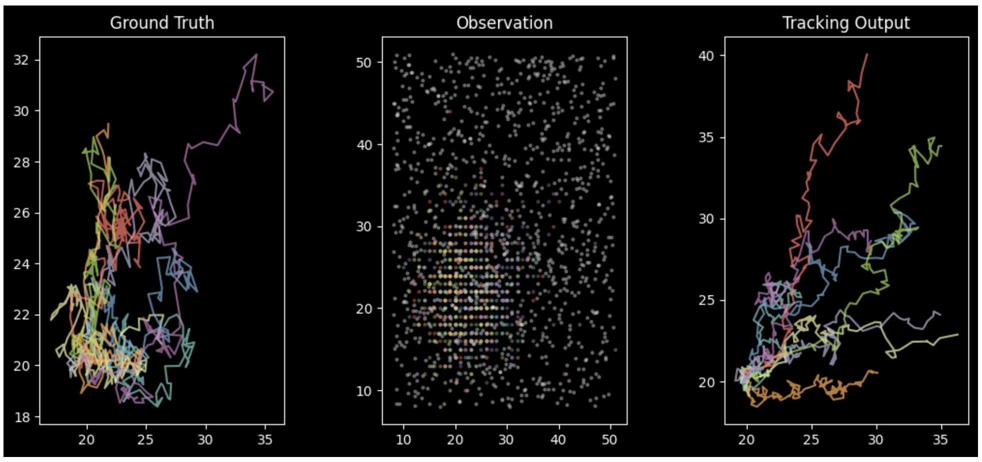

Figure 3.8: A plot of ground truths, simulated microscopy observation and the tracker output for an 8-particle system for 100 time-steps

The GOSPA Metric

GOSPA refers to the Generalised Optimal Subpattern Assignment metric, which is a performance metric used in multi-object tracking [30]. It is a performance metric that is used to evaluate the accuracy and robustness of multi-object tracking algorithms. It is designed to measure the distance between two sets of tracks: a set of ground-truth tracks, and a set of predicted tracks generated by a tracking algorithm. The GOSPA distance is calculated by comparing each predicted track to its best matching ground-truth track, and then summing the distances over all tracks. The resulting value represents the overall tracking error of the algorithm. This means that this can be a valid candidate to be the loss function. The GOSPA metric can handle multiple sources of error in a tracking algorithm, including false positives, missed detections, and track fragmentation. It can also handle situations where the number of tracks in the ground truth and predicted sets are not the same.

The GOSPA metric is defined by four parameters: the gating distance, the order of the metric, the p-norm[g], and the cardinality penalty. The gating distance determines the maximum distance between a predicted track and a ground-truth track for them to be considered a match. The order of the metric and the p-norm control how the distances between predicted and ground-truth tracks are aggregated. The cardinality penalty penalises the algorithm for generating too many or too few tracks compared to the ground truth.

Consider two sets of tracks X and Y, with N and M elements, respectively. Each element in a track set represents an object that has been tracked over time. The GOSPA metric measures the distance between the two sets of tracks by taking into account the following factors:

- the localisation error between each pair of matched tracks
- the cardinality difference between the two sets of tracks, and
- the number of missed detections and false alarms in each set of tracks

We first define a distance matrix D of size $(N+1) \times (M+1)$. $D_{i,j}$ holds the value of the Euclidean distance between the i^{th} track in the set X and the j^{th} track in the set Y. Then, two thresholds, c and p, are defined which represent the maximum allowable localisation error and the cardinality differences, respectively. These thresholds are used to identify the "good" matches between tracks in X and Y. This set of good matches can be defined as:

[g]A p-norm is a measure of the size of a vector or function, defined as the p^{th} root of the sum of the absolute values of its components raised to the power of p, where p is a real number greater than or equal to $1 - \|x\|_p = (\sum_{i=1}^{n} |x_i|^p)^{\frac{1}{p}}$

$$G(c,p) = \{(i,j) : D_{i,j} \leq c, |N - i - M + j| \leq p\} \tag{3.9}$$

Here, evidently $|N - i - M + j|$ represents the cardinality difference between sets X and Y after matching tracks i and j. Then, GOSPA can be computed as:

$$\begin{aligned} GOSPA(c,p,\alpha,\beta) = \alpha c^2 + \sqrt{\frac{\beta}{|G(c,p)|} \sum_{i,j \in G(c,p)} D_{i,j}^2} \\ + (1-\alpha-\beta)\left(\frac{1}{N}\sum_{i=1}^{N}\sqrt{\min_{j=0}^{M} D_{i,j}^2 + c^2} + \frac{1}{M}\sum_{j=1}^{M}\sqrt{\min_{i=0}^{N} D_{i,j}^2 + c^2}\right) \end{aligned} \tag{3.10}$$

Here, α and β are weighing coefficients that determine the importance of the localisation error and cardinality difference terms, respectively. The first term in the equation penalises matches with localisation error greater than c, while the second term penalises matches with cardinality difference greater than p. The third term penalises missed detections and false alarms.

Why is GOSPA better suited than MSE?

One reason is that the MSE treats each predicted track and ground truth track independently, and does not take into account the correspondence between them. In other words, the MSE does not consider which predicted track corresponds to which ground truth track. This can be a problem in tracking applications, where the correspondence between predicted and ground truth tracks is important. We can see this in Fig. 3.6 (top) where, in the 3-particle environment, the turquoise and the yellow tracks seem to have interchanged from $\mathbf{X}$ to $\hat{\mathbf{X}}$.

On the other hand, the GOSPA loss function takes into account the correspondence between predicted and ground truth tracks, and penalises both the cardinality (i.e., the number of tracks) and the distance between them. The GOSPA loss is designed to be more robust to mismatches between predicted and ground truth tracks, and can handle situations where the number of tracks is different between the predicted and ground truth sets.

But GOSPA does not give gradients!

Gradients are a fundamental concept in calculus that represent the rate of change of a function with respect to its input variables. In the context of neural networks and machine learning, gradients are used to determine the direction and magnitude of parameter updates during the optimisation process. Differentiability of loss functions in neural networks is crucial as it enables the computation of gradients, facilitating optimisation algorithms for backpropagation via efficient parameter updates. But GOSPA is not differentiable.

In recent years there has been an increased interest towards the research for using non-differentiable loss functions for training. Many natural language translation models employ metrics like BLEU (Bilingual Evaluation Understudy) and F1-score which are non-differentiable. This can be done in two ways. (1) Instead of using gradient based learning where we optimise the loss function using the said gradients, we can use evolutionary algorithms like the genetic algorithm or the Jaya algorithm [31] for that task. This is a separate domain of study that goes deep into soft-computing. (2) We can, otherwise, use a proxy differentiable metric that gives gradients to update the weights and the biases, but a system of rewards is put into place for the model to minimise the non-differentiable loss function, i.e., we make

