Machine Learning Topological Defects of Liquid Crystals in Two Dimensions

by

Michael Walters

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Science
in
Physics

Waterloo, Ontario, Canada, 2019

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Over the recent few years, condensed matter physics has keenly been testing out the compelling techniques from the toolbox of machine learning. Order parameters, phase transitions, and other thermodynamic quantities have been learned for different systems, including the Ising model and its variants, the XY model, many-fermion systems, and more. In the following work, we apply three different machine learning architectures towards the problem of identification of topological defects, specifically those seen in square-confined, two-dimensional liquid crystals, though the concepts and methods outlined are by no means restricted to this application.

Simulated liquid crystals were modeled by off-lattice, hard, rod-like molecules via Monte Carlo. At high molecular density, geometric frustrations from the square confinement cause the nematic director field to develop an inhomogeneous pattern containing various topological defects as the main physical feature. We show machine learning can capture the correlation between defect positions and the nematic directors around them to classify these different topologies. With a feed-forward neural network, pre-sorting the off-lattice simulation data in a coarse-grained fashion is necessary for effective learning. We also bring to light the often overlooked reason for why pre-sorting was needed in this case. As a more generalizable approach, we show that a recurrent neural network can also learn the topologies without pre-sorting or any other input engineering. Finally, we investigate how unsupervised learning with principal component analysis can be used to accurately pinpoint the location and the type of winding disclinations in our two-dimensional liquid crystal model.

Acknowledgements

This would never have been possible without the stalwart support and mentoring from Jeff Z. Y. Chen, illuminating discussions from Qianshi Wei, and the wellspring of love from friends, family, and my dearest, Liesbeth.

I also thank the Natural Sciences and Engineering Research Council of Canada for financial support and Compute Canada for providing computational resources.

Table of Contents

List of Figures

Abbreviations

CNN	Convolution neural network
DTUX	The D, T, U, and X defect state topologies
FNN	Feed-forward neural network
I-N	Isotropic-nematic transition
LC	Liquid crystal
LSTM	Long short-term memory neural network component
MC	Monte Carlo
MCS	Monte Carlo sweep: N Monte Carlo steps for system size N
ML	Machine learning
NN	Neural network
PCA	Principal Component Analysis
RNN	Recurrent neural network
WA	Winding alignment method introduced in Section 3.3
WD	Winding difference method introduced in Section 3.3

List of Symbols

a	Normal math typeface represents scalars unless otherwise specified
$\mathbf{a}$	Bold-faced lowercase variables represent one-dimensional vectors
$\mathbf{A}$	Bold-faced uppercase variables represent two-dimensional matrices

Even if the open windows of science
at first make us shiver after the cozy
indoor warmth of traditional humanizing myths,
in the end the fresh air brings vigor,
and the great spaces have a splendor of their own.

Bertrand Russell

Chapter 1

Introduction

A new type of matter

In 1888 the Austrian botanist Friedrich Reinitzer remarked the curious melting behaviour of cholesteryl benzoate [75] (English translation: [76]). The chemical displayed two distinct melting points: first melting into a cloudy liquid, then upon further heating, melting again into a clear liquid. Reinitzer promptly corresponded his findings, along with samples of cholesterol benzoate and cholesterol acetate, to physicist Otto Lehmann in March that same year [53]. Lehmann looked more closely at these samples and discovered the crystalline order present in the liquid. Publishing his findings in 1889 [56], liquid crystal (LC) research began in earnest.

As the name implies, liquid crystals are unique in their properties that combine both those of liquids (such as flow, inability to support shear, and the formation/coalescence of droplets) and those of crystals (such as anisotropy of electromagnetic/optical properties and the periodic arrangement of molecules in spatial dimensions) [4]. Having a controllable dynamic state gives LCs interesting applications across many material domains. For example, in the nanosciences, it is typically desirable for carbon nanotubes (CNTs) to have a high degree of alignment for their applications. CNTs can be dispersed in a liquid LC solvent, and transitioning the LC into its aligned state aligns the CNTs with it [26, 27, 62].

Another, exciting emergent technology are liquid crystal elastomers (LCEs) [92]. LCEs combine liquid crystal anisotropy and rubber elasticity to create unique materials that may feature properties like actuation, soft elasticity, birefringence, and may even have applications in 4D printed biomedical devices. These mechanics are often temperature driven, causing volumetric changes or transparency.

Nature had long ago uncovered their utility, though. In the production of spider dragline silk, an aqueous solution of the silk fibroin takes on a nematic phase as an orienting mechanism at an intermediate stage of the process [48, 52, 72]. Cases of LC mesophases have been observed in certain DNA packings: inside the heads of bacteriophages [29], dinoflagellates [60], or plasmid DNA within bacteria [74]. Inside bacteriophages, the packing takes on a columnar shape within concentric rings [11]. In dinoflagellates, the chromosomes assume a twisted cholesteric phase. And at physiological concentrations, *in vitro* plasmid DNA of *Escherichia coli* bacteria show birefringent LC textures of a cholesteric phase [74]. LC phases have been found both *in vivo* as well as *in vitro* for major classes of biological compounds including lipids, proteins, carbohydrates and nucleic acids [37]. Nematic and chiral nematic (cholesteric) phases are most common in this domain, but hexagonal columnar (such as in DNA) and smectic phases (such as in the arrangement of amylopectin side chains in starch) have also been observed [37].

Different topologies may be traversed by the modification of confinement geometry or an external field, as in LC displays [5, 28, 51, 102]. Controlling the nematic director alignment allows control over its optical properties. Defect regions, where the nematic alignment is divergent, can also be utilized in fascinating ways, such as nanomaterial self-assembly. Gurevich et al. [36], simulating a nanoparticle and LC mixture under a variety of quenches and concentrations, found that nanoparticles could be concentrated in defects or in the channels and pockets formed by slower growing regions of the nematic-isotropic interface as nematic regions expanded. Directing the assembly of nanoparticles into addressable arrangements can lead to materials with high processability, self-healing properties, and reversible control [36].

Such defects are at the heart of this study. At nematic densities, many stable or metastable states can exist. We present several states, generated from our Monte Carlo, of LCs in square confinement in Figure 1.1, replicating numerical solutions from Yao, Zhang, and Chen [103]. These comprise the subjects of our work. States are characterized by the different ways in which the local order parameters around the features couple with their locations. Developing a characterization procedure to categorize these defect states is a challenging task, and a neural net presents itself as an ideal candidate.

Machine learning in condensed matter

In the past few years, machine learning (ML) has made a significant foray into condensed matter research and other areas of physics. The merge is not too surprising though since they share some similar methods and goals. Both analyze large amounts of data to learn

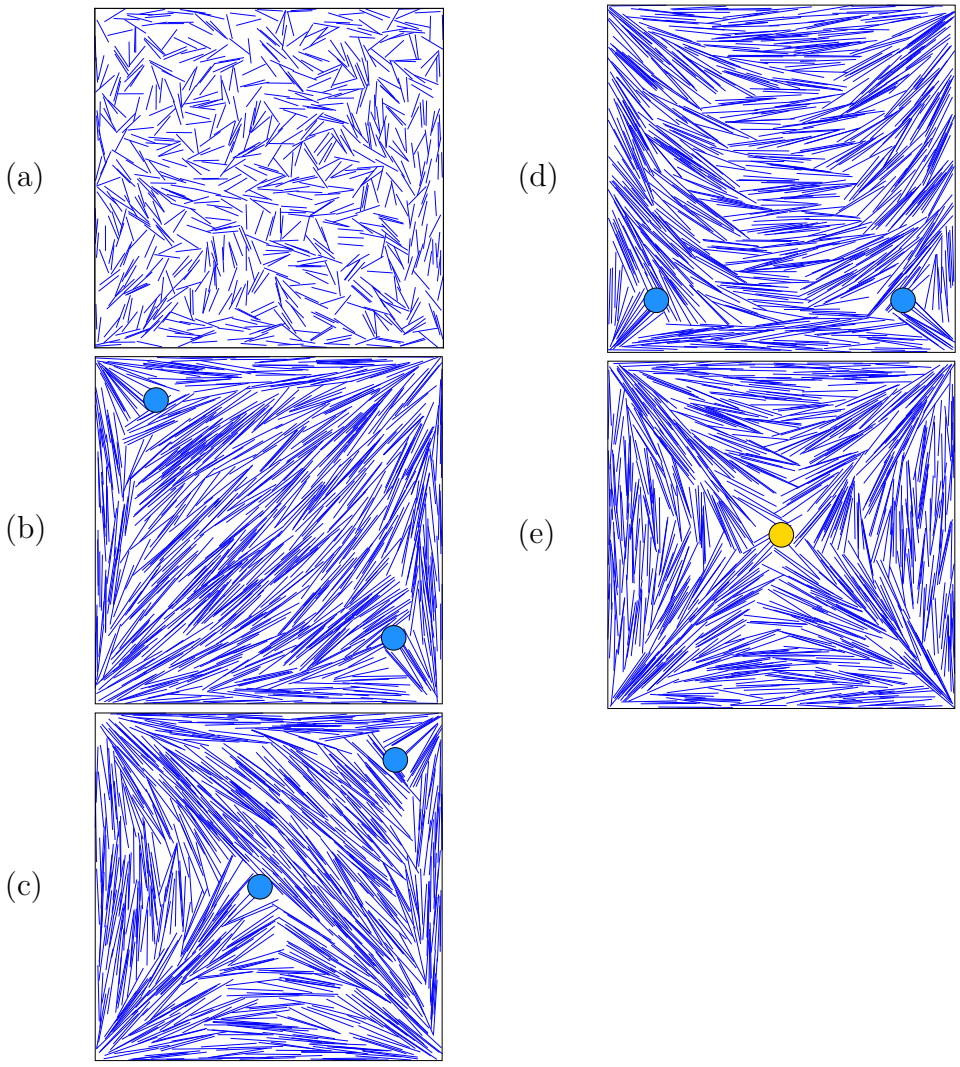


Figure 1.1: Past the critical density, lyotropic LCs phase change from isotropic (a) to nematic. In the nematic phase, different topologies may persist. In (b)-(e) are four examples which will be the topological subjects of our study. The defect state in (b) has a diagonal (D) pattern, and the nematic textures in (c)-(e) resemble the tilted letter T, letter U, and letter X. Blue and yellow circles mark $-1/2$ and -1 winding disclinations, respectively. Images were generated from our Monte Carlo (details given in Section 4.1).

about, and predict the behaviour of complex systems. After all, much of ML is (or at least borrows from) statistical methodology, especially unsupervised learning. Their approaches are quite different, though. Whereas physicists use their knowledge and intuition with sound mathematical steps, ML conversely extracts physics of the system from collected data.

One of the marvels and advantages of this technique is how little statistical-physics information (energies, order parameters, etc.) is needed for the classification of states or the pinpointing of critical physical parameters. Even relatively simple neural network models can learn phase-transition temperatures, order parameters, and quantum-state tomography, from the information of a simple feature such as position coordinates in an off-lattice model or the location and value of spins in an Ising model. This could all be done without any knowledge of the original Hamiltonian or interaction potential energies [6, 8, 9, 10, 41, 45, 63, 89, 93, 95, 96, 97, 98, 99, 104, 105, 106].

Convolutional neural nets (CNNs), the star of modern image recognition, have shown their use in multiple physical systems. Without any knowledge of the energy or locality conditions of the Ising model, a CNN can learn subtly different phases, difficult to distinguish even to the human eye [9]. Learning quantum phase transitions of many-fermion systems was also bested by them [8], even in three dimensions [18]. CNNs were also recently tested to see if the Berezinskii-Kosterlitz-Thouless vortex binding transition in the XY model could be learned, with some interesting results. Beach, Golubeva, and Melko [6], found CNNs appeared to nearly converge on the critical temperature from theory, but found it was likely learning from the magnetization, and not the real spatial relationship of vortices. Some feature engineering of the raw spin data into vortices before input to the NN was then explored to correct for this, with success. In another study, Zhang, Liu, and Wei [105] also used CNNs to learn the percolation transition and BKT transition in the XY and Generalized XY models, but again required, sometimes substantial, feature engineering. Minimal or no feature engineering is a goal in physical science applications of ML, not only for the sake of complexity, but part of the utility and interest in these studies is that the NN is supposed to do all the work in uncovering the underlying physics. Feature engineering aids the NN, pulling from our present understanding of the subject model. Thus, although many conventional, symmetry-breaking, phase transitions can be effectively captured by ML approaches due to the local nature of the order parameter, topological phase transitions are particularly difficult for an NN to learn. The difficulty stems from the fact that topological phase transitions are characterized by the proliferation of non-local, topological defects that are suppressed in the topologically ordered phase [54, 81].

Learning defects in liquid crystals

The problem of learning defects in our LC system is further compounded by the off-lattice nature. The incredible power of CNNs cannot be leveraged in such systems because CNNs require a lattice input structure. Lattice approximations are not an ideal route because such approximations often bring in heuristics (e.g. the distribution of lattice sites) and lose information, especially more local details which are very important for learning topological phase transitions and defects. Hence, we must solve how to capture the main features in a topological defect state when the correlation between defect positions and the physical properties around them is the vital property. In this thesis different ways of incorporating existing NNs for this purpose are discussed.

One of the main obstacle comes from the off-lattice format of our system. Lattice applications can take advantage of the spatially consistent mapping of the physical space to the NN input. As our study shows, this sorting takes care of the position correlating of features. We propose two different architectures for navigating this problem.

First, a modified FNN model performs a coarse grain sorting of rod-molecules before input to the network. This degree of positional sorting was found to be successful even for very coarse sorting, but is certainly application-dependent. We also present a more general method, using a recurrent neural network (RNN), that avoids the need for any presorting. The RNN is a neural network specialized in correlating sequential information (e.g. analyzing a time series of images [66], or predicting upcoming words in a sentence based on previous words [19, 65]). We propose a scheme to feed the Cartesian coordinates and orientational data of molecules sequentially, akin to three time sequenced images, which enables the RNN to correlate these three features. Turning temporal correlation to spatial correlation, an RNN can efficiently identify different states in the original raw data, without any of the coarse-graining or presorting needed for the FNN. Our findings on this matter were recently published in June of this year [94].

The methods so far discussed have all been "supervised" methods wherein each image has an associated label that helps guide the NN in its learning. Another paradigm of ML architectures are considered unsupervised and deal with *unlabeled* data. Unsupervised learning (UL) is, in a sense, a more pure learning, from an artificial intelligence standpoint. In this paradigm, the network learns to classify data not from given labels, but from the actual structure of the data. Some unsupervised methods separate or cluster data. Another recently popular branch of UL called *generative networks* has opened up, where networks essentially try to reproduce data types from a training set. Auto-encoders are a popular language modeling tool [25], and generative adversarial networks can generate high quality reproductions across many data types, recently making waves with their seamless facial

image generators [46, 73].

In our third application of ML, we use Principle Component Analysis (PCA) on a dataset of many samples of small rod clusters to identify what sort of structure is present. PCA is a popular UL technique, but has been at play in the statistics community for over a hundred years since its proposal in 1901 by Karl Pearson [70]. Specifically, we are interested if PCA can identify winding disclinations from the set of $\pm 1/2, \pm 1$ winding numbers. We intend to soon publish these results. The winding plays a crucial role in various contexts, such as in the topological classification of band insulators [88] and superconductors [50]. PCA is a dimension reduction technique that simultaneously transforms data along axes that maximize variance to present the most salient parts of a dataset. These transformations often relate to some underlying feature of the data.

Some successful efforts with PCA have determined phase transitions in Ising models, XY models, Kitaev chains, quantum spin chains, and the same classical LC system studied here [45, 93, 95, 96, 97]. In learning the Ising model temperature phase transition, PCA automatically produced the magnetization order parameter as the primary transformation for separating high and low temperature states [97]. On the XY model, Wang & Zhai [95, 96] found PCA delivered order parameters that (reasonably well) marked the phase transition in the XY model on multiple lattice structures. In another work, Jadrich, Lindquist, and Truskett [45] used PCA on an off-lattice LC system to locate the I-N phase transition density, and again the algorithm produced the classical order parameter used to predict the transition.

We find that PCA can indeed interpret defect sample structures properly and with reasonable effectiveness. Deciding on the input vectors to the algorithm is non-trivial, and similar to the CNN work on the XY model, some feature engineering is necessary. Three different input engineering models were tested: Random, Winding Difference (WD), and Winding Alignment (WA). In short, the Random method cannot differentiate defect types, whereas the WD method can. The WA method cannot identify defect polarity (whether it is $+1$ or -1), but is more robust to unclear samples from the Monte Carlo.

