universität wien

DISSERTATION / DOCTORAL THESIS

Titel der Dissertation / Title of the Doctoral Thesis

„Causally Consistent Abstractions of Time-Series Data“

verfasst von / submitted by

Akshey Kumar

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Doktor der technischen Wissenschaften (Dr. techn.)

Wien, 2023 / Vienna, 2023

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	UA 786 880
Dissertationsgebiet lt. Studienblatt / field of study as it appears on the student record sheet:	Informatik
Betreut von / Supervisor:	Univ.-Prof. Dipl.-Ing. Dr. -Ing. Moritz Grosse-Wentrup
Betreut von / Supervisor:	Ass.-Prof. Dipl.-Ing. Dr.techn.Sebastian Tschiatschek, BSc

Abstract

Understanding complex dynamical systems, particularly in the realm of neuroscience, poses significant challenges due to the high dimensionality and intricacy of available data. While acknowledging the significance of establishing causal relationships, this dissertation contends that a mere causal understanding may not suffice since it may be too complex to be interpretable. Hence, we identify the need for causally consistent abstractions. To address this, we present a mathematical framework outlining key assumptions that facilitate the derivation of causally-consistent high-level models directly from observational data. We then introduce BunDLe-Net – an architecture to learn high-level models directly from neuronal and behavioural time-series data. The efficacy of our architecture is demonstrated across various modalities of neuroscience data, which consistently produces interpretable insights that not only align with existing knowledge but also reveal novel insights about the data. Additionally, this thesis introduces a toolbox for the implementation of BunDLe-Net. Finally, we discuss future avenues for research that our work opens up in a variety of scientific domains, such as causality, data science and neuroscience.

Kurzfassung

Das Verständnis komplexer dynamischer Systeme, insbesondere im Bereich der Neurowissenschaften, stellt aufgrund der hohen Dimensionalität und Komplexität der verfügbaren Daten eine große Herausforderung dar. In dieser Dissertation wird zwar die Bedeutung des Nachweises kausaler Beziehungen anerkannt, doch reicht ein rein kausales Verständnis möglicherweise nicht aus, da es zu komplex ist, um interpretiert werden zu können. Daher sehen wir einen Bedarf an kausal konsistenten Abstraktionen. Um dies zu erreichen, stellen wir einen mathematischen Rahmen vor, der die Ableitung von kausal konsistenten Modellen auf hoher Ebene direkt aus Beobachtungsdaten ermöglicht. Anschließend stellen wir BunDLe-Net vor - einen Algorithmus zum Lernen von High-Level-Modellen direkt aus neuronalen und Verhaltensdaten. Die Wirksamkeit unserer Architektur wird anhand verschiedener Modalitäten neurowissenschaftlicher Daten demonstriert, die durchweg interpretierbare Erkenntnisse liefern, die nicht nur mit dem vorhandenen Wissen übereinstimmen, sondern auch neue Erkenntnisse über die Daten offenbaren. Darüber hinaus wird in dieser Arbeit eine Toolbox für die Implementierung von BunDLe-Net vorgestellt. Schließlich diskutieren wir zukünftige Forschungsmöglichkeiten, die unsere Arbeit in einer Vielzahl von wissenschaftlichen Bereichen wie Kausalität, Datenwissenschaft und Neurowissenschaften eröffnet.

Contents

List of Figures

1. Preamble

In recent times, we have been confronted with increasingly complex systems, such as the brain (Shepherd et al., 1998; Akil et al., 2011), weather phenomena (Runge et al., 2019; Chalupka et al., 2016a), and high-dimensional physical systems (Chen et al., 2022; Raissi and Karniadakis, 2018). These systems, while diverse, share common characteristics such as high dimensionality, stochastic dynamics and intricate causal interactions between their constituent components and the surrounding environment. These traits render these systems challenging to understand and explain in terms of their behaviours and outcomes. Additionally, the complexity is compounded by the limited knowledge we often have about the system's state and its interactions with the external environment. The aim of this dissertation, in its broadest terms, is to explore how one may gain an understanding of complex dynamical systems.

We are particularly interested in understanding brains, where a vast amount of data has become available in recent years (Sejnowski et al., 2014). With the development of various data acquisition techniques like calcium imaging and electrophysiology, it is now possible to collect data at the level of individual neurons, which can be considered the fundamental building blocks of the brain (Wei et al., 2020; De Vico Fallani et al., 2015). This presents the potential for entirely new avenues of insight into the brain's functioning, specifically how individual neurons collaborate to process information and generate behaviour. Furthermore, a growing number of whole-brain recordings are becoming accessible (Migault et al., 2018; Kato et al., 2015; Nguyen et al., 2015; Ahrens et al., 2013). This means we are no longer limited, in terms of data, by partial observability when it comes to describing how information is processed. Despite these advancements in data quality and the abundance of data, the neuroscience community still faces challenges in interpreting data and deriving meaningful and practical insights from it (Jonas and Kording, 2017; Krakauer et al., 2017).

1.1. Outline

We begin this dissertation by formulating the research questions based on current academic knowledge. We introduce our high-level conceptual framework in Section 1.2. After a review of relevant literature in Section 1.3, we formally state our scientific research questions in Section 1.4. We culminate this Chapter by presenting a list of my publications during my doctoral studies that contribute to the theme of this dissertation in Section 1.5.

Chapter 2 presents our theoretical formalism, defining our central notions of dynamical causal consistency and behavioural consistency and outlining the assumptions enabling

the inference of causal abstractions from observational data. To strengthen intuition of these concepts, an example of an analytical solution is provided in Section 2.4. Building on our theoretical formalism, we introduce our novel algorithm BunDLe-Net in Chapter 3, where we use it for Neuronal Manifold Learning. Chapter 4 presents a Python toolbox for BunDLe-Net, emphasising its functionalities for abstracting and visualising complex neuronal and behavioural datasets. Chapter 5 showcases the adaptability and robustness of BunDLe-Net through two variants and experiments on diverse datasets and settings. Concluding this dissertation, Chapter 6 delves into our results and contributions, re-addressing the initial research questions (Section 6.1), highlighting contributions to various fields (Section 6.2), and discussing new research avenues opened by our work (Section 6.3).

1.2. Conceptual Framework

Here, we dissect what it means to *understand a complex dynamical system* and outline its fundamental elements. Understanding a system implies having the capability to reasonably predict its outcomes or dynamics. However, a comprehensive understanding extends beyond mere predictability; it encompasses a causal knowledge of the system's interactions, i.e. it involves understanding how various components causally influence each other to generate its dynamics in time. The strength of a causal understanding lies in its capacity to predict the consequences of alterations to the system (Schölkopf et al., 2021). Moreover, a causal understanding becomes indispensable when we seek to engineer specific behaviours or outcomes within the system.

In this thesis, we adopt the formal language of structural causal models (SCMs), as introduced by Pearl. Causality theory is a framework that describes the mathematics of cause and effect. Conventional statistical models describe associations between a set of variables and are used in machine learning to make predictions. A causal model goes beyond this and describes the system in terms of the causal relationships between its variables. Structural causal models (SCMs), in particular, offer a mathematical framework for modelling a system's variables in terms of their causal interactions. It involves the notion of a directed acyclic graph (DAG) where each of the system's variables is represented by a node, and a directed edge represents a causal interaction between a given pair of variables. In this way, we can reason about what would happen if we made changes, such as removing some variables, severing certain causal links, or intervening to fix the values of some variables (Spirtes et al., 2001).

While the causality theory and the SCM framework are powerful tools, they were primarily developed to deal with independently and identically distributed (iid) data. They do not account for the temporal structure of the data or investigate how causality unfolds in the dynamics of a system. In contrast, the field of physics employs a distinct approach known as dynamical systems, where the time evolution of a system's variables is captured using differential equations (Strogatz, 2019). These equations determine the trajectory of the high-dimensional system in its state space. Unlike the causality framework, the dynamical systems approach lacks the concept of cause and effect and relies

on fundamental physical variables. Knowledge of these variables enables the complete determination of a system's past and future trajectories through differential equations that can be derived from physical laws.

To establish a comprehensive notion of understanding a complex dynamical system, such as the brain, we recognise the need for a framework that combines the robust causal semantics of SCMs with the time-trajectory perspective from the dynamical systems approach. In this thesis, we endeavour to construct a mathematical framework that leverages the strengths of both domains to enhance our understanding of complex dynamical systems.

Thus far, we have discussed what constitutes an understanding of a dynamical system. However, some systems are so complex that even knowledge of its complete SCM or system of differential equations may not greatly further our understanding (Hoel et al., 2016; Hoel, 2017). While causality is a powerful tool in explaining systems with a small to moderate number of variables, more is needed to gain a comprehensive picture of a system with a vast number of variables like the brain. Since causal models at such fine-grained levels can be challenging to interpret, we require simplified models that are not only adept at explaining the phenomena of interest but are also human-interpretable. Consequently, there arises a need to abstract useful concepts either from the complex model or directly from data generated by the system.

To illustrate this, consider the field of neuroscience. The brain can be studied at varying levels of granularity, right from individual neurons to brain regions to cognitive concepts, each serving a specific purpose (Grosse-Wentrup et al., 2023; Haueis, 2022). For instance, causation can be discussed at the individual neuronal level, where we may reason, '*Neuron A causes neuron B to fire*'. However, a psychologist would be less concerned with the firing patterns in neuronal circuits and more focused on the cognitive states of their patient, which determine behaviour and responses to stimuli. These cognitive states serve as abstractions of brain neuronal activity. While they lack the fine-grained detail of the neuronal level, they are of immense value for explaining high-level phenomena. Though these models are of great practical use, they have a limitation in that they lack causal consistency and are not derived from the neuronal level.

To address this, we look to the field of machine learning and data science, where dimensionality reduction techniques have been extensively employed to simplify vast datasets (Lee et al., 2007). Reduced dimensionality allows for more straightforward visualisation, which may, in turn, lead to valuable insights about the system. It is important to note, however, that the majority of existing methods compress data into fewer dimensions in an unsupervised setting, thus potentially discarding relevant information.

Having introduced the problem and identified domains that may offer solutions, we aim to establish theoretical principles that guide us in obtaining meaningful representations tailored to the neuroscience setting. These representations should align with causality, ensuring that they are consistent with the underlying data. In doing so, we strike a balance between simplification for explainability and preserving the integrity of the complex system's causal relationships.

1.3. Background

This review presents an overview of the current state of knowledge in key domains relevant to this thesis. It assumes a basic understanding of causality, in particular, the Structural Causal Model framework (also known as Structural Equation Model). A concise explanation of causality is included, tailored to the requirements of this thesis.

1.3.1. Causality

Causality is a fundamental concept that plays a pivotal role in various domains, including artificial intelligence science (Schölkopf, 2022), neuroscience (Chen et al., 2023; Banerjee et al., 2023), weather (Runge et al., 2019). In essence, it addresses the fundamental question of cause and effect – understanding how one event or variable influences another. Causality provides a powerful lens through which to dissect complex systems and extract meaningful insights. A common challenge in data analysis is distinguishing causality from mere correlation. While correlation implies a statistical association between two variables, causality delves into the mechanisms behind this association. Consider an example where ice cream sales and drowning incidents are correlated, both peaking during the summer. Without causal analysis, one might mistakenly infer that ice cream consumption causes drownings. In reality, both are independently influenced by high temperatures. This example illustrates the dangers of assuming causality solely from correlation (Pearl, 2009).

Causal analysis surpasses purely statistical approaches by offering a deeper understanding of the relationships within a system. Statistical correlations can provide valuable insights, but they often lack the explanatory power needed to make informed decisions or interventions. Causal models, on the other hand, allow us to discern not just that events are correlated but also why they are related, facilitating informed decision-making (Spirtes et al., 2001).

Structural causal models

While several definitions and frameworks of causality exist, we choose to work with the Structural Causal Models framework (Pearl, 2009). For this thesis, we pre-suppose familiarity with basic causality see references. Nevertheless, we still review some basic terms and concepts that are essential for the thesis with the help of an example. This is not meant to be a rigorous but rather an intuitive presentation of the topic. For a more extensive exploration, readers are directed to (Pearl et al., 2016).

Structural causal models are a framework for causality which uses a system of equations to describe causal relations between a given set of variables. Consider a set of variables $X_1, .., X_6$ depicted as nodes in the graph (Figure 1.1). These variables can be real-valued, vector-valued, discrete, or categorical. The edges in the graph depict causal relationships between the variables. For instance, an edge from X_1 to X_3 indicates that X_1 directly influences X_3. We refer to these direct influences as *parents*, denoted as Pa(.). A variable can have zero to multiple parents.

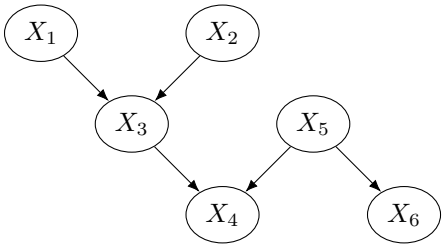

Figure 1.1.: A Directed Acyclic Graph describing the causal structure between variables $X_1, ..., X_6$

Figure 1.1 is a directed acyclic graph (DAG) and depicts the causal structure between the variables in our set. The graph is acyclic because one cannot have cycles in the causal graph. This would lead to a variable being a cause of itself which is not permitted in this framework[1]. In our example, X_1, X_2, and X_5 are *exogenous variables* because they have no parents within this set of variables. Variables with parents within the set, like X_3, X_4 and X_6 are referred to as *endogenous variables*.

A structural causal model (SCM) on the variables $X_1, ..., X_6$ is a set of equations, one for each variable, of the form, $X_i = f(\text{Pa}(X_i), \epsilon_i)$, where $\text{Pa}(X_i)$ denotes the parents of X_i and ϵ_i denotes a noise variable corresponding to variable X_i. It is assumed that we do not have access to these noise terms and that the noise terms are independent of one another. This gives rise to a probabilistic distribution over the variables.

Interventions An intervention involves making a deliberate alteration to one or more variables within the system. In the context of our example, we could intervene by setting X_3 to a specific value. Such an intervention effectively severs the causal links between X_3 and its parents X_1 and X_2, disrupting the normal course of causation. This means that changes in the intervened variable can no longer be attributed to its parents or their causal influence. Interventions play a crucial role in causal modelling, allowing us to explore the effects of controlled changes within a system and assess how these changes propagate through the causal network.

Causal inference at different scales

A major application of causality theory is to infer causal structure between a set of variables from a given dataset. There are a range of causal inference algorithms (Spirtes and Glymour, 1991; Verma and Pearl, 1990). These algorithms take as input the data and output causal structure(s)[2] that are compatible with the observed statistical relationships between variables.

[1] This does not preclude the possibility of a pair of variables mutually influencing each other. In such instances, one must examine their temporally unrolled graph to model their interactions — which means considering the evolution of these variables over time

[2] Sometimes it is not possible to uniquely determine the causal structure from a given dataset, but rather the algorithms would determine an equivalence class of compatible causal structures

Causal structure algorithms assume prior knowledge of *what* the causal variables are. When it comes to complex systems with high-dimensional variables, it is not straightforward how to choose the variables. To illustrate, we take an example from physics. Consider two containers, A and B, in thermal contact, each containing a gas with several molecules undergoing random thermal motion. Let's say we heat container A, increasing the average momentum of its molecules. This leads to a change in container B, with their average momenta rising. If we approach this as a causality problem based on individual variables like position and momentum, it becomes unclear which molecules in container A affect those in container B through this intervention. Even if we infer a causal structure suggesting all variables related to container A influence container B, it offers limited insights. A more effective approach is to focus on the macroscopic level. Instead of examining individual molecule momenta, we could use macrovariables like temperature. Coarse-graining might allow causal statements at the macro-level, which, in some cases, may even offer further insights into the causal mechanisms of the system (Hoel et al., 2013, 2016; Hoel, 2017). Yet, it is unclear how a complex system should be coarse-grained while preserving causal relationships. If we coarse-grain away relevant causal information, we may end up with inconsistencies (Weichwald et al., 2015), and paradoxes against the Causal Markov Condition[3] (Gebharter and Retzlaff, 2020).

Example of a causally inconsistent coarse-graining

As a contribution of this dissertation and to causal abstractions, we present here an SCM scenario of bad coarse-graining. It illustrates how coarse-graining can render variables unfit for causal inference. This example provides insights as to why one should not arbitrarily coarse-grain data or models if one wants to make causal claims. This includes dimensionality reduction techniques like PCA, autoencoders, UMAP, t-SNE, and isomap, which do not take causality into consideration.

Consider the following SCM with three variables,

$$\begin{aligned} X_1 &= \epsilon_1, && \epsilon_1 \sim U(\{1,2,...,10\}) \\ X_2 &= 2X_1 + \epsilon_2, && \epsilon_1 \sim U(\{0,1\}) \\ X_3 &= \epsilon_3 + \begin{cases} 0 & \text{if } X_1 \bmod 2 = 0 \\ 5 & \text{if } X_1 \bmod 2 = 1 \end{cases}, && \epsilon_3 \sim \mathcal{N}(0,1) \end{aligned}$$

where $U(S)$ signifies a uniform distribution over elements of the set S. Let τ be a transformation that is given by $X_i \mapsto Y_i = X_i + \lfloor X_i \rfloor \bmod 2$ for $i \in \{1,2,3\}$.

For the true model at the microscopic level, we can derive the following conditional independencies $(X_2 \perp X_3|X_1)$, $(X_1 \not\perp X_3|X_2)$, $(X_1 \not\perp X_3)$. However, after coarse-graining, Y_1 loses information about X_3, while Y_2 still holds some information about Y_3. Thus the conditional independencies become $(Y_2 \not\perp Y_3|Y_1)$, $(Y_1 \perp Y_3|Y_2)$, $(Y_1 \perp Y_3)$. This would

[3]The Causal Markov Condition is a fundamental assumption made by several causal inference algorithms. It asserts that a given variable is probabilistically independent of all its non-descendant variables when conditioned on its parents in a causal DAG.

lead to learning the wrong Markov equivalence class of DAGs (or causal skeleton) on the macroscopic level.

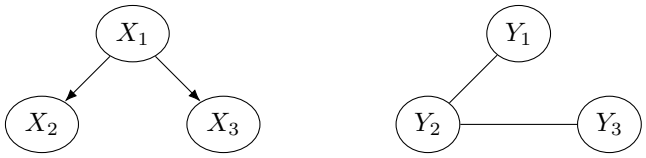

Figure 1.2.: Left: True causal structure. Right: Wrongly inferred causal skeleton after coarse-graining

1.3.2. Abstraction of causal systems

From the previous examples, we have seen that coarse-graining can throw away causal information from the system, leading to inconsistencies, paradoxes, erroneous causal inference and simply wrong models at the abstract level. Abstractions are nevertheless an essential tool for understanding complex systems. To address this, the authors of (Rubenstein et al., 2017) present a mathematical definition of causal consistency between a pair of SCMs. This definition serves to test the compatibility of a high-level SCM with a low-level SCM, determining whether the former can be regarded as a causally consistent abstraction of the latter. However, it does not prescribe a specific *recipe* for appropriately abstracting a high-level model from low-level dynamics; rather, it provides a means for testing. Later in Chapter 2, we build upon the notion of causal consistency and create a framework for abstracting SCMs from time-series data, which is later used in our Publication (Grosse-Wentrup et al., 2023) to bridge the neuronal and cognitive levels.

In a series of papers (Chalupka et al., 2017, 2015, 2016b), Chalupka et al. develop a practical method called causal feature learning (CFL). They present an algorithm to learn discrete causal macrovariables on high-dimensional cause-effect pairs. The causal feature learning algorithm succeeded in learning causal macrovariables in weather science corresponding to the El Nino and La Nina phenomena in the Pacific (Chalupka et al., 2016a). While the visual CFL framework is well-suited for independently and identically distributed (*iid*) image data, there is no straightforward way to extend it to multivariate time-series data, which is typical in neuronal studies. Moreover, the CFL framework makes the rather strict assumption that the macrovariable is discrete, which may not always be the case. Therefore, it is preferable to begin with a continuous macrovariable and observe whether a discrete structure naturally emerges. Such is the approach taken in this thesis.

1.3.3. Dynamical Systems

Dynamical systems are a mathematical framework to describe how variables evolve over time (Strogatz, 2019). It uses a set of differential equations to model the time dynamics of a complex system. It differs significantly from the Structural Causal Model (SCM)

approach, which is directed more towards the iid setting (independently and identically distributed data) and does not take time into account. The dynamical systems approach can offer interesting insights by considering the temporality of the data; for example, we can study the recurrence of certain behaviours or bifurcations in dynamics. While dynamical systems excel at capturing the temporal evolution and patterns of variables, they do not inherently address causality.

There has been theoretical research in deriving structural causal models for dynamical systems[4], thus attempting to bridge these fields that grew out of very different disciplines. The authors of (Mooij et al., 2013; Rubenstein et al., 2018; Bongers et al., 2022) provide a set of assumptions under which an SCM can be derived from a set of differential equations in an equilibrium setting. Janzing et al. derive an analytic macrovariable SCM by coarse-graining variables in a time-series with linear dynamics. While these works offer useful insights into how one can think about dynamical systems in a causal way, they all presuppose knowledge of the underlying dynamics in the form of differential equations. This makes it difficult to apply them to data obtained from experimental studies, where we have little to no prior knowledge of the system's mechanism and dynamics.

1.3.4. Representation Learning in Reinforcement Learning

Several representation learning approaches exist within the field of reinforcement learning. The authors of (Wahlström et al., 2015; Watter et al., 2015; Hafner et al., 2019; Jaques et al., 2020) use generative models to learn low-dimensional variables from pixel data for the control of non-linear dynamical systems. These algorithms use autoencoder-like components which rely on reconstruction of the microscopic state. While this may give us a lossless representation of the data, it may keep unnecessary information thus limiting the compression. In (Zhang et al., 2021), the authors use bisimulation metrics to learn invariant representations without reconstruction of the original state. Impressively, it is able to discard extraneous details and retain only information relevant to the task at hand. Contrastive learning (Kipf et al., 2020; Anand et al., 2019) is yet another method that is able to learn *structured world models* by its ability to extract objects from pixel data. While several of the above works learn representations that are useful for certain tasks such as control or dynamical modelling, few of them allow for a causal semantics. Causal representation learning (Schölkopf et al., 2021) stresses the need to learn high-level causal variables from low-level observations. Our framework takes care of this and also endows a lot of the above-mentioned work with causal interpretations if certain assumptions are fulfilled.

1.3.5. Abstractions in Cognitive Neuroscience

A central question in the field of neuroscience is how neuronal activity generates cognition and behaviour. With the emergence of sophisticated neuro-imaging techniques, there is a vast amount of data available with which we can probe this question. Despite this, bridging the neuronal and cognitive levels remains a challenge.

[4] systems in which the variables have a definite time-structure

Marr's three levels

For a comprehensive understanding of the brain as an information processing system, Marr posited a three-level scheme of understanding: the computational, algorithmic and implementational level, each at a different level of abstraction.

1. **Computational level**: This level addresses the *why question*. It focuses on the high-level function or purpose of a system, abstracting away from the specifics of implementation. At this level, we define what the system is supposed to do and what problems it should solve.

2. **Algorithmic Level**: This level answers the *how question*. It delves into the step-by-step processes and algorithms that the system uses to achieve its computational goals. Here, we break down the system's function into a series of understandable computational steps.

3. **Implementational Level**: This level deals with the *how it's done question*. It deals with details of how the system is physically or biologically realised.

To illustrate, we take the example of a pocket calculator. At the implementation level, we have transistors and diodes that operate on the principles of physical laws. At the algorithmic level, we would have the software, operating system, and algorithms that are required to perform computations. And then, at the computational level, we have the task that the calculator is supposed to accomplish – to perform arithmetic operations between numbers. As we see, this framework is not limited to brains and is generic for information processing systems.

Neuronal manifold learning

Neuronal manifold learning algorithms seek to understand the underlying low-dimensional structure within the high-dimensional neuronal activity. They operate under the assumption that the intrinsic dimensionality of neuronal data is notably lower than the actual number of neurons. Efforts have been made to establish a link between the manifold's dimension and the complexity of the task being studied (Gao et al., 2017). Several widely used algorithms for neuronal manifold learning include PCA (Kato et al., 2015; Ahrens et al., 2012), demixed PCA (Brendel et al., 2011; Kobak et al., 2016), UMAP (McInnes et al., 2018), t-sne (Van der Maaten and Hinton, 2008) autoencoders (Kingma and Welling, 2013), pi-VAE (Zhou and Wei, 2020), and CEBRA (Schneider et al., 2023). The majority of these approaches are unsupervised, focusing on data compression to enable faithful reconstruction. Subsequently, unsupervised methods seek to establish any potential correlations between neuronal activity and behaviour (Rubin et al., 2019).

While unsupervised methods may provide visualisations of the data, they often erroneously discard important information for a specific behaviour. Moreover, the brain is capable of dealing with several processes at once. Hence, an unsupervised approach may only uncover an entangled mesh of processes that is barely interpretable. A study by

Krakauer et al. emphasised that studying the brain in isolation may provide far fewer insights than when complemented with behavioural information. To use Marr's terminology, the implementational level alone might not give us a comprehensive understanding but should rather be supplemented with the computational level so as to reveal underlying algorithms.

Another study by Jonas and Kording posed the question, '*Could a neuroscientist understand a microprocessor?*', where the authors used neuroscience techniques to investigate a microprocessor. Since the algorithms and implementations in a microprocessor are already known, one would hope that neuroscience techniques could be applied to reveal them and thus validate the efficacy of these methods. Unfortunately, aside from revealing some interesting structure in the data, the techniques revealed no meaningful understanding. It notably failed to describe the hierarchy of information processing in the microprocessor. Both these studies highlight the inadequacy of current techniques to reveal insights into the workings of neuronal systems. In our work, we heed the precautions of Krakauer et al. and Jonas and Kording by explicitly considering the behaviour.

C. elegans - the model organism

The roundworm *C. elegans* is a model organism for neuroscience research due to its small nervous system of 302 neurons. All the connections between the neurons have been mapped out and published as a connectome (Varshney et al., 2011). Though we know the entire map, and thus in some sense the causal structure between the neurons, we are still far from understanding how the worm's neurons process information and generate behaviour (Jabr, 2012). This goes to illustrate our point in section 1.2 that even a complete causal characterisation may not greatly increase our explainability of a system. Ideally, we would like to model the neuronal system at the three levels of abstraction as outlined in Marr's framework. Owing to the limited number of neurons, rich behavioural repertoire, and high-quality calcium-imaging data from whole brain recordings (Kato et al., 2015), we choose *C. elegans* neuronal and behavioural data to test and validate our methods in Chapter 3.