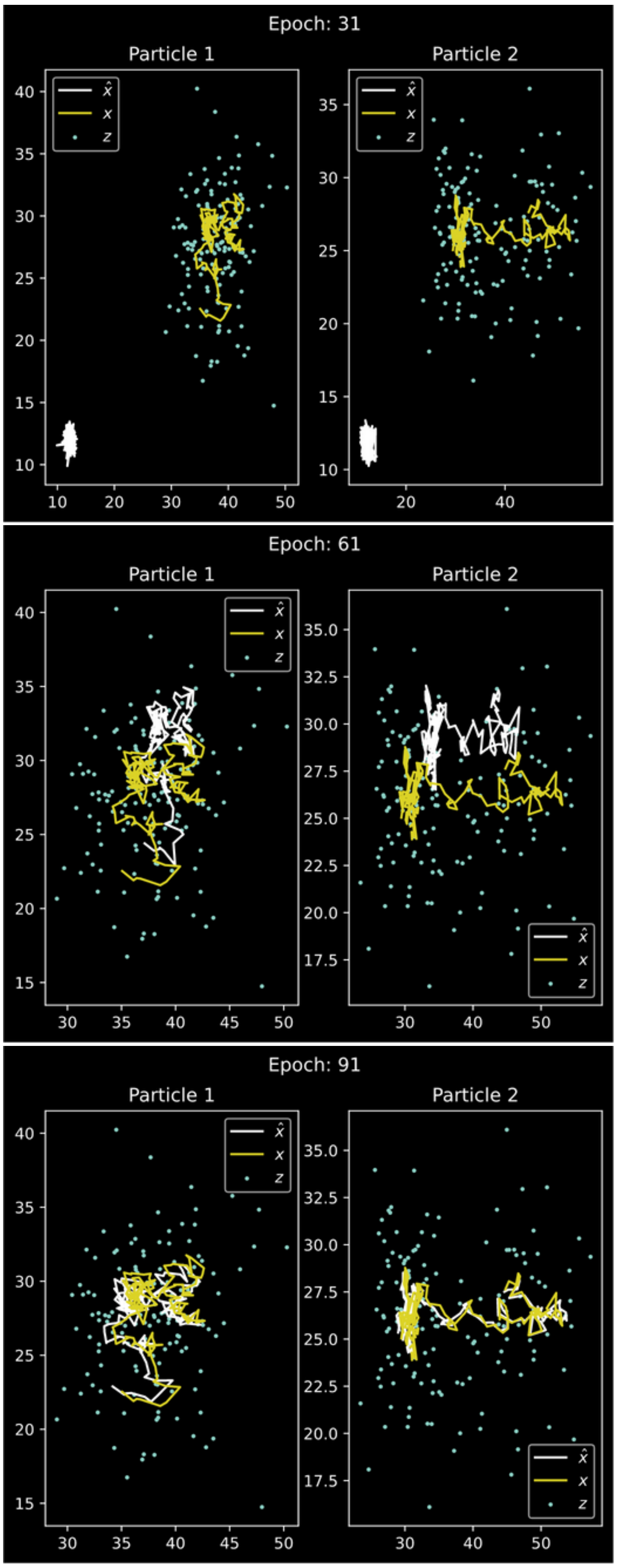

Figure 3.9: Visualisation of the evolution of how the attention-based model learns to map from $\mathbf{Z}$ to $\mathbf{X}$. The turquoise point cloud represents measurements $\mathbf{Z}_p$ (we omitted $\mathbf{Z}_c$ from visualisations for brevity), the yellow curve shows the ground truth $\mathbf{X}$ and the white curve shows the evolution what the predictions $\hat{\mathbf{X}}$ look like in the thirty-first, sixty-first and the ninety-first epochs.

the model learn to maximise its rewards which makes it minimise the loss function. In doing that, we compartmentalise the process of loss minimisation and weight updation. In our experiments, we use the second method.

We see in Fig. 3.9 how this model is able to learn to map from measurements to the ground truth in a 2-particle system. For brevity, we do not show clutter, but keep in mind that the model learns from a cluttered input, as has been discussed before. These visualisations clearly show how the model learns from the entirety of the given data to map to ground truths. This helps us really visualise how attention actually learns, as we are successfully able to represent the evolution of its latent space. Although this is mere eyeballing, more robust arguments will be developed in the next chapter.

3.6 Conclusion

The advancements in deep learning and transformer-based approaches open up new possibilities for extracting meaningful information from complex sequences. By combining the power of deep learning with the ability of transformers to capture bidirectional context, we can push the boundaries of particle tracking performance and contribute to advancements in various scientific disciplines. In this chapter, we focused our attention to the specific application of particle tracking in clutter and how transformers have emerged as a powerful tool in this domain. By leveraging their bidirectional processing and attention mechanisms, transformers show promising results in accurately tracking particles in cluttered environments. This has significant implications for various scientific and engineering fields.

It is not a point of debate that the use cases of attention-based models and Bayesian inference models are not the same. It is, hence, of primal importance to know how to compare the theories discussed in Ch. 2, and the strategies developed in this chapter. In the next chapter, we delve deep into this conundrum in detail. We discuss what regimes are more suitable for Bayesian strategies and what others for attention-based models. We try to understand what it is exactly that the attention is “learning” and what its latent spaces mean, and many other points of interest that help in critically analysing attention-based models against the traditional Bayesian strategies.

Chapter 4

Is Attention All You Need?

Throughout the previous chapters, we talked about the theory behind Bayesian filtering approaches and developed strategies for carying out filtering using attention-based methods. In this chapter, we will explore the regimes where we could and should use either or both of these methods. We will argue why attention-based methods work the way they do, what exactly they need to work well and efficiently, certain limitations that they pose, and also how we can combat these limitations.

4.1 Sanity Check

We expect that in low noise regimes ($\mu \to 0, \omega \to 0$), Kalman filtering would be the best approach for tracking. So, a good question to ask at this point would be, "with enough data, can attention-based models reach Kalman filter-like accuracy?" Our goal is to decrease the loss. Increasing the number of batches processed during training can potentially decrease the loss, but it is not guaranteed.

In general, increasing the number of batches processed during training can help the network learn more accurate representations of the input data and improve its performance. This is because each batch provides the network with additional examples to learn from, and by processing more batches, the network can potentially discover more patterns in the data and better capture the underlying structure of the task. This is exactly what we see in Fig. 4.1, as the number of batches that are processed increases, the loss attains almost the same level as Kalman filters in a clean data, still staying in a 2-particle system.

4.2 Cliffs in Accuracy-Noise or Error-Noise Dynamics

Consider the case where there is absolutely no noise in the observation of fluorescence microscopy ($\mu = 0, \omega = 0$). In this case, any objectively "good" algorithm should have an objectively high accuracy and/or an objectively low error. We expect this high accuracy (and low error) to be sustained as the measure of noise is increased to a certain extent (because the very motive of stochastic filtering methods is to handle noise). However, at certain levels of noise, we would expect certain algorithms to "break", in the sense that their associated accuracy would drastically decrease (or their associated error would drastically increase), forming a "cliff" (Fig. 4.2) on plotting this metric against increasing noise.

If we compare these cliffs while looking at the accuracy-noise or the error-noise dynamics of several algorithms (particularly Bayesian methods against attention-based methods), we can find definite regimes for when one algorithm works better or worse than the other. The principal objective is to be able to see if the attention-based methods are able to prolong the point at which an algorithm "breaks". This

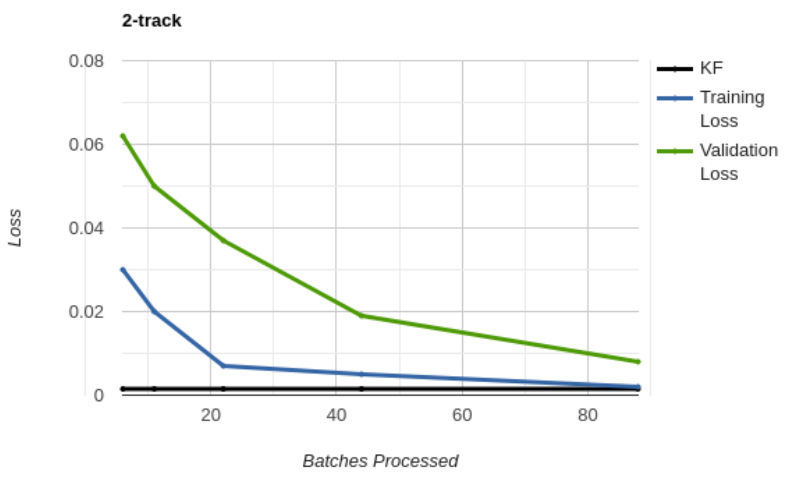

Figure 4.1: With enough data to learn from, an attention-based model can reach Kalman filter-level of accuracy.

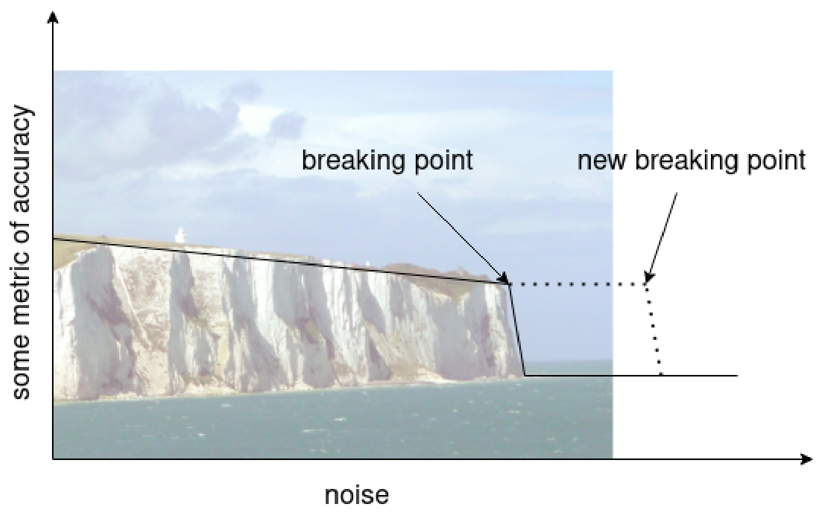

Figure 4.2: As we increase the noise and plot some metric of accuracy against this noise, we should find that for low noise, most models maintain a high accuracy, suddenly breaking down at a particular level of noise, making so called "cliffs". The key idea is to find a method that would prolong this break-point as noise increases.