Following this introduction, we will first look at some essential concepts in supervised machine learning and the methods used to discern topological defect states. In Chapter 3, we move on to unsupervised learning with PCA, providing an overview of its mechanics, some examples of its use in other works, and how it will be used for LC defect learning. Chapter 4 provides an overview of Monte Carlo molecular dynamics and how our liquid crystals were simulated with it, also of continuum-based liquid crystal simulations, with an outline of the Onsager model, and we take a close look at what orientational order is

in LCs and how to quantify it. The proceeding two chapters will then present the results of our supervised and unsupervised studies.

Research in machine learning on problems in physics is in stride. It is already providing computational speed-ups, for example in Monte Carlo sampling both classical [42, 58, 101] and quantum [12, 59, 68, 84]. Unsupervised networks can be used to generate new physical observables and quantum states [20, 78, 82, 90]. Physical structure can be learned to determine useful quantities like critical temperatures, order parameters, etc. [6, 8, 9, 10, 41, 45, 63, 89, 93, 95, 96, 97, 98, 99, 104, 105, 106]. No doubt this direction of research will continue to deliver fascinating insights into state-of-the-art models, and perhaps even shake them up.

Chapter 2

Supervised Machine Learning of Liquid Crystal Defect States

In the following section we will explore a few different NN architectures for learning topological defects as it pertains to our LC system. As was outlined in the above, both FNNs and CNNs have proven largely effective in supervised learning phase transitions, order parameters, and other physical quantities for a handful of systems [6, 8, 9, 10, 18, 89, 98, 99, 104]. It should be noted that all but [98] are on lattice systems, so their methods developed are incompatible with our LC system. On the flip side, there is still fertile ground for research in this direction. To appreciate this roadblock, we will go over some neural net essentials.

The main function of an NN is to read the system configuration through an input layer (e.g. an image, text, or sound bite), process the information in hidden layers, and then generate an output. The output is often a classification estimation of the input, but may be another data structure (another image or sound bite for instance). We first consider classifying system states via an FNN, sketched in Figure 2.1(a).

Each arrow (an "edge") represents a function call that connects nodes in different layers. These nodes, or "perceptrons", are inspired by the neuron model of the brain and are the building blocks of many neural networks, including the FNN, and though individually quite simple, complex functions can be represented by networking many perceptrons together [80]. Going from input to output, data is repeatedly manipulated through function calls at each layer, with each containing their own network parameters — usually referred to as weights and biases. By varying network parameters the final output is consequently affected. Training a network then involves optimizing the network's

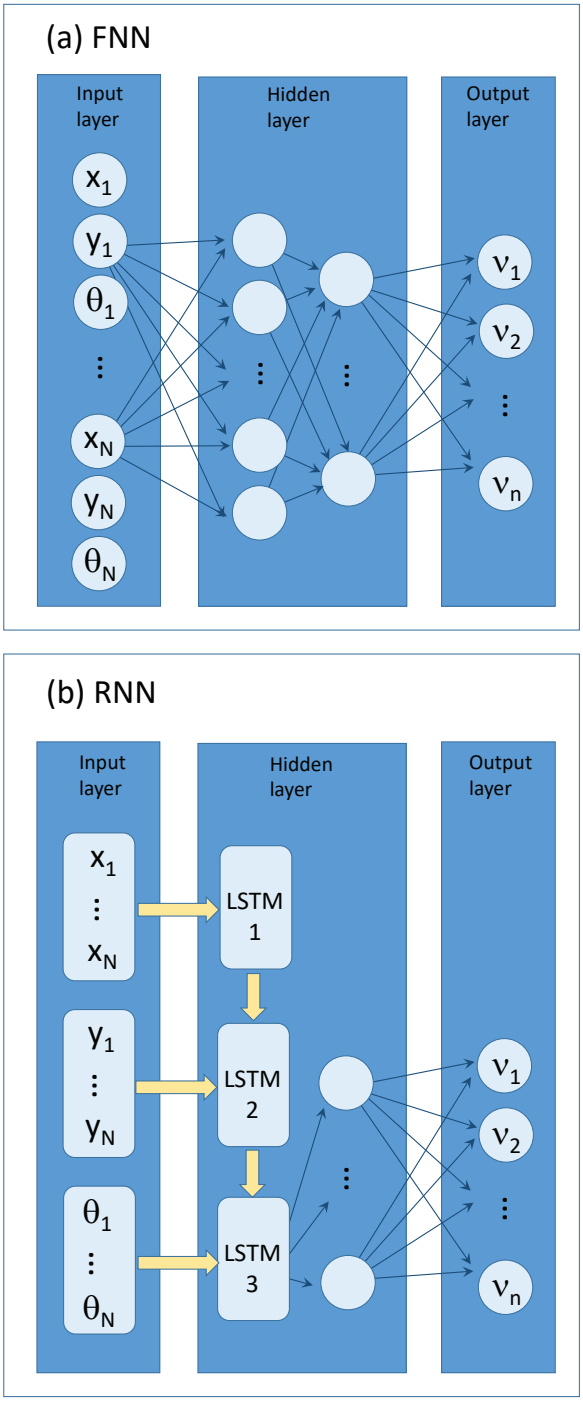


Figure 2.1: Schematics of (a) an FNN with two hidden layers and the sequence (x_l, y_l, θ_l) as input, where $l = 1, 2, ...N$, and (b) an RNN (unrolled) with the sequence x_l $(l = 1, 2, ...N)$ as the input to the first LSTM block, y_l $(l = 1, 2, ...N)$ to the second LSTM block, and θ_l $(l = 1, 2, ...N)$ to the third LSTM block. LSTM blocks also have LSTM-LSTM connections. The final output of the two networks are $\nu_1, \nu_2, ..., \nu_n$, where n is adjusted according to the type of features to be identified.

performance in producing the desirable output with respect to these network parameters. Within one "epoch" the network is trained once on the selected dataset, and in general multiple epochs are needed for a network to converge to an acceptable performance.

2.1 Supervised training and accuracy testing

As shown in Figure 2.1, the output layers of both networks are $(\nu_1, \nu_2, ..., \nu_n)$, each element being a number in the range $(0, 1)$ and $\sum_i \nu_i = 1$ by the normalization (softmax function) of the final output. Hence ν_i may be viewed as the confidence or probability that the tested image is in state i. For a vector of unnormalized outputs $(\nu_1^*, \nu_2^*, \cdots, \nu_n^*)$, softmax is the operation

$$\nu_i = \frac{e^{\nu_i^*}}{\sum_j e^{\nu_j^*}} \tag{2.1}$$

When training to learn the DTUX states, four output nodes ($n = 4$) are needed, and in our I-N transition study (Appendix A), $n = 2$ for isotropic or nematic.

In supervised training, each image carries an identifier α for the known state, and correspondingly the expected values $(\nu_1^\alpha, \nu_2^\alpha, ..., \nu_n^\alpha)$. To train the network on the I-N states, for example, α can be I or N, and

$$(\nu_1^I, \nu_2^I) = (1, 0),$$
$$(\nu_1^N, \nu_2^N) = (0, 1).$$

To train the network for the DTUX states, α can be D, T, U, or X, and

$$(\nu_1^D, \nu_2^D, \nu_3^D, \nu_4^D) = (1, 0, 0, 0),$$
$$(\nu_1^T, \nu_2^T, \nu_3^T, \nu_4^T) = (0, 1, 0, 0),$$
$$(\nu_1^U, \nu_2^U, \nu_3^U, \nu_4^U) = (0, 0, 1, 0),$$
$$(\nu_1^X, \nu_2^X, \nu_3^X, \nu_4^X) = (0, 0, 0, 1).$$

In the process of supervised training, we feed K configuration files to a network. The jth file, where $j = 1, 2, ..., K$, carries a known identifier α_j. Its data is then fed into the network to produce an output $(\nu_1^{(j)}, \nu_2^{(j)}, ..., \nu_n^{(j)})$, whose values depend on the network parameters, known as weights and biases. We attempt to match this output to the identifier α_j listed above. This is done by minimizing the cross entropy cost function,

$$S = -\frac{1}{K} \sum_{j=1}^{K} \sum_{i=1}^{n} \nu_i^{\alpha_j} \log \nu_i^{(j)}, \tag{2.2}$$

with respect to the network parameters. As one can see, in an idealized scenario when the network is perfectly trained, $S = 0$, whereas an untrained network has $S > 0$. Network parameters are updated multiple times per epoch. Plotting S as a function of epoch step can indicate how fast a network is learning (showing a decreasing S) or if the learning is not going well (showing a plateau after multiple epochs).

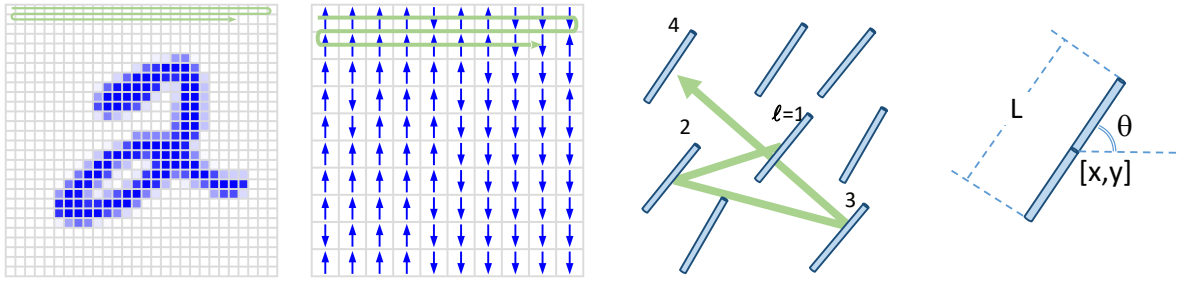
A separate set of test configuration data files, not used in the training, can benchmark the degree of learning during the training by providing a measurement known as the accuracy A. A total of k such data files are used and the jth file also carries a known identifier α_j. We consider an image correctly classified if the node with the maximum output across $(\nu_1^{(j)}, \nu_2^{(j)}, ..., \nu_n^{(j)})$ is the same as the output node with value 1 in $(\nu_1^{\alpha_j}, \nu_2^{\alpha_j}, ..., \nu_n^{\alpha_j})$. For example, if testing on an image with label $\alpha_j = T$ the NN output is $(0.03, 0.91, 0.03, 0.03)$, this would be a correct classification and $A_j = 1$ is assigned. Otherwise, $A_j = 0$. Averaged over all k test files, a mean accuracy A can be defined

$$A = \frac{1}{k} \sum_{j=1}^{k} A_j. \tag{2.3}$$

2.2 Learning topological defects

FNNs are able to accomplish a variety of image recognition tasks [83], a frequent benchmark test being the MNIST hand-written digit data set [7]. Figure 2.2(a) shows one such digit from the MNIST set. Increasing the size and number of layers in an FNN allows more complex features to be learned. A shallow network may learn edges, but a deeper network could identify eyes, for instance. CNNs are much like FNNs but with one crucial difference. Where an FNN has a unique connection to each input channel, a CNN only trains a very small sweeper of weights that sweep across the entire image for input to the next layer. This has the two-fold benefit of much fewer weights, and importantly, a translational invariance to the response of features in the image [55]. CNNs have thus been a natural choice for condensed matter research, and rightly so.

In an ordered domain, particles, molecules, etc., exhibit spatial correlations in long range. Two typical examples are shown in Figure 2.2. In the ferromagnetic state, within an ordered domain spins align in one direction, as illustrated in Figure 2.2(b). In the nematic liquid crystal state, within an ordered domain molecular directions are all aligned towards a common angle (Figure 2.2(c)). Here, an "image" is not a graphic image in the conventional sense. Rather, it is represented by the system configuration data containing physical features of each molecule (values of spins, angles specifying the orientations, etc.).



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 2.2: (a)-(d) Digitization of a two-dimensional hand-written image, Ising model, nematic state, and coordinates for a rod-like molecule, as well as (e)-(h) example configurations of different defect states generated by Monte Carlo. The parameters used to generate these example configurations are $[N, a/L] = [784, 6.32]$. The defect state in (e) has a diagonal (D) pattern, and the nematic textures in (f)-(h) resemble the tilted letter T, letter U, and letter X. The yellow and blue circles mark the defect locations of -1 and $-1/2$ winding numbers, respectively.

These ordered states, on the other hand, sometimes have topological defects in their substantially ordered background. Different patterns can be characterized by different ways in which the local order parameters around the defects couple with the locations of the defects. Typically, developing a characterization procedure to categorize these defects is a challenging task, but we have seen that FNNs and CNNs are more than capable. However, the are not readily appropriate for our off-lattice system.

A typical configuration file generated from the computer simulations has a single-line data structure $[l, x_l, y_l, \theta_l]$ for a given molecule, where l is the label of a molecule, and $[x_l, y_l, \theta_l]$ specify its Cartesian coordinates and angular orientation (see Figure 2.2(d)). Because the molecules are allowed to randomly move in space, the label of a molecule, l, has no relationship at all with the spatial coordinates (see Figure 2.2(c)). This can be contrasted with a simulated data file produced by a lattice model. In such a case, positional information is already embedded in the order in which molecules are labeled (one naturally reads the data in the same order every time), as demonstrated by the arrow in Figure 2.2(b). An NN that attempts to capture position-correlated patterns is thus implicitly aided from this direct mapping of physical position to the ordering of data in a lattice model.

Hence, we must solve how to capture the main features in a topological-defect state when the correlation between defect positions and the physical properties around them is the vital property. As it turns out, an FNN or CNN finds correlation between the order of appearance of data in the input and the physical features to be correlated. In an off-lattice model, index l has no correlation with $[x_l, y_l, \theta_l]$. Even if we use x_l, y_l as an input together with θ_l, an FNN cannot find the correlation between l and the input features $[x_l, y_l, \theta_l]$. This is discussed in Section 5.2.

Can the relationship between l and (x_l, y_l) be re-connected in a way that can be easily used by an FNN for an off-lattice dataset? Our first approach is a course-graining method which cuts the original simulation box into $m \times m$ equally sized cells where $m = 1, 2, 4, 8....$ The NN input is then ordered by cell index M, with the information inside each cell unordered. By such means, it is as though the system is approximated to a lattice form, with increasing accuracy as the cells become smaller and more numerous. An FNN can then begin to correlate physical features with position and identify topological defects with appropriate cell size.

2.2.1 DTUX topologies

The subject LC system has been the focus of recent theoretical and experimental studies due to its practical relevance [22, 31, 57, 86] and interesting theoretical aspects [13, 34, 57, 61, 77, 91, 103]. For example, the possible defect states were recently studied in-depth in terms of the continuum Landau-de Gennes model [77] and the Onsager model [103], producing a comparable ensemble of stable and metastable states as the result of minimizing the free energies in these different models.

The location of the nematic defect points are indicated in Figure 2.2(e)-(h) by colored circles. Here we see only two types of defects: those with winding number $-1/2$ (blue) and -1 (yellow). Although one could argue that the defects are point-like local objects, in a finite system, the overall nematic ordering in separate bulk areas are strongly connected with these local features. Each topology has a unique arrangement of the defect locations and overall nematic texture. Robinson et al. [77] also provide an in-depth look and comparison using several numerical methods of this system (off-lattice, on-lattice, and continuum). From the Landau-de Gennes framework, they found many of the same states the Onsager model produced by Yao, Zhang, and Chen [103].

Returning to the classic example of identifying hand-written numerical digits from 0 to 9, we make a comparison between the darkened pixels in this case with the topological defects in our confined liquid crystal system. In a sense, what a neural network looks for is the feature correlation of the spatial position of the darkened pixels in these images. In Appendix B, we re-enforce some of our concepts mentioned above, by treating the digit recognition problem as a defect state identification problem.

2.3 Preparation of the training and testing data

In total, $N = 784$ rod-like molecules of length L are confined to a square box of dimensions $a \times a$. Mutual crossing of rods and crossing of box boundaries by rods are forbidden to simulate the excluded-volume interactions.

One can show that the parameter that drives the phase transition is the reduced density,

$$\rho \equiv \frac{NL^2}{a^2}. \tag{2.4}$$

Above the critical value ρ^* the system is in a nematic state with directional ordering and below the critical value the system is in an isotropic state with random orientations (except

those near the walls). For system sizes of $N = 20^2$, $N = 24^2$, and $N = 28^2$, our FNN study (Appendix A) suggests $\rho^* = 6.71 \pm 0.25$, 6.95 ± 0.25, and 7.08 ± 0.25, respectively. Such values are comparable to those from previous numerical and experimental studies, which suggest values of ρ^* of ≈ 7.0 [30], 6.5 [23], and $6 - 9$ [32]; the mean-field theory calculation of $\rho^* = 3\pi/2L^2 \approx 4.71$ based on the Onsager theory is a known under-estimate [16, 47].