1.4. Research questions

The central question of this thesis can be broadly stated as follows:

What constitutes a comprehensive understanding of a complex dynamical system like the brain?

In Section 1.2, we introduced a conceptual framework for approaching our central question at a high level. Building upon the research context discussed in the preceding section, we are now prepared to refine our central question into a set of specific scientific inquiries. We split our central question into three research questions.

Q1: Under what conditions can we infer causal abstractions from purely observational data of a dynamical system?

Learning causal models solely from observational data is challenging, as it typically requires interventions and experiments to determine the true causal relationships. However, in fields like neuroscience, ethical and technological constraints often make interventions unfeasible. Causal inference methods can discover some relationships between *already identified* causal variables. Yet, the challenge remains: How can one directly *abstract* such causal variables from observational data?

While Rubenstein et al. present a notion of causal consistency between two SCMs, their work does not tell us whether one can infer such causally consistent abstractions from purely observational data. They also do not deal with dynamical systems but rather with SCMs in the independently and identically distributed (iid) setting. As we have seen in Section 1.3, dynamical systems possess unique causal information in their temporal structure that may be exploited to gain more useful abstractions (for example, capable of making predictions of the dynamics).

This indicates the need for a specialised framework designed for dynamical systems. Within this framework, the objective is to establish a set of assumptions under which an observationally consistent abstraction (inferred from non-interventional data) can also support causal claims. The advantage of an abstraction that supports causal claims lies in its ability to function as an autonomous model directly at the abstract level. This would allow researchers to gain insights into the dynamics of the system and foster a more robust causal understanding of its behaviour.

We tackle this question by first defining and developing a mathematical framework in Chapter 2. Within our framework, we are better equipped to answer this question.

Q2: Given these conditions, how can we *learn* causal abstractions directly from time-series data?

Existing studies have focused on deriving causal models from dynamical systems, mainly by attempting to construct a structural causal model (SCM) from differential equations (Janzing et al., 2018; Rubenstein et al., 2018; Bongers et al., 2022). However, in practical applications, the equations governing a dynamical system are often unknown. Lacking information on the exact physical equations governing the system, researchers have to rely on raw time-series data from the system.

Causal Feature Learning (CFL) is a more practical approach which is successful at learning abstractions from single, cause-effect pairs in independent and identically distributed (iid) data settings (Chalupka et al., 2015, 2017),. However, these iid approaches overlook relevant temporal information in time-series data commonly encountered in neuroscience. Furthermore, in complex systems, the identification of well-defined cause-effect pairs is often elusive, given the dense and global nature of interactions, as is the case in the brain. Hence CFL approaches would be inadequate to deal with data from complex dynamical systems. This creates the need for an algorithm that can *learn* causally consistent abstractions from time-series data. We address this question in Chapter 3

where we develop a data-driven approach to learn causally consistent abstractions.

Q3: How can causally consistent abstractions be used to gain meaningful insights into complex neuronal datasets?

Neuroscience data can pose several challenges for learning causally consistent abstractions. Aside from most of the datasets being observational data (i.e. without interventions), we also have to deal with relatively limited data (for the machine learning setting). Further challenges include the presence of time-correlated noise due to imaging artefacts, partially observable systems, and the individuality of animals and recording sessions. Additionally, multiple abstractions may be causally consistent with the underlying neuronal dynamics. However, not all of them may be relevant to our specific research objectives. Conversely, there might be certain abstractions that capture vital information, such as the behaviour of interest but are not causally consistent, or even derivable, from the ground-truth system. Krakauer et al. emphasise the indispensability of behaviour in gaining meaningful insights into a system. Hence, we would need to learn abstractions that preserve behavioural information.

The diverse forms, shapes, and sizes of neuroscience data present yet another challenge. Behavioural data, in particular, is collected in various formats, encompassing granular details about position, direction, and body configuration, as well as broader information like locomotor states and high-level behaviours. In this context, the aim is to develop a model capable of explaining multimodal behaviours and, in doing so, capture some of the common representations that come into play. This would shed insights into the way information is processed. Hence, to deal with multimodal data, a specific algorithm for a given setting would be inadequate. Rather, we require a flexible architecture that defines a class of algorithms that can be readily tailored to each setting.

We answer this question in Chapter 3, where we use our mathematical framework to develop a Neuronal Manifold Learning algorithm. We perform experiments to investigate this question further in Chapter 5 and demonstrate our method's robustness and flexibility on various data and settings. We make use of our mathematical framework in Publication (Grosse-Wentrup et al., 2023) to bridge the neuronal and cognitive levels.

1.5. Publication list

The following list comprises my research works during my doctoral studies, all contributing to the themes explored in this dissertation. While some works are not yet published, they have been submitted to peer-reviewed journals or are currently in preparation.

1. Akshey Kumar and Moritz Grosse-Wentrup. Learning Cognitive State Representations from Neuronal and Behavioural Data. In *2023 Conference on Cognitive Computational Neuroscience*, 2023. (accepted)

2. Akshey Kumar, Aditya Gilra, Mauricio Gonzalez-Soto, Anja Meunier and Moritz Grosse-Wentrup. BunDLe-Net: Neuronal Manifold Learning Meets Behaviour. *bioRxiv doi:10.1101/2023.08.08.551978 version 3*, 2023. `https://www.biorxiv.org/content/10.1101/2023.08.08.551978v3`. (submitted)

3. Moritz Grosse-Wentrup, Akshey Kumar, Anja Meunier, and Manuel Zimmer. Neuro-Cognitive Multilevel Causal Modeling: A Framework that Bridges the Explanatory Gap between Neuronal Activity and Cognition. *bioRxiv*, 2023. doi: 10.1101/22023.10.27.564404 `https://doi.org/10.1101/2023.10.27.564404`.

4. Pascal Weber, Lukas Miklautz, Akshey Kumar, Moritz Grosse-Wentrup, and Claudia Plant. CaFe DBSCAN: A Density-based Clustering Algorithm for Causal Feature Learning. *The 10th IEEE International Conference on Data Science and Advanced Analytics.* DSAA, 2023. `http://eprints.cs.univie.ac.at/7860/`. (accepted)

5. Akshey Kumar and Moritz Grosse-Wentrup. BunDLe-Net: A Python toolbox for Neuronal Manifold Learning, 2023. (in preparation)

We have chosen to include Publication 2 in this thesis in Chapter 3. We intend to publish the contents of Chapter 4 as Publication 5.

2. Theoretical formalism

In this chapter, we present the first main contribution of this thesis: a theoretical framework for abstracting causally consistent models from dynamical systems. This chapter assumes familiarity with basic probability theory and structural causal models (SCM) (Pearl et al., 2016).

We begin with the definition of the fundamental elements of our framework and our main criteria: dynamical observational consistency (DOC) and dynamical causal consistency (DCC) in Section 2.1. Following these definitions, we present our main theoretical results in the form of two theorems in Section 2.2. Subsequently, in Section 2.3, we define behavioural consistency and discuss its relevance for the neuroscience setting. We conclude this Chapter with the analytical derivation of an abstraction in Section 2.4.1, which demonstrates all of the theoretical concepts introduced in the previous sections.

The two central theorems presented here serve as the foundation of the Behaviour and Dynamics Learning Network (BunDLe-Net) algorithm that we present in Chapter 3. The contents of this chapter are also used as the theoretical foundations for our other Publication, Neuro-Cognitive Multilevel Causal Models Grosse-Wentrup et al. (2023), that bridges the gap between the neuronal and cognitive level.

2.1. Preliminary definitions

Let $E_X = (\epsilon_t : t \in \mathcal{T})$ be an indexed set of exogenous noise variables[1] and let $(X_t \in \mathcal{X} : t \in \mathcal{T})$ be an indexed set of endogenous variables[2].

Definition 1. *(Dynamic SCM) A dynamic SCM (dSCM) is a tuple* $\{\mathcal{T}, P_{E_X}, S_X, \mathcal{I}_\mathcal{X}\}$ *where,*

- $\mathcal{T} \subseteq \mathbb{N}$ *is an index set where the indices correspond to instances in time.*
- P_{E_X} *is the joint probability distribution over the exogenous variables such that for all* $t, t' \in \mathcal{T}$ *such that* ϵ_t *and* $\epsilon_{t'}$ *are independent.*
- S_X *is a set of equations* $X_{t+1} = f[Pa(X_t), \epsilon_t]$ *for* $t \in \mathcal{T}$ *where* $Pa(X_t) \subseteq \{X_{t'} | t' < t\}$.
- $\mathcal{I}_X$ *is a set of allowed interventions*[3] *on the endogenous variables* $X_t : t \in \mathcal{T}$.

[1] Exogenous variables are variables that we cannot access

[2] Endogenous variables are variables that we can access and make interventions on.

[3] An intervention on variable X_t is defined as setting the variable X_t to a specific value, say x, and is denoted $do(X_t = x)$

The essence is that in a *dSCM*, the parents of a variable X_t are restricted to variables that lie in the temporal past of X_t. In our framework, we allow all possible *complete* interventions, $\mathcal{I}_{\mathcal{X}} = \{do(X_t = x) : t \in \mathcal{T}, x \in \mathcal{X}\}$. In other words, if X_t is a high-dimensional vector, we consider interventions on X_t as a whole. Interventions on individual components of X_t is left as an avenue for future work.

Note that the distribution over the exogenous variables, together with the structural equations, induce a distribution over the endogenous variables. We will refer to the latter distribution in our definition of dynamic causal consistency.

Definition 2. *(Markov dSCM) A dSCM is said to be Markov if* $Pa(X_t) = \{X_{t-1}\}$ *for all* $t \in \mathcal{T}$.

$$... \longrightarrow X_{t-1} \longrightarrow X_t \longrightarrow X_{t+1} \longrightarrow ...$$

Figure 2.1.: Causal structure of a Markov dSCM

We denote the conditional probability distribution over endogenous variables by the transition matrix[4] T_X which is given by $[T_X]_{ij} = P(X_{t+1} = j | X_t = i)$ where $i, j \in \mathcal{X}$. We assume a time-homogeneous Markov process, which means the transition matrix does not change with time.

Definition 3. *(Macroscopic interventions) Let* $M_X = \{\mathcal{T}, P_{E_X}, S_X, \mathcal{I}_X\}$ *be a dSCM and let* τ *be a surjective mapping from the state space* $\mathcal{X}$ *to some macrovariable space* $\mathcal{Y}$. *We define interventions on the macroscopic level* $do(Y_t = y)$ *for* $y \in \mathcal{Y}$ *as choosing any intervention in the set* $\{do(X_t = x) | do(X_t = x) \in \mathcal{I}_X, y = \tau(x)\}$. *The set of all possible macrovariable interventions is then given by* $\mathcal{I}_Y = \{do(Y_t = y) : t \in \mathcal{T}, y \in \mathcal{Y}\}$.

Overloading the notation a bit, we denote $\mathcal{I}_Y = \tau(\mathcal{I}_X)$

Definition 4. *(Dynamic observational consistency of dSCMs)*
Let $M_X = \{\mathcal{T}, P_{E_X}, S_X, \mathcal{I}_X\}$ *and* $M_Y = \{\mathcal{T}, P_{E_Y}, S_Y, \mathcal{I}_Y\}$ *be two dSCMs that share the same index set and let* $\tau : \mathcal{X} \to \mathcal{Y}$ *be a surjective mapping between the state spaces of* M_X *and* M_Y. *We say that the triple* $\{M_X, M_Y, \tau\}$ *is dynamically observationally consistent iff for all* $t \in \mathcal{T}$ *we have,* $Y_t = \tau(X_t)$, *and for all* $x' \in \mathcal{X}, y \in \mathcal{Y}$ *we have*[5],

$$P(Y_{t+1} = y | X_t = x') = P(Y_{t+1} = y | Y_t = \tau(x')). \tag{2.1}$$

[4] Note that the transition matrix need not uniquely determine the Markov dSCM since different structural equations S_X and exogenous distributions P_ϵ may induce the same distribution and thus the same transition matrix.

[5] To be more mathematically precise the LHS of the following expression would be $P(X_{t+1} \in \tau^{-1}(y) | X_t = x')$. However, since this is less readable, we shall use $P(Y_{t+1} = y | X_t = x')$ as shorthand for it. This is possible since $Y_{t+1} = y$ and $X_{t+1} \in \tau^{-1}(y)$ are equivalent statements if τ links their spaces. Thus, the LHS lives in the probability space of $\mathcal{X}$ while the RHS lives in the probability space of $\mathcal{Y}$.

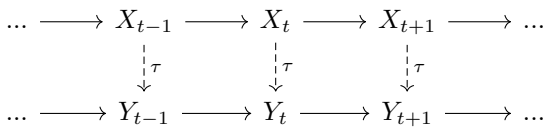

Figure 2.2.: Existence of DCC transformation τ between two dSCMs

Definition 5. *(Dynamic causal consistency of dSCMs)*
Let $M_X = \{\mathcal{T}, P_{E_X}, S_X, \mathcal{I}_X\}$ and $M_Y = \{\mathcal{T}, P_{E_Y}, S_Y, \mathcal{I}_Y\}$ be two dSCMs that share the same index set and let $\tau : \mathcal{X} \to \mathcal{Y}$ be a surjective mapping between the state spaces of M_X and M_Y. We say that the triple $\{M_X, M_Y, \tau\}$ is dynamically causally consistent iff for all $t \in \mathcal{T}$ we have, $Y_t = \tau(X_t)$, and for all $x' \in \mathcal{X}, y \in \mathcal{Y}$ we have[5]*,*

$$P(Y_{t+1} = y | do(X_t = x')) = P(Y_{t+1} = y | do(Y_t = \tau(x'))). \tag{2.2}$$

Note that on the right-hand side, the macroscopic intervention $do(Y_t = \tau(x'))$ comprises any intervention from a set of microscopic interventions as in Definition 3, and not just the intervention $do(X_t = x')$.

2.1.1. Intuitive meaning of DCC and DOC

Figure 2.2 demonstrates the case when two dSCMs in combination with a mapping τ are DCC. The solid arrows represent causal links, and the dotted arrows represent the mapping τ. Later in theorem 2, we will prove that this diagram commutes.

The DOC criterion can be thought of as a conditional independence statement, $(Y_{t+1} \perp X_t | Y_t)$. It ensures that the dynamics at the Y-level is sealed-off from the explicit microvariable state at the X-level. In this way, the dynamics at the Y-level is self-contained, and no additional information from the X-level could tell us anything more about the Y-level dynamics. Thus, the Y-level can be thought of as an independent process in its own right. The DCC criterion makes sure that the above is valid even when performing interventions, thus making the model at the Y-level causal.

Although we have considered a very simple causal structure, namely, the Markov chain, these principles are quite generic and applicable to any kind of time-series causal structure. This includes causation between subsystems of a dynamical system. Such generalisations are beyond the scope of this chapter but will be discussed in Section 6.3.

2.2. Central theorems

The following lemma and theorem establish how we may learn dynamically causally consistent transformations from purely observational data in a Markov setting.

Lemma 1. *Let M_X be a Markov dSCM and let $\tau : \mathcal{X} \to \mathcal{Y}$ be a surjective mapping. Let M_Y be a dSCM that describes the dynamics of $Y_t := \tau(X_t)$. If $\{M_X, M_Y, \tau\}$ is DOC, then M_Y is also a Markov dSCM.*

Proof. [6] The theorem states that a sufficient condition for M_Y to be a Markov dSCM is dynamical observational consistency, i.e for all $x \in \mathcal{X}, y' \in \mathcal{Y}, t \in \mathcal{T}$ we have,

$$P(Y_{t+1} = y' | X_t = x) = P(Y_{t+1} = y' | Y_t = \tau(x)). \tag{2.3}$$

To prove that M_Y is Markov, we will show that the term

$$P(Y_{t+1} = y' | Y_t = y_i, Y_{t-1} = y_j, ...)$$

equals to

$$P(Y_{t+1} = y' | Y_t = y_i)$$

under the assumption of DOC. First, we note the equivalence between the statements $Y_t = y$ and $X_t \in \tau^{-1}(y)$ and use this to expand,

$$P(Y_{t+1} = y' | Y_t = y_i, Y_{t-1} = y_j, ...) = P(Y_{t+1} = y' | X_t \in \tau^{-1}(y_i), X_{t-1} \in \tau^{-1}(y_j), ...)$$

Using the Bayes rule, the additivity rule for mutually exclusive events, and again the Bayes rule, we express the term as a sum over past trajectories,

$$= \frac{\dots \sum_{x_j \in \tau^{-1}(y_j)} \sum_{x_i \in \tau^{-1}(y_i)} P(Y_{t+1} = y' | X_t = x_i, X_{t-1} = x_j, ...) P(X_t = x_i, X_{t-1} = x_j, ...)}{\dots \sum_{x_j \in \tau^{-1}(y_j)} \sum_{x_i \in \tau^{-1}(y_i)} P(X_t = x_i, X_{t-1} = x_j, ...)} \tag{2.4}$$

Now consider the term, $P(Y_{t+1} = y' | X_t = x_i, X_{t-1} = x_j, ...)$. Since (X_n) is a Markov process, it can be simplified as follows,

$$\begin{aligned} P(Y_{t+1} = y' | X_t = x_i, X_{t-1} = x_j, ...) &= \sum_{x' \in \tau^{-1}(y)} P(X_{t+1} = x' | X_t = x_i, X_{t-1} = x_j, ...) \\ &= \sum_{x' \in \tau^{-1}(y)} P(X_{t+1} = x' | X_t = x_i) \\ P(Y_{t+1} = y' | X_t = x_i, X_{t-1} = x_j, ...) &= P(Y_{t+1} = y' | X_t = x_i) \end{aligned} \tag{2.5}$$

Using Equation (2.5), the term (2.4) becomes,

$$= \frac{\dots \sum_{x_j \in \tau^{-1}(y_j)} \sum_{x_i \in \tau^{-1}(y_i)} P(Y_{t+1} = y' | X_t = x_i) P(X_t = x_i, X_{t-1} = x_j, ...)}{\dots \sum_{x_j \in \tau^{-1}(y_j)} \sum_{x_i \in \tau^{-1}(y_i)} P(X_t = x_i, X_{t-1} = x_j, ...)} \tag{2.6}$$

$$= \frac{\sum_{x_i \in \tau^{-1}(y_i)} P(Y_{t+1} = y' | X_t = x_i) \left(\dots \sum_{x_j \in \tau^{-1}(y_j)} P(X_t = x_i, X_{t-1} = x_j, ...) \right)}{\sum_{x_i \in \tau^{-1}(y_i)} \left(\dots \sum_{x_j \in \tau^{-1}(y_j)} P(X_t = x_i, X_{t-1} = x_j, ...) \right)} \tag{2.7}$$

[6] *Note*: This proof is done for the case of discrete state space $\mathcal{X}$. The statement is also thought to hold for continuous state spaces, but the proof would be considerably more involved and is hence omitted.

Now, if we assume that DOC holds, for a given value of $Y_t = y_i$, the distribution $P(Y_{t+1} = y'|X_t = x_i)$ is the same constant as long as $x_i \in \tau^{-1}(y_i)$. It only depends on the value of y_i, and is thus given by $P(Y_{t+1} = y'|Y_t = y_i)$. Since the first summation sums over x_i for a fixed value of y_i, we have,

$$= \frac{P(Y_{t+1} = y'|Y_t = y_i) \sum_{x_i \in \tau^{-1}(y_i)} \left(\dots \sum_{x_j \in \tau^{-1}(y_j)} P(X_t = x_i, X_{t-1} = x_j, ...) \right)}{\sum_{x_i \in \tau^{-1}(y_i)} \left(\dots \sum_{x_j \in \tau^{-1}(y_j)} P(X_t = x_i, X_{t-1} = x_j, ...) \right)} \tag{2.8}$$

$$= P(Y_{t+1} = y'|Y_t = y_i) \tag{2.9}$$

Thus, we started with the term $P(Y_{t+1} = y'|Y_t = y_i, Y_{t-1} = y_j, ...)$, used DOC to show that it simplifies to $P(Y_{t+1} = y'|Y_t = y_i)$, which means that DOC is a sufficient condition for M_Y to be Markov □

Theorem 1. *Let M_X and M_Y be two dSCMs such that $Y_t = \tau(X_t)$ where τ is a surjective mapping. A sufficient condition for dynamical causal consistency of the triple $\{M_X, M_Y, \tau\}$, is for M_X to be a Markov dSCM and for DOC to hold.*

Proof. Since M_X is a Markov dSCM and $\{M_X, M_Y, \tau\}$ statisfies *DOC*, it follows from Lemma (1) that M_Y is also a Markov dSCM.

We proceed by showing that the *RHS* of Equations (2.1) and (2.2) are equal if M_X and M_Y are Markov. Then we show that the *LHS* of these equations are also equal, thus concluding our proof.

Since M_Y is a Markov dSCM, we have $Pa(Y_{t+1}) = \{Y_t\}$. Since Y_t is the only parent of Y_{t+1} we have,

$$P(Y_{t+1}|do(Y_t = x)) = P(Y_{t+1}|Y_t = x) \tag{2.10}$$

Similarly, since M_X is also a Markov dSCM, we have.

$$P(X_{t+1}|do(X_t = x)) = P(X_{t+1}|X_t = x) \tag{2.11}$$

Now consider the LHS of equations (5) and (4). We define the pre-image of $y \in \mathcal{Y}$ with respect to the mapping τ as the set $\tau^{-1}(y) = \{x \in \mathcal{X}|\tau(x) = y\}$. Consequently for any choice of $y \in \mathcal{Y}$,

$$P(Y_{t+1} = y|do(X_t)) = P(X_{t+1} \in \tau^{-1}(y)|do(X_t)) \tag{2.12}$$

$$= \sum_{x \in \tau^{-1}(y)} P(X_{t+1} = x|do(X_t)) \tag{2.13}$$

The summation follows since all the states $x \in \tau^{-1}(y) \subseteq \mathcal{X}$ are mutually exclusive. In a similar way we also have for any $y \in \mathcal{Y}$,

$$P(Y_{t+1} = y|X_t) = \sum_{x \in \tau^{-1}(y)} P(X_{t+1} = x|X_t) \tag{2.14}$$

Since the summands in *RHS* of equations (2.13) and (2.14) are equal by equation (2.11), it follows that,

$$P(Y_{t+1}|do(X_t = x)) = P(Y_{t+1}|X_t = x) \tag{2.15}$$

Thus from equations (2.11) and (2.15), we see that the DOC is sufficient for DCC in our prescribed framework of a Markov microscopic model M_X.

□

Lemma 1 tells us that our DOC requirement ensures that the macroscopic level is Markov. This is an associational statement and not a causal one. Theorem 1 elevates this to a causal statement under certain conditions. This means that we can talk about macroscopic interventions in terms of the *do*-operator.

Moreover, our theorem means that under the assumption of Markov dynamics on the X-level, a causally consistent transformation can be identified from purely observational data if DOC holds. The resulting macroscopic model would have an identical causal structure to the microscopic model. i.e. they both have the same Markov causal structure as in Figure 2.1.

The following theorem will enable us to construct an algorithm for learning DOC transformations from data.

Theorem 2. *If the following diagram commutes, DOC is fulfilled.*

$$\begin{array}{ccc} X_t & \xrightarrow{\mathbf{T}_X} & X_{t+1} \\ \vdots\tau & & \vdots\tau \\ \downarrow & & \downarrow \\ Y_t & \xrightarrow{\mathbf{T}_Y} & Y_{t+1} \end{array}$$

Proof. For this proof, consider a probability vector $P_X \in [0,1]^{|\mathcal{X}|}$ which represents the probability mass function at time t over all possible microstates in space $\mathcal{X}$. We are now interested in how this probability vector can be transformed to give us a probability mass function $P_Y \in [0,1]^{|\mathcal{Y}|}$ over the macrostates in $\mathcal{Y}$ at the time $t+1$. If the diagram commutes, then we may either perform the dynamics on the X-level and then τ transform, or first τ transform and then perform the dynamics on the considerably lower-dimensional Y-level.

$$\widetilde{T_Y}\widetilde{\tau}P_X = \widetilde{\tau}\widetilde{T_X}P_X \tag{2.16}$$

where $\widetilde{T_X}$, $\widetilde{T_Y}$ and $\widetilde{\tau}$ denote matrix representations[7] of the transition and coarse-graining

[7]Note that matrix representations in the probability space $[0,1]^{|\mathcal{X}|}$ and $[0,1]^{|\mathcal{Y}|}$ do not necessarily mean linear-functions in the state space $\mathcal{X}$ and $\mathcal{Y}$. Thus we are not making any restrictions on T_X and τ, which are free to be non-linear on the state spaces

models in the probability spaces of states. We define them as follows,

$$[\widetilde{\tau}]_{ij} := \delta(y_i, \tau(x_j)) \tag{2.17}$$

$$[\widetilde{T_X}]_{ij} := P(X_{t+1} = x_i | X_t = x_j) \tag{2.18}$$

$$[\widetilde{T_Y}]_{ij} := P(Y_{t+1} = y_i | Y_t = y_j) \tag{2.19}$$

Note that fixing τ and T_X induces a transition matrix T_Y on the Y-level, which is given by $T_Y = \tau T_X \tau^R$, where τ^R is the right inverse of non-square matrix τ. For the remainder of the proof, however, it suffices to consider the form in Equation 2.19.

Since Equation 2.16 must hold for any possible choice of the initial probability vector P_X, we have,

$$\widetilde{T_Y}\widetilde{\tau} = \widetilde{\tau}\widetilde{T_X} \tag{2.20}$$

We can simplify this as follows (using the laws of matrix multiplication, the definitions and probability rules),

$$\sum_l [\widetilde{T_Y}]_{il}[\widetilde{\tau}]_{lk} = \sum_j [\widetilde{\tau}]_{ij}[\widetilde{T_X}]_{jk} \tag{2.21}$$

$$\sum_l P(Y_{t+1} = y_i | Y_t = y_l)\delta(y_l, \tau(x_k)) = \sum_j \delta(y_i, \tau(x_j))P(X_{t+1} = x_j | X_t = x_k) \tag{2.22}$$

$$P(Y_{t+1} = y_i | Y_t = \tau(x_k)) = \sum_{x_j \in \tau^{-1}(y_i)} P(X_{t+1} = x_j | X_t = x_k) \tag{2.23}$$

$$P(Y_{t+1} = y_i | Y_t = \tau(x_k)) = P(X_{t+1} \in \tau^{-1}(y_i) | X_t = x_k) \tag{2.24}$$

$$P(Y_{t+1} = y_i | Y_t = \tau(x_k)) = P(Y_{t+1} = y_i | X_t = x_k) \tag{2.25}$$

Hence the diagram commuting implies DOC. □

This theorem forms the theoretical basis of the architecture for the learning algorithm that we introduce in Chapter 3.