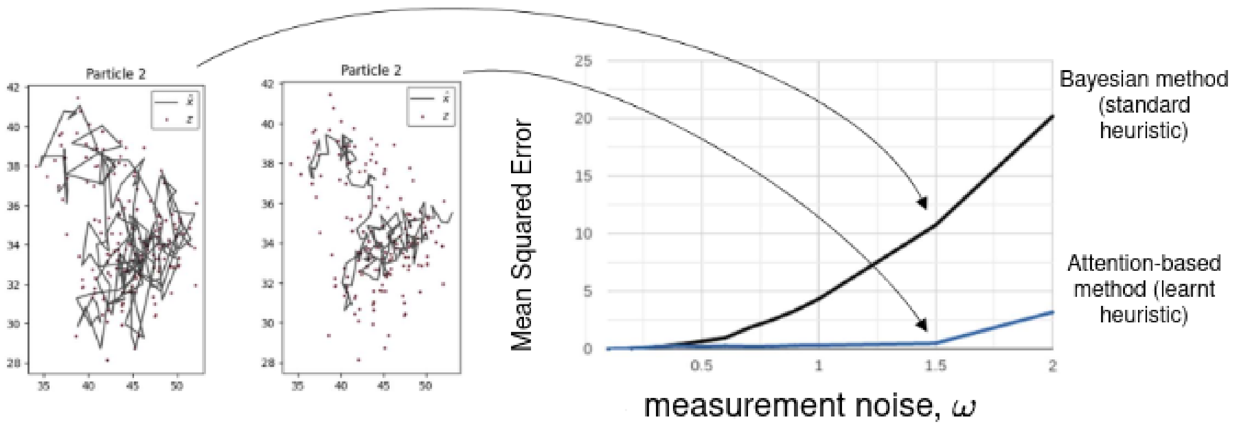

Figure 4.3: Cliffs of MSE observed on increasing measurement noise, ω, for Bayesian and attention-based approaches of filtering in a two-particle system.

would, in its practical sense, mean that the algorithm that prolongs this breakage can handle much higher levels of noise than the algorithm that breaks sooner.

4.2.1 Cliffs of Mean Squared Error

To make the comparisons as comprehensive as possible, we work on very simplistic environments. Staying in a two-particle environment without any other false positive, from Fig. 4.3, it can be seen how the cliffs for MSE are formed on increasing the measurement noise, ω. Initially, both methods are comparable (so it is judicious to use Bayesian methods because of their energy and time effectiveness), but as the measurement noise increases, the Bayesian method starts “breaking” while the attention-based method gives more stable dynamics. Note that these cliffs are “inverted” because a metric of error is plotted instead of a metric of accuracy (opposite of what is shown in Fig. 4.2). More interpretable cliffs will be shown in the following sections.

4.2.2 Cliffs of Degree of Overlap

It was established in the previous chapter that while MSE is a good starting point to carry out the analysis, it is not the most intuitive metric for tracking. We saw in Fig. 4.3 that the method of Bayesian inference breaks much sooner than the attention-based method, but just looking at MSE does not tell us why it is happening.

It is important to note that at each time step, the model has two hypotheses to choose from. In low noise, if the particles are far enough, that would not be an issue. But, as we increase the noise (either measurement noise, or process noise, or both), chances of the model confusing one hypothesis for the other increase, thereby leading to more haphazard predictions. Although we can get a glimpse of this in the visualisations of Fig. 4.3, this can be formalised in a better way. We use two metrics to do that, the state matching metric and the Jaccard coefficient.

- **State matching metric (SMM)**: This is a conservative and a strict metric which considers each track to be independent of the others (i.e., assuming that the predictions have already been assigned the track). The comparison between the two sequences is done by calculating the ratio of the number of elements in the predicted sequence that overlap with the corresponding elements in the ground truth sequence, to the total number of elements in the ground truth sequence. The overlap between elements is determined based on a distance threshold, which is calculated using the Manhattan distance (square threshold, see Fig. 4.4, left). This gives a ratio between 0 and 1, where 1 indicates a perfect match between the predicted and ground truth sequences.

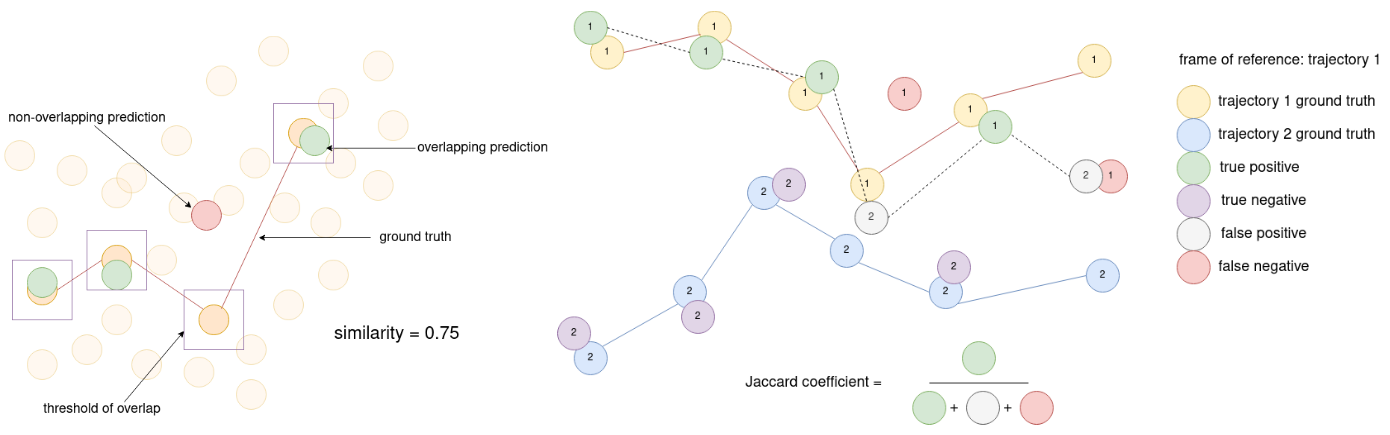

Figure 4.4: Degrees of overlap, left: state matching metric (similarity), right: Jaccard coefficient.

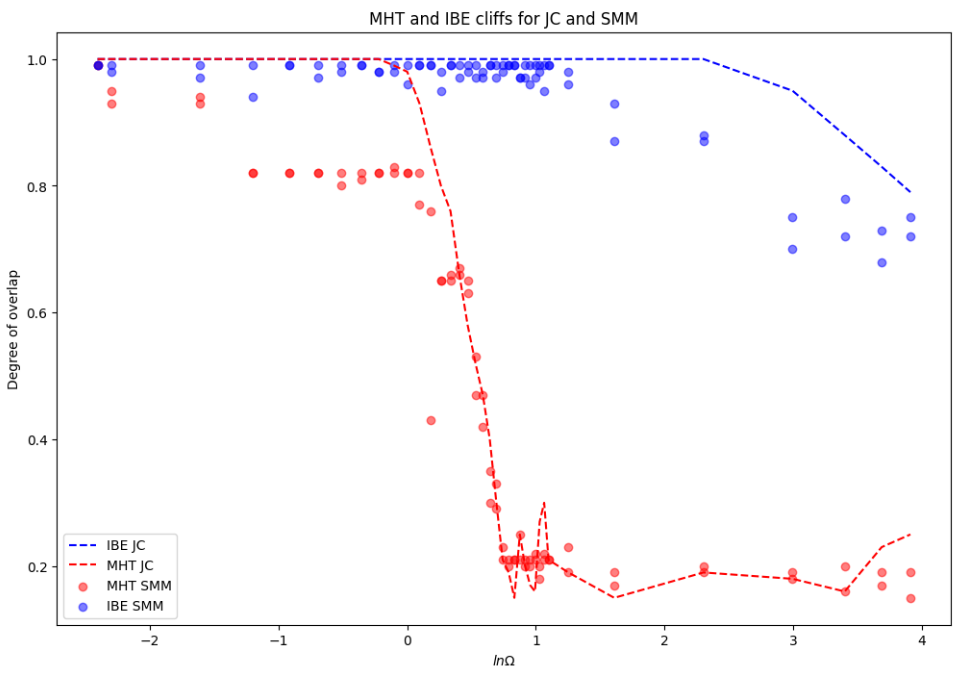

Figure 4.5: Cliffs of JC and SMM on increasing measurement noise, ω, for Bayesian (MHT) and attention-based (IBE) approaches. Note that $ln\Omega$ refers to the natural logarithm of the set Ω of all the measurement noise levels ω.

- **Jaccard coefficient (JC)**: The Jaccard coefficient is a measure of similarity between two sets of elements. It is defined as the ratio of the size of the intersection of the two sets to the size of their union. The Jaccard coefficient ranges from 0 to 1, with 1 indicating that the two sets are identical and 0 indicating that they have no elements in common. In more specific terms (see Fig. 4.4, right), it is the ratio of the number of true positives (the correct predictions) to the sum of the number of true positives, false positives (the predictions that should be assigned to another track but are assigned to the reference track) and false negatives (the predictions that should be assigned to the reference track but are not). This is more useful when there are relatively low tracks[a], but since it makes do without the Manhattan threshold, it is arguably less conservative.

In Fig. 4.5, we see cliffs that are much more akin to what we hypothesised back in Fig. 4.2. We see that attention-based mechanisms are able to handle significantly higher levels of measurement noise whereas Bayesian models break at much lower noise levels. It is important to note that we have taken

[a]In case of a two-particle environment, for example, the prediction that is closer to the reference track in terms of its Euclidean distance is considered to be the correct prediction, so no precision thresholds are required.

the natural logarithm of the noise levels in the x-axis, which means that the difference in the breakage of the Bayesian method and the attention-based method is exponentially higher than what can actually be seen in Fig. 4.5. One might observe that the breakdown of SMM happens in a way that resembles steps. This is likely an artifact of its conservative thresholding. It is also clearly observable that JC is much more "generous" than SMM as the JC cliffs have a smoother and tardier breaking points. Delving deep into the details of the specific behaviours of these metrics might prove to be interesting in the future.