Returning to defect topologies, within the nematic state the system can display a stable D-defect pattern, or can be trapped in the free energy minima corresponding to one of the X-, T-, or U-defect patterns, due to the finite confinement effects. The nature of the metastability has been recently addressed extensively in References [13, 22, 31, 34, 57, 61, 86, 91]. In order to explore the best machine learning techniques in identifying the defect states, we established a database from the MC simulations at a fixed $\rho = 19.63$ (produced by setting $a = 6.32L$). The relatively high ρ enables the trapping of the metastable defect patterns during simulation runs. In total, 4400 independent configurational snapshots were collected for each of the defect states (see Section 5.1). Of these, 400 snapshots were reserved for the testing dataset.

For each defect type in Figs. 2.2(e)-(h), 4000 snapshots were used for training. A typical defect pattern can be rotated by a $\pi/2$-, π-, or $3\pi/2$-angle and maintains its topological structure, because of the square boundary geometry. Since these 4000 snapshots were taken from MC simulations, they already contained all different orientations of the defect pattern naturally. To ensure that all four orientations of the square boundary are indeed equally treated, we further rotated snapshots by these angles. The raw dataset of each snapshot contained coordinate data ordered by the label of molecules, l, and followed by $[x_l, y_l, \theta_l]$ (see Figure 2.2(d)), where $l = 1, 2, ..., N$. The treatment differs from the pixel approach of a digital image, for which a pixel grid system would be established.

2.4 Recurrent Neural Networks

A typical structure of the recurrent neutral network is represented in Figure 2.1(b). The crux of this RNN is the long short-term memory (LSTM) module [39]. An RNN can be made with different types of modules, but the LSTM is a popular choice and is well-sufficient for this work. Except for the first block, an LSTM cell has two input channels: an LSTM-external connection to new raw data from the system to be studied, and an LSTM-LSTM connection, taking its own output and internal weights from the previous step as input, hence the recurrent aspect. Schematically, it helps to represent this as a series of LSTM cells for each raw data input. To complete the RNN, a single layer of perceptrons

is appended to act as a final interpretive layer of the LSTM output, compressing the large LSTM output to the smaller prediction output $(\nu_1, \nu_2, ..., \nu_n)$.

An LSTM takes raw data at each iteration of input, and considers it together with its own previous state simultaneously. That is, an LSTM establishes data correlations with those fed into earlier cell states through LSTM-LSTM connections. In its composition, like an FNN, an RNN (including the LSTM) is still merely an ensemble of floating-point network parameters. The logic of supervised training is the same here as with an FNN: determination of the network parameters through optimization of a cost function. The RNN can be coded in terms of Tensorflow [1] libraries efficiently.

LSTM units have several working parts, schematically displayed in Figure 2.3. The memory ability is thanks to the LSTM's "cell state" $\mathbf{C_t}$. The cell state is updated at each iteration t to account for new information received $\mathbf{I_t}$. In defect learning, the inputs are the three feature vectors as in Figure 2.1, $\mathbf{I_0} = \mathbf{x}$, $\mathbf{I_1} = \mathbf{y}$, and $\mathbf{I_2} = \theta$.

Inside the cell, $\times$ and $+$ indicate element-wise multiplication and addition. There are also three "gates" made of sigmoid, σ, and tanh operations. The sigmoid function (Figure 2.4) can be used as a gate since it suppresses negative inputs and allows positive inputs, and the tanh function is used to help normalize and regulate values by compressing input to the range $(-1, 1)$. In Figure 2.3, each green bubble marks a set of trainable network weights and biases: $\mathbf{W_1}$, $\mathbf{b_1}$, $\mathbf{W_{2a}}$, $\mathbf{b_{2a}}$, etc.

The first gate, marked by the blue 1, serves as a sort of "forgetting" layer. Based on the input and previous output (concatenated together), some cell state units are turned off or left alone (or somewhere in between) from the sigmoid multiplication. Mathematically,

$$\mathbf{C^{(1)}_{t-1}} = \mathbf{C_{t-1}} * \sigma\left(\mathbf{W_1} \cdot [\mathbf{I_t}, \mathbf{h_{t-1}}] + \mathbf{b_1}\right).$$

It is considered a forgetting layer because it does not add new information to the cell state but drops elements. The next layers, 2a and 2b, handle that.

The sigmoid gate of 2a chooses which information to add to the cell state, with the tanh gate controlling their normalized magnitudes.

$$\mathbf{C^{(2)}_{t-1}} = \sigma\left(\mathbf{W_{2a}} \cdot [\mathbf{I_t}, \mathbf{h_{t-1}}] + \mathbf{b_{2a}}\right) * \tanh\left(\mathbf{W_{2b}} \cdot [\mathbf{I_t}, \mathbf{h_{t-1}}] + \mathbf{b_{2b}}\right).$$

Adding the cell update $\mathbf{C^{(2)}_{t-1}}$ to the forgotten cell state $\mathbf{C^{(1)}_{t-1}}$ provides the new cell state $\mathbf{C_t} = \mathbf{C^{(1)}_{t-1}} + \mathbf{C^{(2)}_{t-1}}$.

Finally, a third sigmoid gate decides what variables to output based on $\mathbf{h_{t-1}}$ and $\mathbf{I_t}$, and multiplies this by the (compressed) cell state:

$$\mathbf{h_t} = \tanh(\mathbf{C_t}) * \sigma\left(\mathbf{W_3} \cdot [\mathbf{I_t}, \mathbf{h_{t-1}}] + \mathbf{b_3}\right).$$

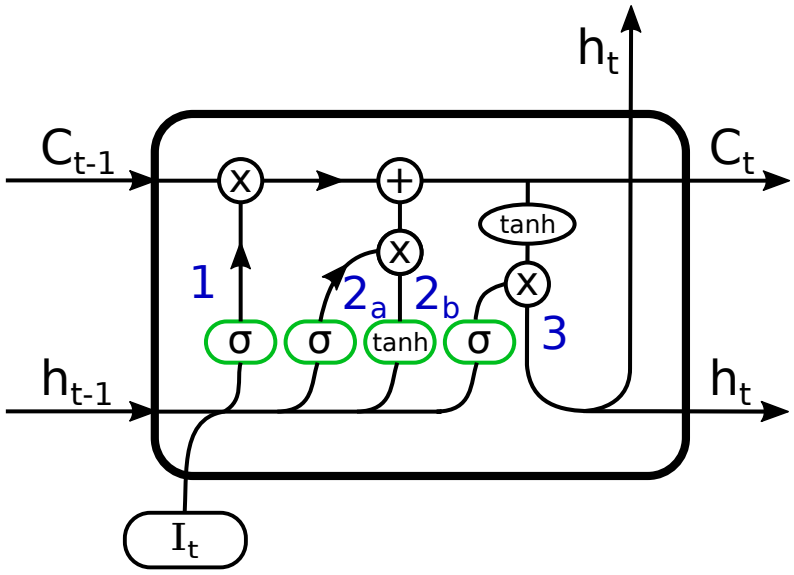


Figure 2.3: LSTM unit schematic. Output $\mathbf{h_t}$ is generated from the integration of its cell state $\mathbf{C_{t-1}}$, previous output $\mathbf{h_{t-1}}$, and input $\mathbf{I_t}$. The details on interior workings are given in text.

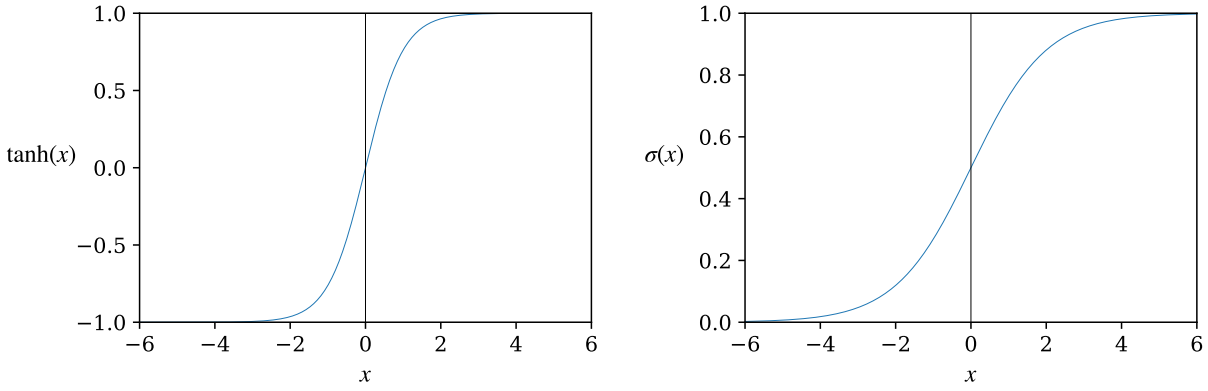


Figure 2.4: $\tanh(x)$ and sigmoid function $\sigma(x) = \frac{e^x}{1+e^x}$.

Chapter 3

Unsupervised Machine Learning of Defects with PCA

3.1 Principal Component Analysis

In this section we explore Principal Component Analysis (PCA) as a technique for detecting defect regions. PCA is a multipurpose statistical method that performs a dimension reduction on a dataset, while simultaneously this transformation results in a form that better distinguishes features in the data. It falls into the category of unsupervised learning since it reveals features in the data without any label-directed goals.

The goal of PCA, in this context, is to find a transformation on our dataset that redistributes it in a way that highlights certain features of the data. This transformed set may then be amenable to classification, either by discrete clustering or sorting along a continuum. To do so, PCA determines the vectors, or "components", which maximize variance in a dataset. Components are ordered based on their associated variance. Once the first component is determined, subsequent components are found by subtracting out all previous components and finding the maximal variance of the result.

One might wonder: Is variance really proportional to the significance of a feature? To this, we must recognize that the significance of a feature is subjective. Really, the question should be reversed: Is what I am placing significance on discoverable by the variance of my dataset? In support of variance equating importance, it helps to note that a dimension with very low variance is very unimportant–scrapping it would minimally effect the information within the data.

To provide a mathematical backing and further explanation, we'll follow the outline offered by Johnathan Shlens [85]. We let an $m \times n$ matrix $\mathbf{X}$ represent our dataset of m observations across n dimensions. The goal of PCA is to find the matrix $\mathbf{W}$ that *best* transforms our dataset to a new set of orthogonal axes, or "principal components":

$$\mathbf{Y} = \mathbf{X}\mathbf{W}.$$

The $m \times n$ matrix $\mathbf{Y}$ then is our dataset projected onto this new set of axes. The *best* axes of course are not arbitrary, but are such that they capture the maximal variance within the original dataset, with the variance monotonically decreasing with increasing axis index. In other words, the first PCA axis captures the largest variance, the second axis captures the next largest variance, and so on. The column vectors $\mathbf{w_i}$ of $\mathbf{W}$ are in fact the unit vectors that describe the orientation of these new axes in the original n-dimensional space, and hence describe how each observation in $\mathbf{X}$ projects onto them.

To begin, we note the covariance matrix of $\mathbf{X}$ (normalized to zero-mean and unit variance)

$$\mathbf{C_X} = \frac{1}{m}\mathbf{X}^T\mathbf{X}$$

Normalizing puts every feature on the same playing field. The diagonal elements of $\mathbf{C_X}$ are the variances of the n variables, and the off-diagonal elements are their covariances. We note that high covariances indicate a redundancy in our dataset, and we wish to minimize these.

We can now define a goal for optimizing $\mathbf{W}$: to diagonalize the covariance matrix $\mathbf{C_Y}$ of our transformed dataset. Such a matrix minimizes redundancy across variables and maximizes variance. There are multiple ways to diagonalize $\mathbf{C_Y}$, but PCA opts for a simple route by having PCA components form an orthonormal basis. To find $\mathbf{W}$ we perform an eigendecomposition on $\mathbf{C_Y}$ and find that the PCA components are in fact the eigenvectors of $\mathbf{C_X}$. We begin by writing $\mathbf{C_Y}$ in terms of $\mathbf{W}$:

$$\begin{aligned}
\mathbf{C_Y} &= \frac{1}{m}\mathbf{Y^T}\mathbf{Y} \\
&= \frac{1}{m}(\mathbf{W}^T\mathbf{X}^T)(\mathbf{X}\mathbf{W}) \\
&= \mathbf{W}^T\left(\frac{1}{m}\mathbf{X}^T\mathbf{X}\right)\mathbf{W} \\
&= \mathbf{W}^T\mathbf{C_X}\mathbf{W}.
\end{aligned} \tag{3.1}$$

We may draw on the property that any symmetric matrix $\mathbf{A}$ is diagonalizable, $\mathbf{A} = \mathbf{SDS}^T$, for diagonal eigenvalue matrix $\mathbf{D}$ and orthonormal eigenvector matrix $\mathbf{S}$. If we now define $\mathbf{W}$ as this matrix of eigenvectors for $\mathbf{C_X}$ (since $\mathbf{C_X}$ is symmetric), we have

$$\begin{aligned}\mathbf{C_Y} &= \mathbf{W}^T(\mathbf{WDW}^T)\mathbf{W} \\ &= (\mathbf{W}^T\mathbf{W})\mathbf{D}(\mathbf{W}^T\mathbf{W}) \\ &= \mathbf{D},\end{aligned}$$

where we also use the orthonormality of $\mathbf{W}$. The eigenvalues, λ, are also called the "explained variance" in the context of PCA. Their normalized values, $\lambda_l^* = \lambda_l / \sum \lambda_k$, or explained variance ratios (EVRs), are used as a metric as to the importance of each axis and are equivalent to the fraction of variance in the dataset captured by component l. Values close to 1 indicate more significance and those closer to 0 being less significant. Generally, one expects only a small number of dimensions to be significant and will see a noticeable drop in λ^* indicating where component axes may be ignored. We can verify that our conditions are met:

- By Eq. 3.1, the ith diagonalized variance of $\mathbf{C_Y}$ is the variance of $\mathbf{X}$ along the ith principal component.
- $\mathbf{W}$ indeed diagonalizes $\mathbf{C_Y}$.

Going forward, we will index PCA components starting from 1. As usual, vectors will be bold lower-case, and matrices will be bold upper-case. So the first, most significant component would be $\mathbf{w_1}$.

3.2 PCA learning of order parameters

Recently, PCA has seen some fascinating use on condensed matter systems. In 2016, Wang [97], with a dataset of simulated Ising spin systems (each observation was an instance of a system and each spin location was a feature dimension of $\mathbf{X}$), PCA could locate phase transitions in both the standard Ising model, but also the conserved order parameter (COP) Ising model. In the standard Ising model, reduction showed that only the first PCA component was significant for capturing the distinguishing information of the system. Indeed, the corresponding eigenvector $\mathbf{w_1}$ was simply a constant function, meaning that it was in fact the magnetization order parameter, and cluster analysis of the data along this component (although it was also visually evident) showed two clusters corresponding

to the high and low temperature states. The author took things a step further with the COP model because each system had a net-zero magnetization, so the magnetization order parameter would not suffice and PCA would need to find other indicators of the phase transition. Overcoming this restriction, PCA found that four components were all that was necessary and together could form the structure factor indicative of the phase transition.

In 2018, Jadrich, Lindquist, and Truskett [45] applied PCA to detect phase transitions of multiple off-lattice systems, including the fluid-nematic transition of a liquid crystal system much like the one of focus here (although with ellipsoid molecules as opposed to rod-like). Their methods were similar to those just described, but instead of each observation being a whole system, they were smaller local samples centered on random probe molecules (e.g. the orientations of 20 nearest neighbours). Additionally, the off-lattice nature carried with it some ambiguity as to the structuring/ordering of the nearest neighbours that formed the $\mathbf{X}$ dimensions. Using an intuitive distance-to-probe ordering, they found again only the first PCA component was needed to learn the transition, and that it modeled the classical order parameter used to locate the transition.

What is crucial to note here, and what is so exceptional about this method, is that, apart from the raw particle information (location, spin, orientation, etc.), it requires no additional information about each system such as Hamiltonians or interaction potentials. On top of this, PCA does not cluster observations in some black-box style, but discovers the precise order parameters that are classically used to indicate the phase change.