2.3. The behavioural setting

DOC and DCC ensure that our transformations preserve the dynamical causal structure of the ground truth model. In many cases, there may be an enormous number of possible abstractions that fulfil DCC, many of which may not be useful for our specific needs. For example, one possible transformation that respects DCC but tells us nothing interesting about the dynamics is a function mapping every element of $\mathcal{X}$ to a constant.

To demonstrate how DCC models can be made useful *and* interpretable, we need to consider the neuroscience setting. Neuronal activity serves an organism the purpose of interacting with its environment. A good cognitive model would summarise the neuronal dynamics while preserving information about how it interacts with its environment.

Krakauer et al. suggest studying neuronal activity in the context of the organism's behaviour instead of studying it in isolation. Following this, we supplement our notion of a dSCM to include a behaviour that results from the neuronal activity.

Let $M_X = \{\mathcal{T}, P_{E_X}, S_X, \mathcal{I}_X\}$ be a dSCM. Let $(B_t \in \mathcal{B} : t \in \mathcal{T})$ be an indexed set of behaviour variables. S_B is a set of equations $B_t = g(X_t, \epsilon_B)$ for $t \in \mathcal{T}$. $M_X^B = \{\mathcal{T}, P_{E_X}, S_X, \mathcal{I}_X, S_B\}$ is defined as the corresponding *dSCM in a behavioural context.*

Definition 6. *(Behavioural consistency) Let M_X^B and M_Y^B be two dSCMs in the same behavioural context. The triple $\{M_X^B, M_Y^B, \tau\}$ are said to be behaviourally consistent (BC) iff for all $t \in \mathcal{T}$ we have $Y_t = \tau(X_t)$ and for all $b \in \mathcal{B}$, $x \in \mathcal{X}$ we have,*

$$P(B_t = b | X_t = x) = P(B_t = b | Y_t = \tau(x)) \tag{2.26}$$

Our definition ensures that both the microscopic and macroscopic level contain the same amount of information about the behaviour. In the neuroscience setting, we assume that the behaviour is always an effect of the neuronal activity, i.e. $B_t = g(X_t, \epsilon_B)$, and not vice-versa. Making the assumption allows the macroscopic level to be causal with respect to the behaviour as well. Subsequent work could extend this framework to scenarios involving feedback loops where the behaviour can further act as a stimulus and induce neuronal activity.

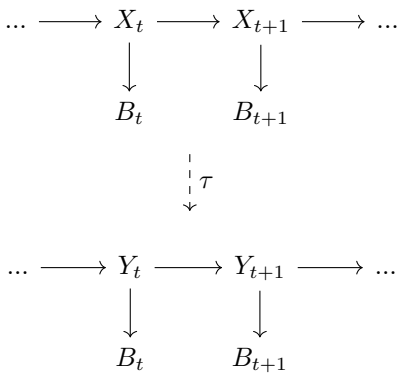

Figure 2.3.: Abstraction of a Markov dSCM while preserving behavioural information

2.4. Analytical solution example

Here, we demonstrate how one can construct a macroscopic dSCM for any system with linear dynamics. Note that this is not a unique solution, and there may be other abstractions that fulfil our requirements. We use the theoretical devices developed in the previous sections to present an analytical solution for the linear case and prove how it satisfies our criteria. The resulting macrovariable preserves the Markov causal structure and is sealed off from the microscopic dynamics since it preserves sufficient information about its own dynamics.

2.4.1. Analytic solution for linear models

Here, we present a transformation of a linear dynamical system in a behavioural context and prove that it is both DCC and BC. Let $M_X = \{\mathcal{T}, P_{E_X}, S_X, \mathcal{I}_X\}$ be a Markov dSCM where X_t, μ, ϵ_t^X are vectors in $\mathbb{R}^n$ and M, Σ are $n \times n$ matrices:

$$\mathcal{T} : \mathbb{N} \tag{2.27}$$

$$S_X : X_{t+1} = X_t + MX_t + \epsilon_t^X \tag{2.28}$$

$$P_{E_X} : \epsilon_t^X \sim \mathcal{N}(\mu, \Sigma) \tag{2.29}$$

$$\mathcal{I}_X : \{do(x) | x \in \mathbb{R}^n\} \tag{2.30}$$

M represents the adjacency matrix of directed connections between various X^i i.e. $M_{ij} \neq 0$ iff there is a causal connection, $X_t^j \to X_{t+1}^i$. We do not allow any cycles in graph defined by M. We also define a behaviour based on the microvariable state as follows,

$$B_t = g(K^T X_t) + \epsilon_B, \qquad K \in \mathbb{R}^n \tag{2.31}$$

where g can be any arbitrary functional mapping and ϵ_B is a noise variable. Note that the behaviour depends on X_t only through the linear combination $K^T X$.

Now let $\tau : \mathbb{R}^n \to \mathcal{F}$ be a surjective coarse-graining[8] to $\mathcal{F} \subseteq \mathbb{R}^m$, a lower-dimensional space. We choose m to be equal to the length of the longest path in the DAG defined by M. We represent τ as an $m \times n$ matrix such that $Y_t := \tau(X_t)$.

$$\tau = \begin{pmatrix} K^T M \\ K^T M^2 \\ \vdots \\ K^T M^m \end{pmatrix} \tag{2.32}$$

Let $M_Y = \{\mathcal{T}, P_{E_Y}, S_Y, \mathcal{I}_Y\}$ be a Markov dSCM

$$\mathcal{T} : \mathbb{N} \tag{2.33}$$

$$S_Y : Y_{t+1} = T_Y Y_t + \epsilon_t^Y \tag{2.34}$$

$$P_{E_Y} : \epsilon_t^Y \sim \mathcal{N}(\tau\mu, \tau\Sigma\tau^T) \tag{2.35}$$

$$\mathcal{I}_Y : \{do(x) | \tau(x) \in \mathbb{R}^m\} \tag{2.36}$$

where T_Y is an $m \times m$ matrix

$$T_Y = \begin{pmatrix} 1 & 1 & 0 & \dots & 0 \\ 0 & 1 & 1 & & 0 \\ 0 & 0 & 1 & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix} \tag{2.37}$$

Proof. We begin by showing that the diagram in Theorem 2 commutes, thus ensuring dynamical observational consistency (DOC). We then invoke Theorem 1 to prove dynamical causal consistency (DCC).

[8] mapping to a lower dimensional space

Upper arm of the commutativity diagram First, we apply the transition model as stipulated by M_X to X_t, followed by coarse-graining with τ to obtain,

$$Y_{t+1}^{(U)} = \tau X_{t+1} = \begin{pmatrix} K^T X_t + K^T M X_t \\ K^T M X_t + K^T M^2 X_t \\ K^T M^2 X_t + K^T M^3 X_t \\ \vdots \\ K^T M^m X_t \end{pmatrix} + \tau \epsilon_t^X \tag{2.38}$$

Note that the final entry in the column vector simplifies to $K^T M^m X_t$ since $M^{m+1} = 0$. This is because M is the adjacency matrix of an acyclic graph and m is the length of the longest possible walk in the graph.

Lower arm of the commutativity diagram For this path, we first coarse-grain X_t with τ followed by applying the transition model on the Y-level as stipulated by M_Y to obtain,

$$Y_{t+1}^{(L)} = T_Y Y_{t+1} + \epsilon_t^Y = T_Y \tau X_t + \epsilon_t^Y = \begin{pmatrix} K^T X_t + K^T M X_t \\ K^T M X_t + K^T M^2 X_t \\ K^T M^2 X_t + K^T M^3 X_t \\ \vdots \\ K^T M^m X_t \end{pmatrix} + \epsilon_t^Y \tag{2.39}$$

By definition of ϵ_t^Y, $Y_{t+1}^{(U)}$ and $Y_{t+1}^{(L)}$ are identically distributed, thus concluding our proof of DOC. Since M_X is a Markov dSCM by definition, we invoke Theorem 1 to prove DCC. Behaviour causal consistency is ensured since the first component of Y_t is $K_T X_t$, which contains all information about B_t in X_t. □

For an arbitrary coarse-graining of variables at the X-level, it may not be possible to have a description of the variables on the Y-level in terms of a dSCM. In some cases, a dSCM description might be possible, but only with a different causal structure or with latent confounders. With the above example, we have clearly shown how a dSCM can be abstracted to the Y-level in such a way that causal structure and behavioural information are preserved.

Having seen that it is possible to construct consistent transformations for linear dynamical systems, we now move on to the non-linear case, where a closed-form analytical solution may not be possible. To tackle this, we develop a generic architecture for learning dynamically consistent macroscopic models in Chapter 3 based on the commutativity diagram of Theorem 2.

3. BunDLe-Net: Neuronal manifold learning meets behaviour

In this chapter, we leverage our theoretical formalism and the commutativity diagram of Theorem 2 to construct an architecture that facilitates learning causally consistent abstractions from time-series data. The following publication introduces this advancement to the neuroscience community as BunDLe-Net, a Neuronal Manifold Learning algorithm. Validating its efficacy, we assess its performance using neuronal and behavioural data obtained from the roundworm *C. elegans* (Kato et al., 2015). Additionally, we conduct comparative analyses, pitting BunDLe-Net against other widely-used dimensionality reduction techniques and state-of-the-art neuronal manifold learning methods applied to the same dataset. We extensively discuss the insights gained from BunDLe-Net's abstractions that reveal distinct topologies of the neuronal manifold. While some findings reinforce existing knowledge, confirming the validity of our method, we also unveil novel insights in this work.

3.1. Bibliographic information

Akshey Kumar, Aditya Gilra, Mauricio Gonzalez-Soto, Anja Meunier and Moritz Grosse-Wentrup. BunDLe-Net: Neuronal Manifold Learning Meets Behaviour. *bioRxiv*, doi:10.1101/2023.08.08.551978 version 3, 2023. `https://www.biorxiv.org/content/10.1101/2023.08.08.551978v3`.

3.2. My contribution

- Creating the algorithm
- Implementing the algorithm in Python and conducting various experiments
- Creating an evaluation scheme for comparisons between algorithms
- Writing the manuscript (except the Introduction and Discussion sections)

BunDLe-Net: Neuronal Manifold Learning Meets Behaviour

Akshey Kumar[1,4], Aditya Gilra[2,3], Mauricio Gonzalez-Soto[1], Anja Meunier[1,4], and Moritz Grosse-Wentrup[1,5,6]

[1] *Research Group Neuroinformatics, Faculty of Computer Science, University of Vienna, Austria*
[2] *Machine Learning Group, Centrum Wiskunde & Informatica, Amsterdam, Netherlands*
[3] *Department of Computer Science, The University of Sheffield, Sheffield, United Kingdom*
[4] *UniVie Doctoral School Computer Science DoCS, University of Vienna, Austria*
[5] *Vienna Cognitive Science Hub*
[6] *Data Science @ Uni Vienna*

October 18, 2023

Abstract

Neuronal manifold learning techniques represent high-dimensional neuronal dynamics in low-dimensional embeddings to reveal the intrinsic structure of neuronal manifolds. Common to these techniques is their goal to learn low-dimensional embeddings that preserve all dynamic information in the high-dimensional neuronal data, i.e., embeddings that allow for reconstructing the original data. We introduce a novel neuronal manifold learning technique, BunDLe-Net, that learns a low-dimensional Markovian embedding of the neuronal dynamics which preserves only those aspects of the neuronal dynamics that are relevant for a given behavioural context. In this way, BunDLe-Net eliminates neuronal dynamics that are irrelevant to decoding behaviour, effectively de-noising the data to reveal better the intricate relationships between neuronal dynamics and behaviour. We demonstrate the quantitative superiority of BunDLe-Net over commonly used and state-of-the-art neuronal manifold learning techniques in terms of dynamic and behavioural information in the learned manifold on calcium imaging data recorded in the nematode *C. elegans*. Qualitatively, we show that BunDLe-Net learns highly consistent manifolds across multiple worms that reveal the neuronal and behavioural motifs that form the building blocks of the neuronal manifold.

1 Introduction

Advances in neuronal imaging techniques have increased the number of neurons that can be recorded simultaneously by several orders of magnitude [1, 2]. While these advances greatly expand our abilities to study and understand brain function, the complexities of the resulting high-dimensional data sets pose non-trivial challenges for data analysis and visualisation. Fortunately, individual neurons are embedded in brain networks that collectively organise their high-dimensional neuronal activity patterns into lower-dimensional neuronal manifolds [3, 4].

This work was supported under the CHIST-ERA grant (CHIST-ERA-19-XAI-002), by the Austrian Science Fund (FWF) (grant reference I 5211-N) and the Engineering and Physical Sciences Research Council United Kingdom (grant reference EP/V055720/1), as part of the Causal Explanations in Reinforcement Learning (CausalXRL) project.

To understand the collective organisation of individual neurons into brain networks, we require algorithms that learn neuronal manifolds from empirical data.

The goal of neuronal manifold learning is to find low-dimensional representations of data that preserve particular data properties. In neuroscience, a broad range of classical dimensionality reduction techniques is being employed, including but not limited to principal component analysis (PCA), multi-dimensional scaling (MDS), Isomap, locally linear embedding (LLE), Laplacian eigenmaps (LEM), t-SNE, and uniform manifold approximation and projection (UMAP) [5]. More recently, advances in artificial intelligence in general and deep learning methods, in particular, have given rise to a new class of (often non-linear) dimensionality reduction techniques, e.g., based on autoencoder architectures [6, 7, 8] or contrastive learning frameworks [9].

Common to all these techniques is their goal to reduce the data dimensionality while preserving particular properties of or information in the data. For instance, autoencoder-based frameworks typically focus on finding low-dimensional data representations that allow a good (or even perfect) reconstruction of the original, high-dimensional data. In contrast, we argue that reconstruction quality is only one out of several desirable features for neuronal manifold learning. First, and in line with the argument by Krakauer et al. [10] that neuroscience needs behaviour, we argue that a neuronal manifold learning algorithm should not aspire to represent all but only those characteristics of high-dimensional neuronal activity patterns that are relevant in a given behavioural context. For instance, when studying an animal's ability to navigate a maze using visual cues, neuronal activity patterns that carry auditory or olfactory information are irrelevant in the behavioural context and should be abstracted away to better reveal the intricate relationships between neuronal representations of the visual cues and motor behaviour. Second, we argue that the reconstruction of the dynamics of the neuronal activity patterns should also take into account whether the low-dimensional embedding is causally sufficient in terms of the system's dynamics. To elaborate on this issue, consider the example of using a dimensionality reduction technique to learn the physical state description of a simple pendulum from a video stream showing the pendulum in action. Ideally, the dimensionality reduction technique should learn to represent the position and momentum of the pendulum for each video frame because these two variables constitute a full description of the system's physical state. In contrast, a dimensionality reduction technique that learns to represent the positions of the pendulum in the current and the past video frame only (without representing the pendulum's momentum) would also allow for a good reconstruction of the dynamics of the pendulum. This is the case because the pendulum's momentum, which is required to predict in which direction it will swing, can be approximately reconstructed from the difference in position across two video frames. However, this representation would not constitute a complete description of the actual physical state of the system. In analogy, a neuronal manifold learning technique should attempt to learn a complete physical state description of the underlying neuronal dynamics. Mathematically, this goal can be formulated as learning neuronal state trajectories that form a Markov chain because, in a Markov chain, the current state of the chain is causally sufficient for predicting the next state (in mathematical terms, the past and future states of the chain are statistically independent given the current state).

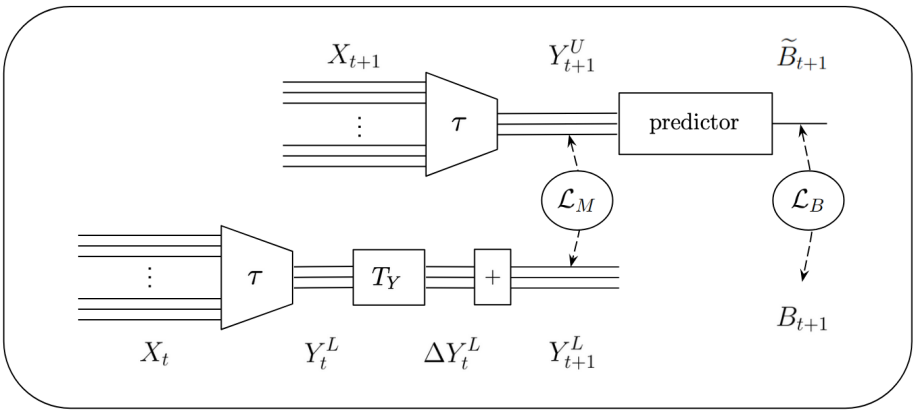

Figure 1: The BunDLe-Net architecture

Here, we introduce a novel framework for neuronal manifold learning, termed the Behaviour and Dynamics Learning Network (BunDLe-Net). BunDLe-Net learns a low-dimensional Markovian representation of the neuronal dynamics while retaining all information about a given behavioural context. It is based on the architecture shown in Fig. 1, which consists of two branches. In the lower branch, the high-dimensional neuronal trajectories X_t are first projected via a mapping τ to a lower-dimensional, latent trajectory Y^L_t. A first-order transition model T_Y then predicts the difference ΔY^L_t between the current and the next state to arrive at an estimate Y^L_{t+1} of the latent state at time $t+1$. This predicted latent state is compared to the true latent state at time $t+1$ in the upper branch, which is obtained by mapping the observed neuronal state X_{t+1} at time $t+1$ via the same τ as in the lower branch to the latent state Y^U_{t+1} via the loss function $\mathcal{L}_{\text{Markov}}$. By jointly learning the mapping τ and the first-order transition model T_Y that minimise the loss function $\mathcal{L}_{\text{Markov}}$ we obtain a latent, low-dimensional time-series Y_t that is Markovian by construction. This is the case because the transition model T_Y acts as a bottleneck that constrains the class of functions for τ for which the current state of the system is sufficient to predict the next state, in the sense that previous states do not provide any additional information. However, this architecture is not yet sufficient to learn a meaningful latent data representation because a mapping τ that projects the neuronal state trajectories to a constant ($Y_t = c$) would also fulfil the criterion of Markovianity. To obtain a meaningful latent representation, we also require that the behavioural context must be decodable from the latent representation Y_t by adding the loss function $\mathcal{L}_{\text{Behaviour}}$ that measures the reconstruction error between the true behavioural labels (B_{t+1}) and those predicted from the latent representation ($\tilde{B}_{t+1}$). By jointly learning that mapping τ and the first-order state transition model T_Y that minimise the two loss functions $\mathcal{L}_{\text{Markov}}$ and $\mathcal{L}_{\text{Behaviour}}$, the BunDLe-Net architecture learns low-dimensional Markovian representations of those aspects of the high-dimensional neuronal state trajectories that are relevant for a given behavioural context.

We remark that BunDLe-Net is a generic architecture in the sense that each of its modules (the mapping τ, the state transition model T_Y, and the prediction model for the behaviour)

can be realised by whatever models, e.g., linear or non-linear mappings which may be realised via (deep) neuronal networks or other modelling techniques, are most suitable for a certain type of neuronal data. The BunDLe-Net architecture is available as a *Python* toolbox at https://github.com/akshey-kumar/BunDLe-Net.

In the following, we compare the BunDLe-Net architecture with other state-of-the-art neuronal manifold learning techniques on calcium imaging data recorded in the nematode *C. elegans* [11] and demonstrate its ability to uncover intricate relationships between neuronal activity patterns and behaviour that are not revealed by competing techniques.

2 Results

Here, we demonstrate how BunDLe-Net preserves vital information about behavioural dynamics while simultaneously enabling visually interpretable insights into the data. We start with a quantitative evaluation of BunDLe-Net and compare it with existing state-of-the-art neuronal manifold learning techniques. We then examine the visual interpretability of the embeddings of BunDLe-Net and competing algorithms. To ensure the robustness of our findings, we apply BunDLe-Net to five different worms and analyse the consistency of the embeddings in terms of their topology. The results highlight the generalisation abilities of BunDLe-Net, revealing similar patterns while maintaining individual differences across recordings. Finally, we show that BunDLe-Net is capable of embedding behaviours in distinct motifs based on the neuronal basis of the behaviour and its dynamics.

2.1 Description of data

We apply BunDLe-Net to calcium-imaging whole brain data from the nematode *C. elegans* from the work by Kato et al. [11]. This dataset is ideal for demonstrating the capabilities of BunDLe-Net due to its high-dimensional neuronal recordings labelled with motor behaviour[1], multiple animal recordings, eight different behavioural states, and multiple repetitions of behavioural states over time. It includes time-series recordings of neuronal activation from five worms with human-annotated behaviours for each time frame. The recordings consist of approximately 2500-3500 time samples, spanning around 18 minutes (sampled at ~ 2.9 Hz) in which around 100 - 200 neurons are recorded. A low-pass filter with a cut-off frequency of 0.07 Hz is applied to mitigate high-frequency noise in the raw neuronal traces. Not all recorded neurons could be identified; hence, only a subset is labelled for each worm, with different yet overlapping subsets identified across worms. The human-annotated behaviours B denote the motor state of the worm at a given instant of time and can take on one of eight states: forward, slowing, dorsal turn, ventral turn, no-state, sustained reversal, reversal-1, and reversal-2.

2.2 Quantitative evaluation against competing methods

We evaluate a latent space representation based on how well it preserves behavioural and dynamical information. To estimate the *behavioural information* of an embedding, we train a simple

[1]The motor behavioural labels were inferred from the activity of the neurons AVAR, AVAL, SMDVR, SMDVL, SMDDR, SMDDL, RIBR, RIBL while the worms were immobilised. Hence, we removed these neurons from the dataset to ensure we are not inferring behaviours directly from these neurons.

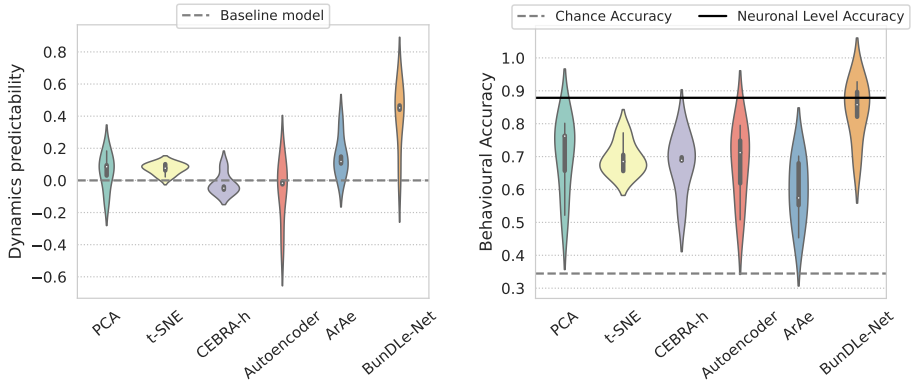

Figure 2: Quantitative comparison of BunDLe-Net with other commonly-used manifold learning techniques. (left) Evaluation of how well dynamical information is preserved in the embedding. The dashed line represents a baseline autoregressor which copies its input to the output. (right) Evaluation with respect to how well behavioural information is preserved. The dashed line represents the chance decoding accuracy, estimated by randomly shuffling the behavioural labels. The solid line represents the behaviour decoding accuracy from the raw neuronal traces.

feed-forward neural network in a supervised setting[2] to predict behaviour from the embedding. The decoding accuracy is then used as a metric for the information content about B in the embedding, with the decoding accuracy obtained on the raw, high-dimensional neuronal traces serving as the baseline. To evaluate the *dynamical information* in the embedding, we train an ANN autoregressor to predict Y_{t+1} from Y_t. The mean squared error between the predicted and true Y_{t+1} is estimated. From this, we compute a predictability metric for the dynamics, defined as $1 - \text{MSE}_m/\text{MSE}_{io}$, where MSE_m is the mean squared error of the model, and MSE_{io} is the mean squared error of a trivial autoregressor that copies its input to the output. We trained all evaluation models on a training set of the embedded data and performed the evaluation on a held-out test set to prevent overfitting (for more details, see Model validation in Section 4.3).

With the stage for evaluation set, we compare BunDLe-Net with other algorithms that are commonly used to learn high-level representations in the field of neuroscience such as PCA, t-SNE, autoencoder, an ANN autoregressor with an autoencoder architecture (ArAe)[3] and CEBRA-hybrid[4]. A description of these methods can be found in Section 4.4. All embedding spaces were chosen to be 3-dimensional for ease of comparison across algorithms and visualisation purposes.

Figure 2 presents the outcomes of our quantitative comparison, showcasing dynamical and behavioural prediction metrics in the left and right panels, respectively. Each panel depicts the

[2]We use a simple architecture consisting of a single linear layer since it already demonstrated a high decoding accuracy (~ 0.94) on the raw neuronal traces. Hence, more complex models are not required to evaluate the embeddings.

[3]The ArAe would preserve dynamical information and embed it in a lower dimensional space due to the autoencoder architecture.

[4]Note that CEBRA as an algorithm was designed for continuous-valued behaviours. We cast our categorical behaviour (int) into a continuous behaviour (floating-point) and ran CEBRA on it.

predictability metric on the y-axis and the manifold learning technique on the x-axis, while the violin plots portray the metric's distribution across all five worm datasets. The substantial variability across these plots underscores the diverse behavioural and dynamical attributes inherent in the dataset of each worm. For the dynamics evaluations, we compare all the models to a baseline model, which simply copies the input Y_t to the output as the predicted value for Y_{t+1}. For the behaviour evaluation, we compare with a chance level behavioural decoding accuracy obtained by randomly shuffling the behavioural labels. We also compare it with the behavioural decoding accuracy from the raw neuronal traces.

Turning to the results, we see that BunDLe-Net outperforms all other methods, including the state-of-the-art CEBRA, by a large margin. In the left panel, unsupervised methods like PCA, t-SNE, and the autoencoder show limited improvement over the baseline in predicting dynamics. Since they try to preserve maximum variance in the data in a low-dimensional space, they neglect to preserve minor details that may be crucial in determining future time dynamics. CEBRA-hybrid, which also takes temporal information into consideration, does not perform better than the baseline model. The autoregressive-autoencoder, which seeks to reconstruct X_{t+1} from X_t, preserves some dynamical information and is seen to outperform PCA, t-SNE, and the autoencoder. Nonetheless, ArAe's reconstruction of the entire neuronal state at time $t+1$ can lead to irrelevant details persisting in latent space embedding. In contrast, BunDLe-Net's design focuses exclusively on retaining information pertinent to the latent space state at time $t+1$, which results in a markedly superior performance even compared to ArAe.