4.3 Attention has an energy problem

A FLOP (Floating-Point Operation) is a mathematical operation that involves floating-point numbers, which are numbers with a decimal point that can represent both large and small values with high precision. Examples of floating-point operations include addition, subtraction, multiplication, and division of floating-point numbers, as well as more complex mathematical functions such as trigonometric functions, logarithms, and exponentials. FLOPs are used as a unit of measurement to quantify the processing power of a computer[b], particularly in applications that involve intensive numerical computation such as scientific simulations, data analysis, and machine learning.

Estimating the FLOPs required for a particular algorithm involves understanding the mathematical operations involved and the number of times they are performed. In general, the FLOPs required for an algorithm depend on the input size and the complexity of the algorithm.

Consider a relatively simple environment with 2 particles. Recall from Ch. 2 that each state corresponding to the motion model of a single particle has 2 values, x_t and y_t, describing their positions at time instant t, and for each particle, we have, say, 150 time steps. All of this information is stored in a matrix of size $(2, 2, 150)$. The parameter size of the hidden layers used in the models is 512 and that of the feed-forward network is 1024. Estimating the FLOP requirements for each step in the process of an attention-based model, we have:

- for input embedding, $(2 \times 2 \times 150) \times 512 = 307200$ FLOPs,
- for multi-head attention, $3(Q, K, V) \times (2 \times 2 \times 150) \times 3 \times 8 = 43200$ FLOPs, where there are 3 attention heads and the size of the attention vectors is 8,
- for context creation, $(2 \times 2 \times 150)^2 \times 8 = 2880000$ FLOPs,
- for concatenation and linear projection, $(2 \times 2 \times 150) \times 512 = 307200$ FLOPs,
- for fully connected feed-forward neural network, $(2 \times 2 \times 150) \times 1024 \times 3 = 1843200$ FLOPs, where 3 includes 2 linear layers and 1 activation layer,
- and for output embedding, $(2 \times 2 \times 150) \times 512 = 307200$ FLOPs.

On accounting for 3 encoder layers, we estimate an upper bound of about 10.35 million FLOPs required for running an attention-based encoder model.

A typical desktop CPU uses around 100-200 watts of power while running at full load [32]. This corresponds to a power consumption rate of around 0.1-0.2 J/s per watt. Assuming a conservative power consumption rate of 0.1 J/s per watt, and assuming that the 10.35 million FLOPs are executed on a desktop CPU at full load, the amount of energy used for these operations would be around 5.175 J[c].

[b]A computer, besides the general terminology, in this context, refers to any sort of mechanism that computes. So, since an algorithm involves carrying out computations, it is a computer.

[c]To put this into perspective, the average energy consumption of a 60-watt light bulb over the course of 1 hour is approximately 216,000 J (or 216 kilowatt-seconds). So, the amount of energy used for 10.35 million FLOPs is over 40,000

While this is seemingly not a very high amount of energy, we must remember that this estimate corresponds to a system with only two particles and no false positives drawn from a Poisson point process in it. This significantly reduces the number of hypotheses that the model would have to deal with. Further, comparing it with a Kalman filter based MHT method, which requires only about 512 FLOPs per component of position per particle per timestep, 10.35 million FLOPs is huge. Considering more realistic cases would require substantially more FLOPs (in the order of billions) and thus, would expend much more energy footprint. If, hence, we develop algorithms that consume large amounts of energy, we should have a very justifiable reason to do so.

4.4 The Fair Comparison Conundrum

We have been alluding many times to the fact that the fundamental ways in which Bayesian methods and attention-based methods work/learn are drastically different. From the perspective of Bayesian filtering algorithms, we look at the current state to predict the next. However, from the perspective of attention-based learning, we look at all the states together to understand the particle behaviour, to then predict how a similar particle would behave. So, a fairer way to compare both of these methods would be for them to have the same conditions and constraints as each other – what happens when the attention-based method has the same amount of information as the Bayesian method, and vice versa? This kind of analysis also helps us to test the limits of computations for both kinds of models, which would help in future questions of scalability.

4.4.1 How to include multiple previous frames for Bayesian analysis?

As we saw in the previous chapter, attention-based methods can effortlessly leverage the entire context available to make informed predictions. When comparing attention-based methods and MHT using Bayesian filtering, it is crucial to ensure a fair comparison by granting MHT access to as much context as possible. In order to present MHT in the most favourable light, we would ideally like to augment its capability by giving it access to a more extensive history of hypotheses. This approach entails providing MHT with all previous states, allowing it to consider the entire context throughout the sequences. By doing so, we aim to provide MHT with the maximum amount of information available, akin to the comprehensive context that attention-based models naturally incorporate. However, we quickly encounter a significant obstacle when attempting to equip MHT with extensive historical context – the computational complexity of MHT grows exponentially as the number of previous states increases. Each additional state introduces a new dimension of possibilities, requiring exhaustive comparisons and associations.

This can be illustrated with an example (Fig. 4.6). Staying with our previous case of only 2 particles without any false positives, using MHT to predict the next frame based on the previous frame, if we just look at the association step, to associate a prediction to its correct corresponding trajectory, we need to carry out 4 computations. Now, if we use the previous 2 frames, the association step will require 8 computations. Let us refer to the number of previous frames that we are looking at as n. We see that when $n = 1$, the number of possible associations is $2^{1+1} = 4$; when $n = 2$, the number of possible associations is $2^{2+1} = 8$. We can, hence, generalise that the number of possible associations when we consider the previous n frames is 2^{n+1}.

Let us now take a very conservative assumption that a single association step takes $1\mu\text{s} = 10^{-6}\text{s}$. Then, for $n = 29$, the time taken for only the association step would be $2^{29+1} \times 10^{-6}\text{s} \approx 12.5$ days. Similarly, for $n = 39$, the association step would take around 34.84 days, and for $n = 50$, the association step

times less than the amount of energy used by a single 60-watt light bulb in 1 hour.

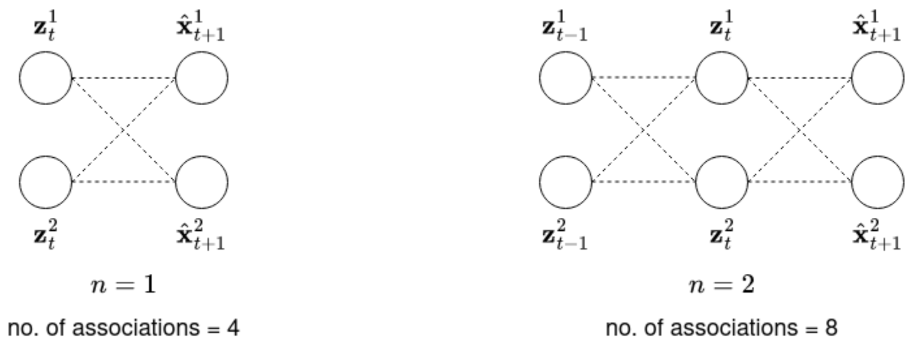

Figure 4.6: In a case where there are two trajectories and the data is false-alarm-free, just associating the correct predictions with their respective trajectories would take 2^{n+1} computations if we work with Bayesian models, where n is the number of previous frames

would take around 35.71 years! This is an unreasonably high amount of time[d], especially when in our experiments, attention-based methods are easily able to handle $n = 150$ frames. Thus, in our pursuit of a fair comparison, we must acknowledge the limitations of MHT when dealing with extensive historical context.

One way to do this is to nevertheless consider multiple hypotheses in several frames. While we establish that this is costly, we should still do it for as many frames as we deem reasonable. Stretching the capacity of MHT to match that of attention can be done using reasonable amount of frames (say, $n = 20$), minimal number of trajectories (say 2), cutting off unlikely scenarios after n frames, and parallelising the evaluation of a hypothesis. We are currently working on this approach for a fair comparison.

Another approach to use more of the past information is to break the Markovian assumption. Attention also breaks the Markovian assumption which states that the future state of a system depends only on the current state and is independent of the past states, given the current state. In a Markovian process, the history of states beyond the immediate previous state does not affect the prediction of the next state. On the other hand, attention inherently assumes that the importance of a state can be dependent on far-away contexts, be it in the past or in the future.

In this pursuit, we come up with a non-Markovian, lighter alternative to MHT which we call the moving averages-based filtering (MAF).