3.3 PCA learning of disclinations

We now put forth the questions: Can PCA use nearest neighbour information of our LC system to identify defects within nematic samples? Can it go beyond that and learn which *type* of defect? Figure 3.1 shows the first four winding disclinations that are simple and complete in the sense that a $\pm k\pi/2$ rotation about the rod axis occurs after a full procession. The following discusses these questions, with results on the endeavor presented in Section 6.

The results of Jadrich et al. [45] are certainly encouraging, though their objective was somewhat different. PCA showed that a long-range angular correlation appeared signaling the nematic phase. For this they used every 10^{th} nearest neighbour. Here we are concerned with detailed local features. In theory, the nematic vs. non-nematic states should be easily differentiable by the bulk angular order. For a given probe i (either a rod or a point

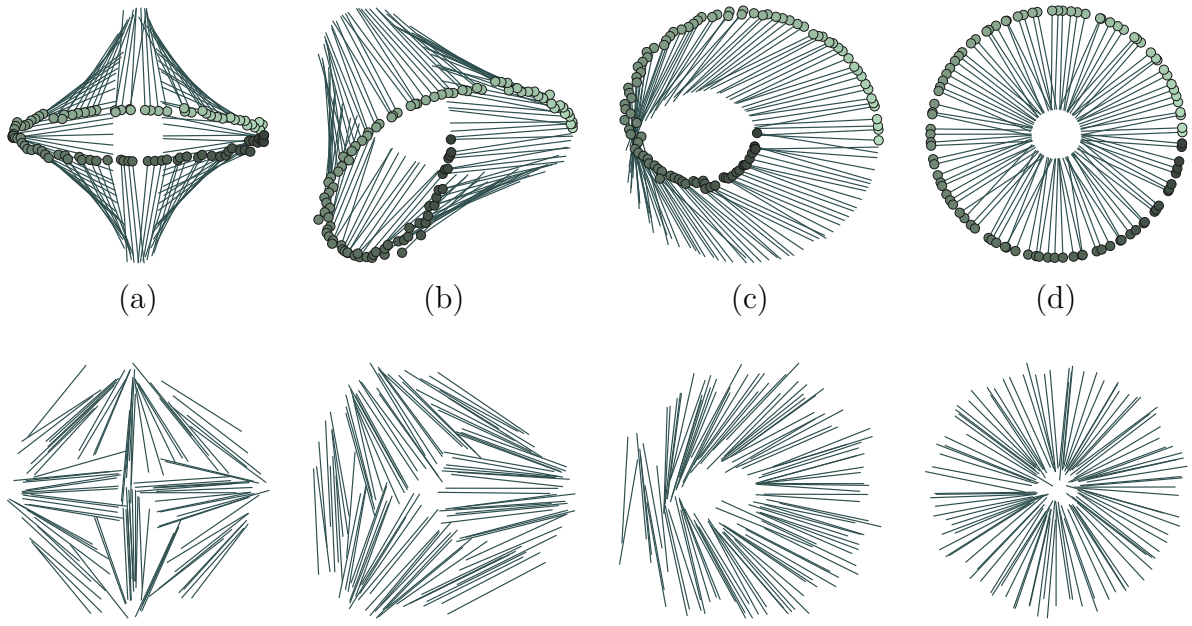


Figure 3.1: (a)-(d) Examples of initialized disclinations with winding numbers -1, $-1/2$, $+1/2$, and $+1$ respectively. In proceeding counter-clockwise around a defect, winding polarity indicates the direction of rotation about the rod axis, and magnitude indicates how many rotations about this axis. Coloured dots are included as a visual aid. (e)-(h) Below, uncrossed samples of the above disclinations. Such samples would be included in the PCA dataset.

in space), we may construct a feature vector $\mathbf{x}^{(i)} = [x_1^{(i)}, x_2^{(i)}, \ldots, x_{nn}^{(i)}]$ (the i^{th} row of $\mathbf{X}$) of some physical quantity (or quantities) x. Rod orientation relative to the probe rod, θ_{ij}, or, to account for the rod symmetry, the transformed $\cos(2\theta_{ij})$ may be sufficient, but the probe rod orientation is highly variable within a characteristic defect orientation, so defects may appear significantly different in their feature vectors. A counter to this is to take neighbouring rod orientations with respect to a local nematic director α instead

A random ordering of neighbours may work in detecting nematic vs. defect states, but should not be expected to work for the more detailed defect type detection. In a perfectly shaped defect, a maximal $\theta_{\alpha j}$ or frequency of certain angles may signal a defect type, but the Monte Carlo defect samples would have too much overlap/similarity for a randomly ordered feature vector. Thus, in the same way that humans visually identify winding numbers, a feature vector that is ordered by a counter-clockwise procession around the defect center should be a more effective approach. Starting the procession from the α director is also important for a consistent feature vector representation.

Using rods as probes introduces a spatial bias to the probing. For instance, an ideal defect center contains no rods, so using rod probes biases the probe centers to be on the edges of defects. We thus use an evenly spaced lattice probe array of 20×20 probes, and assemble nearest neighbours based on these probe centers. After collecting neighbours around a given probe, the center of mass is used as the point of rotation for ordering neighbours in non-random feature vectors.

Three different feature construction methods were tested: Random, Winding Alignment (WA), and Winding Difference (WD). Their results provide an interesting insight into the problem at hand.

Random: In the Random method, $\mathbf{x}^{(i)}$ is constructed by a random ordering of neighbours, and $x_j^{(i)} = \cos(2\theta_{j\alpha})$. This method captures any degree of alignment among neighbours if present, but loses positional ordering information.

Winding Difference: In the WD method, index j is ordered by occurrence in a counterclockwise procession about the defect center of mass and starting from the direction α. The feature value of each neighbour is an adjusted relative orientation, $x_j^{(i)} = \widetilde{\theta}_{j\alpha}$. The adjusted value is a correction of $k\pi, k \in \mathbb{Z}$, due to the rod symmetry. This rod adjustment can actually be problematic. "False rotations" may occur in which a few rods may increase/decrease the accumulated winding difference by a large amount and signal a rotation, when really those rods are outliers from the sample pattern. This problem is looked at in more depth in Section 6.2. However, in well-behaved samples, this method gives all the necessary information for characterizing defects: rod orientation as a function of the procession around the defect.

Winding Alignment: This method bears the same ordering of index j around the defect as the WD method, but uses a different feature value: $x_j^{(i)} = \cos(2\theta_{j\alpha})$. Unlike the WD method, which gauges the difference in rod orientation, the WA method avoids the symmetry issue by measuring neighbour j's alignment with $\alpha^{(i)}$. This method sacrifices winding polarity for overcoming the value adjusting problem in the WD method.

When calculating $\alpha^{(i)}$, we use a discretized version of the method outlined in Section 4.3 that produces the Q-matrix in equation 4.12:

$$Q_{nn} = \frac{1}{2}\begin{pmatrix} S_{nn}(x,y) & T_{nn}(x,y) \\ T_{nn}(x,y) & -S_{nn}(x,y). \end{pmatrix}$$

where S_{nn} and T_{nn} are averaged over N_{nn} neighbours,

$$S_{nn} = \frac{1}{N_{nn}} \sum_j \cos(2\theta_j)$$
$$T_{nn} = \frac{1}{N_{nn}} \sum_j \sin(2\theta_j),$$

with θ_j simply being rod orientation about the x-axis. By diagonalizing Q_{nn}, the leading eigenvalue is equivalent to the order parameter derived in Eq. 4.15, and its corresponding eigenvector is the nematic director $\alpha^{(i)}$ [4, 14].

Chapter 4

Computational Methods and Order in Liquid Crystals

Two main approaches for simulating liquid crystals are molecular dynamic (MD) and continuum phenomenological models. MD simulations are generally more computationally intensive, but can capture finer details at the microscopic level. Importantly, they also simulate the dynamic evolution of the system, unlike static continuum models. Both approaches are used to search for stable solutions of confined LCs. In the following, we will first detail the Monte Carlo MD approach used here for generating liquid crystal states. Then we will go over the Onsager continuum model used by Yao et al. [103] to produce these DTUX topological solutions. Lastly, we will discuss the concept of order in liquid crystals as it is an important concept central to topological states and defects.

4.1 Monte Carlo molecular dynamics

The Monte Carlo technique has been a widespread method for simulating liquid crystals and molecular dynamics at large. With credit generally awarded to John von Neumann, Stanislaw Ulam, and Nicholas Metropolis, the method was devised in the late 1940s as part of the Second World War effort by studying the diffusion of neutrons in fissionable material ([2] pg. 147). The principle is to treat a determinate mathematical problem with a probabilistic analogue and solve it by stochastic sampling. More specifically, we use the Metropolis Monte Carlo method published in 1953 by Metropolis et al. [64].

The method aims to explore the state space, while also tending towards states of higher statistical probability. We let t count the number of "Monte Carlo Sweeps" (MCSs), where

one MCS attempts N random movements of molecules, either by iterating over the molecule list or randomly selecting N molecules, though Hastings [38] found iterating over the list equally valid with the benefit of being that bit simpler. We also let $\mathbf{\Gamma}(t)$ represent the system state at time step t, with $\mathbf{\Gamma}(0)$ being the initial state. In the case of our liquid crystal system of hard rods, a random rod movement would mean

$$\begin{aligned} \mathbf{r}_i(t+1) &= \mathbf{r}_i(t) + \delta\mathbf{r} \\ \theta_i(t+1) &= \theta_i(t) + \delta\theta \end{aligned} \tag{4.1}$$

for rod label i, and small spatial and angular displacements $\delta\mathbf{r}$ and $\delta\theta$. In the Metropolis algorithm, we must create a rule for the acceptance or rejection of a proposed move. We start by considering abstractly the state energy at time t, $E(t)$. The first condition in our rule is that if $E(t+1) < E(t)$, then this is energetically favourable and we accept the move (postponing the calculation of $E(t)$ for now). In the case of $E(t+1) > E(t)$ we do not immediately reject this. Instead we consider the Boltzmann factor, that is the probability of the system being in state $\mathbf{\Gamma}(t+1)$:

$$p_\Gamma(t+1) = \frac{1}{Z} e^{-E(t+1)/k_B T} \tag{4.2}$$

with the canonical partition function Z. Further, we may compare the probability of this state with $\mathbf{\Gamma}(t)$

$$\begin{aligned} \frac{p_\Gamma(t+1)}{p_\Gamma(t)} &= \frac{e^{-E(t+1)/k_B T}}{e^{-E(t)/k_B T}} \\ &= e^{-(E(t+1)-E(t))/k_B T} = e^{-\Delta E/k_B T}. \end{aligned} \tag{4.3}$$

We then base the decision of acceptance/rejection off this relative probability. To do so, a random number $\varepsilon \in [0,1]$ is generated, and compared with the probability ratio $p_\Gamma(t+1)/p_\Gamma(t)$. The rule is to accept the proposed move if $\varepsilon < p_\Gamma(t+1)/p_\Gamma(t)$.

Figure 4.1 shows the chances of acceptance/rejection of a move for a given ΔE. Moves that are energetically costly always have a finite chance of being accepted, though it is exponentially unlikely as ΔE increases. This feature is essential for the simulation to explore the phase space properly. Notice also that the acceptance approaches unity as $\Delta E \to 0$ as we expect.

In calculating $\Delta E/k_B T$, we needn't calculate the total system energies $E(t)$ and $E(t+1)$, rather since we only need their difference we simply calculate the energy change associated

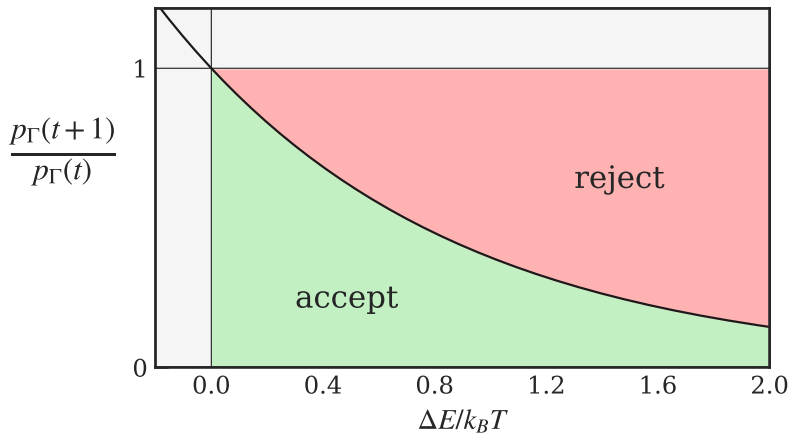


Figure 4.1: Probability ratio of Eq. 4.3 showing chances of acceptance or rejection upon generating ε for a given ΔE.

with the proposed move. As described in Allen & Tildesly [2] (pg. 156), the change in potential energy is calculated by summing over the interaction energies, ϵ_{ij}, between our selected molecule i and the other relevant molecules j,

$$\Delta E = \sum_j \epsilon_{ij}(t+1) - \sum_j \epsilon_{ij}(t). \tag{4.4}$$

This also has the advantage that interactions beyond a certain cutoff distance for both molecular positions, such as a mean field, cancel each other out.

Returning to our liquid crystal model of steric rods, the situation gains a simplification. Recall that the interaction potential of neighbouring molecules is approximated as a delta function: it is zero everywhere except for if there is an overlap at which point it is infinite. Thus, when a proposed move causes an overlap this results in $\Delta E = +\infty$, and conversely if the move grants no overlap, the system perceives no change in potential energy and $\Delta E = 0$. As these are the only two scenarios, we may circumnavigate the entire ε generation and comparison phase, with an overlap having no probability of success and a non-overlap always being accepted.

It is up to the simulator to determine which magnitudes of $\delta\mathbf{r}$ and $\delta\theta$ from Eq. 4.1 work best for their system. There is no theoretic solution for an optimal choice ([2] pg. 159). Generally, the smaller a degree of movement, the more likely a move is to be accepted. However, there is an inverse consequence in that the system generally explores the phase space more slowly. As briefly discussed in Allen & Tildesly ([2] pg. 159), often simulators

will choose a failure rate of around 0.5, but a study by Wood & Jacobson [100] found 0.9 maximized the root-mean-square displacement of atoms as a function of computer time. Allen & Tildesly soundly point out though, this study was carried out on a first generation computer (1959), with 32 hard spheres at a particular packing fraction, and there is no reason to believe that their optimum is universal, or even that root-mean-square displacement is always the desired metric for phase space exploration.

Our Monte Carlo simulated system contained N rigid rods, each having length L, confined to a 2D square area of side-length a. In 2D, a single rod can be represented by a straight line, described by the center-of-mass coordinates, $[x, y]$, and the direction that the rod makes with respect to the x-axis, θ. No rod thickness is considered here, as in a 2D space, two infinitesimally thin rods already interact with each other by an excluded "volume", namely, an excluded area. A successfully generated configuration contains the $[l, x_l, y_l, \theta_l]$ coordinates of all N rod-like molecules, line by line for $l = 1, 2, 3...N$. The wall-confinement effect is enforced by disallowing the intersection of a rod-like molecule with the wall boundaries. Although the macrostate of a system is specified by three parameters N, L, and a, only two are relevant, N, and the ratio L/a. The magnitudes of $\delta\mathbf{r}$ and $\delta\theta$ from Eq. 4.1 were determined by trial and error to maintain acceptance rates close to 50% to allow sufficient system evolution. To speed up the simulation, the cell-index technique [2] (pg. 195) was incorporated in the algorithm.

4.1.1 At the computational crossroads

Confined LC topologies have been well documented and observed both experimentally [22, 31, 57, 86, 91] and theoretically through various models [13, 34, 57, 61, 77, 91, 103]. On the simulation front, unifying the results of off-lattice MD models and continuum mathematical models is a sound goal. As the recent work of Robinson et al. [77] shows, there is still progress to be made there. In their work, they provide a close look at how MD and continuum methods compare on a square-confined LC. They looked to replicate experimental states found by Tsakonas et al. [91] (which match the D and U states discussed here) using Hard Gaussian Overlap (HGO) [3] and Gay-Berne (GB) [35] off-lattice, molecular level models, and a continuum Landau-de Gennes model [24]. For example, the authors found that it can be tricky to produce higher energy, metastable states with MD methods, especially states in extreme or narrow parameter fields. In their study, both off-lattice methods failed to produce the metastable U state.

4.2 Onsager continuum model

In studying LCs there are three main continuum theories which use different order parameters to capture the anisotropic behaviour: the Oseen-Frank model, the Onsager model, and the Landau-de Gennes model. Going through each continuum model would be exhausting and divergent from the goal of this work. However, the Onsager model, being the most appropriate for hard rod molecules, was the choice of Yao, Zhang, and Chen [103] in the preliminary work which produced the DTUX solutions as energy minima. It is fitting to outline its framework and see how the DTUX states are arrived at.