Shifting our attention to the right panel, all models surpass chance-level behaviour decoding accuracy. Notably, both CEBRA-h and the unsupervised methods (PCA, t-SNE, autoencoder) exhibit roughly the same performance on average. Despite this, their average decoding accuracy remains notably lower than neuronal-level decoding accuracy, indicating an inability to capture behavioural information at the neuronal level completely. Although ArAe worked slightly better at preserving dynamical information, it falls short in preserving behavioural information. This suggests that unsupervised preservation of dynamical attributes alone does not suffice for constructing behaviourally relevant models. In this regard, BunDLe-Net stands out by retaining all behavioural information, as originally intended. On average, it even rivals the decoding performance achieved with raw neuronal data.

Of particular interest is the comparison between CEBRA-h and BunDLe-Net in terms of their respective performances. Despite incorporating behavioural information in addition to dynamics, CEBRA-h demonstrates only marginal improvements over other models. In contrast, BunDLe-Net rises above all other methods, excelling in both behavioural and dynamical metrics. This highlights BunDLe-Net's proficiency in effectively retaining crucial neuronal-level information relevant to behaviour analysis and modelling. For further evaluation of behavioural and dynamical performance of BunDLe-Net's embedding, please refer to Appendix A.

2.3 Visual interpretability of embeddings

In this section, we analyse the embeddings of BunDLe-Net and other competing neuronal manifold learning techniques. We visualise the embeddings of the Worm-1 in 3D and evaluate them qualitatively based on their structure and interpretability. We generalise the insights to all worms in the next section. Figure 3 shows the embeddings of Worm-1 by a) PCA, b) t-SNE, c) Autoencoder, d) Autoregressor-Autoencoder (ArAe), e) CEBRA-hybrid, and f) BunDLe-Net. In a), b) and c), we observe a noticeable drift in the PCA, t-SNE, and autoencoder embeddings. This

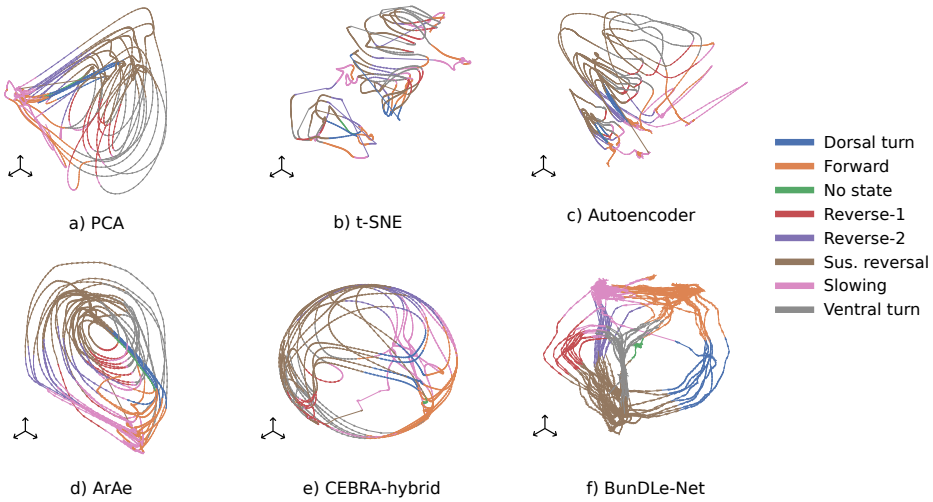

Figure 3: Neuronal manifolds learnt by various algorithms viz. a) PCA b) t-SNE c) Autoencoder d) Autoregressor-Autoencoder (ArAe) e) CEBRA-hybrid f) BunDLe-Net.

drift drags out the dynamics in time, which is undesirable since we are searching for consistent mappings independent of time. The drift is also seen to obscure the recurrent nature of the dynamics to a large extent in b). The source of this drift could be a calcium imaging artefact or some neuronal dynamics irrelevant to our behaviour of interest. Since these models aim to preserve maximum variance for full-state reconstruction, they inadvertently embed the drift.

In contrast, in Figure 3 d), e), f), we see that this drift is largely absent, and the recurrent dynamics are more evident. These models have a common characteristic: they consider dynamics without attempting to reconstruct the entire neuronal state. Among the three methods shown, ArAe is unsupervised, while CEBRA and BunDLe-Net take behaviour into account. In both d) and e), we observe reasonably separated behaviours with minor trajectory overlaps. However, both embeddings demonstrate high variance *within* a trajectory of a given behaviour. In contrast, BunDLe-Net produces compact bundles that are well-separated from one another. The variance is low within each bundle, while a high variance is observed between different bundles. Consequently, BunDLe-Net's embedding exhibits distinct behavioural trajectories that are well-separated and along which the dynamics recur in an orbit-like fashion.

Additionally, in e), we observe that CEBRA-h tends to embed the neuronal activity on the surface of a sphere, which may be an artefact resulting from the contrastive learning paradigm. As a consequence, trajectories may be forced to intersect at certain points. Such intersection points are generally undesirable because they introduce ambiguity about the future trajectory. Ideally, intersection points should only occur when there is genuinely no information available about the subsequent behavioural trajectory.

In stark contrast, BunDLe-Net's trajectories demonstrate a markedly different pattern, characterised by high compactness and sparse intersections. Figure 3 f) reveals precisely three intersection points: sustained reversal ●, ventral turn ●, and forward ●. (See supplementary material

https://github.com/akshey-kumar/BunDLe-Net/tree/main/figures/rotation_comparable_embeddings for rotating 3-D plots). These intersections and bifurcations could be interpreted as instances where BunDLe-Net encountered a lack of information about future trajectories.

2.4 Consistency of neuronal manifolds across worms

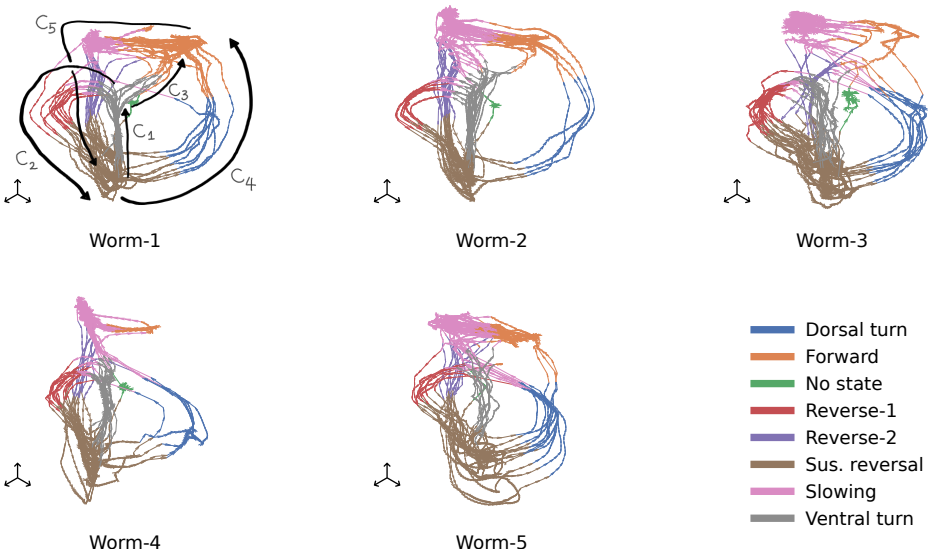

Figure 4: BunDLe-Net embeddings on five different *C. elegans* worm datasets which include neuronal recordings and behavioural labels. 3-D animations are available in the supplementary section.

Here, we apply BunDLe-Net to all five worms in the dataset to visually compare the embeddings regarding their consistency and/or any differences that arise across worms.

To produce comparable embeddings[5], we first trained a model on each worm separately. We then extracted the T_Y layer and behaviour predictor layer from the model with the least loss (Worm-1, in this case). We then trained fresh models on each worm, with the chosen T_Y and behaviour predictor layers from Worm-1 frozen in, until the losses converged. Thus the new models would only have to learn the mapping τ for each worm while the other layers remained unchanged throughout the learning process. Notably, this approach was feasible despite recording different neurons from each worm. By adopting this strategy, we ensured consistent geometries across the worms, allowing us to effectively compare differences in topology, should they be present.

The embeddings are illustrated in Figure 4. A latent dimension of three was again chosen for ease of visualisation, and can also be justified by a graph-theoretical argument detailed

[5]We could also simply fit separate models on each worm's data, as was done for the evaluation in Figure 2. Due to differing initialisations of BunDLe-Net this would result in visually different embeddings. These embeddings however, share the same topology independent of the initialisation. For ease of visual comparison between embeddings, we adopt the above procedure to have latent spaces that can be mapped to one another.

in Section 4.3. Examining Figure 4, we observe a branching structure in the trajectories of all the worms. For now, let us consider Worm-1. The dynamics exhibit bundling of several segments, leading to recurring patterns along these bundles. Within each branch, the dynamics are predominantly deterministic, while probabilistic *decisions* occur only at specific bifurcation points in the trajectories. We disregard bundles consisting of fewer than one or two segments and identify five prominent bundles in Worm-1, which can be described as follows,

(C_1) : sustained reversal ● → ventral turn ●

(C_2) : ventral turn ● → slowing ● → reversal-1 ● → sustained reversal ●

(C_3) : ventral turn ● → forward ●

(C_4) : sustained reversal ● → dorsal turn ● → forward ●

(C_5) : forward ● → slowing ● → reversal-2 ● → sustained reversal ●

These five motifs define the generic building blocks of the neuronal manifold in the sense that the neuronal trajectories are almost deterministic within each motif, and probabilistic bifurcations occur at the transitions between motifs. As can be readily checked in Figure 4, these building blocks are highly consistent across worms, with similar behavioural motifs emerging across all worms. For example, motif C_2 is consistently present in the embeddings of all worms, forming a loop (● → ● → ● → ●). The same holds true for motifs C_1 and C_5. However, motif C_4 is not present in all worms and is notably absent in Worm-4. Instead, both Worm-4 and Worm-5 exhibit a slightly different motif (sustained reversal ● → dorsal turn ● → slowing ●). This variation in motifs may be due to the recording times, which may have been too short to capture all possible transitions for a given animal.

It is noteworthy that even though the individual worm recordings do not share an identical subset of neurons, the embeddings share a basic topological structure with only minor variations in transitions and bifurcation points. These results demonstrate consistency in the embeddings across worms while preserving individuality in the behavioural dynamics in each worm and recording session.

2.5 Embedding of states in distinct behavioural motifs

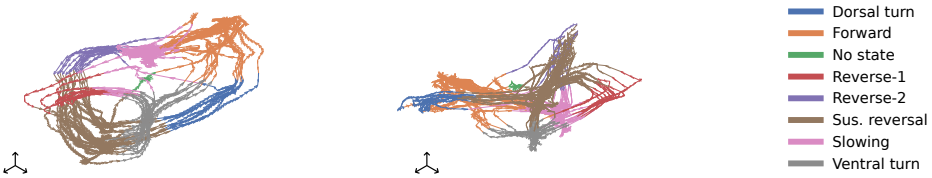

Figure 5: (left) BunDLe-Net trajectory of Worm-1 highlighting the embedding of slowing behaviour (in pink) within two distinct bundles or behavioural motifs. (right) In contrast, the sustained reversal state (in brown) is represented by a single intersection point on the right side.

Behaviour can be modelled at different levels of granularity. In the present data set, the worms' behaviour is described in terms of high-level behavioural patterns such as forward and reversal movements. Alternatively, one could analyse the angular positions and velocities of the various

segments of the worms' bodies, resulting in a more fine-grained representation. Both fine-grained and coarse-grained models hold value in specific contexts. However, it is crucial to maintain consistency within a model's state space to describe the dynamics accurately. If we utilise a model to understand fine-grained elements but only have access to coarse-grained information, the resulting model will be incomplete or inconsistent in the sense that it lacks the essential information required to predict features of the behavioural dynamics at the desired level of granularity. Here, we demonstrate how BunDLe-Net adeptly handles the coarse-graining of data while still preserving the crucial distinctions between states that are instrumental in explaining the overall dynamics.

We present the discovery of two distinct behavioural states with identical labels based on BunDLe-Net's neuronal embedding concerning the given set of behaviours. Consider branch $\boldsymbol{C_2}$ ($\rightarrow \rightarrow \rightarrow$) and $\boldsymbol{C_5}$ ($\rightarrow \rightarrow \rightarrow \rightarrow$) of the trajectory in Figure 5. The *slowing* behaviour (in pink) occurs in both these branches, i.e., they are represented distinctly in the latent space and are not fused together even though they have been assigned the same behavioural label. Branch $\boldsymbol{C_2}$ has a much shorter *slowing* segment than branch $\boldsymbol{C_5}$. We name the new behavioural states corresponding to $\boldsymbol{C_2}$ and $\boldsymbol{C_5}$ as *slowing 1* and *slowing 2*, respectively. These different types of slowing movements are embedded in distinct behavioural motifs since they differ in their neuronal realisation and their relevance for the model dynamics, i.e., one would predict different future trajectories depending on whether the state is *slowing 1* or *slowing 2*. We note that this is not the case for other behavioural states, e.g., the sustained reversal (in brown) for which all trajectories form one coherent bundle in the embedding. This implies that in the behavioural state of a sustained reversal BunDLe-Net found no information at the neuronal level to predict whether a dorsal or ventral turn is more likely to occur next. In summary, BunDLe-Net can maintain distinct representations or fuse trajectories depending on whether dynamical information about future behaviours is present. Accordingly, if provided with a set of behaviours that are not consistent or complete for the construction of a full dynamical model, BunDLe-Net can discover extra distinctions or *states* that complete this set of behaviours, provided this information is present in the neuronal level.

3 Discussion

We have demonstrated the superiority of BunDLe-Net to other neuronal manifold learning techniques on calcium imaging data recorded in *C. elegans*. However, BunDLe-Net can easily be extended to other imaging modalities and model organisms by adapting its learning modules (for the latent embedding function τ, the state transition model T_Y, and the behavioural decoding layer) while maintaining the overall structure shown in Figure 1. As such, BunDLe-Net is not one algorithm but a generic architecture for learning consistent state representations from neuronal data based on simple but vital principles. In the following, we further elaborate on the relevance of these principles for neuronal manifold learning.

On a fundamental level, the concept of a neuronal manifold can be interpreted as a scientific discovery that sheds new light on how large numbers of neurons coordinate their activities to represent information, implement computations, and generate behaviour. In this view, the goal of neuronal manifold learning techniques is to reveal the true, intrinsic structure of the neuronal manifold from empirical data. Alternatively, neuronal manifold learning algorithms can be interpreted as data compression and visualisation techniques. In this view, the particular shape of a neuronal manifold results from a model-based dimensionality-reduction technique that

attempts to preserve certain data properties. Notably, these two viewpoints are not mutually exclusive, i.e., the observed shape of the neuronal manifold may be influenced by its intrinsic structure as well as by the particularities of the dimensionality reduction technique.

Indeed, our results in Figure 3 show substantial qualitative differences in the manifolds across various learning techniques, indicating that different model assumptions inherent to the various algorithms influence the shapes of the learned manifolds. On the other hand, the results obtained by BunDLe-Net shown in Figure 4 demonstrate that highly consistent manifolds can be learned across multiple animals, supporting the concept of an intrinsic structure of the neuronal manifold.

Remarkably, BunDLe-Net achieves this consistency despite only 22 out of more than 100 neurons per animal being shared across the five data sets. We attribute this ability to reconstruct consistent manifolds to the time-delayed embedding of the neuronal dynamics for learning the latent dynamics (cf. Section 4.1), which due to Taken's theorem [12] allows the reconstruction of a Markovian representation of a dynamical system (i.e., the neuronal dynamics on the manifold) regardless of the specific observation function (i.e., the recorded neurons for each worm). We note that the number of time lags that need to be considered in this embedding is determined in BunDLe-Net by minimising the Markovian loss function $\mathcal{L}_{\text{Markov}}$, i.e., the number of time lags is increased until no further decrease in the loss function is observed.

Together with the constraint that the behavioural information must be preserved, BunDLe-Net's ability to learn a Markovian latent embedding results in almost deterministic trajectories that only exhibit a high degree of randomness at a discrete number of bifurcation points (cf. Section 2.3 and Figure 4). This distinction in the neuronal dynamics between periods of high certainty with apparently random behaviour at a discrete number of bifurcation points renders the neuronal manifold of *C. elegans* particularly interesting. Specifically, we interpret the almost deterministic trajectory bundles as the basic building blocks of the neuronal manifold that are fused together at the bifurcation points to create the manifold's intrinsic structure.

The bifurcations act as decision points regarding the worm's future behaviour. However, it is presently unclear how *C. elegans* makes these decisions. In general, the randomness in the bifurcation points could be due to intrinsic randomness in the neuronal activity or due to latent, unobserved neurons, i.e., observing these neurons could disentangle the bifurcation points and again result in deterministic trajectories. However, BunDLe-Net's ability to learn Markovian representations would disentangle the bifurcation points if such information were present in the time delay embeddings of the neuronal dynamics. Because this is not the case, our empirical results align with an interpretation in which the randomness in the bifurcation points is intrinsic neuronal noise. However, we remark that such randomness might be overwritten by external stimuli, which were not part of the experimental design.

Regardless of the nature of the noise in the bifurcation points, the learned neuronal manifolds reveal the behavioural flexibility of *C. elegans* in the context of its neuronal dynamics. In particular, they reveal when, i.e., at which points on the neuronal manifold, *C. elegans* makes decisions about its future behaviour. As such, we predict that external perturbations of the neuronal activity, e.g., by optogenetic stimulation, are most effective when applied at times when the neuronal state is in one of the bifurcation points. Conversely, we hypothesise that the neuronal dynamics are more robust against external perturbations if these are applied when the neuronal dynamics follow one of the highly deterministic trajectory bundles. To generalise from this argument, we consider neuronal manifold learning algorithms in general and BunDLe-Net in particular to be of extraordinary utility in neuroscience because these methods allow us to

make empirically testable predictions on how large-scale neuronal dynamics are coordinated to generate behavioural flexibility.

To conclude this article, we outline several potential extensions of BunDLe-Net. First, we note that we have only presented the application of BunDLe-Net to discrete behaviours. Extensions to continuous behaviours can be implemented by adapting the behavioural prediction layer or, in a less elegant fashion, by discretising continuous behaviours. Second, it would be interesting to consider the extension of BunDLe-Net to multiple non-mutually exclusive behaviours to study how large-scale neuronal activity coordinates multi-dimensional behaviours. Naturally, this approach could be extended to include stimuli to study how external information is encoded in neuronal manifolds and translated into behaviour. Each of these changes would merely require adapting the behavioural prediction layer. Regarding the learning module for the latent embedding, we note the growing body of literature on the topic of (causal) representation learning. Representation learning addresses the problem of learning high-level (causal) variables from low-level observations [13, 14]; a topic with potentially rich synergies with neuronal manifold learning that are yet to be explored.

4 Methods

In this section, we first provide further information on the theoretical principles that motivate BunDLe-Net. Subsequently, we elaborate on the architectural framework that arises from these principles. We then proceed to provide a comprehensive overview of BunDLe-Net's implementation, encompassing the learning modules and the details of the training process. Finally, we present the competing methods that serve as benchmarks for evaluating the performance of BunDLe-Net.

4.1 Theoretical principle

BunDLe-Net employs a fundamental theoretical principle to embed neuronal data with respect to a given set of behaviours. The core idea is to ensure that the resulting embedding Y contains all information about the dynamics and behaviour that is present at the neuronal-level X. To elucidate this concept, consider the diagram in Figure 6, where T_X denotes a transition model at the X level. For illustrative purposes, we presently assume that the X level is Markov, but will later relax this assumption. The embedding Y is obtained by applying a function τ on the X level. Generally, the resulting transition model at the Y level may not be Markov, implying that Y_t might not fully capture the information about Y_{t+1} present in the system, either at the X level and/or in the past states Y_{t-n}, where $n \in \mathbb{Z}^+$. Such an embedding would be of limited use since one might need to refer back to the X level to answer certain questions about the Y level.

$$\begin{array}{ccc} X_t & \xrightarrow{\mathbf{T}_X} & X_{t+1} \\ \downarrow \tau & & \downarrow \tau \\ Y_t & \xrightarrow{\mathbf{T}_Y} & Y_{t+1} \end{array}$$

Figure 6: Commutativity diagram where the X and Y level signify the neuronal and latent space dynamics respectively.

To ensure a more comprehensive and self-contained embedding, we aim for Y to be Markov and independent of the X level. This requires the diagram (Figure 6) to commute, i..e. it should not make a difference if we first time-evolve and then transform with τ, or the other way round. Put in terms of conditional independence, our requirement takes the form $Y_{t+1} \perp X_t | Y_t$, meaning that knowledge of X_t provides no additional information about Y_{t+1} beyond what is already known from Y_t. In this way, the dynamics at the Y level are self-contained and *sealed-off* from the details at X level. This is what makes our embedding so useful and interpretable: our embedding has all the relevant information from the X level, enabling it to be viewed as a distinct and meaningful dynamical process in its own right.

Non-Markovian neuronal dynamics To handle non-Markov neuronal dynamics at X, we consider time windows that include the previous n time steps, i.e., $(X_t, ..., X_{t-n})$ as input to our model. By choosing a large enough value for n, we can ensure that the resulting process becomes Markov [12], allowing us to model it in the same way as described above. Note that while earlier we were mapping a single time slice to a point in latent space, now we are mapping an entire time window of length n to a single point in latent space. Such a transformation does not merely coarse-grain over the neuronal or *spatial* level of granularity but also over the *temporal* domain of patterns.

Learning meaningful embeddings While the requirement of a Markov embedding may be very useful in terms of elegance and interpretability, it is not sufficient to ensure meaningful embeddings. For example, consider a transformation τ that uniformly maps every neuronal state to a constant. In this scenario, the resultant process would exhibit Markov dynamics as a single-state process. However, such an embedding fails to yield any meaningful insights regarding the underlying dynamics or behaviour. Remarkably, for BunDLe-Net, such a process would yield a perfect $\mathcal{L}_{\text{Markov}}$ loss, irrespective of the input data.

An additional requirement must be imposed to avoid such *trivial* embeddings. We demand that the behaviour B can be decoded from the embedding, thereby preventing the transformation from reducing everything to a mere constant. By upholding this crucial condition, we preserve the behavioural intricacies that render the embedding purposeful and informative, aligning with the ideals espoused by Krakauer et al. [10].

4.2 BunDLe-Net architecture

Here, we explain how the BunDLe-Net's architecture in Figure 1 arises from the commutativity diagram of Figure 6. The upper and lower arms in the architecture correspond to the possible paths from X_t to Y_{t+1} in the commutativity diagram. The lower arm in the architecture involves first coarse-graining X_t, followed by implementing a transition model on the Y-level. In practice, the transition model outputs ΔY_t from which Y_{t+1} is estimated as $Y_t + \Delta Y_t$. Since the transition model T_Y outputs Y_{t+1} with only Y_t as input, the Y-level is first-order Markov by construction. The upper arm of BunDLe-Net coarse-grains the time-evolved X_{t+1}[6]. Both arms result in estimates of Y_{t+1} which we distinguish by upper indices Y^L_{t+1} and Y^U_{t+1}. We add a mean-squared error term to our loss function $\mathcal{L}_{\text{Markov}}$ that forces Y^L_{t+1} and Y^U_{t+1} to be equal, thus ensuring that

[6]Since we have time-series data, we need not learn T_X of the commutativity diagram, but simply feed X_{t+1} directly into the network.

our requirement of commutativity in Figure 6 is satisfied,

$$\mathcal{L}_{\text{Markov}}(Y^U_{t+1}, Y^L_{t+1}) = \left\| Y^U_{t+1} - Y^L_{t+1} \right\|^2.$$

The estimated Y_{t+1} is then passed through a predictor layer which learns to output the behaviour B_{t+1} given Y_{t+1}. Correspondingly, we add a term $\mathcal{L}_{\text{Behaviour}}$ to our loss function, which forces the predicted behaviour to match the true behaviour. This ensures that Y_t contains the same amount of information about B_t as X_t. Here, we use the cross-entropy loss where $B^{(j)}_{t+1}$ represents the j-th component of a one-hot encoded label vector of B_{t+1}, and $\tilde{B}^j_{t+1}$ is the softmax output of the predicted $\tilde{B}_{t+1}$.

$$\mathcal{L}_{\text{Behaviour}}(B_{t+1}, \tilde{B}_{t+1}) = -\sum_{j=1}^{8} B^{(j)}_{t+1} \log(\tilde{B}^{(j)}_{t+1}).$$

Both terms are weighted by a hyper-parameter γ and the loss function is given as,

$$\mathcal{L} = (1-\gamma)\mathcal{L}_{\text{Markov}} + \gamma \mathcal{L}_{\text{Behaviour}}.$$

All the layers in BunDLe-Net are learned simultaneously, and both loss terms ensure that the learned τ and T_Y preserve information about the behavioural dynamics. An open-source Python implementation of the BunDLe-Net architecture is available at https://github.com/akshey-kumar/BunDLe-Net.

4.3 Learning modules

Architecture parameters The τ layer (encoder) of our network consists of a series of ReLU layers [15], followed by a normalisation layer. An encoder of identical architecture is used later in the autoregressor-autoencoder (ArAe) model to facilitate comparison across models. For the *predictor* and T_Y layer, we use a single dense layer each. In the case of our dataset, this sufficed to achieve good performance. For other data sets, more complex layers may be required. For T_Y, we also add a normalisation layer so that the output remains in the scaling of the latent space learned by τ. The details of the individual layers are provided in the Python code in Appendix B.

Gaussian noise against overfitting To safeguard against overfitting of the model, we introduce Gaussian white noise in the latent space by incorporating it in the τ layer. Injecting Gaussian white noise is a well-established regularisation technique that makes the model robust to overfitting [16, 17]. Since we are working with relatively limited data in the context of artificial neural networks, guarding against overfitting becomes particularly crucial.

Latent space dimensionality We choose the dimensionality of the Y-level to be three. This is because, in 3-D, we can connect any finite number of points without the edges crossing each other. This allows for embeddings of neuronal activity in the form of trajectories with nodes and edges that do not intersect. This might not always be possible in 2-D, where one can have a constellation of data points that cannot be connected without crossings. It is possible however, to embed any arbitrary graph in three dimensions without the edges having to intersect. [18].

Intersection points are undesirable for the embedding of a dynamical process due to the ambiguity they introduce. A meaningful embedding should exhibit smooth trajectories without

self-intersections. An intersection point of two trajectories would mean that the past state at time $(t-1)$ contains additional information about the future state $(t+1)$ than the present state at (t), thus rendering the dynamics non-Markovian. Avoiding such intersections and non-Markovian dynamics enhances the interpretability of the embedded data and allows an enhanced prediction of future dynamics.