Consider $\mathbf{x}_t^i$ to be the state of the i^{th} trajectory, where $i \in [1, M]$ where M is the total number of trajectories, at the t^{th} time step. With MAF, using an elementary assumption, we choose the hypothesis $\boldsymbol{\eta}_{1:t}^i$. This hypothesis could or could not be optimal, which we represent by $\boldsymbol{\eta}_{1:t}^*$. By naively selecting the hypothesis, we circumvent the problem of exploding hypotheses, but it means that it would perform worse:

$$p(\mathbf{x}_t^i | \boldsymbol{\eta}_{1:t}^i, \mathbf{Z}_{1:t}) \leq p(\mathbf{x}_t^i | \boldsymbol{\eta}_{1:t}^*, \mathbf{Z}_{1:t}) \tag{4.1}$$

where equality holds if and only if $\boldsymbol{\eta}_{1:t}^i = \boldsymbol{\eta}_{1:t}^*$. Now, to correct this factor of error, we take the moving averages of the previous n frames:

[d]Transformers do not have to deal with the 2^{29} hypotheses because they smartly prune them (we discuss this later). We can do the same with MHT. We can consider hypotheses for a limited amount of time frames at the time then keep only the highest probabilities.

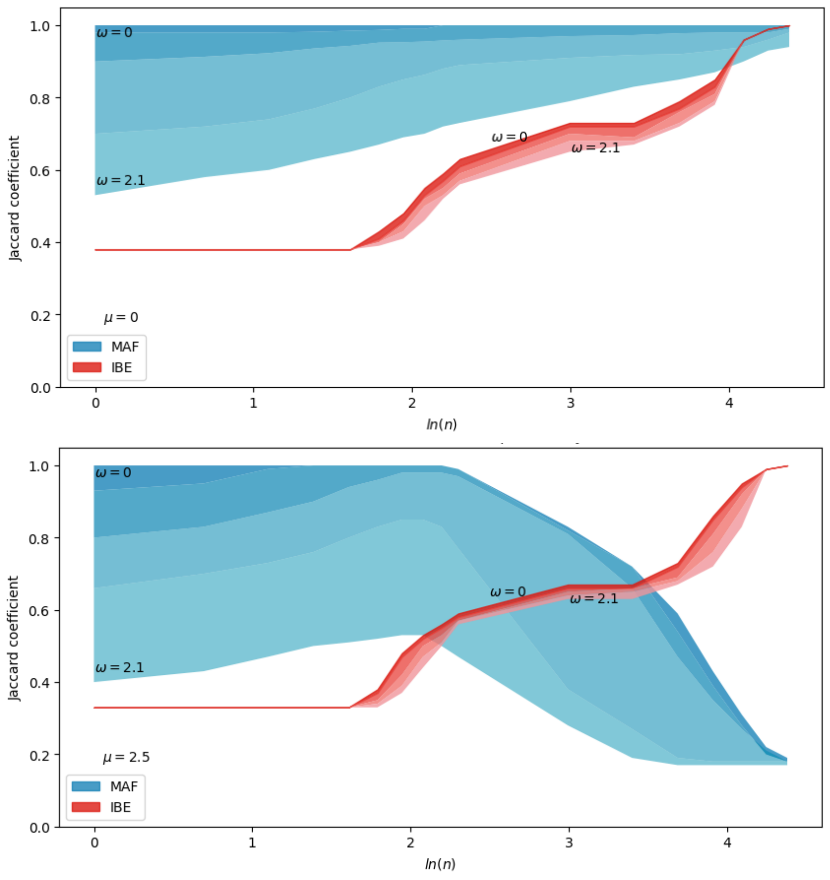

Figure 4.7: Comparison of moving averages-based filtering with IBE: we plot the JC for a false-alarm-free data of 2 particles against $ln(n)$ where n refers to the number of previous frames that are being used for prediction; top: $\mu = 0$, bottom: $\mu = 2.5$

$$\hat{\mathbf{x}}_t^i = \frac{1}{n} \sum_{\tau=t-n}^{t} \mathbf{x}_\tau^i \tag{4.2}$$

When $n = 1$, it works exactly the same as MHT as discussed in section 2.3, but as we increase $n > 1$, the expected value for the state $\mathbf{x}_t^i$ estimated is replaced by $\hat{\mathbf{x}}_i^t$ before each iteration. In doing so, it also breaks the Markovian assumption.

4.4.2 Learnt heuristics perform better in the presence of large historical information

The comparative performance is shown in Fig. 4.7. To unpack this, we have to look at many things separately. When there is no process noise (Fig. 4.7, top, $\mu = 0$), and there is no measurement noise either ($\omega = 0$), the Bayesian method (MAF, blue) is unbeatable ($JC = 1$). This has already been established before. But as the number of frames, n increases, it stays undefeated.

Staying with the condition that there is no process noise, but gradually increasing measurement noise ($\omega > 0$), we see that for lower values of n, JC for the Bayesian method reduces drastically. But as n increases, it is able to catch up to some extent and improve its performance. At this stage, we hypothesise that this “catching up” is likely because when $\mu = 0$, the probability for two particles to collide is negligible (see Fig. 4.8.a), which makes it easier for the model to predict, especially when there is more historical context available.

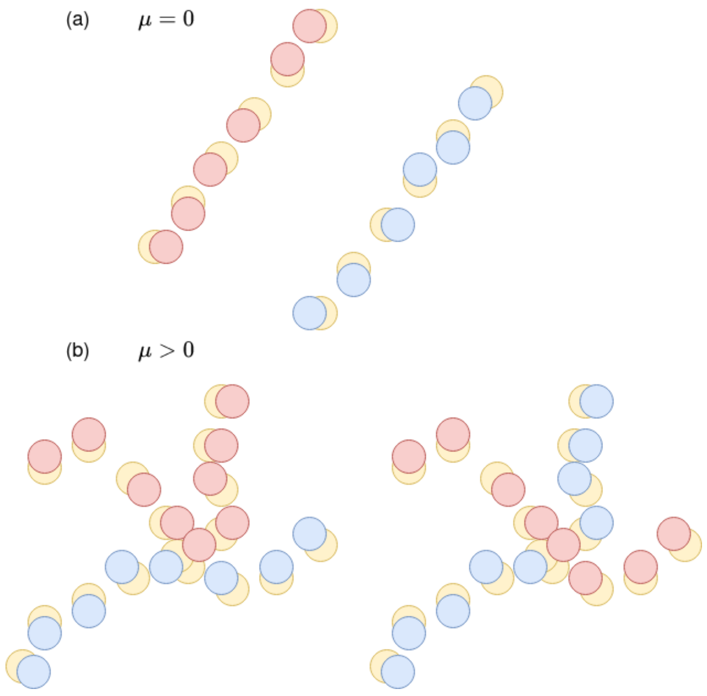

Figure 4.8: No process noise makes it less likely for two particles to collide, making the choice of hypothesis easier

On the other hand, the attention-based method (red) performs much worse when n is low – because it does not have enough data to learn from. After a certain threshold of n, it is able to better its performance. What is remarkable is that an increase in measurement noise practically does not even affect the attention-based mechanism as dramatically as it did the Bayesian method. This is because the heuristics in this case are learnt, so it is able to adapt to different levels of measurement noise.

Moving on to a more realistic scenario where there is indeed some process noise ($\mu > 0$, Fig. 4.7, bottom), even in the case where there is no measurement noise ($\omega = 0$), for large n, even the Bayesian method (blue) starts to break. As we increase the measurement noise ($\omega > 0$), it reduces the JC for Bayesian methods more dramatically than it did in the previous case. As n increases, this JC first increases, reaches a maximum and then breaks. We can list two reasons for why this is happening. (1) This is an artifact of the moving averages aspect of this computation – as we increase n, initially that helps in smoothing out the noise in the measurements and produce more accurate predictions, but setting n too high can lead to the predictions becoming too stale and not keeping up with the rapid variations in motion. (2) We also hypothesise that as we increase the process noise ($\mu > 0$), the chance of particles to cross each other increases (see Fig. 4.8.b), which would as a consequence make the association step for the Bayesian methods worse, leading to a worse JC.

The behaviour of JC for the attention-based method (red) as n increases is fairly similar to the case that we discussed above – initially with less information, it performs much worse than the Bayesian method, but as information increases, it gets better to reach a perfect JC. Now, this shows that an increase in process noise does not affect the behaviour of transformers either. What really affects their performance is data – an increase in data gives the transformer more information to work with and to learn from.