Put forth by Lars Onsager in 1949 [69], the model has been a popular choice in describing the interaction of anisotropic colloidal particles. Here we focus our attention on the model as it applies to confined rod molecules under no external potential. Via this model we can gain insight into system properties, in particular the I-N transition and solving for stable configurations.

To begin, we define a density distribution $\rho(\mathbf{r}, \mathbf{u})$ where $\mathbf{r}$ represents the location of a given rod's center, and $\mathbf{u}$ its orientation. This density distribution is normalized such that $\int \rho(\mathbf{r}, \mathbf{u}) d\mathbf{r} d\mathbf{u} = N$. The Onsager model then prescribes a free energy functional

$$\beta F = \int \rho(\mathbf{r}, \mathbf{u}) \ln \left[L^2 \rho(\mathbf{r}, \mathbf{u}) \right] d\mathbf{r} d\mathbf{u} + \frac{1}{2} \int \rho(\mathbf{r}, \mathbf{u}) w(\mathbf{r}, \mathbf{u}; \mathbf{r}', \mathbf{u}') \rho(\mathbf{r}', \mathbf{u}') d\mathbf{r} d\mathbf{u} d\mathbf{r}' d\mathbf{u}', \quad (4.5)$$

where $\beta = 1/k_B T$ for Boltzmann constant k_B and temperature T. The first term relates to the positional and orientational entropies. Spatial entropy prefers a uniform distribution of particles, and orientational entropy prefers a uniform distribution of angles. The second term emerges from the second-virial approximation and contains the interaction effects between molecules, represented by the kernel function $w(\mathbf{r}, \mathbf{u}; \mathbf{r}'\mathbf{u}')$. Via a Mayer cluster expansion,

$$w(\mathbf{r}, \mathbf{u}; \mathbf{r}'\mathbf{u}') = 1 - \exp\{-\beta \nu(\mathbf{r}, \mathbf{u}; \mathbf{r}', \mathbf{u}')\}, \quad (4.6)$$

where $\nu(\mathbf{r}, \mathbf{u}; \mathbf{r}'\mathbf{u}')$ is our non-overlapping interaction potential

$$\nu(\mathbf{r}, \mathbf{u}; \mathbf{r}', \mathbf{u}') = \begin{cases} +\infty & \text{if rods overlap} \\ 0 & \text{otherwise} \end{cases} \quad (4.7)$$

This means $w(\mathbf{r}, \mathbf{u}; \mathbf{r}', \mathbf{u}') = 1$ if rods overlap, increasing F, and 0 if not. The interaction integral in 4.5 can be simplified for a spatially homogeneous system (with thin rods of high aspect ratio) to

$$\int w(\mathbf{r}, \mathbf{u}; \mathbf{r}', \mathbf{u}') dr' = L^2 |\mathbf{u} \times \mathbf{u}'|, \quad (4.8)$$

(in two dimensions). It is clear that parallel rods optimize this orientation term. However, at the same time the entropy term is pushing for a uniform spread of angles and positions. From the competition of these terms then emerges the isotropic-nematic transition.

As discussed in Chen (2016) [14] (pg. 25), truncating 4.5 at the second virial term is valid in three dimensions—this was already resolved in Onsager's original work where he argued there was no need for higher-order terms for small thin-rod systems—but is less appropriate in two dimensions where the higher-order terms are closer to the second order magnitude. This does render the equation for the free energy *quantitatively* less accurate, but importantly the principal physics are still captured in this form. Indeed, this underestimates the critical density, ρ_c, of the I-N transition at $NL^2/A = 3\pi/2 \approx 4.71$. Values from previous numerical studies suggest $\rho_c \approx 7.0$ [30], 6.5 [23]; or in a recent macro-scale experiment of confined vibrated needles, $6 - 9$ [32].

In their separate works, Chen (2013) [13] and Yao et al. [103] determine numerical solutions from a mathematically equivalent self-consistent field theory model. This model introduces a mean-field acting on rod segments and enacts Euler's forward scheme on a propagator of a modified diffusion equation to carry out the computation and locate energy minima. Additionally, special care was taken in incorporating the hard-wall boundary effects. This derivation is more involved and not essential for our purposes.

The two free phenomenological system parameters are the reduced density $\rho^* = NL^2/a^2$ and L/a [14] (Yao [103] introduce a third quantity, b/a, for rectangular confinements with width b). The ratio L/a is really a finite size scaling parameter, and the reduced density essentially determines the system topology as either isotropic or nematic, and which (meta-) stable topologies are realizable. Phase diagrams as functions of these three parameters are shown in Figure 4.2. We can see that regardless of b/a and L/a, a low enough reduced density will produce an isotropic state, and above which the specific nematic topology depends on all three parameters. The L phase is another topology, but is not considered in this work.

4.3 Orientational order

Naturally, of great interest is a quantitative way to mark whether a system is nematic or isotropic. That is, to mark when the higher symmetry of the isotropic phase is broken in the nematic phase. Qualitatively we describe this less symmetric nematic phase as having more order. The task then is to create a quantitative order parameter that, for example, vanishes to zero in the isotropic phase and approaches unity in the nematic

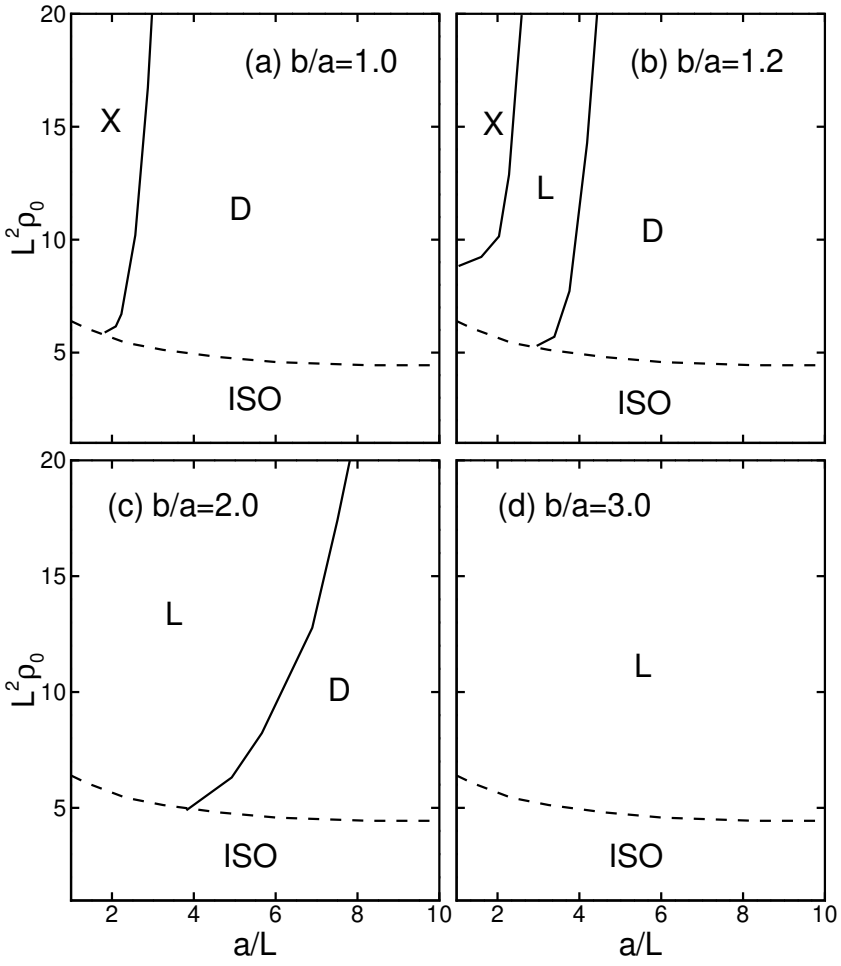


Figure 4.2: Phase diagrams of the rectangular confined rod system, reproduced with permission from Yao et al. [103]. Here, $\rho_0 = N/a^2$ and the dashed line indicates a second-order phase transition.

phase. In a ferromagnetic system, for instance, the magnetization is conveniently readily an appropriate order parameter. However, conjuring one for liquid crystals is more complicated.

We proceed with our derivation in three dimensions, referencing Andrienko's LC summary [4], and then consider the two dimensional case. Naively one may opt for a directional average but since rods have a head-tail symmetry the average of $\mathbf{u}$ will cancel out. The next invariant is then the tensor

$$Q_{\alpha\beta} = \frac{1}{N}\sum_i \left(u_\alpha^{(i)} u_\beta^{(i)} - \frac{1}{d}\delta_{\alpha\beta} \right), \tag{4.9}$$

in d dimensions, where α and β represent the Cartesian variables x, y and z, $\delta_{\alpha\beta}$ is the Kronecker delta, and the summation is a thermal average over a small but macroscopic volume. This Q-tensor has several useful properties:

- It is symmetric since $u_\alpha^{(i)} u_\beta^{(i)} = u_\beta^{(i)} u_\alpha^{(i)}$.

- It has zero trace:

$$\begin{aligned} \mathrm{Tr} Q_{\alpha\beta} &= Q_{xx} + Q_{yy} + Q_{zz} \\ &= \frac{1}{N}\sum_i \left[(u_x^{(i)})^2 + (u_y^{(i)})^2 + (u_z^{(i)})^2 - 1 \right] \\ &= 0 \end{aligned}$$

  since $\mathbf{u}$ is a unit vector.

- As the nematic approaches perfect alignment,

$$Q = \begin{pmatrix} -1/3 & 0 & 0 \\ 0 & -1/3 & 0 \\ 0 & 0 & 2/3 \end{pmatrix},$$

  since $Q_{zz} = u_z u_z - 1/3 = 1 - 1/3 = 2/3$, then with Q being traceless and the symmetry of x and y, $Q_{xx} = Q_{yy} = -1/3$.

A final and important property emerges, that $Q_{\alpha\beta} = 0$ in the isotropic phase. To see this we note in spherical coordinates

$$\begin{aligned} u_x &= \sin\theta\cos\phi, \\ u_y &= \sin\theta\sin\phi, \\ u_z &= \cos\theta. \end{aligned}$$

We also introduce $f(\theta, \phi)$ as the probability of finding a molecule with angles θ and ϕ in the given region. Now, $Q_{\alpha\beta}$ may be equivalently expressed as

$$Q_{\alpha\beta} = \int_0^{2\pi} d\phi \int_0^{\pi} \sin\theta d\theta f(\theta, \phi) \left(u_\alpha u_\beta - \frac{1}{3}\delta_{\alpha\beta} \right).$$

In the isotropic phase $f(\theta, \phi) = 1/4\pi$. Thus, any $Q_{\alpha\beta}$ term of x or y is zeroed because of the periodic integral in ϕ. This only leaves

$$\begin{aligned} Q_{zz} &= \frac{1}{4\pi} \int_0^{2\pi} d\phi \int_0^{\pi} \sin\theta d\theta \left(\cos^2\theta - \frac{1}{3} \right) \\ &= \frac{1}{6} \left(x^3 - x \right) \Big|_{-1}^{1} = 0. \end{aligned}$$

Reducing to two dimensions is fairly straight forward. Our Q tensor becomes

$$Q_{\alpha\beta} = \int_0^{2\pi} f(\theta) \left(u_\alpha u_\beta - \frac{1}{2}\delta_{\alpha\beta} \right) d\theta, \tag{4.10}$$

where we now let θ describe the angle a rod makes with the x-axis. To generalize the expression for a given region centered on (x, y), we simply have

$$Q_{\alpha\beta}(x, y) = \int_0^{2\pi} f(x, y, \theta) \left(u_\alpha u_\beta - \frac{1}{2}\delta_{\alpha\beta} \right) d\theta, \tag{4.11}$$

with $f(x, y, \theta)$ now as the probability of a rod having orientation θ at the location (x, y). We may concisely express Q as the matrix

$$Q = \frac{1}{2} \begin{pmatrix} S(x, y) & T(x, y) \\ T(x, y) & -S(x, y). \end{pmatrix} \tag{4.12}$$

Given $\mathbf{u} = (\cos\theta, \sin\theta)$, we define the elements

$$\begin{aligned} S(x, y) = -2Q_{yy} = 2Q_{xx} &= 2 \int_0^{2\pi} f(x, y, \theta) \left(\cos^2\theta - \frac{1}{2} \right) d\theta \\ &= \int_0^{2\pi} f(x, y, \theta) \cos(2\theta) d\theta \end{aligned} \tag{4.13}$$

and

$$\begin{aligned} T(x, y) = 2Q_{xy} = 2Q_{yx} &= 2 \int_0^{2\pi} f(x, y, \theta) \left(\cos\theta \sin\theta \right) d\theta \\ &= \int_0^{2\pi} f(x, y, \theta) \sin(2\theta) d\theta. \end{aligned} \tag{4.14}$$

Qualitatively, these elements describe the strength of alignment along the x and y axes, and their diagonals at $\theta = \pi/4$ and $3\pi/4$. $S(x, y)$ responds to alignment with the x and y axes, yielding 1 for x-axis alignment and -1 for y-axis alignment. With $T(x, y)$, the response is 1 for $\pi/4$-axis alignment and -1 for $3\pi/4$-axis alignment. We may also observe that both elements go to zero in isotropic phases. Finally, the largest eigenvalue of Q pertains to the main order parameter, and its eigenvector is the nematic director ([2], pg. 93),

$$\Lambda(x, y) \equiv \sqrt{S^2(x, y) + T^2(x, y)}. \tag{4.15}$$

An intensity map of T and S on the stable diagonal D and metastable X solutions in Figure 4.3 show how these parameters can be used to characterize different topologies.

Should we wish to determine a bulk nematic order parameter, we may take the mean values of $S(x, y)$ and $T(x, y)$,

$$\overline{S}(x, y) = \int_{-a/2}^{a/2} \int_{-a/2}^{a/2} \frac{S(x, y)}{a^2} dxdy, \tag{4.16}$$

$$\overline{T}(x, y) = \int_{-a/2}^{a/2} \int_{-a/2}^{a/2} \frac{T(x, y)}{a^2} dxdy, \tag{4.17}$$

for box edge length a. A bulk order parameter

$$\overline{\Lambda} \equiv \sqrt{\overline{S}^2 + \overline{T}^2} \tag{4.18}$$

may then be utilized to gauge if the whole system is in a nematic or isotropic state.

In Figure 4.4, $\overline{T}$ is shown for the D state at various a/L ratios. The critical density shows a sudden onset of order $\overline{T}$. The reduced free energy difference per particle $\beta\Delta F/N$ in (b) is for the D state relative to the X state, indicating the D state to be more energetically favourable. Our Monte Carlo simulations also support this, as all other non-D nematic states eventually melted into the D state.

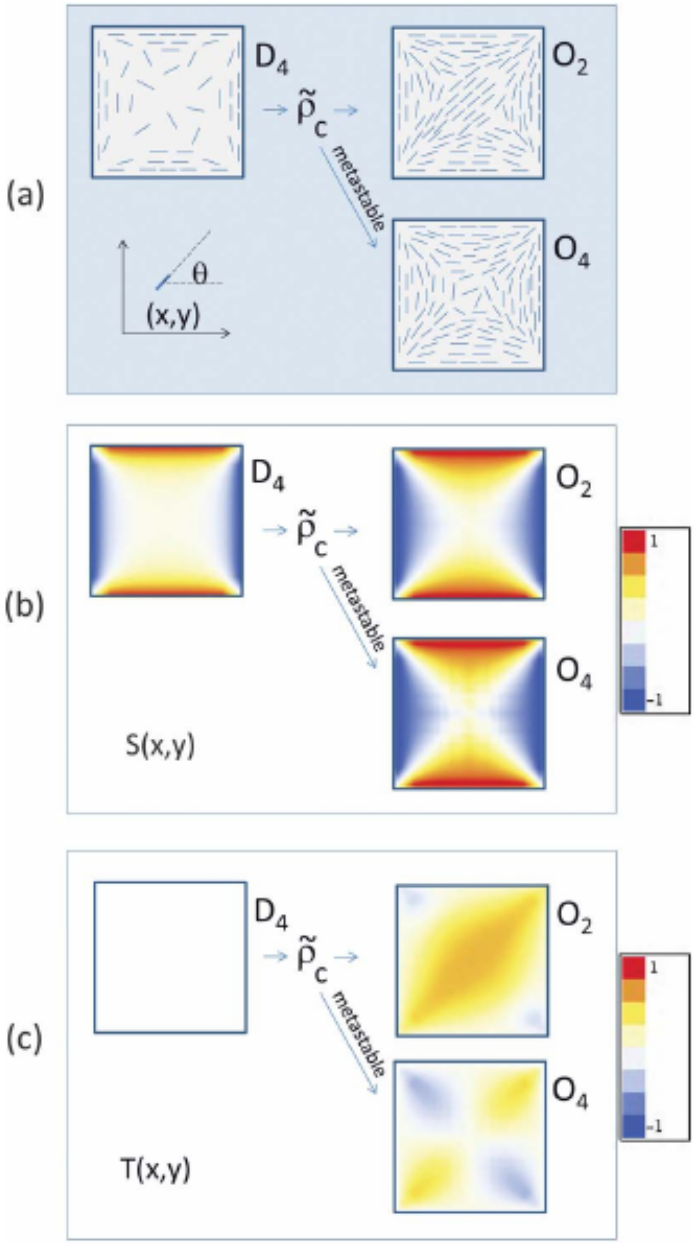


Figure 4.3: Stable D and metastable X solutions from Chen (2013) [13], reproduced with permission. In this work the D state is labeled O_2 and the X state is O_4, with D_4 labeling the isotropic state. $\tilde{\rho}_c$ indicates a transition past the critical transition density. Order parameters S and T are also given. T presents itself as an clear way to distinguish these two topologies.