Model validation / parameter tuning To determine the optimal parameters for the model, including the number and types of layers, we use a held-out validation set on Worm-1. The neuronal and behavioural data of Worm-1 is partitioned into seven folds along the time axis, and one fold is randomly selected as the validation set from the time-ordered dataset. The remaining data forms the training set. By choosing an entire fold in the data as a validation set, we ensure that the model performs as well on unseen data. This would not be the case if we created our validation set by *iid* (independent and identically distributed) sampling due to high time correlations in the time series. After selecting the optimal model parameters through validation on Worm-1, we train models with the same parameters on the other worms. Since we only use Worm-1 for parameter tuning, if the model performs well on other worms, we can be confident that its success is not due to overfitting.

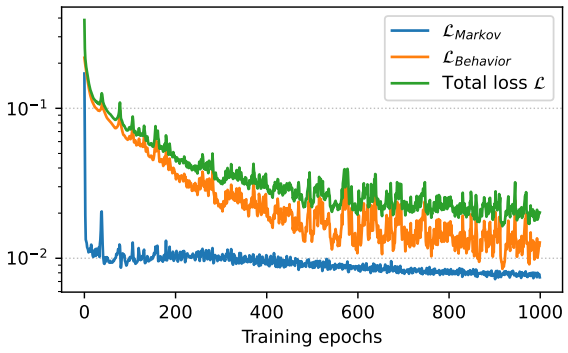

Figure 7: Markov and behavioural loss during training of BunDLe-Net on a log plot

Training details Since the neuronal data was found to be non-Markovian[7], we use time-windows of length 15 as input to BunDLe-Net. Reducing the window length decreased model performance while increasing it further had no significant effect. Training was performed with the ADAM optimiser [19] with a learning rate of 0.001 and batch size of 100. The γ parameter of BunDLe-Net was chosen to be 0.9 to ensure that $\mathcal{L}_{\text{Markov}}$ and $\mathcal{L}_{\text{Behaviour}}$ are roughly the same order of magnitude during training (see Figure 7). We trained BunDLe-Net until the losses converged.

[7] We tested for non-Markovianity using an autoregressor model and found that including multiple time steps from the past boosted the prediction performance of the model.

4.4 Description of competing methods

Here, we describe the other commonly-used neuronal manifold learning algorithms used in the comparison. All models are used to project the *C. elegans* data to a three-dimensional space for purposes of fair comparison. A full implementation of the various models, training process, and evaluation procedures can be found at https://github.com/akshey-kumar/comparison-algorithms.

PCA Principal component analysis [20] has been applied to neuronal datasets to enable visualisation and interpretation of the data. It is a linear transformation that aligns the data along the directions of maximum variance. Typically, the first three principal components are chosen and plotted in 3-D space [11]. The resulting trajectories can provide a rough perspective of the neuronal dynamics at a high level. Since this is a commonly-used method to coarse-grain data, we use PCA as our first baseline model.

t-SNE t-distributed stochastic neighbour embedding is a popular tool for visualising high-dimensional data, including neuronal data [21, 22]. It is essentially a non-linear dimensionality reduction method that tries to preserve distances between the data points.

Autoencoder Arguably, autoencoders (or some variant thereof) are currently one the most predominant method for learning low-dimensional representations of data [6, 7]. Typically, an autoencoder learns a representation by attempting to reconstruct the training data using an ANN composed of an encoder and decoder [23]. Here, we consider the deterministic vanilla autoencoder with a deep encoder and decoder. The depth of the layers, number of neurons, and other training-related hyperparameters were tuned to obtain reasonably optimal performance.

Autoregressor-autoencoder (ArAe) An autoregressor is generally used on time-series data to predict the future state based on the past. Here we implemented an autoregressor with an ANN with an autoencoder-like architecture[8] and refer to it as ArAe. Such architectures have been used before to learn low-dimensional representations of time-series data [6, 24]. We implement our ArAe as an ANN with a deep encoder and decoder that tries to predict X_{t+1} given X_t as input with Y_t as the latent space as seen in Figure 9.

CEBRA CEBRA [9] is a state-of-the-art neuronal manifold technique. It uses contrastive learning to optimise the encoding of data by maximising the similarity between related samples and minimising the similarity between unrelated samples. The algorithm employs neural network encoders and a similarity measure to optimise the embeddings based on user-defined or time-only labels. In our experiments, we used CEBRA-hybrid, which takes both behaviour and time dynamics into account for the embedding.

5 Acknowledgements

We would like to thank Sebastian Tschiatschek and Simon Rittel for enriching discussions at the Causal Representation Workshop 2021 which was hosted at the Faculty of Computer Science,

[8] We use an autoencoder architecture since an autoregressor, in general, need not map the data to a low-dimensional space. Hence, we use an encoder to obtain an embedding that can be compared with the other methods

University of Vienna. We would also like to thank Manuel Zimmer and his lab, especially Kerem Uzel, for collaborating with us and providing the neuronal calcium imaging data from *C. elegans*. We also thank Verity Cook for discussions about the illustrations and figures.

References

[1] Jason ND Kerr and Winfried Denk. Imaging in vivo: watching the brain in action. *Nature Reviews Neuroscience*, 9(3):195–205, 2008.

[2] Misha B Ahrens, Michael B Orger, Drew N Robson, Jennifer M Li, and Philipp J Keller. Whole-brain functional imaging at cellular resolution using light-sheet microscopy. *Nature Methods*, 10(5):413–420, 2013.

[3] Mark M Churchland, John P Cunningham, Matthew T Kaufman, Justin D Foster, Paul Nuyujukian, Stephen I Ryu, and Krishna V Shenoy. Neural population dynamics during reaching. *Nature*, 487(7405):51–56, 2012.

[4] Carsen Stringer, Marius Pachitariu, Nicholas Steinmetz, Matteo Carandini, and Kenneth D Harris. High-dimensional geometry of population responses in visual cortex. *Nature*, 571 (7765):361–365, 2019.

[5] Rufus Mitchell-Heggs, Seigfred Prado, Giuseppe P Gava, Mary Ann Go, and Simon R Schultz. Neural manifold analysis of brain circuit dynamics in health and disease. *Journal of Computational Neuroscience*, 51(1):1–21, 2023.

[6] N. Wahlström, Schön T. B., and M. P. Deisenroth. From pixels to torques: Policy learning with deep dynamical models. *Deep Learning Workshop at the 32nd International Conference on Machine Learning ICML 2015*, 2015.

[7] Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/a1afc58c6ca9540d057299ec3016d726-Paper.pdf.

[8] Chethan Pandarinath, Daniel J. O'Shea, Jasmine Collins, Rafal Jozefowicz, Sergey D. Stavisky, Jonathan C. Kao, Eric M. Trautmann, Matthew T. Kaufman, Stephen I. Ryu, Leigh R. Hochberg, Jaimie M. Henderson, Krishna V. Shenoy, L. F. Abbott, and David Sussillo. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature Methods*, 15(10):805–815, September 2018. doi: 10.1038/s41592-018-0109-9. URL https://doi.org/10.1038/s41592-018-0109-9.

[9] Steffen Schneider, Jin Hwa Lee, and Mackenzie Weygandt Mathis. Learnable latent embeddings for joint behavioural and neural analysis. *Nature*, 617(7960):360–368, May 2023. doi: 10.1038/s41586-023-06031-6.

[10] John W Krakauer, Asif A Ghazanfar, Alex Gomez-Marin, Malcolm A MacIver, and David Poeppel. Neuroscience needs behavior: correcting a reductionist bias. *Neuron*, 93(3): 480–490, 2017.

[11] Saul Kato, Harris S. Kaplan, Tina Schrödel, Susanne Skora, Theodore H. Lindsay, Eviatar Yemini, Shawn Lockery, and Manuel Zimmer. Global brain dynamics embed the motor

command sequence of caenorhabditis elegans. *Cell*, 163(3):656–669, October 2015. doi: 10.1016/j.cell.2015.09.034. URL https://doi.org/10.1016/j.cell.2015.09.034.

[12] Floris Takens. Detecting strange attractors in turbulence. In David Rand and Lai-Sang Young, editors, *Dynamical Systems and Turbulence*, volume 898 of *Lecture Notes in Mathematics*, pages 366–381. Springer-Verlag, 1981. doi: 10.1007/BFb0091924.

[13] Thomas N. Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=H1gax6VtDB.

[14] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021. doi: 10.1109/JPROC.2021.3058954.

[15] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[16] Alexander Camuto, Matthew Willetts, Umut Simsekli, Stephen J Roberts, and Chris C Holmes. Explicit regularisation in Gaussian noise injections. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 16603–16614. Curran Associates, Inc., 2020.

[17] Richard M. Zur, Yulei Jiang, Lorenzo L. Pesce, and Karen Drukker. Noise injection for training artificial neural networks: A comparison with weight decay and early stopping. *Medical Physics*, 36(10):4810–4818, 2009. doi: https://doi.org/10.1118/1.3213517.

[18] Robert F. Cohen, Peter Eades, Tao Lin, and Frank Ruskey. Three-dimensional graph drawing. In Roberto Tamassia and Ioannis G. Tollis, editors, *Graph Drawing*, pages 1–11, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg. ISBN 978-3-540-49155-2.

[19] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. URL https://arxiv.org/abs/1412.6980.

[20] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11): 559–572, 1901. doi: 10.1080/14786440109462720.

[21] George Dimitriadis, Joana P. Neto, and Adam R. Kampff. t-SNE visualization of large-scale neural recordings. *Neural Computation*, 30(7):1750–1774, July 2018. doi: 10.1162/neco_a_01097. URL https://doi.org/10.1162/neco_a_01097.

[22] Jinke Liu and Martin Vinck. Improved visualization of high-dimensional data using the distance-of-distance transformation. *PLoS Computational Biology*, 18(12):e1010764, December 2022. doi: 10.1371/journal.pcbi.1010764. URL https://doi.org/10.1371/journal.pcbi.1010764.

[23] Mark A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37(2):233–243, 1991. doi: https://doi.org/10.1002/aic.690370209. URL https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690370209.

[24] Boyuan Chen, Kuang Huang, Sunand Raghupathi, Ishaan Chandratreya, Qiang Du, and Hod Lipson. Automated discovery of fundamental variables hidden in experimental data. *Nature*

Computational Science, 2(7):433–442, July 2022. doi: 10.1038/s43588-022-00281-6. URL https://doi.org/10.1038/s43588-022-00281-6.

A Further evaluation of BunDLe-Net's embedding

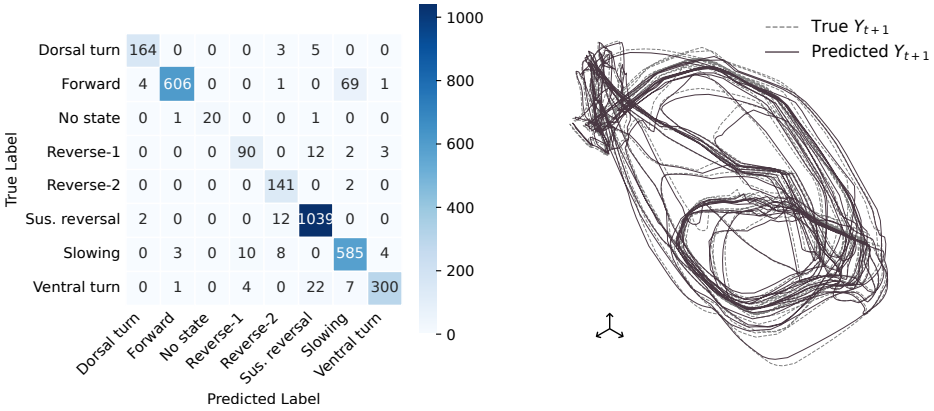

Figure 8: (left) Confusion matrix of behaviour predictions from BunDLe-Net's embedding for Worm-1 (right) True dynamics and dynamics predicted by BunDLe-Net for Worm-1.

In the following, we provide further information to build an intuition for the behavioural and dynamic prediction performance of BunDLe-Net. In Figure 8 a), we present the confusion matrix for BunDLe-Net's behavioural *prediction layer* from the ANN architecture. BunDLe-Net achieves a decoding accuracy of 94.3%, with the few decoding errors dominated by confusion of forward and slowing, two behaviours that are qualitatively similar and only quantitatively differ in the speed of the motion. To evaluate the dynamical performance of the model, we use the *transition model layer* T_Y to estimate Y_{t+1} from Y_t and compare it with the true Y_{t+1}, obtained as $\tau(X_{t+1})$. Figure 8 b) shows that the predicted dynamics indeed track the true dynamics rather well. These results indicate that the behaviour predictor and transition model within BunDLe-Net do well to preserve dynamical and behavioural information as intended.

B BunDLe-Net architecture

```
class BunDLeNet(Model):
    """Behaviour and Dynamical Learning Network (BunDLeNet) model.

    This model represents BunDLe-Net's architecture for deep
    learning and is based on the commutativity diagrams. The
    resulting model preserves information relevant to the
    behavioural dynamics

    Args:
        latent_dim (int): Dimension of the latent space.
    """
    def __init__(self, latent_dim):
        super(BunDLeNet, self).__init__()
        self.latent_dim = latent_dim
        self.tau = tf.keras.Sequential([
            layers.Flatten(),
            layers.Dense(50, activation='relu'),
            layers.Dense(30, activation='relu'),
            layers.Dense(25, activation='relu'),
            layers.Dense(10, activation='relu'),
            layers.Dense(latent_dim, activation='linear'),
            layers.Normalization(axis=-1),
            layers.GaussianNoise(0.05)
        ])
        self.T_Y = tf.keras.Sequential([
            layers.Dense(latent_dim, activation='linear'),
            layers.Normalization(axis=-1),
        ])
        self.predictor = tf.keras.Sequential([
            layers.Dense(8, activation='linear')
        ])

    def call(self, X):
        # Upper arm of commutativity diagram
        Yt1_upper = self.tau(X[:,1])
        Bt1_upper = self.predictor(Yt1_upper)

        # Lower arm of commutativity diagram
        Yt_lower = self.tau(X[:,0])
        Yt1_lower = Yt_lower + self.T_Y(Yt_lower)

        return Yt1_upper, Yt1_lower, Bt1_upper
```

B.1 BunDLe-Net loss function

```
def bccdcc_loss(yt1_upper, yt1_lower, bt1_upper, b_train_1, gamma):
    """Calculate the loss for the BunDLe Net

    Args:
        yt1_upper: Output from the upper arm of the BunDLe Net.
        yt1_lower: Output from the lower arm of the BunDLe Net.
        bt1_upper: Predicted output from the upper arm of the BunDLe Net.
        b_train_1: True output for training.
        gamma (float): Tunable weight for the DCC loss component.

    Returns:
        tuple: A tuple containing the DCC loss, behaviour loss, and total
                                                loss.
    """
    mse = tf.keras.losses.MeanSquaredError()
    scce = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True)

    DCC_loss = mse(yt1_upper, yt1_lower)
    behaviour_loss = scce(b_train_1, bt1_upper)
    total_loss = gamma*DCC_loss + (1-gamma)*behaviour_loss
    return gamma*DCC_loss, (1-gamma)*behaviour_loss, total_loss
```

C Other architectures of ANN models

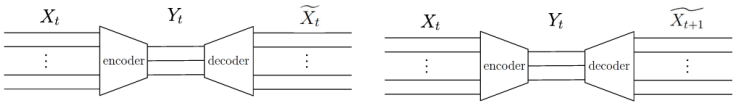

Figure 9: Architecture of Autoencoder and autoencoder-autoregressor (ArAe)

D Learning process

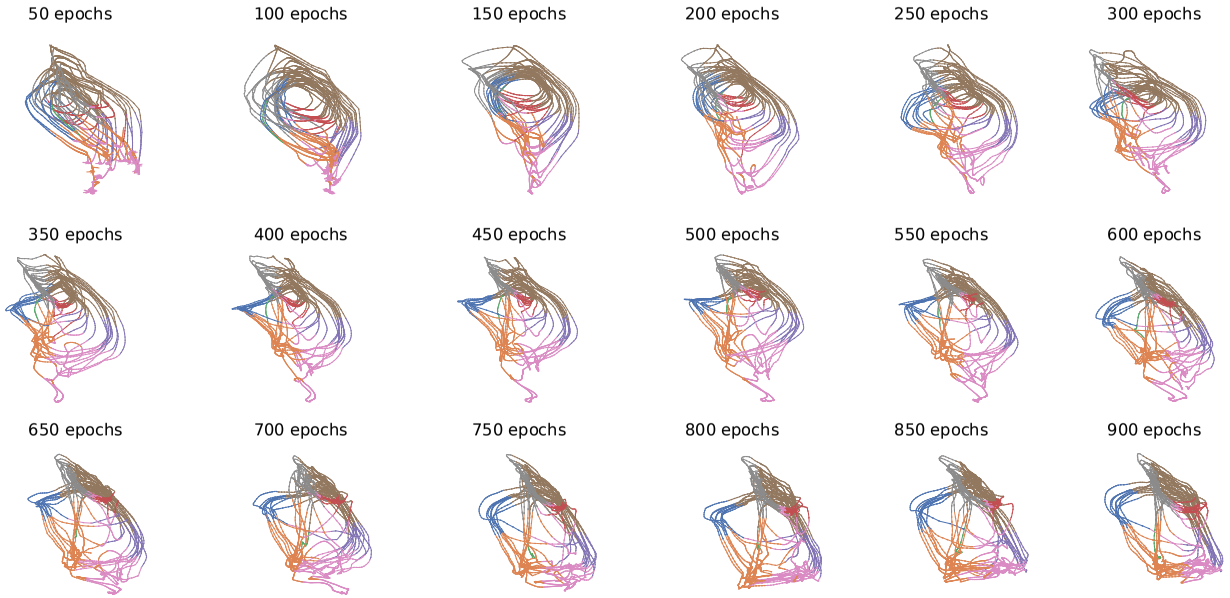

Figure 10: Visualisation of the learning process as a function of epochs

4. BunDLe-Net toolbox

In Chapter 3, we presented the algorithm BunDLe-Net and demonstrated its performance on embedding neuronal data. Here, we present a Python toolbox which includes an implementation of BunDLe-Net along with other tools to process and visualise neuronal data. This toolbox is intended to assist neuroscientists in the analysis of high-dimensional neuronal recordings and gain automated visual insights about the data.

4.1. Dependencies

BunDLe-Net is implemented using an Artificial Neural Network (ANN) architecture, which was created using TensorFlow (Abadi et al., 2015) and Keras (Chollet et al., 2018) modules. We extensively utilise NumPy (Harris et al., 2020) and SciPy (Virtanen et al., 2020) for numerical operations. To facilitate various visualisations, including embeddings and other plots, we use Matplotlib (Hunter, 2007). Given that the *C. elegans* data is stored in MATLAB files, we use the Mat73 library to load and process them. Additionally, for algorithm comparison and evaluation purposes, we rely on functionalities provided by scikit-learn (Pedregosa et al., 2011).

4.2. Features of the toolbox

To demonstrate the toolbox's capabilities, we provide a concise showcase of BunDLe-Net's functionality, covering the entire process from initial data loading and preprocessing to model training, data embedding, and subsequent visualization. A schematic of the data processing and code structure can be found in Figure A.1 in the appendix. The following code snippets, extracted from the `main.py` file, offer a glimpse into BunDLe-Net's capabilities.

```
import sys
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from sklearn.decomposition import PCA
from functions import Database, preprocess_data, prep_data, BunDLeNet,
    train_model, plotting_neuronal_behavioural, plot_latent_timeseries,
    plot_phase_space, rotating_plot

sys.path.append(r'../')
```

We begin by importing essential modules. All the functions and classes that are contributed as part of the BunDLe-Net toolbox are located in the `function.py` script. To use these

functionalities, we import them into our `main.py` script.

4.3. Loading neuronal datasets

```
### Load Data (excluding behavioural neurons) and plot
worm_num = 0
b_neurons = [
  'AVAR',
  'AVAL',
  'SMDVR',
  'SMDVL',
  'SMDDR',
  'SMDDL',
  'RIBR',
  'RIBL'
]
data = Database(data_set_no=worm_num)
data.exclude_neurons(b_neurons)
X = data.neuron_traces.T
B = data.states
state_names = ['Dorsal turn', 'Forward', 'No state', 'Reverse-1', '
    Reverse-2', 'Sustained reversal', 'Slowing', 'Ventral turn']
```

Here, we demonstrate a class specifically designed for loading standard neuronal time-series recordings, along with associated behavioural data, while enabling the selective identification of individual neurons by their designated names. The neuronal data is retrieved from `.mat` files, which is a common format for calcium imaging recordings from neuroscience research labs.

Database Class This class is responsible for loading data associated with one of the five worms within our dataset. The `data_set_no` argument specifies the worm number, ranging from 0 to 4. This class loads time-series neuronal and behavioural data, including corresponding neuronal labels. In the above code, the neuronal data, `X` is an array of shape `(t, n)`, and `B` is an array of shape `(t)`, where t is the number of time steps, and n is the number of neurons.

Excluding Neurons For the convenience of investigation, we provide a method called `exclude_neurons` that allows the exclusion of specific neurons from our analysis. For instance, in Chapter 3, certain neurons were excluded because they were directly related to behavioural inference. This exclusion capability is valuable for neuroscientists seeking to examine the influence of specific neurons or neuron groups on behavioural dynamics.

4.4. Pre-processing and data preparation

```
### Preprocess and prepare data for BundLe Net
time, X = preprocess_data(X, data.fps)
X_, B_ = prep_data(X, B, win=15)
```

`preprocess_data`: The `preprocess_data` function is responsible for implementing any required pre-processing steps on the neuronal traces `X`. In this particular case, we applied a low-pass filtering technique. If further pre-processing steps are required for different datasets, they can easily be incorporated into the `preprocess_data` function to accommodate specific data needs.

`prep_data` This function takes the pre-processed neuronal data `X` and behavioural data `B` and transforms them into a format suitable for input into the BunDLe-Net model. The parameter `win` allows us to select a window of width `win` for embedding into the latent space. This means that, apart from the capability of embedding a single time-slice of neuronal activity, we also have the option to embed an entire window of time-slices. To embed a single time-slice, set `win=1`. The `prep_data` function arranges the neuronal data `X_` in a shape `(t - win, 2, win, n)` and `B_` behavioural data in a shape `(t - win)` such that they are synchronised. In the second dimension of `X_`, we have `X[:,0,:,:]` and `X[:,1,:,:]` corresponding to X_t and X_{t+1}, respectively, where one is a time-delayed version of the other. This relation holds for all t , `X[t+1, 0, :, : ] = X[t, 1, :, : ]`.

4.5. Training BunDLe-Net on the data

```
### Deploy BunDLe Net
model = BunDLeNet(latent_dim=3)
model.build(input_shape=X_.shape)
optimizer = tf.keras.optimizers.legacy.Adam(learning_rate=0.001)

loss_array = train_model(
  X_,
  B_,
  model,
  optimizer,
  gamma=0.9,
  n_epochs=2000,
  pca_init=False,
  best_of_5_init=False
)

# Training losses vs epochs
plt.figure()
for i, label in  enumerate(["$\mathcal{L}_{{Markov}}$", "$\mathcal{L}_{{
    Behavior}}$","Total loss $\mathcal{L}$" ]):
  plt.semilogy(loss_array[:,i], label=label)
plt.legend()
plt.show()
```

First, the `BunDLeNet` model needs to be instantiated. While doing this, we need to set the hyper-parameter of the model, `latent_dim`, which corresponds to the latent dimension of the embedding space. The latent dimension is fixed to be three for graph theoretical reasons that are detailed in the methods section (4.3) of the publication in Chapter 3. Hence, for neuronal manifold learning, we advise leaving the latent dimension unchanged.

The model is then built, for which it requires the input shape `(t-win, 2, win, n)`. We use Adam optimiser (Kingma and Ba, 2014) with a fixed learning rate of 0.001.

Finally, we begin training by calling `train_model` function on the prepared neuronal `X_` and behavioural data `B_`, the optimiser, and other hyper-parameters. The `gamma` hyper-parameter is the weight that balances the effect of the behavioural and dynamics loss. For all experiments in this thesis, we have kept `gamma` fixed at 0.9 since it ensures that both losses are in the same order of magnitude. The number of epochs is chosen to be 2000, which was chosen to ensure convergence of the losses.

Initialisation schemes : We offer two novel initialisation schemes for the parameters (weights and biases) of BunDLe-Net, aside from the default randomised initialisation of TensorFlow networks (Glorot initialization).

- `pca_init=True` uses PCA initialisation of the network. It initialises the `tau` layer of BunDLe-Net in such a way that the embedding (output of `tau`) corresponds to the principal components of the input neuronal data. Even though PCA may not be the best final embedding, it may give a good starting point from where to search for better embeddings of the data. This may save time when compared with a randomised initialisation. The idea of PCA initialisation has been applied to neural networks where it was shown to improve stability and generalisation (Suzuki and Sakanashi, 2019; Ren et al., 2016). However, to the best of our knowledge, it has not yet been applied to manifold learning techniques.

- `best_of_5_init=True` uses the initialisation of the best of 5 runs (in terms of their loss functions). Whether or not BunDLe-Net converges to a good solution (with a low loss) can often strongly depend on the initialisation of the network parameters. This could be due to the highly non-linear and non-convex landscape of the loss functions, replete with local minima. To deal with this, we allow for 5 random initialisations of BunDLe-Net and train for 200 epochs. Our rationale is that if the initialisation is a good one, the loss functions should begin to converge after a few epochs of training; fine-tuning can then be done in further training. From these 5 initialisations, we then pick the one with the lowest loss. This is then used to instantiate BunDLe-Net, after which training proceeds for 2000 epochs. Thus, we use a mix of randomised trial-and-error and gradient descent to reach an optimal solution.

- If both `pca_init` and `best_of_5_init` are set to `False`, BunDLe-Net is initialised with the default TensorFlow initialiser.

After training is complete, we plot the training curves to inspect the training process. For BunDLe-Net to have successfully fitted, we expect both the losses to converge to a constant value that is close to zero.

4.6. Embedding the neuronal data

```
### Projecting into latent space
Y0_ = model.tau(X_[:,0]).numpy()
```

Following successful training of `BunDLeNet`, we use the `tau` layer of the fitted model to embed the neuronal data into the latent space.

4.7. Plotting functions

```
### Plotting latent space dynamics
plot_latent_timeseries(Y0_, B_, state_names)
plot_phase_space(Y0_, B_, state_names = state_names)
rotating_plot(Y0_, B_,filename='figures/rotation_'+ algorithm + '_worm_'+
    str(worm_num) +'.gif', state_names=state_names, legend=False)

plotting_neuronal_behavioural(X, B, state_names=state_names)
```

One of the powers of our method is that it enables neuroscientists to gain visual insights about their data. `plot_latent_timeseries` plots the three latent dimensions as a time-series with time along the x-axis and the behavioural label `B_` as the background colour. `plot_phase_space` space produces a 3-dimensional plot of the trajectory in latent space. The parameter `state_names` is meant to provide the behavioural labels so that the trajectory is coloured according to the behavioural state at that time point. This allows one to understand the connection between behaviour and neuronal activity through the embeddings. `rotating_plot` produces a rotating plot of the same and saves it as a `.gif` file.