4.4.3 Decreased chance of particle collision increases precision

In the previous section, we hypothesised that one of the reasons why the JC was decreasing for MAF was that the particles were likely colliding in cases of high process noise. So, we wanted to see whether

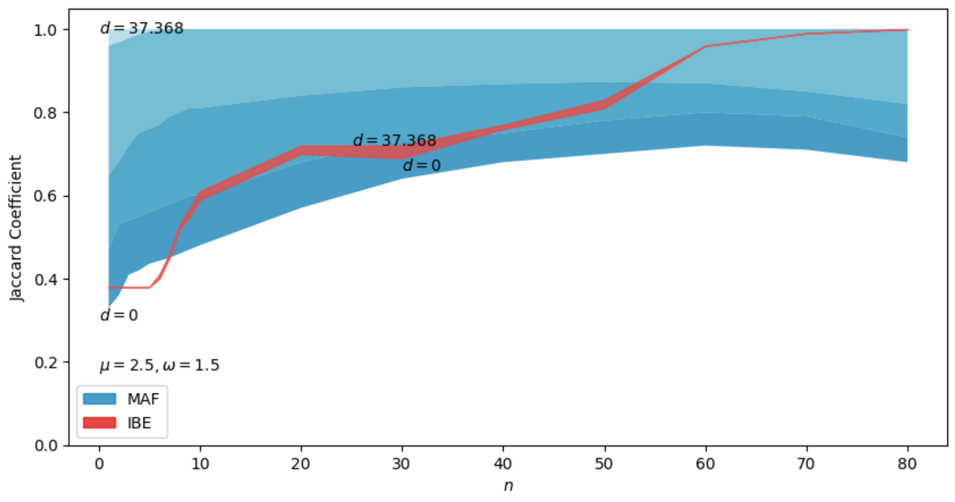

Figure 4.9: Behaviour of the JC as n increases for MAF and IBE on varying the initial distance d between the particles in a two-particle system

we would further be able to control this collision probability. In the previous experiments, the starting point for all the particles was random, as a result the initial Euclidean distance, d, between the particles was also random. In this case, we decided to carry out several experiments while controlling this initial distance d between the two particles – i.e. subsequently increasing it, while keeping noise constant. One caveat with this experiment is that no matter how far we start the particles, there is no way of controlling if they will collide or not when there is high process noise, we only decrease their chance of colliding by increasing d.

These results can be seen in Fig. 4.9. In this case, we again see a dramatic difference in JC for MAF (blue) as we increase the initial distance from $d = 0$ to $d = 37.368$. In the former case, it performs badly, albeit first increasing the score as n increases and then decreasing for large n, as explained in the previous section. On the other hand, in the latter case, since the initial distance between the particles is large enough, the score is always perfect, because there is no problem in the step where predictions are associated to the correct trajectories – the trajectories are far enough, so the correct prediction gets associated to the corresponding trajectory, leading to a perfect JC.

Not unlike the experiments in the previous section, a very different behaviour is observed for how the JC changes as we increase n for IBE (red) – the explanation of this behaviour is the same as the one in the previous section.

So, from these experiments we can conclude that while process and measurement noise can dramatically affect the performance of Bayesian methods, that is a non-issue for attention-based methods. The issue for attention-based methods is lack of information, where Bayesian methods perform much better in any case. We, hence, have a clear idea of the regimes in which both the methods succeed and fail.

4.5 Hypothesis Pruning with Attention

We see in Fig. 4.10 that as we increase n, in general, the time taken for the attention-based method to learn[e] decreases. From an auto-regressive perspective, this does not make sense. In an auto-regressive model, the prediction at each time step depends on the previous time steps. As the auto-regressive window (n) increases, the model would need to consider a longer history of previous states to make

[e]We define that a model has learnt by using the checkpoint $\mathcal{L}(e) - \mathcal{L}(e-1) = \lim_{\epsilon \to 0} \epsilon$ where $\mathcal{L}(e)$ refers to the loss function at epoch e.

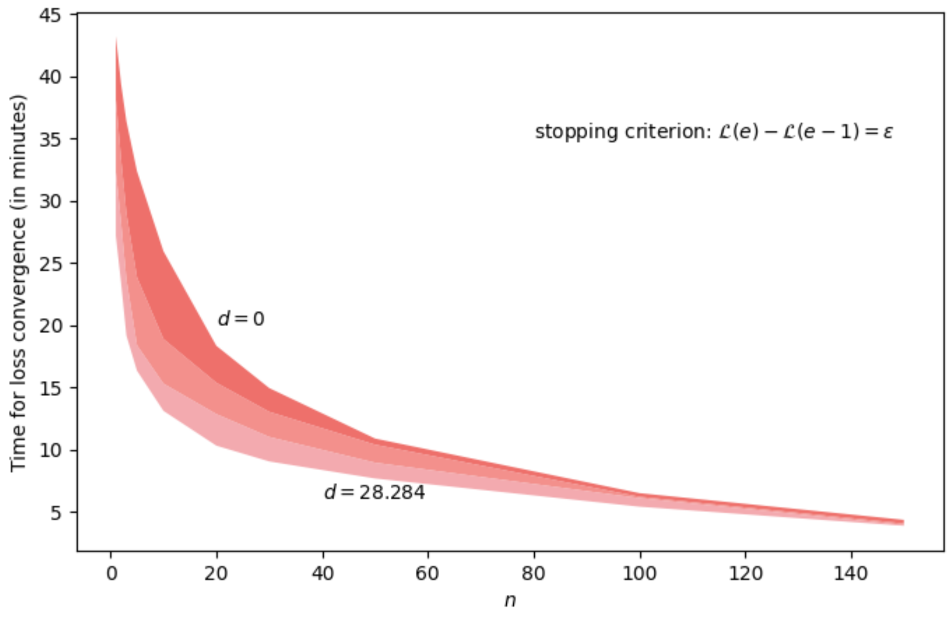

Figure 4.10: Time taken for IBE to learn against the number of frames n used to predict the next state, while varying the initial distance d between the two particles in a two-particle system

accurate predictions. We would expect that this increased dependency on past states would lead to increased computational complexity and training time.

However, recall from Ch. 3 that IBE is an encoder-only model. In an encoder-only transformer model, where the inputs are fixed-length sequences, inherently there cannot be an auto-regressive behaviour. Encoder-only transformer models are highly parallelisable by design – the self-attention mechanism allows each frame within each sequence to be processed independently, as it attends to all other frames in parallel. Of course, when there aren't enough frames (small n), it takes longer to develop this context understanding. By using larger sequences, we also take advantage of batching more effectively. When training with batches of sequences, the model can process multiple sequences simultaneously, utilising the parallel processing power of the GPU. This parallel processing enables more efficient training, as computations can be performed simultaneously across a sequence, and simultaneously between sequences. With larger sequences, the efficiency of parallel computation can actually offset the increased computational cost due to longer sequences, leading to shorter training times.

We also see in Fig. 4.10 that this decrease in training time is faster as the starting distance, d increases. This is to say that if we reduce the chance of the two particles colliding, then it is easier for the attention-based model to learn their underlying interactions. As the particles move further apart, the model can discern clearer trajectories and interactions, enabling more effective hypothesis pruning and reducing the computational burden of training. The observation that the training time decreases as the initial distance between the two particles increases suggests an intriguing phenomenon in attention-based models. It indicates that the model requires more time to learn and understand the intricate interactions between the particles when they are initially close to each other. This insight highlights the potential of attention mechanisms in effectively selecting the right hypotheses during multiple hypothesis tracking. By taking the time to comprehend the complex dynamics of nearby particles, attention enables the model to prune erroneous hypotheses and focus on the most relevant and accurate predictions. This ability to learn and capture subtle interactions between particles not only enhances the tracking performance but also showcases the power of attention in optimizing hypothesis selection for challenging tracking scenarios.

4.6 Conclusion

One of the key advantages of attention-based methods is their ability to handle large-scale and complex problems. Traditional Bayesian filtering approaches often struggle with computational efficiency when dealing with high-dimensional data and large observation spaces. In contrast, attention-based methods can scale efficiently to handle such challenges, thanks to their parallelisable nature and the ability to selectively attend to relevant parts of the input sequence. Furthermore, attention-based methods have the potential to achieve state-of-the-art performance on a wide range of tasks. Their ability to learn complex patterns and dependencies in the data allows them to capture intricate relationships that may not be easily captured by traditional filtering approaches. By utilising attention mechanisms, these methods can selectively focus on important features or regions of the input, effectively reducing noise and enhancing the discriminative power of the model.

It is important to note that attention-based methods are not intended to replace Bayesian filtering approaches entirely. Both approaches have their strengths and limitations, and the choice between them depends on the specific problem and context. However, the emergence of attention-based methods has opened up new possibilities and, as we saw throughout this chapter, has expanded our toolkit. Their ability to handle large-scale problems efficiently, achieve state-of-the-art performance, and provide interpretability makes them a crucial component in the field of machine learning and artificial intelligence.

Chapter 5

Future Scope

In this chapter, we explore the future scope of our strategies discussed throughout the text, aiming to enhance the capabilities of tracking multiple particles in dense and cluttered environments. Building upon the foundations of existing techniques, we delve into several key aspects that hold immense potential for advancing the state-of-the-art in particle tracking. By incorporating innovative approaches, including attention-based methods and the integration of normal multiple hypothesis tracking with attention, we aim to address the challenges associated with tracking in complex scenarios. This chapter presents a comprehensive overview of these aspects, discussing their significance, potential benefits, and outlining the methodologies that will be employed.