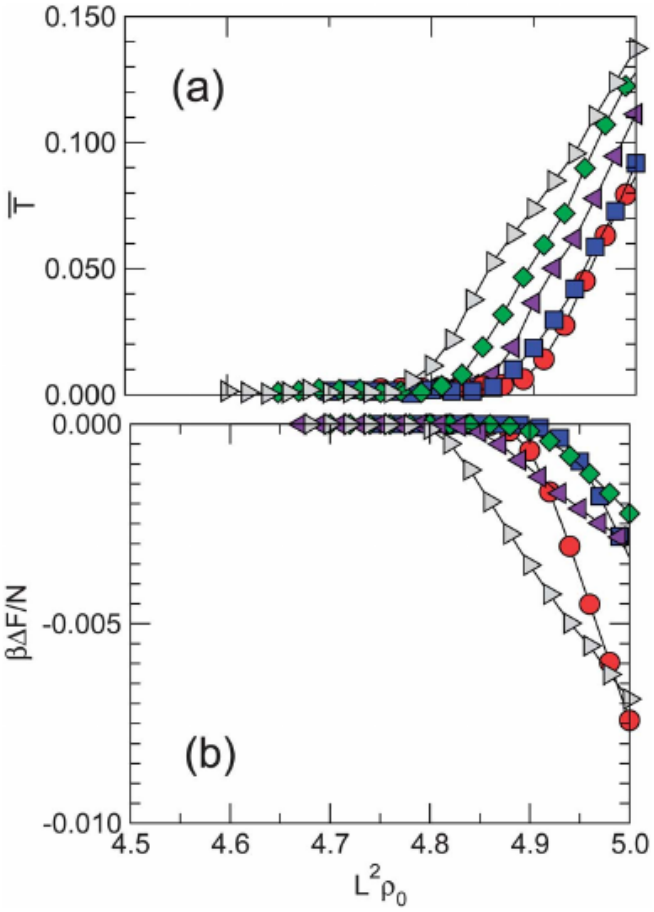


Figure 4.4: (a) Mean order parameter $\overline{T}$ and (b) reduced free energy difference per particle $\beta\Delta F/N$ as functions of reduced density for the D state at $a/L = 3$ (circles), 5 (squares), 7 (diamonds), 9 (left triangles), 11 (right triangles). Energy in (b) is relative to the X state. Results are from an Onsager mean-field model by Chen (2013) [13]. Figure reproduced with permission.

Chapter 5

Results: Supervised Learning of Topological States

5.1 Monte Carlo data generation

All DTUX data sets had $N = 784$ rods and a box-edge to rod length ratio $a/L = 6.32$, amounting to a density $\rho = 19.63$. The relatively high density ensures the lifetime of metastable XTU states. Trapping of a particular type of defect state depends on the initial condition. We take the approach of randomly placing rods in the box to start with and then align the directions of the rods according to a particular defect pattern. After this, each rod was given a small random positional and orientational nudge to add some randomization. Generally, there would still be substantial crossing between the rods and between rods and walls. An uncrossing MC period was then conducted; typically for density $\rho \sim 19$, this period was approximately 5×10^4 MCS. After that, the systems were run for 10^4 MCS to further equilibrate the system. This was followed by taking a "snapshot" (i.e. writing the rod coordinates and angles to a file). For each topology, 4400 snapshots were taken from independent runs, each having a different initial condition from the random nudging. Among these, 4000 snapshots were used for training and 400 set aside for testing. To cover rotational degeneracy of the topologies, an equal number of images were rotated $\pi/2$, π, and $3\pi/2$.

5.2 Application of FNN

A few technical details are provided. We used a multilayer perceptron FNN of modest size, having two hidden layers: the first of size 128, and the second of size 32. With the implementation of Tensorflow, exponential linear units were the chosen neurons for their proven effectiveness and quick learning [21, 1], and an early stopping technique determined sufficient training time [67]. Dropout was used at a 50% drop rate to reduce overfitting while training [87]. The Adam algorithm was used for optimization [49], and Softmax was applied to the output neurons to normalize the output set (see Section 2.1). For evaluating NN performance, a separate test dataset of images not seen during training is needed. A useful NN model needs to be able to correctly classify images it has not been trained on, otherwise it may merely have found a set of parameters that only work for the training set.

We used cross entropy S as the cost function to measure the training quality on the training data set; plotted as a function of epoch, we can see how the model learns over time and estimate its learning trajectory. When the network is adequately trained, S approaches 0. Another unique insight into the network's performance is the accuracy A, defined to measure the network performance on the unseen test set. An accuracy of 1 is scored for correct classification on the entire test set. Both S and A are quantitatively defined in Section 2.1.

A naive approach would be directly taking an FNN for identifying these defect states, sequentially feeding the raw $[x_l, y_l, \theta_l]$ data into the $3N$ input nodes, and training the network to recognize the four different topologies by supervised learning. This approach has seen success in learning phase transitions of Ising systems [9], polymer systems [98], and the XY model to a degree [6]. However, this method showed a pronounced failure in the current application. As we show in Figure 5.1(a), indicated as $m = 1$, this approach does not come close to an acceptable performance, producing a plateau in an undiminishing cost. In addition, A reaches a plateau at an unsatisfactory level of approximately 60%.

Why is this so? One of the essential features the network needs to learn for these defect configurations is the correlation between the position of a topological defect and the molecular orientation in the vicinity of the defect. A typical raw snapshot datafile records the $[x, y, \theta]$ data sequentially according to the order of the label of the rod-like molecules l. Because there is no a priori knowledge of which molecules show up in the defect regions, the labels of the defect-region molecules differ from file to file. Indeed, in a statistically independent set of files, such as the ones produced here from different initial conditions, there are no label-position correlations of the defect-region molecules among the learning data files. This all addresses a crucial, but often unappreciated, aspect of image

classification: by filling input vectors in a positionally sorted fashion (Figure 2.2(a),(b)), positional information, and indeed its correlation to whichever feature is being written to the input vector, is consequently encoded. If this sorting is destroyed, even if we give the positions (such as $[x, y]$) as part of the input data, the position-feature correlation is destroyed. This unseen property, and its essential importance can also be demonstrated via the digit recognition problem in Appendix B.

Hence, the key information is the position sorting in the initial data input, as the FNN relates features with the ordering of the input data. We develop the following coarse-graining procedure to train an FNN in identifying the DTUX defect states.

A schematic in Figure 5.2 shows the coarse graining method used here. The confinement box is divided into $m \times m$ cells, where m is an integer. The cells are labeled $M = 1, 2, ...m \times m$ in a left-right, top-down order. The raw data in every snapshot is consequently presorted according to the center-of-mass coordinates of the molecules, $[x, y]$, so that molecules belonging to the $M = 1$ cell show up first, $M = 2$ cell show up second, etc. Within a cell, the order of data is still random and no further presorting is made. The fixed size of the NN input vector means each rod must only appear once, even if it extends into multiple cells. Using the center of mass of the rod to decide its cell is a natural choice. The $m = 1$ case returns to the raw data format. By the end, the order of appearance of molecular information is no longer according to l, but, according to M for all coarse-graining degree $m \geq 2$. The presorted data is then used in supervised training.

The presorting procedure worked well with an FNN. Figures 5.1(a) and (b) show how the cost function and accuracy quickly approach the ideal value of 0 and 1 respectively, as we presort the data beyond $m = 2$. In the case of $m = 4, 8$, less than 20 epochs (surprisingly short) are needed to adequately train the FNN. This can be attributed to the defect patterns in Fig. 2.2 themselves. By dividing the square box in $m = 4$ cells, for example, one can already distinguish the defect structures, by ignoring fine details inside a given cell. Of course, in general we expect that the degree of coarse-graining, m, needs to increase for a more complicated defect pattern with more defect features.

In summary, to effectively train an FNN to identify features in an image or simulation data file, the ordering of data points (pixels in image, spins in the Ising model, and rod-like molecules in the current study) contains vital information of the data. An off-lattice model usually produces data with a random order and it must be presorted according their approximate $[x, y]$ coordinates, if we are looking for the correlation between coordinates and the physical features.

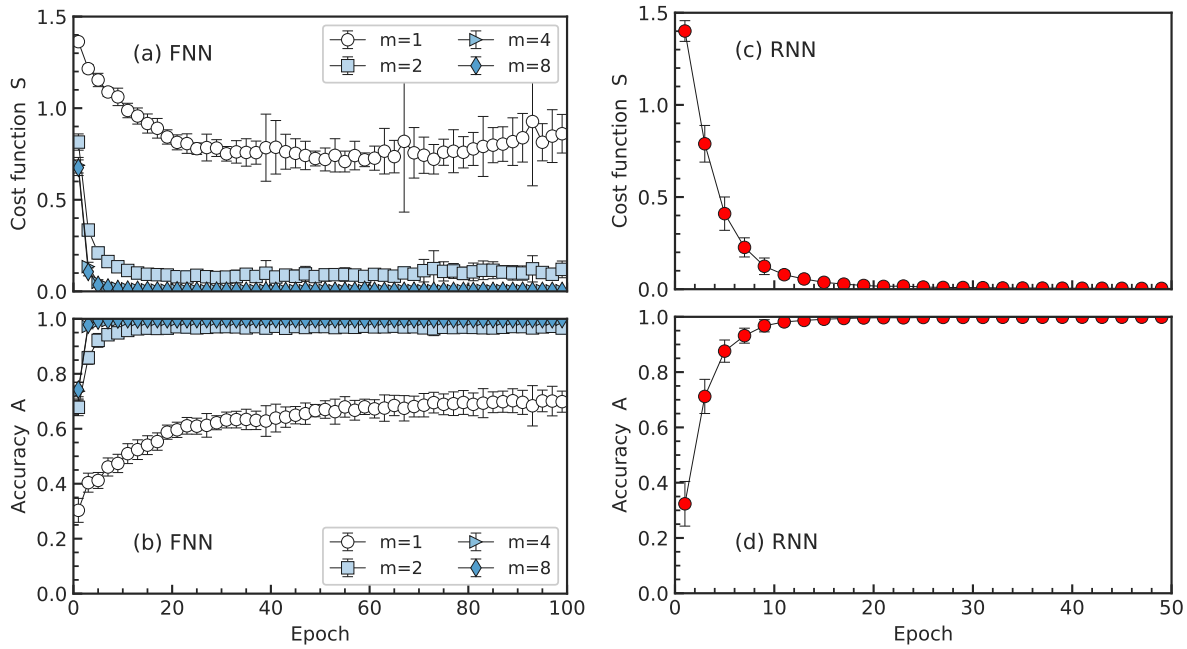


Figure 5.1: Cost function S and accuracy A, defined on the training and test datasets respectively, monitored on an FNN [(a) and (b)] and RNN [(c) and (d)] as functions of epoch step. Symbols in the plots represent the averaged S and A produced from 20 repeated training runs, from which error bars are also estimated. Circles, squares, triangles, and diamonds in (a) and (b) correspond to the degrees of coarse-graining in the presorting procedure used: $m = 1$ (unsorted raw data), 2, 4, and 8, respectively. For coarse-graining, the original simulation box in Fig. 2.2(e)-(h) is divided into $m \times m$ cells [see Section 5.2]. The circles in (c) and (d) also represent the same averaged S and A, where an RNN was used with raw, unsorted data as the input (i.e., $m = 1$).

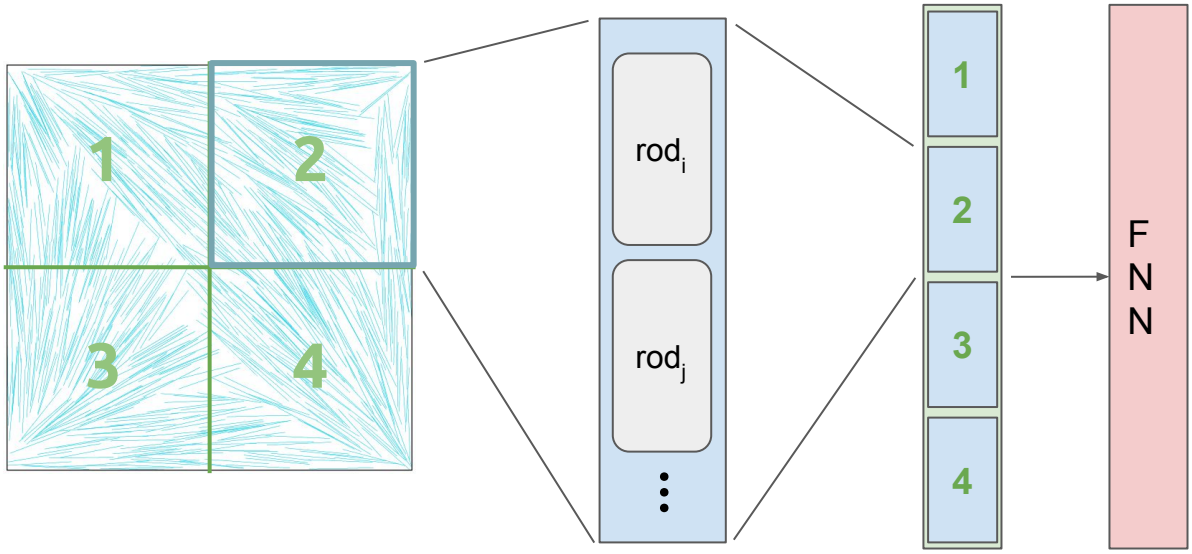


Figure 5.2: Schematic of the coarse graining method. Within a cell, rods are randomly ordered, however cells are positionally ordered for input to the network.

5.2.1 FNN as a universal approximator

There is a matter that cannot be ignored on this subject. The Universal Approximation Theorem for neural networks. Rigorously proved by Hornik, Stinchcombe, and White [40], the theorem's main points are:

- Multilayer FNNs, even those with a single hidden layer, are universal approximators: they can approximate any continuous function to arbitrary accuracy.
- The above result is independent of choice of activation function so long as it is bounded and non-constant.
- Functions with finite support can be approximated exactly with a single hidden layer.
- Lack of success is either due to an insufficient number of hidden neurons *or lack of a deterministic relationship between input and output.*

We emphasize the last point because this is the root cause of the failure of the FNN in learning the DTUX topologies. There is no consistency in what will appear in a given input

node. For example, due to the randomness of the input loading, the same configuration may be loaded completely differently on separate instances.

Each input node to the NN represents a dimension along which the network is trying to recognize patterns. If there is no consistency in the relationship (the ordering) between dimensions then the network does not stand a chance of recognizing patterns. We show this explicitly with our MNIST testing in Appendix B.

5.3 Application of RNN

Here we demonstrate a novel usage of an RNN in condensed matter systems. Exploiting its ability to correlate physical features through LSTM blocks, we adopt a triple iterative structure shown in Fig. 2.1(b) to find the correlation between the spatial coordinates x, y and the orientational coordinate θ. The three main features, x_l, y_l and θ_l are input into the network model iteratively through the LSTM-external layer, one after another. As a technical note, we used a rather small RNN having only a single layer LSTM of 64 hidden neurons, and a single perceptron layer of 128 neurons. Dropout, Adam optimization, and early stopping were again used throughout the supervised training [49, 67, 87].