4.8. Experimental and future developments

Future developments will focus on extending the architecture of BunDLe-Net to various settings. While the behaviour in the present datasets has been discrete, extensions would involve accommodating continuous-valued behaviours, angular data, and multimodal behavioural data within the architecture. The first experiments have been carried out and documented in Section 5.1.

Currently, all neuronal dimensions are jointly embedded into the three-dimensional latent space. By adjusting BunDLe-Net's architecture, specific neuronal components could be mapped to designated dimensions, providing interpretability to the latent space. Pertaining to this, axis decomposition experiments have been conducted, as outlined in 5.2.

Further adaptations could augment the BunDLe-Net architecture with more sophisticated layers, including convolutional layers and probabilistic layers (Kingma and Welling, 2013). This would expand the domain of BunDLe-Net into the computer vision domain and open avenues for simulating high-dimensional systems in the latent space.

5. Applications of BunDLe-Net architecture

In Chapter 3, we introduced the BunDLe-Net architecture and used it as a Neuronal Manifold Learning technique on Calcium imaging data from the roundworm *C. elegans*, with a single discrete-valued behavioural variable. This revealed an intricate structure of the underlying neuronal manifold and surpassed other state-of-the-art methods in terms of behavioural decoding performance and dynamical predictability. In this Chapter, we broaden the scope of BunDLe-Net, demonstrating its ability to generalise well both to new kinds of data and new behavioural settings, with minimal adaptation. This demonstrates that BunDLe-Net is not merely a superior algorithm but rather a versatile and flexible architecture that is easily adapted to different data modalities.

While calcium imaging is a widely used method for acquiring single-neuron data, it represents just one of many approaches. Electrophysiology techniques, capturing spiking activity in individual neurons, offer qualitatively distinct data (Wei et al., 2020). Furthermore, neuroscience experiments span various modalities, often involving the simultaneous recording of multiple behaviours or continuous-valued behaviour variables like the position of an animal.

5.1. BunDLe-Net on multimodal behavioural data

Since BunDLe-Net was originally developed and validated using the *C. elegans* calcium imaging data, our aim was to assess its performance in a markedly distinct neuronal dataset and experimental setting. This exploration not only tests the robustness of our theoretical foundations but also its ability to generalise to novel datasets. For this purpose, we have chosen the rat hippocampus dataset (Grosmark and Buzsáki, 2016) that has often been used to validate other embedding techniques (Schneider et al., 2023; Zhou and Wei, 2020).

5.1.1. Spiking neuronal vs. Calcium-imaging data

The *C. elegans* data was obtained through calcium imaging. This method uses a calcium-sensitive indicator (GCaMP), which fluoresces when the calcium concentration within a neuron changes (de Melo Reis et al., 2020). When a neuron fires, calcium ions enter the neuron, causing fluorescence, which is then detected by a microscope. In this way, it is possible to obtain time-series recordings of populations of neurons at the single-neuron resolution. An advantage of calcium imaging is that it allows a large number of neurons to be recorded simultaneously, like the whole brain imaging in *C. elegans* (Kato et al., 2015).

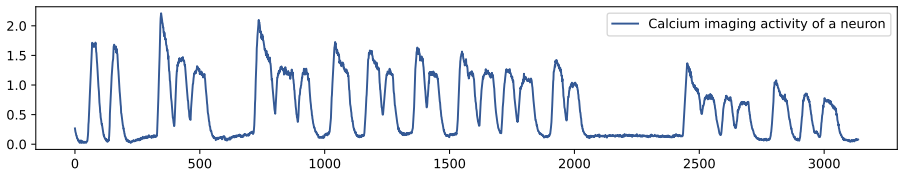

Figure 5.1.: Calcium imaging activity of a neuron from the *C. elegans* dataset (Kato et al., 2015).

However, this method has a lower signal-to-noise ratio when compared to electrophysiology data. This is because it records the spikes indirectly, and hence it results in a non-linear, low-pass filtered, and time-delayed version of neuronal activity (Wei et al., 2020). This may have implications in the behaviour decoding accuracy and introduce artefacts in understanding the time dynamics. In terms of data, calcium-imaging data consists of continuous-valued, slowly-modulating time-series data, as can be seen in Figure 5.1.

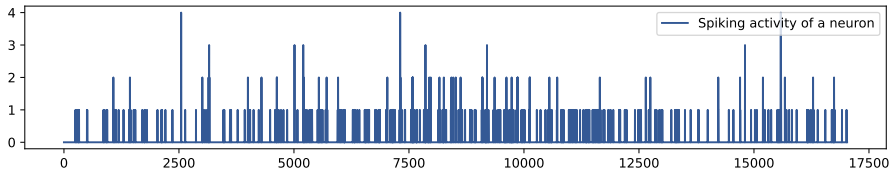

Figure 5.2.: Spiking activity of a neuron from the rat hippocampus dataset (Grosmark and Buzsáki, 2016).

Another widely used technique of recording neuronal data is through electrophysiology experiments. This method uses electrodes to directly measure the electrical activity of individual neurons, offering us information about the spiking activity of a neuron. An advantage is that this method has a better signal-to-noise ratio compared to Calcium imaging and often better temporal resolution (Wei et al., 2020). However, this method usually only records sparse subsets of neurons in specific brain regions. Consequently, the data is generally lower-dimensional and does not constitute whole-brain imaging.

In terms of data, spiking data is a time-series of spike events (discrete-valued), as can be seen in Figure 5.2. Note that this publicly available spiking data has been pre-processed, which included binning the ensemble spike activities into 25 millisecond intervals. This binning process transforms the data from a binary-valued time series (representing either a spike or no spike) into a format where each interval may contain varying numbers of spikes, such as two spikes, three spikes, and so on.

5.1.2. Challenges for BunDLe-Net in a new context

Since spiking data differs significantly from neuronal data, we would expect BunDLe-Net to face a number of challenges in this new domain. Firstly, unlike continuous-valued data

from calcium imaging, spiking data is discrete-valued. Secondly, the sampling time in relation to the rate of the activity is different, affecting the data's time resolution. Aside from these aspects, there is a crucial difference in these data modalities. While the *C. elegans* dataset is whole-brain imaging, the spiking data has been recorded from a small subset of neurons within a sub-region of the rat's brain. With whole-brain imaging, one could still argue that there is no missing information about the system or, to use the language of causality, no unobserved hidden confounders. However, when recording only a small subset of neurons, it is likely that they are substantially influenced by neurons and circuits lying beyond the scope of our dataset.

Consequently, certain aspects of the dynamics may remain inaccessible due to a lack of information, and this may render the dynamics non-Markovian. This qualitative distinction presents a significant challenge for any neuronal manifold learning technique. However, it is worth noting that BunDLe-Net incorporates past time steps into the embedding process. According to Taken's theorem (Shalizi, 2003), there may be theoretical promise for reconstructing some of this missing information.

5.1.3. Description of the data

In this study, we used a publicly available rat electrophysiology dataset from a study conducted by Grosmark et al. (Grosmark and Buzsáki, 2016). This dataset has been widely employed to assess different neural embedding techniques (Schneider et al., 2023; Zhou and Wei, 2020). It contains both spiking neuronal data and behavioural data that are simultaneously recorded as the rat navigates a 1.6-meter linear track. The neuronal activity was recorded from putative pyramidal neurons in the hippocampal CA1 area. Such recordings have been carried out on four rats (named Achilles, Gatsby, Cicero and Buddy) from between 48 - 120 neurons.

For our analysis, we use the pre-processed data sourced from (Schneider et al., 2023) to facilitate a fair comparison. The dataset was curated to encompass the periods during which the rat actively traversed the track. Then, the ensemble spike activities were discretised into 25-millisecond intervals through a binning process.

Both the neuronal and behavioural data can be seen in Figure 5.3. The rat's position and running direction (left or right) were encoded as three separate behaviour variables, one containing continuous-valued position information and the other two being binary variables indicating the running direction. Note that the last two behaviours contain the same information and can thus be merged into a single variable. However, for purposes of consistency to compare with other algorithms, we keep them as distinct behaviours, as in the original paper.

5.1.4. Adapting BunDLe-Net for multiple and continuous-valued behaviours

In previous work, BunDLe-Net was applied to a dataset with a single discrete behavioural variable. However, in the rat hippocampus dataset, we encounter a different scenario with three behavioural variables, one of which is continuous, while the others are discrete. To

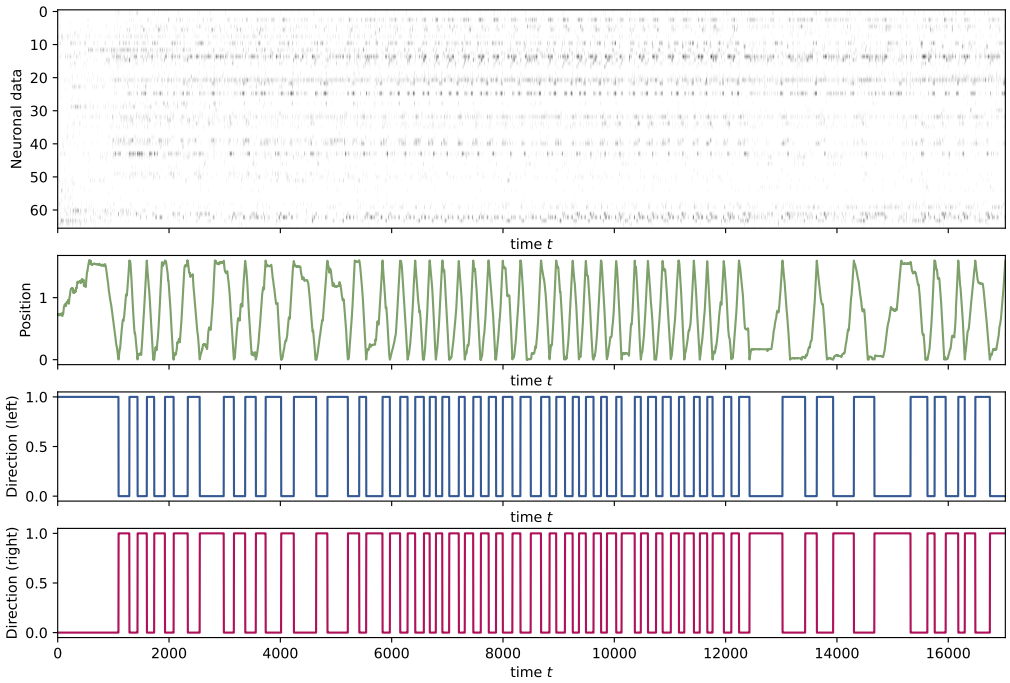

Figure 5.3.: Spiking neuronal and behavioural data recorded from rat-1 (Achilles). Neuronal activity was recorded simultaneously with three behavioural variables, position (continuous-valued), direction-left (binary) and direction-right (binary)

accommodate this new behavioural modality, we make minimal adaptations to BunDLe-Net's architecture. We augment the behaviour predictor unit of BunDLe-Net (see architecture in Chapter 3) with three output channels, each corresponding to one of the three behaviours. We then modify the behaviour loss $\mathcal{L}_B$ to a mean squared error loss to accommodate the continuous-valued behaviour. Apart from these adjustments, all other aspects of BunDLe-Net, including the hyperparameters, remain unchanged. The adapted BunDLe-Net is separately trained on each of the rat datasets[1].

5.1.5. Results - Comparison of the BunDLe-Net and CEBRA embeddings

We compare BunDLe-Net with CEBRA, a state-of-the-art algorithm for neuronal manifold learning (Schneider et al., 2023). CEBRA uses contrastive learning to embed neuronal activity based on behaviour, time, or a hybrid of both. For this comparison, we use the CEBRA-hybrid setting to ensure a fair assessment with BunDLe-Net, which inherently considers both dynamics and behaviour in its learning process. CEBRA-hybrid was trained on all rat datasets using parameters identical to those specified in the original demo notebook from CEBRA's webpage `https://cebra.ai/docs/demo_notebooks/Demo_hippocampus.html`. Since the CEBRA publication demonstrated their algorithm on the same rat hippocampus data, we can be fairly certain that the parameters for CEBRA are optimally tuned.

Figure 5.4, shows the BunDLe-Net and CEBRA-hybrid embeddings of all four rat datasets (each in separate gray boxes). For each algorithm, we have plotted the same embedding twice, with one labelled with a gradient and the other with a binary colour map, which correspond to the position and direction behaviour variable, respectively. We do this in order to visualise how behaviours relate to the embedded neuronal activity. In the first plot, we plot the embedded data points, and in the second, we plot line segments between data points at time t and $t+1$. The former plot helps us visualise the distribution of the embedded points in latent space, and the latter lets us visualise how their trajectories evolve with time.

From the embedding of rat-1 (Achilles), we see that embeddings are *geometrically* different, with CEBRA embedding on the surface of a sphere and BunDLe-Net embedding on a plane. Despite these geometrical differences, BunDLe-Net and CEBRA's embedding manifest the same *topological structure* – a single closed loop. This is noteworthy, as it suggests that both algorithms effectively capture the underlying topological structure of neuronal activity concerning the associated behaviour. In contrast to the embedding of *C. elegans* in Chapter 3, there are no discernible bifurcations in the trajectory of the rat's neuronal dynamics. Instead, a single deterministic loop is apparent, reflecting the rat's straightforward task of moving back and forth along the track to obtain rewards from either end, as indicated by human-annotated behaviour data.

[1]Note that here we fitted BunDLe-Net on each of the rats separately. This is different from our methods in publication 3 on the *C. elegans* data where we used a specific training sequence to ensure comparable embeddings. For the rat data, the embeddings revealed considerably simpler topologies, which were easier to compare to one another. Hence, the more sophisticated fitting pipeline used in Chapter 3 were not required.

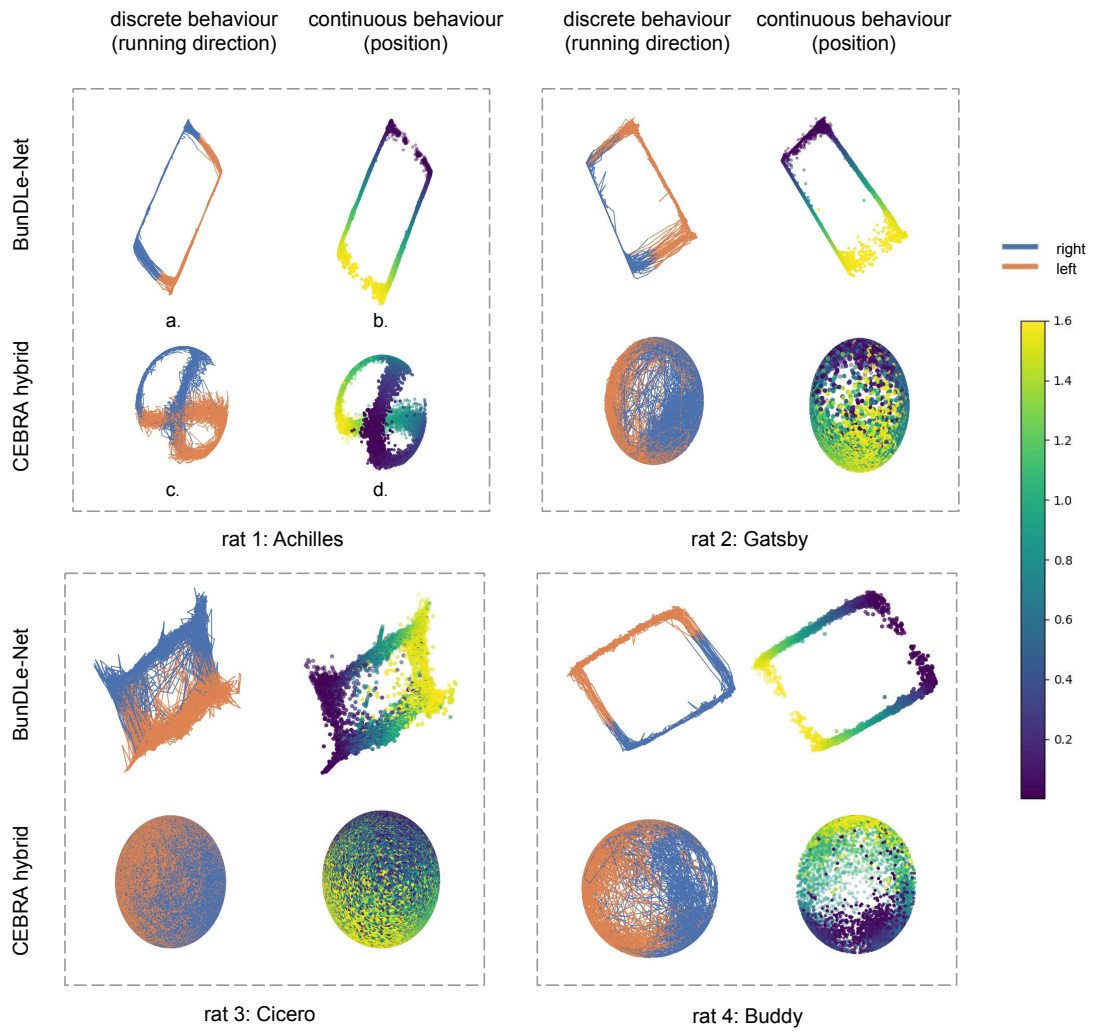

Figure 5.4.: BunDLe-Net and CEBRA-hybrid embeddings of four rats in the rat hippocampus dataset. The embeddings are colour-labelled with the behavioural variable of interest. The continuous gradient colour corresponds to the position variable, and the discrete (blue, orange) corresponds to the binary variable of direction (right, left).

Further, we see from BunDLe-Nets embeddings that both position and direction are well-preserved, as can be seen by the coloured labelling of the points. This implies that the recorded neurons indeed encode information about both the rat's position and direction in their spiking patterns. Our implicit behaviour decoding module within BunDLe-Net is clearly able to decode the positional and directional information from the neuronal activity. This finding is consistent with what is known about the hippocampus from decades of biological research (Grosmark and Buzsáki, 2016).

Now, turning our attention to the embeddings of the other three rats, named Gatsby, Cicero, and Buddy, we see that BunDLe-Net's embedding remains quite consistent in terms of topological structure, i.e. we see a rectangle where one dimension encodes position and the other encodes direction. In contrast, CEBRA's embedding becomes obscured by noise, resulting in a topological structure that is barely discernible. The data points in CEBRA's embedding are dispersed diffusely, while in BunDLe-Net, they follow a more compact trajectory[2]. Additionally, we observe that BunDLe-Net's trajectories (upper left in each of the three boxes) are far smoother in terms of transition from time step t to $t+1$. In contrast, CEBRA's embeddings (lower left in each of the three boxes) show numerous discontinuous jumps in the trajectory that obscure any discernible dynamic structure. This contrast may be attributed to BunDLe-Net's dynamical loss function $\mathcal{L}_D$ and its very architecture, which is designed to preserve dynamical information.

These results demonstrate several strengths of BunDLe-Net over CEBRA: the ability to capture the underlying topological structure, robustness to noise in the data, smoother trajectories in terms of dynamics, orthogonal embedding of independent behaviours, and consistency of embeddings over animals.

5.2. Interpretability of latent dimensions

Thus far in our explorations of neuronal data with BunDLe-Net, we have embedded all the features (neurons) jointly into a three-dimensional latent space[3]. All three dimensions jointly contain information about the dynamics of the system. Such an embedding was seen to be interpretable in terms of the system's dynamics as a whole by revealing bundled trajectories with distinctive topologies.

However, this joint embedding tells us little about the interactions of the parts of the system. Complex systems often consist of various fairly modular subsystems, and we may be interested in studying how these subsystems interact in terms of their information. For example, in the context of the brain, neuroscientists are keen to understand how various regions of the brain interact with one another to process sensory information and generate behaviour.

To address this, we propose endowing each latent dimension with interpretability by

[2]It should be noted that CEBRA's tendency to embed data over the surface of a sphere is an inherent feature of the algorithm itself and should not be interpreted as a specific characteristic of the neuronal manifold associated with this data.

[3]Three dimensions were seen to be sufficient since we can always embed a finite graph in three-dimensional space, without the edges intersecting.

relating it to a known subsystem of interest. To this end, we embed the features of each subsystem into a separate dimension of the latent space. Previously, we had a high-dimensional vector X that we jointly embedded to a three-dimensional vector Y through a transformation $Y = \tau(X)$. We now consider n subsystems within the original system, which splits our feature space $\mathcal{X}$ into smaller feature spaces $\mathcal{X}^{(1)}, \mathcal{X}^{(2)}, ...\mathcal{X}^{(n)}$ where $\mathcal{X}^{(i)}$ is a subspace of $\mathcal{X}$. Now, to map each subsystem to a separate latent dimension, we use several $\tau^{(i)}$ such that $Y^{(i)} = \tau^{(i)}(X^{(i)})$. In this way, we maintain distinctions between subsystems of interest and move towards making causal statements between various components of the system.

5.2.1. Subsystems - neuron categories

Traditionally, in neuroscience, neurons have been categorised into three types based on their functional role in the nervous system. Sensory neurons are responsible for detecting external stimuli and relaying them to other neurons. Motor neurons are responsible for executing commands of the nervous system by conveying them to muscles that then get actuated into physical movement. Interneurons have neither sensory detectors nor muscle connections; instead, they process information received from sensory neurons within their circuits and subsequently relay it to motor neurons. This classical view of neuronal information processing suggested a linear pathway: sensory neuron $\rightarrow$ interneuron $\rightarrow$ motor neuron.

However, a recent study (Kaplan et al., 2018) has challenged this linear information processing model, proposing a more integrated pathway with feedback mechanisms among these neuronal units. Here, we use these sensory, inter- and motor categories to partition the set of neurons into three *subsystems*. We then embed the neuronal data accordingly and investigate how information is processed and relayed between these groups.

5.2.2. Description of the data

We use the same *C. elegans* data (Kato et al., 2015) as in Chapter 3. In addition, we utilise the categories of neurons – sensory, motor and interneuron. Since only a few neurons were identified in the original dataset, we could only reliably label the categories of these identified neurons. This substantially reduced the number of neurons from which we could learn an embedding as compared to Chapter 3. Based on the categories, we partition the set of identified neurons into three subsets $\mathcal{X}^{(1)}$, $\mathcal{X}^{(2)}$ and $\mathcal{X}^{(3)}$ corresponding to the sensory neuron, interneuron, and motor neuron, respectively.

5.2.3. BunDLe-Net with anatomically interpretable latent layers

To further enhance the interpretability of the BunDLe-Net architecture, we introduce a novel adaptation that allows for the separate embedding of distinct feature subsets as seen in Figure 5.5. The full implementation and code can be found in the BunDLe-Net GitHub repo `https://github.com/akshey-kumar/BunDLe-Net/tree/main/experimental/anatomical_interpretability_of_axes`.

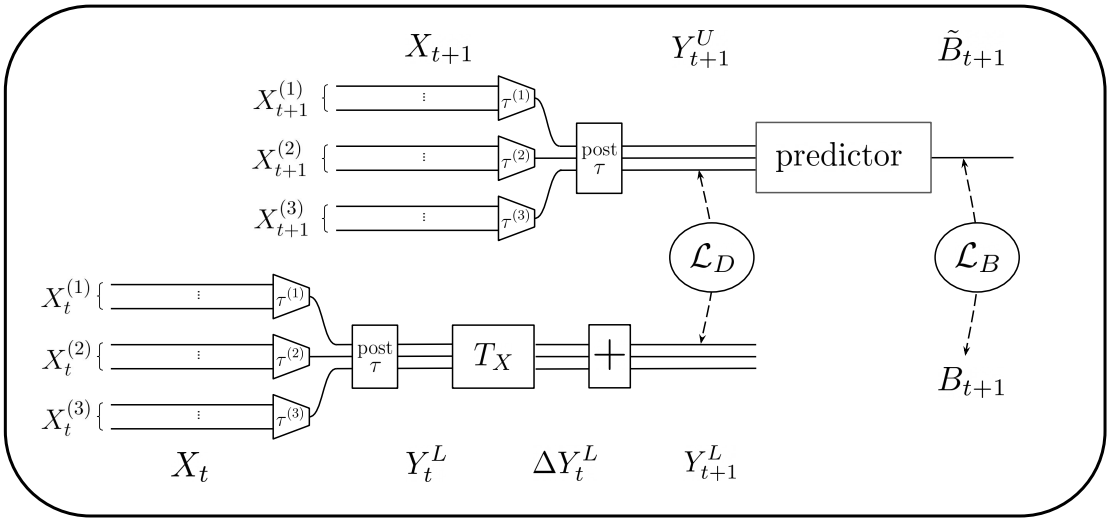

Figure 5.5.: BunDLe-Net adaptation for embedding distinct subsets of features (neurons) into separate latent dimensions

We modify the τ layer to incorporate three independent mappings denoted by $\tau^{(1)}$, $\tau^{(2)}$ and $\tau^{(3)}$. These mappings operate on the vectors $X^{(1)}$, $X^{(2)}$, and $X^{(3)}$ respectively. Each $\tau^{(i)}$ consists of a series of dense layers with ReLu activation that map the input to a single-dimensional output (see GitHub implementation for further details). The output from all three $\tau^{(i)}$ are then passed through another layer, post-τ, which concatenates the three outputs, normalises it, and adds Gaussian noise[4].

The same three $\tau^{(i)}$ along with the post-τ layer are applied to both X_t and X_{t+1}. This ensures that we have a consistent embedding across time-steps. Apart from the modification to the τ mapping, all other aspects of the architecture remain unchanged from our previous work.

5.2.4. Results - Anatomically interpretable latent dimensions

Figure 5.6 shows the embedding of worm-1 with each axis explicitly labelled according to the neuron subset it encodes. The colour coding correspond to the behavioural label (sustained reversal ●, ventral turn ●, dorsal turn ●, forward ●, slowing ●, reverse-1 ●, reverse-2 ●, no state ●). For 3-D rotating plots of all five worms, please see `https://github.com/akshey-kumar/BunDLe-Net/tree/main/figures/rotation_axis_decomp`.

Remarkably, we observe the same branching structure and topology as in Chapter 3, despite embedding only a much smaller subset of identified neurons from the dataset. This observation underscores the presence of globally encoded dynamics within the entire

[4]The incorporation of normalisation and Gaussian noise aligns with our previous work in Chapter 3 and serves the same purpose of regularisation to reduce overfitting.