5.1 Scalability of Pure Attention

A future scope of this project involves the development of an advanced tracking system for multiple particles within a highly dense and cluttered environment. To achieve this, an attention-based encoder-only transformer model will be employed, leveraging the power of self-attention mechanisms to capture the spatio-temporal relationships between particles. The key innovation lies in the introduction of a learnable bucketing step (see Fig. 5.1), which plays a crucial role in transforming the high-dimensional input into a lower-dimensional latent representation. This latent representation allows for efficient attention computations, reducing computational complexity and memory requirements.

The learnable bucketing step is not a trivial task, as it requires a thoughtful design and training strategy. It involves partitioning the input space into several spatio-temporal buckets, each representing a specific region of interest within the environment. These buckets serve as local receptive fields, capturing the interactions and dynamics within their respective regions. By making this bucketing step learnable, the model can adaptively allocate particles to the most appropriate buckets based on their spatial and temporal characteristics. This adaptability is crucial for handling dynamic environments where the density and clutter of particles may vary over time.

Once the input has been bucketed into lower-dimensional spatio-temporal buckets, attention mechanisms are applied to enable the model to selectively focus on relevant interactions between particles within each bucket. By attending to the most informative regions, the model can extract rich contextual information and capture complex dependencies between particles, even in challenging cluttered environments. This attention-based approach enhances the model's ability to discriminate between different particle trajectories and track them accurately over time.

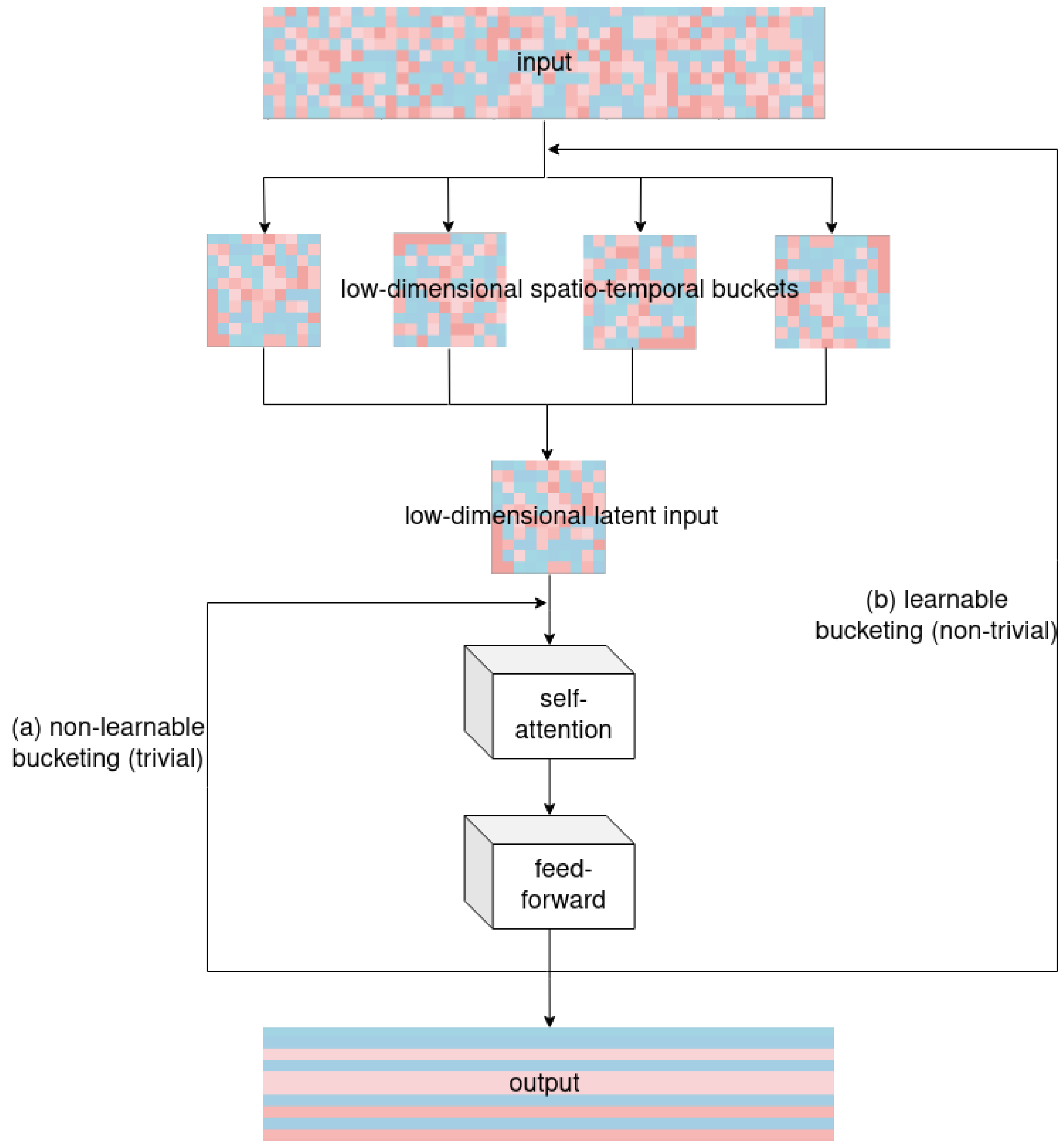
input
low-dimensional spatio-temporal buckets
low-dimensional latent input
(b) learnable bucketing (non-trivial)
(a) non-learnable bucketing (trivial)
self-attention
feed-forward
output

Figure 5.1: Reducing the dimensionality of the input into smaller buckets to have a lighter attention; (a) making the workflow learnable from the attention step is easy but it would require human intervention for bucketing, (b) ideally, we would want the workflow to learn how to bucket so that it can decide for itself which spatio-temporal aspects to give more priority to, but this is non-trivial

5.2 Bayesian Tracking with Attention

Another crucial aspect of the future scope of this project involves the integration of normal multiple hypothesis tracking with attention-based methods to enhance particle tracking performance. While Bayesian methods have proven to be effective for filtering particles and generating multiple hypotheses, the challenges lie in the association and hypothesis pruning stages, which, as we have discussed earlier, are computationally demanding due to the NP-hard nature of the problem. Here, attention mechanisms can play a pivotal role in improving the efficiency and accuracy of these stages.

By incorporating attention-based methods into the association and hypothesis pruning steps, the model can selectively focus on the most relevant particles and their interactions, reducing the search space and improving the computational efficiency. We saw in the previous chapter that attention allows the model to weigh the importance of different hypotheses and dynamically assign attention scores to potential associations between particles, based on their spatio-temporal relationships and contextual information. We saw that attention could effectively capture long-range dependencies and complex interactions between particles, even in dense and cluttered environments. By attending to informative regions and considering both local and global contexts, the model can make more informed decisions regarding association and hypothesis pruning, improving the overall tracking accuracy.

By leveraging attention to guide the decision-making process, the model can effectively handle the NP-hard aspects of the problem, reducing computational complexity and improving tracking accuracy. This hybrid approach combines the strengths of both techniques and has the potential to significantly advance particle tracking capabilities.

5.3 Pretraining

Pretraining [33], a common practice in deep learning, involves training a model on a large-scale dataset to learn generic features or representations before fine-tuning it on a task-specific dataset. By leveraging pretraining, we can leverage the vast amount of data available to learn rich representations that capture essential spatio-temporal patterns and dynamics.

In the context of particle tracking, pretraining can be particularly beneficial for several reasons. The availability of large-scale unlabelled or weakly labelled datasets allows the model to learn generalisable features that are applicable across different tracking scenarios. By capturing high-level patterns and concepts, the pretrained model can provide a strong foundation for subsequent fine-tuning and adaptability to specific tracking tasks. Pretraining can, hence, alleviate the need for an excessive amount of labelled training data, which is often expensive and time-consuming to obtain. By utilising unsupervised or weakly supervised pretraining approaches, the model can learn from readily available data without relying on costly annotations. This reduces the annotation burden and makes the tracking system more scalable and accessible.

Moreover, pretraining can aid in addressing challenges related to the high dimensionality and variability of particle tracking data. By learning low-dimensional latent representations through pretraining, the model can effectively capture salient features and reduce noise and redundancy in the input data. This dimensionality reduction facilitates more efficient computations and improves the tracking system's robustness and generalisation ability.

Furthermore, pretraining enables the transferability of learnt representations across related tasks or domains. By training on a diverse set of data that spans different environments and particle types, the model can acquire knowledge that can be effectively transferred to new tracking scenarios. This transfer learning capability reduces the need for extensive retraining or fine-tuning when deploying the system in novel environments, saving time and computational resources.

5.4 Conclusion

The attention-based methods discussed in this text offer a powerful framework for capturing spatio-temporal relationships and extracting contextual information, enhancing tracking accuracy and efficiency. By leveraging self-attention mechanisms and incorporating a learnable bucketing step, the model can adaptively allocate particles to lower-dimensional spatio-temporal buckets, reducing computational complexity and improving tracking performance in dynamic environments.

The integration of normal multiple hypothesis tracking with attention addresses the association and hypothesis pruning challenges. By selectively focusing on relevant particles and their interactions, attention-based methods reduce the search space, improving computational efficiency and tracking accuracy. This hybrid approach combines the strengths of both techniques, facilitating more informed decisions and robust tracking in dense and cluttered environments.