The performance of this RNN is exceptionally good in identifying the defect states. Figures 5.1(c) and (d) demonstrate that within an initial 20 epochs, our RNN efficiently captures the main features of the four defect states, evaluated on an unseen test dataset. We stress here that unlike the procedure used to produce Figs. 5.1(a) and (b), we used the raw, unsorted data as the input on our RNN experiment.

Without other analysis tools it is difficult to say which *exact* topological features the RNN is learning to distinguish the DTUX image set. One could argue that the location of a defect in a larger system is a point-like object, but the DTUX topologies here, complete in their square confinement, are compositions of bulky nematic patterns interrupted at defect points (Figures refFIG1(e)-(h)). In a finite system such as the one we take here, the exact division between defect points and the overall nematic pattern becomes artificial. We intend to believe that the RNN examines their entire topologies by correlating the spatial-angular information.

Beyond liquid crystals, many other classes of materials contain topological defects that are of practical or fundamental interest; some are detectable by the naked eye and some are subtle to visualize. Especially when a molecular configuration file is given, either produced directly from computer simulations or indirectly reproduced from real experiments, the fluctuating microscopic configurations of all molecules could obscure the existence of a

certain topological feature. Hence it would be desirable to have a computer algorithm to deal with the nontrivial task of identifying these states.

In order to design a classical algorithm to identify the topological states, understanding the main features (spatial dimensionality, line defect versus point defect, definition of winding number, etc.) is required to describe a specific nature of the state. Well-thought mathematical procedures would be needed to capture the correlation (or discorrelation) between defect regions for different states. Classification would then require tools that identify defects, locate them reliably, and finally try to correctly classify this to their human-defined templates. The NN model here presents a universal and simple method that sidesteps this non-trivial process by requiring only the raw data as input. The RNN can masterfully learn these defect topologies by using its ability of making correlations between data features, without asking the question of the specific mathematical and physical properties such as where to look for the defects and what kind of defects a system may contain.

The neural network approach we suggest here is simple, automated, and universal. Additionally, neural nets can provide a near instantaneous computing time for classification, of course minus the time required for training. In our case though, this was agreeable, requiring only tens of minutes.

Chapter 6

Results: Learning and Locating Disclinations by PCA

We show here how PCA can be very effective at distinguishing nematic, isotropic, and defect regions. Three different methods, described in Section 3.3, are showcased here: Random, Winding Difference (WD), and Winding Alignment (WA). The Random method is the simplest, but is limited to only discerning strength of nematic ordering. The WD method takes us to the doorstep of our goal, but may misread edge-cases with false rotations and the like. We find the WA method is ignorant of the rod correcting problem, but sacrifices the ability to learn $\pm$ winding polarity.

In the following, the PCA-reduced datasets are analyzed to glean how, and to what degree, each method is separating the data. By viewing the component eigenvectors and explained variance ratios (EVRs), we also understand what features of the data are used for its separation, and how significant each component is. Finally, we consider the component "responses" (their $\mathbf{y}$ output vector) to several test inputs of isotropic, nematic, or isolated defect samples, and full boxed topologies from Section 5.

6.1 Preparation of data

The dataset was prepared from a combination of images from the isotropic-nematic (I-N) transition study in Appendix A, and images from a new set of isolated defects created for this purpose. Generation of the LC images is done by Monte Carlo, as described in Section 5.1, with parameters particular to the I-N images described in Appendix A. For a diversity

of samples, I-N images at densities $\rho = 16.0, 11.6, 8.7, 7.2, 5.7$, and 4.2 were used. Figure A.1 shows where each density lies along the I-N spectrum.

For the isolated defect portion of the dataset, four different initial conditions were used, one for each defect shown in Figure 3.1, where images (a)-(d) show the rod initialization, and (e)-(h) show the systems after a rod uncrossing phase plus an additional 10 MC sweeps. Only the uncrossed, partially relaxed snapshots were added to the dataset. Equilibration is not the goal here since defects will fall apart, so only a few MC sweeps are performed. The number of rods in the defect files had to equal N_{nn} so that the entire defect is captured in a given probe. Since the MC simulation is initialized in a lattice, we heuristically chose $N_{nn} = 6^2$ for this PCA study. As can be seen in Figure 3.1, this number produces clear defects that are well filled out. Different values of N_{nn} may be worth investigating, however, smaller values would miss important rods that define a defect. In the other direction, too large a number may capture conflicting topologies, for example if the sample were near a boundary wall or corner.

When extracting samples from the I-N images, a 20×20 uniform lattice of probes would grab 400 samples. Doing this for 60 snapshots of 6 different files amassed 144 000 I-N samples. Taking 400 probes for a given isolated defect snapshot would produce redundant images, so only a single sample would be taken for each defect snapshot. During the MC production run, each isolated defect was regenerated from scratch with randomization applied to their positions and orientations so they have no correlation. For each defect type, 24 000 individual samples were collected. In total, the complete dataset included 240 000 samples.

6.2 Results of PCA

In the series of Figures 6.2, 6.5, and 6.11, observations in the PCA reduced space and their EVRs are shown for each method. The Python library `sklearn` was used for the PCA reduction, and the necessary data normalization. In addition to such information, the first few component eigenvectors are plotted in Figure 6.1 to see which aspects of the feature vectors the model is extracting.

Random method

With the Random mode of Figure 6.2, only the first component is significant, as indicated by the EVRs, and the random-noise components for $l > 1$ in Figure 6.1. The constant

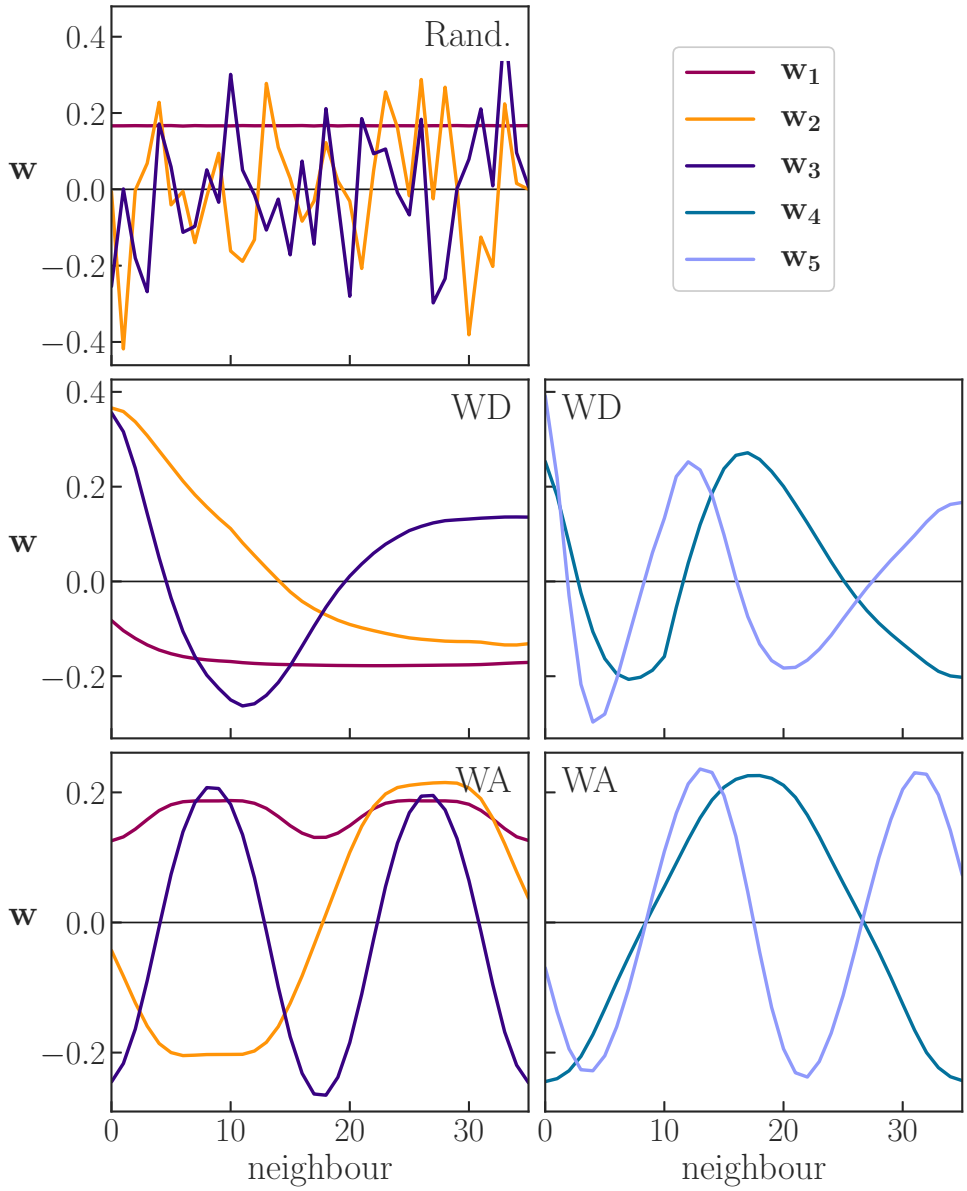


Figure 6.1: The first three (Random) or five (WD/WA) component eigenvectors. $\mathbf{w_1}$ always captures an average angular alignment. The PCA algorithm fails to learn any other features in the Random mode. The text discusses the higher order waveforms in WD & WA.

form of $\mathbf{w_1}$ indicates it performs a summation operation (with some multiplicative scalar) on $\mathbf{x}^{(i)}$. As $x_j^{(i)} = \cos(2\theta_{j\alpha})$, this amounts to an effective alignment along $\alpha^{(i)}$. Although this can detect whether a sample is nematic or not, more is needed to differentiate defects, as will become apparent when tested on isolated defects.

The results of Figures 6.3 & 6.4 confirm the limited utility of the Random mode for detecting nematic samples. Figure 6.3 shows the feature vectors of a nematic and isotropic sample, and their corresponding PCA responses. The nematic sample produces a strong positive $\mathbf{y_1} \approx 5$ response, and reciprocally the isotropic produces a negative $\mathbf{y_1} \approx -2$ response.

In Figure 6.4, we can see response averages across nematic, isotropic, and defect samples. This information reveals that the large mass of samples surrounding $\mathbf{y_1} = -4$ in Figure 6.2 is from the defect dataset. Defects are, in this sense, maximally unaligned because of their uniform angular distribution. The negative value comes from standardizing of the feature vectors. As we can see in Figure 6.3, standardization always reduces feature values, so an isotropic or defect sample will average about a negative center (Figure 6.3(b)). Dotting such a vector with the positive $\mathbf{w_1}$ component of Figure 6.1 produces a negative response.

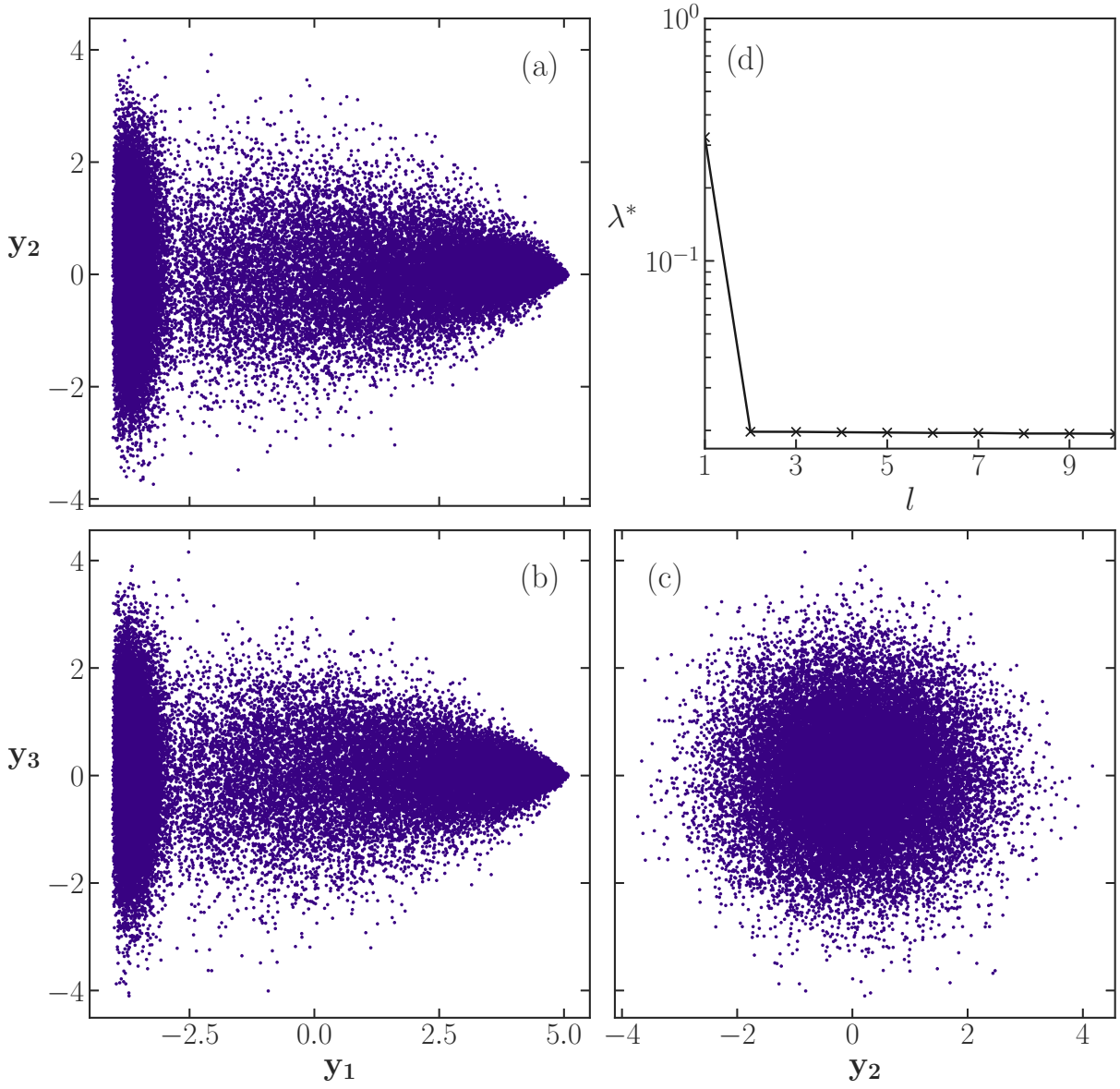


Figure 6.2: Samples in the PCA reduced space for the first three components, and the EVRs for the first 10 components, using the Random method. Every fifth data point is plotted to reduce saturation.

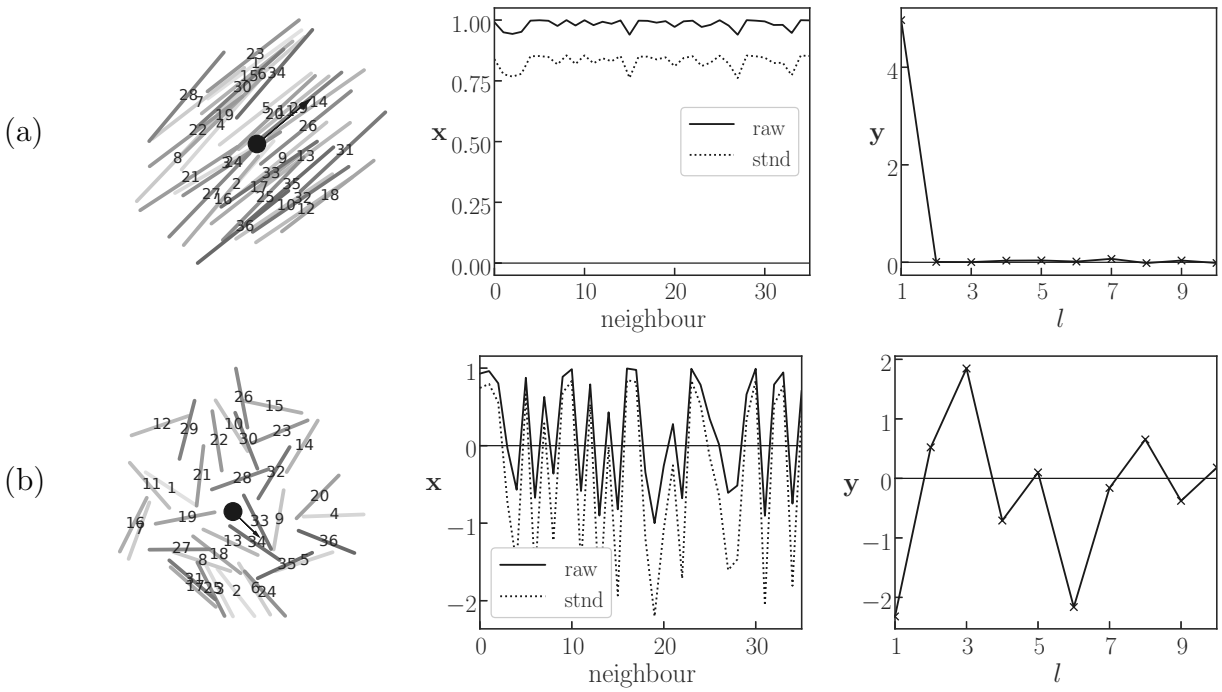


Figure 6.3: Feature vectors (middle column) and their PCA-basis response (right column) for nematic (a) and isotropic (b) samples under the Random model. Neighbour numbers are overlain on top of the rods and the nematic director is indicated with a black arrow. Dotted lines are the feature vector plots are after standardization. We can see that $\mathbf{y_1}$ captures isotropic and nematic information by the strong opposing responses.