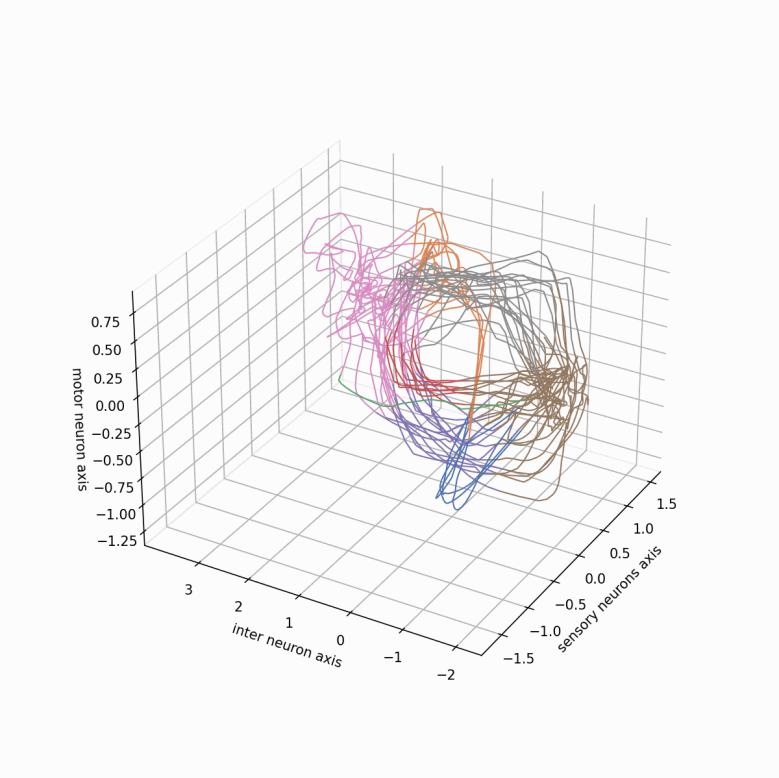

Figure 5.6.: Embedding of C. elegans neuronal data in three-dimensional latent space where each dimension corresponds to a specific subset of neurons.

neuron population, indicating robust representations in the data. This finding aligns with existing knowledge in neuroscience concerning C. elegans neuronal circuitry (Kaplan et al., 2018).

While the overall structure of the dynamics is maintained, the embedding is seen to be more noisy. This could be attributed to the reduced number of neurons mapped into each embedding dimension. Another potential factor is that different dimensions may encode varying amounts of information about specific behaviours. For instance, some behavioural trajectories may be tightly bundled along certain dimensions while being more loosely bundled along others. This suggests that dimensions with tighter bundling may contain predictive information about the dynamics of those behaviours.

We do not attempt a rigorous analysis of the embeddings in this section but point out some noteworthy observations that can spark future research along these lines. Firstly, note how certain behaviours are projected along a specific direction or confined to a given plane. For example, we see that the slowing ● and sustained reverse ● are mostly separated along inter-neurons axis. This is something we observe across all worms. On the other hand, when the dorsal turn ● occurs, it differs from the other behaviours along the sensory neuron axis. Secondly, notice how the dynamics often switches its orientation

and alignment with axes through the evolution. For example, consider the transition from sustained reversal ● to ventral turn ●. Initially, the ventral turn dynamics ● progresses vertically along the motor neuron axis. It then makes a sharp turn and continues along the inter-neuron axis for the latter phase, before which it segues into the slowing movement ●.

This may allow us to make statements such as, *The dynamics is initiated in neuron-subset $\mathcal{X}^{(1)}$, which then relays the information to neuron-subset $\mathcal{X}^{(2)}$; the information is then shared with neuron-subset $\mathcal{X}^{(3)}$ which jointly orchestrates the behaviour with neurons-subset $\mathcal{X}^{(2)}$ which is then again relayed to $\mathcal{X}^{(1)}$, and so on....* From such embeddings, one could thus investigate interactions between various subsystems, their relationships with one another, and the flow of information between them. Such an approach builds up to uncovering algorithms within neuronal systems. A further and more comprehensive biological and information-processing analysis of these embeddings is beyond the scope of this thesis. Instead, this experiment serves as a foundational step for future research endeavours. Further discussions can be found in Section 6.3.

6. Discussion

We begin this section by answering the original research question posed in 1.4. By answering these specific questions, we show how we have contributed to various domains such as causality, data science, and neuroscience in Section 6.2. The implications of our research pave the way for several novel avenues of exploration, some of which we discuss in Section 6.3. This includes extensions of BunDLe-Net, further techniques to improve causal attractions and generalisations of our theoretical framework. We finally summarise the scope and achievements of this research project and conclude this dissertation in Section 6.4.

6.1. Research questions answered

Q1: Under what conditions can we infer causal abstractions from purely observational data of a dynamical system?

From our theoretical framework, we see that Markovianity at the X-level is sufficient to infer causally consistent abstractions from purely observational data of a dynamical system. This is a direct consequence of Theorem 1, which states that Markovianity and dynamical observational consistency are jointly sufficient conditions for dynamical causal consistency. This means that if we are able by any means to obtain an observationally consistent abstraction (see def. 4) of the X-level, then if the X-level is Markov, we can directly conclude that our abstraction is dynamically causally consistent as well. This allows us to talk about interventions directly at the Y-level without referring to the ground-truth X-level.

Our main contribution is to identify a setting where it is possible to make causal abstractions from purely observational data. This setting includes our definitions of Markov dSCM (def. 2), our notion of DOC (def. 4) and DCC (def. 5), and Macroscopic intervention (def. 3). With these definitions in place, we can prove our causal claims in Lemma 1 and Theorem 1. In general, inferring DCC from DOC may not always be possible. For example, suppose our model M_X at the X-level is not Markov. In that case, even if the Y-level M_Y is compatible with $Y_t = \tau(X_t)$ for all t, it is not guaranteed to preserve causal information about the dynamics at the of X-level. Note that our requirement of Markovianity at the X-level is weaker than requiring Markovianity both at the X and Y levels (which would intuitively mean the same linear causal structure at both levels). This is due to the elegant way that DOC is defined. Lemma 1 ensures that if observational consistency is fulfilled, then the Y-level automatically takes on the Markov causal structure of the X-level. Thus, we have presented a minimal set of conditions,

which, up to our knowledge, are the first of their kind, which allow causally consistent abstractions of data from dynamical systems.

Q2: Given these conditions, how can we *learn* causal abstractions directly from time-series data?

As an answer to this question, we present the learning algorithm BunDLe-Net in Chapter 3 which is designed specifically to extract causally consistent abstractions from time-series data in a behavioural context. Here, we outline how this question is tackled through this thesis. First, in our theoretical framework, we present and prove Theorem 2. This theorem is based on the commutativity diagram, which says that it does not make a difference between first abstracting with τ and then time-evolving or first time-evolving and then abstracting. If the commutativity diagram is satisfied, then DOC is fulfilled. Guided by this principle, we design an architecture specifically tailored to learn DOC transformations as outlined by Theorem 2. If the Markov condition of the X-level is fulfilled, we invoke Theorem 1 which guarantees that the learned abstraction is not only observationally but also causally consistent.

Additionally, we provide an analytical solution within our mathematical framework, catering to systems characterised by linear dynamics and linear behaviour. This gives us a closed-form solution for abstracting a Y-level that is dynamically causally consistent with the underlying dynamics. Hence, if we have data from a linear dynamical system and can determine its transition model (say through a linear autoregressor), we can construct such an abstraction directly from data.

For non-linear systems that show much more complex and interesting dynamics, obtaining an analytical closed-form solution may not be feasible in most cases. Hence, we leverage the architecture based on the Theorem 2. This architecture is flexible and can be equipped with arbitrarily non-linear layers for the abstracting layer τ, the transition model T_Y and/or the behaviour predictor. Since neural networks offer an innate flexibility of adding arbitrarily non-linear layers, we utilise them in our architecture in Chapter 3. BunDLe-Net is, therefore, a generic architecture that can be endowed with layers of desirable non-linearity in order to learn abstractions directly from time-series data.

Q3: How can causally consistent abstractions be used to gain meaningful insights into complex neuronal datasets?

We were previously faced with the challenge that there may exist several causally consistent abstractions, most of which may not be useful; for example, a mapping to a single-state Markov process. Hence, we require that behaviour should be decodable from the abstraction. Causally consistent abstractions that preserve behavioural information prove to be a powerful tool to gain insights into neuronal data. This is demonstrated in both Chapters 3 and 5. Since they preserve behavioural information, they serve as a high-level model of behaviour that is derived from neuronal activity. We can use this to understand the way information is represented and processed in the brain. Our theoretical

framework in Chapter 2 is used to bridge the neuronal and cognitive levels in our joint work (Grosse-Wentrup et al., 2023).

To make sense of how BunDLe-Net's abstraction helps us understand neuronal data, we consider the various components of BunDLe-Net (see architecture in Chapter 3) and see what it learns about the data. Firstly, since BunDLe-Net has a behaviour predictor, it is implicitly endowed with the ability to decode behavioural information from neuronal data. This tells us about how representations of the behaviour are encoded at the neuronal level. The abstracting layer τ distils neuronal activation patterns down to a low-dimensional space which yields powerful visual insights into the data, such as bifurcations and convergences that can be related to decision-making. Finally, the transition model, allows us to predict the trajectory of the embedded brain state in latent space giving us insights into the dynamics of the system. In the experiments with *C. elegans* data, we saw how BunDLe-Net neatly separates out the deterministic parts of the dynamics from the probabilistic ones which other methods fail to do.

The versatile nature of the BunDLe-Net architecture allows it to handle neuronal data, which comes in a variety of shapes and sizes. In Chapter 5, we demonstrate BunDLe-Net in diverse settings involving discrete-valued neuronal data, continuous-valued and multimodal behaviours. It outperforms state-of-the-art methods and reveals consistent embeddings on both roundworm and rat data. Finally, we show how BunDLe-Net can incorporate anatomical information about various subsystems in the brain in Section 5.2. Hence, we show that BunDLe-Net is a flexible architecture to gain robust and consistent results that broaden our understanding of neuronal data.

6.2. Contributions to various fields

In this section, we reflect upon the scope of our work and discuss its impact across multiple fields. The interdisciplinary nature of this project contributes to various domains such as causality, data science, and neuroscience and, in doing so, enriches the broader field of artificial intelligence.

6.2.1. Statistics and Causality

From our brief survey of causality in Section 1.3, we saw that it is poorly understood how to define (high-level) causal variables in complex systems. We presented a case to illustrate how coarse-graining causal variables may lead to variables that are unfit for causality. Thus, we found that the variables resulting from an arbitrary coarse-graining are not causal. Only certain transformations preserve the causality in the system.

Also, the majority of causal models focus on the independent and identically distributed (iid) causal variables. We argue that temporal information often contains valuable causal information. On this basis, we develop a framework for causally consistent abstractions of dynamical systems in Chapter 2. The crux of our contribution lies in a simple but vital principle – preserving dynamical information and, in doing so, preserving temporal causality. We formalise this insight through our mathematical framework and prove two

theorems. The first theorem gives us conditions under which it is possible to learn a causally consistent transformation of a dynamical system from purely observational data, should such a transformation exist. Our second theorem provides the basis for constructing a generic architecture to learn DOC abstractions from purely observational data. If the data fulfils the Markov assumption, then we are assured by the first Theorem that the learned abstraction is also dynamically causally consistent (DCC). While this architecture is not a causal inference algorithm in a traditional sense, it provides a new causal way to think about time-series data. This can be a valuable new perspective that works in settings where causal inference may not be feasible or informative (high-dimensional systems with dense causal interactions).

6.2.2. Machine learning and Data Science

Our contribution to the field of data science is a novel architecture for dimensionality reduction and manifold learning. While most previously existing methods are tailored to independent and identically distributed (iid) data within unsupervised settings, few protocols are designed for time-series data. Even among the limited methods targeting time-series data, many still incorporate iid-based approaches such as PCA within their pipelines, which results in a loss of valuable information inherent in the system's dynamics (Ali et al., 2019). Our BunDLe-Net architecture addresses this critical gap by preserving these intricate temporal relationships. In this dissertation, we have implemented BunDLe-Net with a neural net, but in general, our contribution is a generic architecture that is readily adaptable to a variety of modalities.

In recent times, autoencoder architectures have gained prominence for dimensionality reduction and representation learning. These methods encode information to optimise the reconstruction of the original data. They prove effective when the data exhibits significant redundancy, as is often the case in image and video data. However, it is important to note that autoencoders are primarily designed for scenarios where the data naturally conforms to a low-dimensional manifold, which does not always hold true for all high-dimensional data. In complex information-processing systems, such as the brain, there are numerous processes occurring simultaneously. Therefore, it is not a reasonable assumption that the data strictly conforms to a low-dimensional manifold. More often, we are interested in specific aspects of the system, and while other information is not redundant, it is not relevant to our objectives. This is where our algorithm's strength lies because it is a supervised algorithm designed to decode a specific behaviour[1]. In the context of neuroscience and causality, the supervisory variable is chosen as behaviour, but in other applications, it could be any target variable of interest, such as an external stimulus, a response, an internal system component, or any human-annotated aspect of interest. BunDLe-Net would then provide a lower-dimensional embedding that can be used to

[1] If one wishes to use the BunDLe-Net architecture in the unsupervised setting, it can be readily adapted. Instead of decoding a behavioural variable, the network can be configured to reconstruct the entire system state at time $t + 1$. This would still preserve dynamical information but of the system as a whole rather than for a specific component. This adaptation would render BunDLe-Net equivalent to an autoencoder setting where we input X_t and reconstruct X_{t+1}

model the target variable.

Besides the supervised component, BunDLe-Net has autoregressive capabilities. Each of the components of BunDLe-Net, viz. a behaviour decoder, τ and T_Y, can be used individually for specific tasks, including decoding behaviour from the low-dimensional representation, embedding new data and simulating the system in the embedded space. In particular, we shall discuss future avenues of research which relate to simulating complex systems on a low-dimensional manifold in Section 6.3. Aside from contributing a dimensionality reduction technique, our method serves as a powerful visualisation tool for complex time-series data. These visualisations provide insights into the topological structure of the dynamics with regard to a specific task or behaviour.

6.2.3. Neuroscience

Our first contribution to the field of neuroscience is a Neuronal Manifold Learning technique that takes into account behaviour in accordance with the guidelines of Krakauer et al.. This method is not merely a visualisation tool but a means to gain automated insights about how the brain processes information. Our algorithm is particularly developed to deal with multimodal neuronal data that often involves simultaneous neuronal and behavioural recordings. Such data, combined with the abstraction techniques of BunDLe-Net, can reveal intriguing insights about information processing in the brain. This would reduce efforts of manual data processing and searching for patterns by automating it to a certain degree. Among the various insights we can gain from studying the dynamics of neuronal activity are decision making. For example, the application of BunDLe-Net on *C. elegans* data revealed an intricate branching trajectory that was not evident from the neuronal traces or a pure behavioural analysis. The bifurcations could be linked to decision-making. Several interesting questions remained to be explored in this direction. Furthermore, BunDLe-Net is not restricted to calcium imaging data but works equally well on spiking data from the rat hippocampus and outperforms all state-of-the-art methods as shown in Figure 5.4.

On a higher level, we offer the theoretical foundations that give rise to the NC-MCM framework to bridge the neuronal and cognitive levels (Grosse-Wentrup et al., 2023). This builds on our theoretical framework and presents a mathematically grounded definition of a cognitive state so that we can learn it from data. It also presents the basis for learning discrete cognitive concepts from the BunDLe-Net's embeddings in latent space.

6.2.4. Artificial Intelligence

An important and vital component of an intelligent agent is the ability to reason in terms of abstract concepts or representations. This includes the ability to group diverse instances into higher-level abstract concepts, reason at the abstract level and, subsequently, make predictions or decisions in the real world. By abstracting, we essentially discard information to the extent that we can make generalisations that can then be reused in different scenarios. In general, there is an entire hierarchy of levels of abstraction that can be present, but the essence is the ability to reason at varying levels of abstraction

(Hofstadter, 1979). Schölkopf et al point out that a central challenge for AI pertains to causal representation learning (Schölkopf et al., 2021). This challenge involves the discovery of high-level causal variables from low-level observations.

By specifically contributing a framework for learning causal high-level variables in the time-series setting, we directly make a contribution to the field of Causal representation learning, which is a step towards artificial intelligence. While this thesis focuses on dynamical systems, it lays the foundation for work in more generic settings, which will be discussed in Section 6.3.2. We also locate this research amidst representation learning work in reinforcement learning (Zhang et al., 2021).

6.3. Outlook

This section explores various new research directions unveiled by this dissertation. Additionally, we highlight the current work's limitations and delineate future research prospects. Our discussion of future research is structured into three categories: extensions of the BunDLe-Net architecture, generalisations of the mathematical framework, and other potential avenues for exploration.

6.3.1. Extensions of BunDLe-Net architecture

Simulating high-dimensional dynamics in the latent space

In all current versions of BunDLe-Net, we used a linear and deterministic transition model T_Y at the Y-level (latent space). A linear model was chosen, so we had the simplest model that would not be prone to overfitting. For our purposes of learning a DCC and BC embedding of the neuronal data, a linear T_Y is seen to suffice, as the dynamics are approximately linear for small differences.

However, working with a linear T_Y considerably limits the complexity of dynamics that we can have at the Y-level. While a linear model suffices for ensuring short-term predictability of the dynamics, it is prone to fail in making long-term predictions. Long-term predictability will be an important aspect if we want to simulate the dynamics in the latent space. Apart from having good long term-predictability, we would also need to allow for probabilistic transitions. From the *C. elegans* embeddings in 3, we see several bifurcations in dynamics. Hence, we would also need to supplement T_Y with probabilistic layers, such as in a variational autoencoder (Kingma and Welling, 2013). Thus, endowing T_Y with nonlinear and probabilistic layers for T_Y would open avenues to creating simulations at the latent space.

Simulating the system in latent space would allow us to capture complex dynamics of the neuronal activity in a much lower dimensional latent space. This is also computationally much lighter than attempting a full dynamical simulation of a high-dimensional system such as the brain. Simulations would also enable us to test the robustness of our method by comparison with real experiments. For example, one could simulate interventions in the model and check if the resulting dynamics align with interventional experiments.

Embedding subsystems separately

In Section 5.2, we saw how BunDLe-Net's architecture τ layer can be readily adapted to embed sub-systems separately. In doing so, the dimensions in latent space acquire further interpretability since we can relate them to specific components of the system. In this way, we can study how various subsystems interact with one another in terms of the dynamics since BunDLe-Net's embedding gives us a way to visualise the presence (or absence) of information in a given subsystem. We can potentially use this to study how information is processed and relayed between various components.

In our proof-of-principle example, we considered three subsystems based on the neuron category (sensory, motor and interneurons) with a single embedding dimension for each subsystem. Future work would involve creating ways to visualise and/or quantify results in systems containing more than three subsystems. One can then ask the question of how the subsystem embeddings causally interact with each other. Two approaches can be taken here: one could try to infer causal interactions directly from the transition model T_Y. Alternatively, we could engineer T_Y to match the known rules of how the subsystems interact (domain knowledge) and learn embeddings consistent with this.

So far, we have discussed cases where we already had clear demarcations of subsystems of our interest. These demarcations could be on the basis of anatomical, functional modularity, or simply convention. In general, for unexplored complex systems, it may not be obvious how to partition them into separate subsystems. To tackle this, future research can incorporate disentanglement methods along with traditional causal inference into our architecture to identify modular subsystems in an automated way.

BunDLe-Net on video data

Our work can potentially address questions in the domain of computer vision and the analysis of complex systems. Specifically, we envision its applicability in the context of extracting fundamental variables from video frames (pixel data), which has garnered much attention in recent times (Chen et al., 2022; Kipf et al., 2020). This has widespread applicability, starting from simple physics-based systems to more general artificially intelligent computer vision systems. A simple physics-based example would be learning variables such as position and momentum from the video of an oscillating pendulum. Learning high-level representations of video data that are dynamically consistent would enable us to simulate the system, identify important image features and understand how different image features affect the dynamics.

Our architecture can be readily adapted to this problem setting by using convolutional layers, which have been seen to excel at learning features from image data. The flexibility offered by our method is that we do not need to reconstruct the entire original video frame, as most existing methods do. Rather, we can use a behaviour of interest (or any target variable) to determine the level of granularity of our embedding. This allows us to ignore irrelevant details or other processes that co-occur but are not of interest in the video.

Topological data analysis

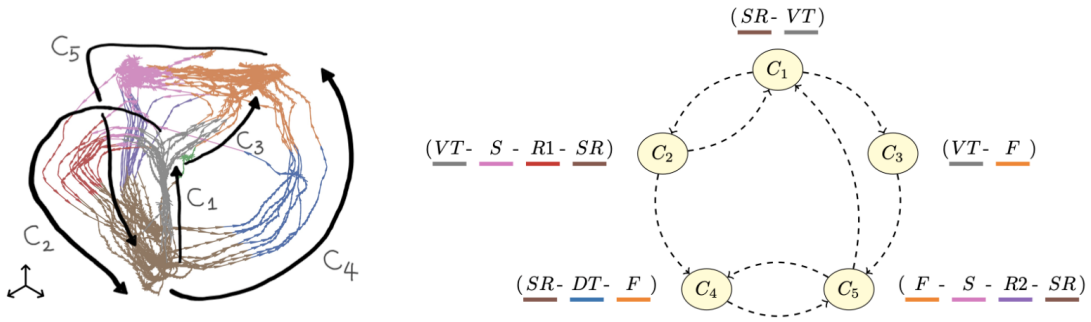

Figure 6.1.: Cognitive state transition diagram from BunDLe-Net's embedding of *C. elegans* data

It is evident from Figures 5.3, 5.6 and Chapter 6.1 that the BunDLe Net embeddings exhibit a definite topological structure. While the rat embeddings took the topological form of a loop, the *C. elegans* embeddings showed a more complex topology with several bifurcations and holes. In both cases, the dynamics were seen to recur along this topological form.

In Chapter 3 we used each *branch* of the trajectory to naturally define a set of discrete cognitive states $C_1, ..., C_5$. Our reasoning was that, along each branch of the trajectory, the dynamics were predominantly deterministic. Probabilistic decisions occurred only at the bifurcations. Hence, we could simplify our embedding to a Markov transition process that stochastically switches between deterministic nodes if we take each branch to be a node in our graph. In our publication, we manually annotated the branches of the embedding (see left Figure 6.1).

One can automate this process by harnessing Topological Data Analysis (TDA) to infer the topological structure directly from the embedding. There are a host of existing TDA methods (Chazal and Michel, 2021; Skaf and Laubenbacher, 2022), but most of them are for iid data. One could either use a TDA method as it is or refine it to a time-series setting. Either way, we can determine the topology of our trajectories and thus obtain a state-transition diagram over the discrete states, as seen in Figure 6.1.

While this is not an extension of the BunDLe-Net architecture, it would supplement the analysis pipeline and enhance the capabilities of the BunDLe-Net toolbox. In this way, we would not only obtain continuous-valued embeddings from BunDLe-Net but also obtain a discrete high-level model. These discrete models can be interpreted as cognitive states through the NC-MCM framework. Discrete cognitive states offer the advantage of familiarity within the field of cognitive science, rendering them notably more useful and easier to interpret.

6.3.2. Generalisations of the mathematical framework

In our current mathematical formalism, we have considered the dynamical setting where $X_t \to X_{t+1}$ and settings where transformations τ preserve this causal structure. Future work may attempt to generalise this formalism to settings with more complex causal structures. For example, one may have already identified a causal structure at the X-level (such as the connectome at the neuronal level) and wish to learn a Y-level in such a way that it respects the original causal structure. By doing this, we would create a set of high-level causal variables that model causal relationships between real-world entities.

Our current requirement of DOC requires that $P(Y_{t+1}|X_t) = P(Y_{t+1}|Y_t)$. This can be viewed as a conditional independence statement that says: $(Y_{t+1} \perp X_t|Y_t)$, which means that Y_t screens off Y_{t+1} from the X-level. In this way, we obtain a *self-contained* model within the Y-level. This reasoning can be generalised as follows.

Consider a given causal structure at the X-level (determined by experiments, for example). The causal Markov condition (CMC) on the X-level states that every variable is independent of its non-descendants when conditioned on its parents: $X_i \perp X_j|Pa(X_i)$ where $X_j \in \mathrm{ND}(X_i)$. This CMC thus yields a set of conditional independence relations that follow from a graph (identical to the CIs obtained by d-separation) (Scheines, 1997). Intuitively, the CMC means that given the direct causes, a variable is screened off from all other variables except its effects. Now, let's say we want to create a Y-level $Y_i = \tau_i(X_i)$ such that the Causal Markov Condition also holds on the transformed variables at the Y-level. i.e. $Y_i \perp \tau_j(X_j)|\tau(Pa(X_i))$ where $X_j \in \mathrm{ND}(X_i)$, where we use the shorthand $\tau(Pa(X_i)) = \{\tau(X_k) : X_k \in Pa(X_i)\}$. For the CMC to be valid at the Y-level, we would need to ask the question: What is a sufficient condition or set of assumptions on the transformation τ such that the Causal Markov Condition on the X-level holds on the holds Y-level as well?

We would then get a series of conditional independencies (similar to DOC in the dynamical setting). It may then be possible to impose the dependencies in the transformations through certain architectures, as was possible in the dynamical setting. Generalisations of our framework have the potential to give general and direct answers to paradoxes such as (Gebharter and Retzlaff, 2020) and may help avoid causal inconsistencies on coarse-graining (Weichwald et al., 2015). It would also greatly broaden the application beyond the time-series chain to causally interacting subsystems of dynamical systems that allow interventions on parts of the system.

6.3.3. Other research directions

Decision-making in neuroscience

The BunDLe-Net trajectories on *C.elegans* reveal a mixture of deterministic and probabilistic dynamics. Along a given branch, the trajectory was nearly deterministic, while the probabilistic choices were confined to the bifurcation regions in the trajectory. We could use this to study decision-making within neuronal circuits. Notably, we observe from the embeddings that a bifurcation may occur *even before* the new behaviour is externally observed. For example, in Figure 6.1, see the sustained reversal (●) branching

off into the dorsal turn (●) and ventral turn (●). This can indicate that the information of the decision is present in the neuronal activity even before we observe it externally as a locomotory behaviour. This can be investigated more rigorously using tools from statistical testing to validate such claims.

Future research could address questions such as: Are these decisions truly probabilistic, or are there some hidden factors that influence these decisions? One could conduct experiments by comparing an embedding of the total neuron population with an embedding of a subset of neurons. We can then study the effect of the inclusion of these neurons on the bifurcations, which would then give us information such as whether the given subset of neurons influences the given decision. Of course, causal statements can only have a grounding under very specific settings or with interventional experiments, which lead us to our next direction.