The exploration of pretraining offers the potential to learn generic representations from large-scale unlabelled or weakly labelled data. By capturing essential spatio-temporal patterns and reducing dimensionality, pretraining enhances generalisation, scalability, and robustness. Leveraging transfer learning, pretrained models can be effectively adapted to new tracking scenarios, reducing the need for extensive retraining and fine-tuning.

The future scope of this project, however, is not without challenges. Careful architectural design, loss function selection, and training strategies are essential for successful integration and realisation of these advancements. Additionally, limitations and potential trade-offs must be considered, such as computational resources, dataset availability, and the need for domain-specific fine-tuning.

Bibliography

[1] B. Biermann, S. Sokoll, J. Klueva, M. Missler, J. Wiegert, J.-B. Sibarita, and M. Heine, "Imaging of molecular surface dynamics in brain slices using single-particle tracking," *Nature communications*, vol. 5, no. 1, p. 3024, 2014.

[2] Q. Wang, H. He, Q. Zhang, Z. Feng, J. Li, X. Chen, L. Liu, X. Wang, B. Ge, D. Yu, *et al.*, "Deep-learning-assisted single-molecule tracking on a live cell membrane," *Analytical Chemistry*, vol. 93, no. 25, pp. 8810–8816, 2021.

[3] C. Manzo and M. F. Garcia-Parajo, "A review of progress in single particle tracking: from methods to biophysical insights," *Reports on progress in physics*, vol. 78, no. 12, p. 124601, 2015.

[4] A. F. David, P. Roudot, W. R. Legant, E. Betzig, G. Danuser, and D. W. Gerlich, "Augmin accumulation on long-lived microtubules drives amplification and kinetochore-directed growth," *Journal of Cell Biology*, vol. 218, no. 7, pp. 2150–2168, 2019.

[5] M. L. Azoitei, J. Noh, D. J. Marston, P. Roudot, C. B. Marshall, T. A. Daugird, S. L. Lisanza, M.-J. Sandí, M. Ikura, J. Sondek, *et al.*, "Spatiotemporal dynamics of gef-h1 activation controlled by microtubule-and src-mediated pathways," *Journal of Cell Biology*, vol. 218, no. 9, pp. 3077–3097, 2019.

[6] A. Gennerich and D. Schild, "Finite-particle tracking reveals submicroscopic-size changes of mitochondria during transport in mitral cell dendrites," *Physical biology*, vol. 3, no. 1, p. 45, 2006.

[7] S. L. Mironov, "Adp regulates movements of mitochondria in neurons," *Biophysical journal*, vol. 92, no. 8, pp. 2944–2952, 2007.

[8] R.-L. Zhang, F. W. Pratiwi, B.-C. Chen, P. Chen, S.-H. Wu, and C.-Y. Mou, "Simultaneous single-particle tracking and dynamic ph sensing reveal lysosome-targetable mesoporous silica nanoparticle pathways," *ACS applied materials & interfaces*, 2020.

[9] L. C. Kapitein, M. A. Schlager, W. A. Van der Zwan, P. S. Wulf, N. Keijzer, and C. C. Hoogenraad, "Probing intracellular motor protein activity using an inducible cargo trafficking assay," *Biophysical journal*, vol. 99, no. 7, pp. 2143–2152, 2010.

[10] M. T. Gallagher, T. D. Montenegro-Johnson, and D. J. Smith, "Simulations of particle tracking in the oligociliated mouse node and implications for left–right symmetry-breaking mechanics," *Philosophical Transactions of the Royal Society B*, vol. 375, no. 1792, p. 20190161, 2020.

[11] R. Scheda, S. Vitali, E. Giampieri, G. Pagnini, and I. Zironi, "Study of wound healing dynamics by single pseudo-particle tracking in phase contrast images acquired in time-lapse," *Entropy*, vol. 23, no. 3, p. 284, 2021.

[12] Y. Li, J. Schnekenburger, and M. H. Duits, "Intracellular particle tracking as a tool for tumor cell characterization," *Journal of biomedical optics*, vol. 14, no. 6, pp. 064005–064005, 2009.

[13] E. Meijering, O. Dzyubachyk, and I. Smal, “Methods for cell and particle tracking,” *Methods in enzymology*, vol. 504, pp. 183–200, 2012.

[14] P. Vallotton, G. Danuser, S. Bohnet, J.-J. Meister, and A. B. Verkhovsky, “Tracking retrograde flow in keratocytes: news from the front,” *Molecular biology of the cell*, vol. 16, no. 3, pp. 1223–1231, 2005.

[15] H. M. Van Der Schaar, M. J. Rust, C. Chen, H. van der Ende-Metselaar, J. Wilschut, X. Zhuang, and J. M. Smit, “Dissecting the cell entry pathway of dengue virus by single-particle tracking in living cells,” *PLoS pathogens*, vol. 4, no. 12, p. e1000244, 2008.

[16] C. Vonesch, F. Aguet, J.-L. Vonesch, and M. Unser, “The colored revolution of bioimaging,” *IEEE signal processing magazine*, vol. 23, no. 3, pp. 20–31, 2006.

[17] S. Biedzinski, G. Agsu, B. Vianay, M. Delord, L. Blanchoin, J. Larghero, L. Faivre, M. Théry, and S. Brunet, “Microtubules control nuclear shape and gene expression during early stages of hematopoietic differentiation,” *The EMBO Journal*, vol. 39, no. 23, p. e103957, 2020.

[18] G. Camargo Ortega, S. Falk, P. A. Johansson, E. Peyre, L. Broix, S. K. Sahu, W. Hirst, T. Schlichthaerle, C. De Juan Romero, K. Draganova, *et al.*, “The centrosome protein akna regulates neurogenesis via microtubule organization,” *Nature*, vol. 567, no. 7746, pp. 113–117, 2019.

[19] R. Kumar, G. Charpiat, and M. Thonnat, “Multiple object tracking by efficient graph partitioning,” in *Computer Vision–ACCV 2014: 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part IV 12*, pp. 445–460, Springer, 2015.

[20] A. Roshan Zamir, A. Dehghan, and M. Shah, “Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs,” in *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part II 12*, pp. 343–356, Springer, 2012.

[21] P. Emami, P. M. Pardalos, L. Elefteriadou, and S. Ranka, “Machine learning methods for data association in multi-object tracking,” *ACM Computing Surveys (CSUR)*, vol. 53, no. 4, pp. 1–34, 2020.

[22] R. Spilger, A. Imle, J.-Y. Lee, B. Mueller, O. T. Fackler, R. Bartenschlager, and K. Rohr, “A recurrent neural network for particle tracking in microscopy images using future information, track hypotheses, and multiple detections,” *IEEE Transactions on Image Processing*, vol. 29, pp. 3681–3694, 2020.

[23] R. Spilger, J.-Y. Lee, V. O. Chagin, L. Schermelleh, M. C. Cardoso, R. Bartenschlager, and K. Rohr, “Deep probabilistic tracking of particles in fluorescence microscopy images,” *Medical image analysis*, vol. 72, p. 102128, 2021.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.

[25] N. B. Toomarian and J. Barhen, “Learning a trajectory using adjoint functions and teacher forcing,” *Neural networks*, vol. 5, no. 3, pp. 473–484, 1992.

[26] L. N. Smith, “Cyclical learning rates for training neural networks,” in *2017 IEEE winter conference on applications of computer vision (WACV)*, pp. 464–472, IEEE, 2017.

[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[28] S.-H. Bae, J. Park, and K.-J. Yoon, “Joint estimation of multi-target signal-to-noise ratio and dynamic states in cluttered environment,” *IET Radar, Sonar & Navigation*, vol. 11, no. 3, pp. 539–549, 2017.

[29] Á. F. García-Fernández, J. L. Williams, K. Granström, and L. Svensson, “Poisson multi-bernoulli mixture filter: Direct derivation and implementation,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 4, pp. 1883–1901, 2018.

[30] A. S. Rahmathullah, Á. F. García-Fernández, and L. Svensson, “Generalized optimal sub-pattern assignment metric,” in *2017 20th International Conference on Information Fusion (Fusion)*, pp. 1–8, IEEE, 2017.

[31] P. Mishra, P. Mohapatra, T. K. Patra, and P. Subham, “Disease diagnosis in grapevines–a hybrid resnet-jaya approach,” in *International Advanced Computing Conference*, pp. 39–56, Springer, 2021.

[32] L. Lannelongue, J. Grealey, and M. Inouye, “Green algorithms: quantifying the carbon footprint of computation,” *Advanced science*, vol. 8, no. 12, p. 2100707, 2021.

[33] B. Zoph, G. Ghiasi, T.-Y. Lin, Y. Cui, H. Liu, E. D. Cubuk, and Q. Le, “Rethinking pre-training and self-training,” *Advances in neural information processing systems*, vol. 33, pp. 3833–3845, 2020.