Interventional experiments and BunDLe-Net

While it is still challenging to perform interventions at the neuronal circuit level, there are an increasing number of experiments that offer this ability. By optogenetic techniques, one can even stimulate single neurons in a controlled setting (Antinucci et al., 2020; Shemesh et al., 2017). Interventional experiments offer us a much richer range of possibilities than purely observational data. Such experiments would be a perfect test bed to check whether BunDLe-Nets' embeddings are indeed causal at the level of subsystems. Furthermore, we could adapt BunDLe-Net to take into account the experimental interventions and accordingly adjust the embedding. In Chapters 3, 5, we had to rely on the assumption that the underlying data is Markov. With interventional data, we would not need this assumption. This would broaden the scope of our algorithm since, with access to interventional data, we are not bound to the causal sufficiency assumption and, hence, apply our methods to settings where whole-brain imaging is not always available settings, viz. in more complex brains like that of zebrafish, mouse, monkey, and ultimately the human brain.

Future research could also incorporate our representation learning method into closed-loop settings with neuronal stimulation. This would involve learning the embedding dynamically and adapting it when we gain new interventional information. This is an exciting avenue of research where one can try to engineer interventions on the basis of the embeddings (and vice-versa).

Explainable AI on RL agents

Thus far, we have focussed on understanding brains through our formalism and BunDLe-Net algorithm. However, we can also apply the same models to understand and explain the behaviour of artificially intelligent agents. The need for explanations may be crucial in certain scenarios where human life may depend on AI decisions, such as in self-driving cars or healthcare applications. While AI has been seen to achieve a range of complex activities, it currently lacks the ability to explain its actions in terms that are interpretable by a human being.

It is envisionable that BunDLe-Net can be applied to a reinforcement learning agent to understand its interactions with an environment. By doing so, we can gain access to internal representations of the reinforcement learning agent and observe how it processes information. In place of the behaviour B, we could decode human-annotated actions of interest. This would then give us a model that preserves information about the actions of interest while also giving insights into how these decisions are arrived at and at which point in time such decisions are reached within the agent.

6.4. Conclusion

We commenced this dissertation with a fundamental question: *What does it take to gain an understanding of a complex dynamical system like the brain?* This question is motivated by the enormous amounts of data available from complex systems. Yet, it is unclear how to use this data to gain a holistic understanding of the system. We proposed a conceptual framework to deal with this problem and identified the need for causally-consistent abstractions to gain an understanding of the system. After thoroughly reviewing existing literature and state-of-the-art methods, we recognised a research gap: the need for a framework and algorithm to learn causally consistent abstractions from data. To address this research gap, we broke down our problem into three research questions, each of which falls into the domain of causality, algorithms and cognitive neuroscience.

To address our first question, that of inferring causal abstractions of dynamical systems without access to interventions, we develop a mathematical framework. This introduces the notion of Dynamical Causal Consistency (DCC) of a low-level system with an abstract model. Within our mathematical framework, we prove a theorem that allows a causal interpretation of the model derived from purely observational data under specific conditions (Markovianity), thus answering the first question. Unlike previous works, our framework also offers an elegant solution of how to learn causally consistent abstractions from data through our commutativity diagram Theorem.

Based on our mathematical framework, we create a learning architecture called BunDLe-Net and implement it in a Python toolbox in Chapter 4, thus effectively answering our second question. BunDLe-Net is presented as a neuronal manifold learning in Chapter 3, where we use it to learn causal abstractions of neuronal activity of the nematode *C. elegans* in a behavioural context. Quantitative evaluations of BunDLe-Net based on behavioural decoding and autoregressive capabilities show that it outperforms other state-of-the-art techniques. Our embedding also provides the most visually interpretable embeddings, which is a direct consequence of being a causal abstraction that preserves behavioural information. The topology of the trajectory tells us about behavioural decision-making at the level of neuronal activity. Our analysis in the publication both confirms existing knowledge and reveals novel insights.

Apart from validating our algorithm on Calcium imaging from *C. elegans*, we tested it on different settings, including rat data. Despite encountering spiking neuronal data and a distinct behavioural paradigm, BunDLe-Net outperforms state-of-the-art methods. In our subsequent experiments, we employ a novel approach, adapting BunDLe-Net

to embed subsystems separately, thus providing each embedding axis with anatomical interpretability. Throughout these experiments, BunDLe-Net consistently reveals a clear topology of the dynamics that is seldom captured by other methods. The underlying topology of the manifolds provides insights into the encoding and dynamic processing of information in the brain, along with the decision-making process. Comparable topologies in the neuronal activities of different individuals within the same setting indicate a similar information processing approach, with occasional minor individual differences. BunDLe-Net's capacity to visually illuminate how neuronal circuits process information and generate behaviour effectively addresses our third research question.

Our exploration of the research questions has yielded contributions across the domains of causality, computer science, and neuroscience while opening up numerous avenues for new research. While our focus was on dynamical systems in neuroscience, there exists potential for further extension of our work into more generalised settings and the broader field of artificial intelligence. Starting from basic but vital foundational principles, we created a mathematical framework, paving the way for developing a class of computational algorithms. This convergence of mathematical and computational methodologies found its purpose in the domain of neuroscience, offering insights into the complexities of real-world neuronal data. Thus, this interdisciplinary work of research has made the long journey from fundamental principles to mathematical framework to algorithm to practical application in neuroscience.

Understanding the complexities of the human brain is a challenging pursuit, with no clear consensus on what constitutes a comprehensive understanding. Amidst the various approaches, this thesis introduces our unique perspective through causal abstractions. We view this work as a small but foundational step in contributing to the broader question of deciphering systems as complex and intricate as the human brain.

Bibliography

Saul Kato, Harris S Kaplan, Tina Schrödel, Susanne Skora, Theodore H Lindsay, Eviatar Yemini, Shawn Lockery, and Manuel Zimmer. Global brain dynamics embed the motor command sequence of caenorhabditis elegans. *Cell*, 163(3):656–669, 2015.

Andres D. Grosmark and György Buzsáki. Diversity in neural firing dynamics supports both rigid and learned hippocampal sequences. *Science*, 351(6280):1440–1443, 2016. doi: 10.1126/science.aad1935. URL `https://www.science.org/doi/abs/10.1126/science.aad1935`.

Gordon M Shepherd, Jason S Mirsky, Matthew D Healy, Michael S Singer, Emmanouil Skoufos, Michael S Hines, Prakash M Nadkarni, and Perry L Miller. The human brain project: neuroinformatics tools for integrating, searching and modeling multidisciplinary neuroscience data. *Trends in neurosciences*, 21(11):460–468, 1998.

Huda Akil, Maryann E. Martone, and David C. Van Essen. Challenges and opportunities in mining neuroscience data. *Science*, 331(6018):708–712, February 2011. doi: 10.1126/science.1199305. URL `https://doi.org/10.1126/science.1199305`.

Jakob Runge, Sebastian Bathiany, Erik Bollt, Gustau Camps-Valls, Dim Coumou, Ethan Deyle, Clark Glymour, Marlene Kretschmer, Miguel D Mahecha, Jordi Muñoz-Marí, et al. Inferring causation from time series in earth system sciences. *Nature communications*, 10(1):2553, 2019.

Krzysztof Chalupka, Tobias Bischoff, Pietro Perona, and Frederick Eberhardt. Unsupervised discovery of el nino using causal feature learning on microlevel climate data. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, UAI'16, page 72–81, Arlington, Virginia, USA, 2016a. AUAI Press. ISBN 9780996643115.

Boyuan Chen, Kuang Huang, Sunand Raghupathi, Ishaan Chandratreya, Qiang Du, and Hod Lipson. Automated discovery of fundamental variables hidden in experimental data. *Nature Computational Science*, 2(7):433–442, 2022.

Maziar Raissi and George Em Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

Terrence J Sejnowski, Patricia S Churchland, and J Anthony Movshon. Putting big data to good use in neuroscience. *Nature neuroscience*, 17(11):1440–1441, 2014.

Ziqiang Wei, Bei-Jung Lin, Tsai-Wen Chen, Kayvon Daie, Karel Svoboda, and Shaul Druckmann. A comparison of neuronal population dynamics measured with calcium imaging and electrophysiology. *PLoS Comput. Biol.*, 16(9):e1008198, September 2020.

Fabrizio De Vico Fallani, Martina Corazzol, Jenna R. Sternberg, Claire Wyart, and Mario Chavez. Hierarchy of neural organization in the embryonic spinal cord: Granger-causality graph analysis of in vivo calcium imaging data. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 23(3):333–341, 2015. doi: 10.1109/TNSRE.2014.2341632.

Geoffrey Migault, Thijs L van der Plas, Hugo Trentesaux, Thomas Panier, Raphaël Candelier, Rémi Proville, Bernhard Englitz, Georges Debrégeas, and Volker Bormuth. Whole-brain calcium imaging during physiological vestibular stimulation in larval zebrafish. *Current Biology*, 28(23):3723–3735, 2018.

Jeffrey P. Nguyen, Frederick B. Shipley, Ashley N. Linder, George S. Plummer, Mochi Liu, Sagar U. Setru, Joshua W. Shaevitz, and Andrew M. Leifer. Whole-brain calcium imaging with cellular resolution in freely behaving caenorhabditis elegans. *Proceedings of the National Academy of Sciences*, 113(8), December 2015. doi: 10.1073/pnas.1507110112. URL `https://doi.org/10.1073/pnas.1507110112`.

Misha B Ahrens, Michael B Orger, Drew N Robson, Jennifer M Li, and Philipp J Keller. Whole-brain functional imaging at cellular resolution using light-sheet microscopy. *Nature methods*, 10(5):413–420, 2013.

Eric Jonas and Konrad Paul Kording. Could a neuroscientist understand a microprocessor? *PLoS computational biology*, 13(1):e1005268, 2017.

John W. Krakauer, Asif A. Ghazanfar, Alex Gomez-Marin, Malcolm A. MacIver, and David Poeppel. Neuroscience needs behavior: Correcting a reductionist bias. *Neuron*, 93(3):480–490, 2017. ISSN 0896-6273. doi: https://doi.org/10.1016/j.neuron.2016.12.041. URL `https://www.sciencedirect.com/science/article/pii/S0896627316310406`.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021. doi: 10.1109/JPROC.2021.3058954.

Judea Pearl. *Causality*. Cambridge University Press, September 2009. doi: 10.1017/cbo9780511803161. URL `https://doi.org/10.1017/cbo9780511803161`.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, prediction, and search*. Adaptive Computation and Machine Learning series. Bradford Books, Cambridge, MA, 2 edition, January 2001.

Steven H Strogatz. *Nonlinear dynamics and chaos*. CRC Press, London, England, 2 edition, May 2019.

Erik P. Hoel, Larissa Albantakis, William Marshall, and Giulio Tononi. Can the macro beat the micro? integrated information across spatiotemporal scales. *Neuroscience of Consciousness*, 2016(1), 08 2016. ISSN 2057-2107. doi: 10.1093/nc/niw012. URL https://doi.org/10.1093/nc/niw012. niw012.

Erik P Hoel. When the map is better than the territory. *Entropy*, 19(5):188, 2017.

Moritz Grosse-Wentrup, Akshey Kumar, Anja Meunier, and Manuel Zimmer. Neurocognitive multilevel causal modeling: A framework that bridges the explanatory gap between neuronal activity and cognition. *bioRxiv*, 2023. doi: 10.1101/2023.10.27.564404. URL https://www.biorxiv.org/content/early/2023/10/27/2023.10.27.564404.

Philipp Haueis. Descriptive multiscale modeling in data-driven neuroscience. *Synthese*, 200(2):129, 2022.

John A Lee, Michel Verleysen, et al. *Nonlinear dimensionality reduction*, volume 1. Springer, 2007.

Bernhard Schölkopf. *Causality for Machine Learning*, page 765–804. Association for Computing Machinery, New York, NY, USA, 1 edition, 2022. ISBN 9781450395861. URL https://doi.org/10.1145/3501714.3501755.

Xiaowen Chen, Faustine Ginoux, Martin Carbo-Tano, Thierry Mora, Aleksandra M Walczak, and Claire Wyart. Granger causality analysis for calcium transients in neuronal networks, challenges and improvements. *eLife*, 12:e81279, feb 2023. ISSN 2050-084X. doi: 10.7554/eLife.81279. URL https://doi.org/10.7554/eLife.81279.

Amitava Banerjee, Sarthak Chandra, and Edward Ott. Network inference from short, noisy, low time-resolution, partial measurements: Application to c. elegans neuronal calcium dynamics. *Proceedings of the National Academy of Sciences*, 120(12), March 2023. doi: 10.1073/pnas.2216030120. URL https://doi.org/10.1073/pnas.2216030120.

Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. *Causal inference in statistics: A primer*. John Wiley & Sons, 2016.

Peter Spirtes and Clark Glymour. An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*, 9(1):62–72, April 1991. doi: 10.1177/089443939100900106. URL https://doi.org/10.1177/089443939100900106.

Thomas Verma and Judea Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, pages 255–270, 1990.

Erik P. Hoel, Larissa Albantakis, and Giulio Tononi. Quantifying causal emergence shows that macro can beat micro. *Proceedings of the National Academy of Sciences*, 110(49):19790–19795, November 2013. doi: 10.1073/pnas.1314922110. URL https://doi.org/10.1073/pnas.1314922110.

Sebastian Weichwald, Timm Meyer, Ozan Özdenizci, Bernhard Schölkopf, Tonio Ball, and Moritz Grosse-Wentrup. Causal interpretation rules for encoding and decoding models in neuroimaging. *Neuroimage*, 110:48–59, 2015.

Alexander Gebharter and Nina Retzlaff. A new proposal how to handle counterexamples to markov causation à la cartwright, or: fixing the chemical factory. *Synthese*, 197 (4):1467–1486, Apr 2020. ISSN 1573-0964. doi: 10.1007/s11229-018-02014-7. URL `https://doi.org/10.1007/s11229-018-02014-7`.

Paul K. Rubenstein, Sebastian Weichwald, Stephan Bongers, Joris M. Mooij, Dominik Janzing, Moritz Grosse-Wentrup, and Bernhard Schölkopf. Causal consistency of structural equation models. *Uncertainty in Artificial Intelligence: proceedings of the Thirty-Third Conference*, 2017.

Krzysztof Chalupka, Frederick Eberhardt, and Pietro Perona. Causal feature learning: an overview. *Behaviormetrika*, 44:137–164, 2017.

Krzysztof Chalupka, Pietro Perona, and Frederick Eberhardt. Visual causal feature learning. *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, pages 181–190, 2015.

Krzysztof Chalupka, Frederick Eberhardt, and Pietro Perona. Multi-level cause-effect systems. *Artificial Intelligence and Statistics*, 2016b.

J. Mooij, D. Janzing, and B. Schölkopf. From ordinary differential equations to structural causal models: the deterministic case. In *Proceedings of the Twenty-Ninth Conference Annual Conference on Uncertainty in Artificial Intelligence*, pages 440–448, Corvallis, Oregon, 2013. AUAI Press.

P. K. Rubenstein, S. Bongers, B. Schölkopf, and J. M. Mooij. From deterministic ODEs to dynamic structural causal models. In *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence (UAI)*, August 2018. URL `http://auai.org/uai2018/proceedings/papers/43.pdf`.

Stephan Bongers, Tineke Blom, and Joris M. Mooij. Causal modeling of dynamical systems, 2022.

Dominik Janzing, Paul Rubenstein, and Bernhard Schölkopf. Structural causal models for macro-variables in time-series, 2018.

N. Wahlström, Schön T. B., and M. P. Desienroth. From pixels to torques: Policy learning with deep dynamical models. *Deep Learning Workshop at the 32nd International Conference on Machine Learning ICML 2015*, 2015.

Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in*

Neural Information Processing Systems, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/a1afc58c6ca9540d057299ec3016d726-Paper.pdf.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

Miguel Jaques, Michael Burke, and Timothy Hospedales. Physics-as-inverse-graphics: Unsupervised physical parameter estimation from video. In *Eighth International Conference on Learning Representations*, pages 1–16, 2020.

Amy Zhang, Rowan Thomas McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=-2FCwDKRREu.

Thomas N. Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=H1gax6VtDB.

Ankesh Anand, Evan Racah, Sherjil Ozair, Yoshua Bengio, Marc-Alexandre Côté, and R Devon Hjelm. Unsupervised state representation learning in atari. *Advances in neural information processing systems*, 32, 2019.

David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. MIT Press, 1982.

Peiran Gao, Eric Trautmann, Byron Yu, Gopal Santhanam, Stephen Ryu, Krishna Shenoy, and Surya Ganguli. A theory of multineuronal dimensionality, dynamics and measurement. *BioRxiv*, page 214262, 2017.

Misha B Ahrens, Jennifer M Li, Michael B Orger, Drew N Robson, Alexander F Schier, Florian Engert, and Ruben Portugues. Brain-wide neuronal dynamics during motor adaptation in zebrafish. *Nature*, 485(7399):471–477, 2012.

Wieland Brendel, Ranulfo Romo, and Christian K Machens. Demixed principal component analysis. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper_files/paper/2011/file/f4a331b7a22d1b237565d8813a34d8ac-Paper.pdf.

Dmitry Kobak, Wieland Brendel, Christos Constantinidis, Claudia E Feierstein, Adam Kepecs, Zachary F Mainen, Xue-Lian Qi, Ranulfo Romo, Naoshige Uchida, and Christian K Machens. Demixed principal component analysis of neural population data. *eLife*, 5, April 2016. doi: 10.7554/elife.10989. URL https://doi.org/10.7554/elife.10989.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2018. URL https://arxiv.org/abs/1802.03426.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Ding Zhou and Xue-Xin Wei. Learning identifiable and interpretable latent models of high-dimensional neural activity using pi-vae. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7234–7247. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/510f2318f324cf07fce24c3a4b89c771-Paper.pdf.

Steffen Schneider, Jin Hwa Lee, and Mackenzie Weygandt Mathis. Learnable latent embeddings for joint behavioural and neural analysis. *Nature*, pages 1–9, 2023.

Alon Rubin, Liron Sheintuch, Noa Brande-Eilat, Or Pinchasof, Yoav Rechavi, Nitzan Geva, and Yaniv Ziv. Revealing neural correlates of behavior without behavioral measurements. *Nature communications*, 10(1):4745, 2019.

Lav R. Varshney, Beth L. Chen, Eric Paniagua, David H. Hall, and Dmitri B. Chklovskii. Structural properties of the caenorhabditis elegans neuronal network. *PLoS Computational Biology*, 7(2):e1001066, February 2011. doi: 10.1371/journal.pcbi.1001066. URL https://doi.org/10.1371/journal.pcbi.1001066.

Ferris Jabr. The connectome debate: Is mapping the mind of a worm worth it. *Scientific American*, 18, 2012.

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

François Chollet et al. Keras: The python deep learning library. *Astrophysics source code library*, pages ascl–1806, 2018.

Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.

J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

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

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. URL `https://arxiv.org/abs/1412.6980`.

Aiga Suzuki and Hidenori Sakanashi. Pci: Principal component initialization for deep autoencoders. In Igor V. Tetko, Věra Kůrková, Pavel Karpov, and Fabian Theis, editors, *Artificial Neural Networks and Machine Learning – ICANN 2019: Deep Learning*, pages 165–169, Cham, 2019. Springer International Publishing. ISBN 978-3-030-30484-3.

Xu-Die Ren, Hao-Nan Guo, Guan-Chen He, Xu Xu, Chong Di, and Sheng-Hong Li. Convolutional neural network based on principal component analysis initialization for image classification. In *2016 IEEE First International Conference on Data Science in Cyberspace (DSC)*, pages 329–334, 2016. doi: 10.1109/DSC.2016.18.

Ricardo Augusto de Melo Reis, Hércules Rezende Freitas, and Fernando Garcia De Mello. Cell calcium imaging as a reliable method to study neuron–glial circuits. *Frontiers in Neuroscience*, 14:569361, 2020.

Cosma Rohilla Shalizi. Methods and techniques of complex systems science: An overview. *arXiv*, 2003. doi: 10.48550/ARXIV.NLIN/0307015. URL `https://arxiv.org/abs/nlin/0307015`.

Harris S. Kaplan, Annika L.A. Nichols, and Manuel Zimmer. Sensorimotor integration in caenorhabditis elegans: a reappraisal towards dynamic and distributed computations. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1758): 20170371, September 2018. doi: 10.1098/rstb.2017.0371. URL `https://doi.org/10.1098/rstb.2017.0371`.

Mohammed Ali, Mark W Jones, Xianghua Xie, and Mark Williams. Timecluster: dimension reduction applied to temporal data for visual analytics. *The Visual Computer*, 35 (6-8):1013–1026, 2019.

Douglas R. Hofstadter. *Gödel, Escher, Bach: An Eternal Golden Braid*, chapter Brains and Thoughts. Basic Books Inc., 20th-anniversary edition edition, 1979.

Frédéric Chazal and Bertrand Michel. An introduction to topological data analysis: fundamental and practical aspects for data scientists. *Frontiers in artificial intelligence*, 4:108, 2021.

Yara Skaf and Reinhard Laubenbacher. Topological data analysis in biomedicine: A review. *Journal of Biomedical Informatics*, 130:104082, 2022. ISSN 1532-0464. doi: https://doi.org/10.1016/j.jbi.2022.104082. URL https://www.sciencedirect.com/science/article/pii/S1532046422000983.

Richard Scheines. An introduction to causal inference. 1997. doi: 10.1184/R1/6490904.V1. URL https://kilthub.cmu.edu/articles/An_Introduction_to_Causal_Inference/6490904/1.

Paride Antinucci, Adna Dumitrescu, Charlotte Deleuze, Holly J Morley, Kristie Leung, Tom Hagley, Fumi Kubo, Herwig Baier, Isaac H Bianco, and Claire Wyart. A calibrated optogenetic toolbox of stable zebrafish opsin lines. *eLife*, 9:e54937, mar 2020. ISSN 2050-084X. doi: 10.7554/eLife.54937. URL https://doi.org/10.7554/eLife.54937.

Or A. Shemesh, Dimitrii Tanese, Valeria Zampini, Changyang Linghu, Kiryl Piatkevich, Emiliano Ronzitti, Eirini Papagiakoumou, Edward S. Boyden, and Valentina Emiliani. Temporally precise single-cell-resolution optogenetics. *Nature Neuroscience*, 20(12): 1796–1806, November 2017. doi: 10.1038/s41593-017-0018-8. URL https://doi.org/10.1038/s41593-017-0018-8.

A. Appendix

A.1. Links

BunDLe-Net toolbox
https://github.com/akshey-kumar/BunDLe-Net

Code for experiments - anatomical interpretable latent dimensions
https://github.com/akshey-kumar/BunDLe-Net/tree/main/experimental/anatomical_interpretability_of_axes

BunDLe-Net - Rotating plots for all worm embeddings - comparable embeddings
https://github.com/akshey-kumar/BunDLe-Net/tree/main/figures/rotation_comparable_embeddings

BunDLe-Net - Rotating plots for all worm embeddings - independent embeddings
https://github.com/akshey-kumar/BunDLe-Net/blob/main/figures/worms_separately/rotation_BunDLeNet_worm_3.gif

BunDLe-Net - Rotating plots for all worm embeddings - interpretable latent dimensions
https://github.com/akshey-kumar/BunDLe-Net/tree/main/figures/rotation_axis_decomp

Competing Neuronal Manifold Learning techniques - comparison and evaluation
https://github.com/akshey-kumar/comparison-algorithms

A.2. Evaluation pipeline for BunDLe-Net

This schematic shows the pipeline that we used for the evaluation of BunDLe-Net, and various other embedding algorithms in Publication 3. The flow of information processing in the schematic is from top to bottom. The code for the implementation can be found at https://github.com/akshey-kumar/comparison-algorithms. Note that in the BunDLe-Net toolbox Chapter 4 however, we are not performing evaluation, but only embedding. Hence the train and test set of the schematic will be the same dataset here.

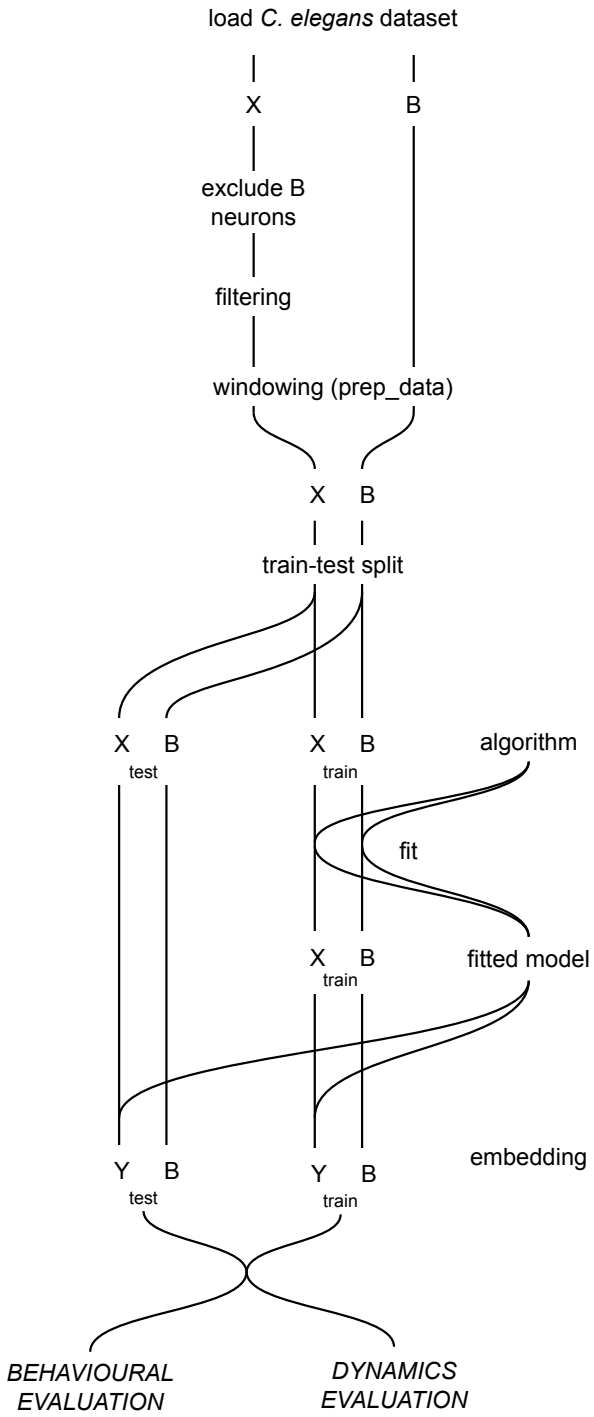

Figure A.1.: Schematic of BunDLe-Net toolbox pipeline and evaluation