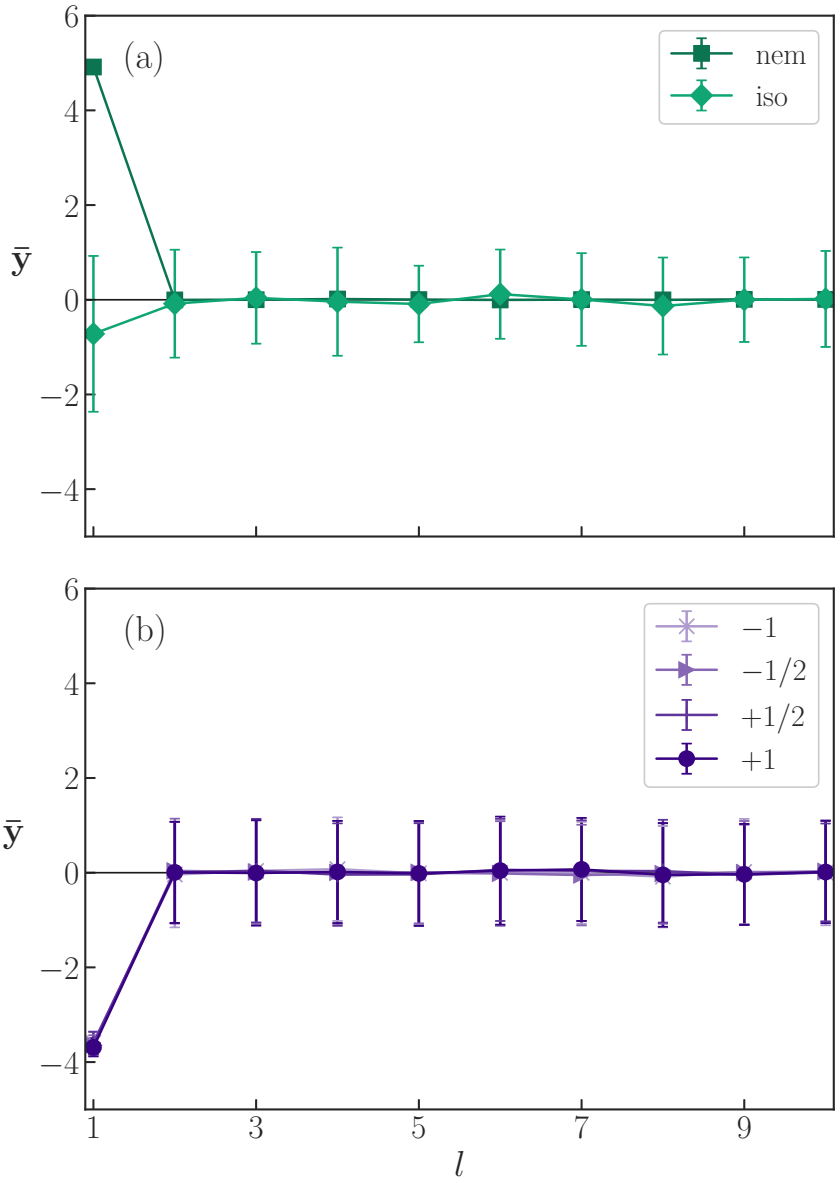


Figure 6.4: Response curves from the WA method on (a) nematic and isotropic samples, and (b) defect state samples. Each point is an average of 800 samples, with error bars measuring the standard deviation.

WD method

The PCA-reduced samples of the WD method are shown in Figure 6.5 along with the EVRs. The most prominent features of the reduced samples are several clear strikes or bands and a large rotated-S curve that spans $\mathbf{y}$. There is also a concentrated cluster around $[\mathbf{y_1}, \mathbf{y_2}] = [-10, -2]$. We can understand these clusters by considering the eigenvectors of Figure 6.1, and by inverting the PCA decomposition to find what a feature vector looks like for a given point (since $\mathbf{W}$ is invertible). Before doing so, it is worth viewing the feature vectors of nematic, isotropic, and defect samples.

Figures 6.8 & 6.9 show feature vectors and response curves for these six samples. We recall that the WD feature vector is the adjusted rod orientations relative to the nematic director,

$$\mathbf{x}^{(i)} = [\widetilde{\theta}_{1,\alpha}, \widetilde{\theta}_{2,\alpha}, \ldots, \widetilde{\theta}_{N_{nn},\alpha}].$$

The adjustment is required so that rotations can be signaled, and is due to the rod symmetry. Without adjustment the feature vector signal would be scattered from rods that are some factor of π from the angle that they *should* be: the angle that makes sense with their surrounding neighbours and in step with the winding, if present. Consider several rods that are adequately aligned with angle θ. From the Monte Carlo, some rods will reliably be a factor of π away from this orientation. However, the PCA algorithm needs to have consistency in the representation of aligned rods. It is no good for any given sample of aligned rods to have a random number of them be degrees of π away from the group alignment, they must all be expressed as the angle closest to θ.

From this point the solution suggests that while proceeding around the sample, reexpress each rod angle as that which is closest to the neighbour just before it. This solves much of the problem, but unfortunately has its limits. In detecting defects, the sticky issue of "false rotations" thus emerges. It is occasional for some number of rods to be disruptive of what seems to be the general trend of a sample. Consider a uniformly aligned nematic sample. It only takes a few deviant rods that are even only somewhat perpendicular to the general alignment to produce a false rotation. For example, if each deviant rod is $\sim +\pi/4$ from its neighbour, it only takes a few to cause a significant positive net rotation. After returning to the other rods that are well aligned, a $+\pi$ or even $+2\pi$ rotation could have been falsely signaled.

Indeed, what the alignment *should* be is not always clear and is the Achilles heel of this method. In some cases it is even ambiguous to human scrutiny what the correct angle of such a rod is, and whether a rotation or defect is really present. Two such examples are

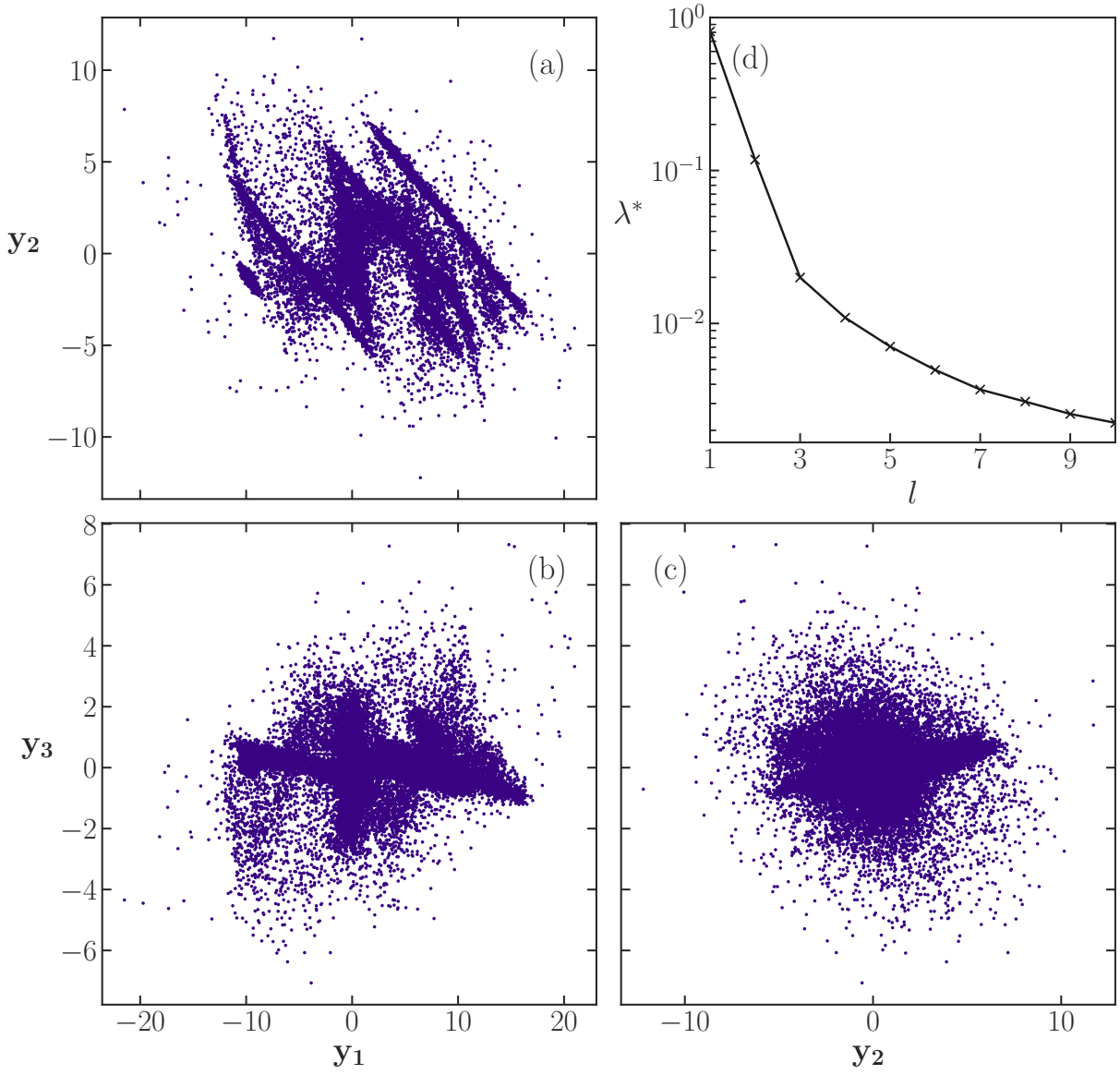


Figure 6.5: Samples in the PCA reduced space for the first three components, and the EVRs for the first 10 components, using the WD method. Every fifth data point is plotted to reduce saturation.

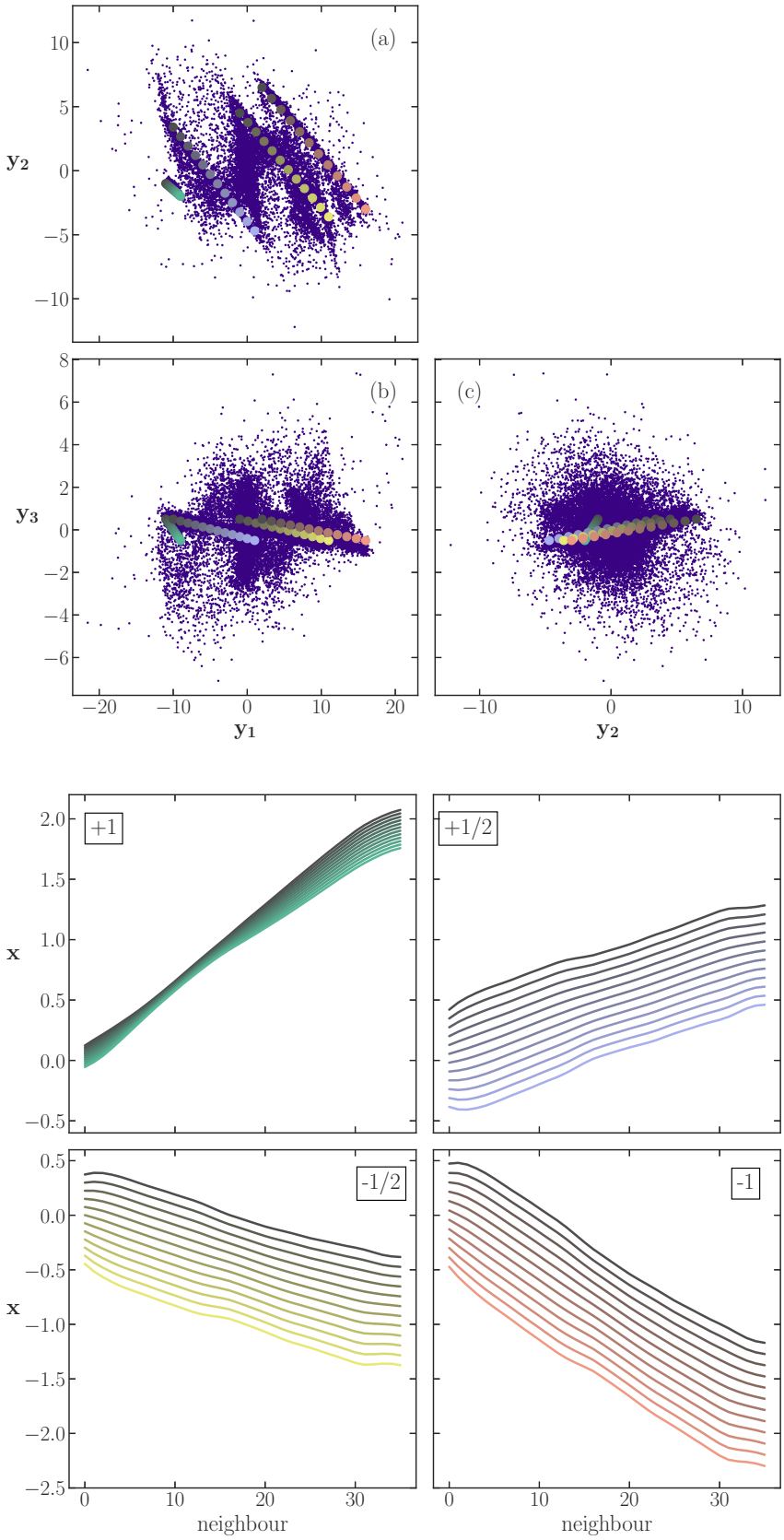


Figure 6.6: Inverting the PCA process along the coloured bands shows they correspond to the four winding defects.

shown in Figure 6.7. In 6.7(a), rods 16-19 cause a clear false $-\pi$ rotation, and in 6.7(b) rods 2-4 cause a false $+\pi$ rotation.

In Figure 6.8 are two examples of the WD response to nematic and isotropic states. The nematic state in (a) shows a minimal response as desired. The behaviour of isotropic states is undefined. As in (b), a -2π rotation is signaled, but the sample is not actually a -1 winding defect.

Figure 6.9 shows responses from four defect samples, each producing a clear signal in the desired form. Averages for isotropic, nematic, and defect samples are shown in Figure 6.10. Each point is an average over 800 images of each type of sample. Nematic aligned samples all strongly have a zero response, and isotropic samples have the large variance predicted. The unpredictable response on isotropic samples is actually not of concern since defect detection of the sort considered here only applies to nematic domains.

Most interestingly, the four defect states are relatively well distinguished. The $+1$ defect samples have very little variance. Because its initialized form has (almost) no crossing, the relaxed form looks highly similar. Additionally, its radial symmetry means that it is not sensitive to the direction of α. These two factors cause minimal variance for $+1$ defect types. Conversely, other states may lose their shape somewhat after uncrossing, and are also sensitive to α. Thus, they have a higher variance in their feature vectors and responses.

This variance can also be seen in Figure 6.6. $+1$ defects have very little variance. Additionally, the $\mathbf{w_2}$ component responds to vertical shift in feature vectors corresponding to the starting direction of α. Each of the higher order components ($\mathbf{w_3}, \mathbf{w_4}$,...) respond to feature vectors similar to their shape. $\mathbf{w_3}$ responds to a single bump in the first half, etc.

Although the WD method is susceptible to giving false responses for edge cases, it is very effective at sensing and classifying defect samples if they are reasonably clear. On this point, we note that defect detection is inherently imperfect with plenty of cases being unclear or partial defects, and even being ambiguous to human classification.

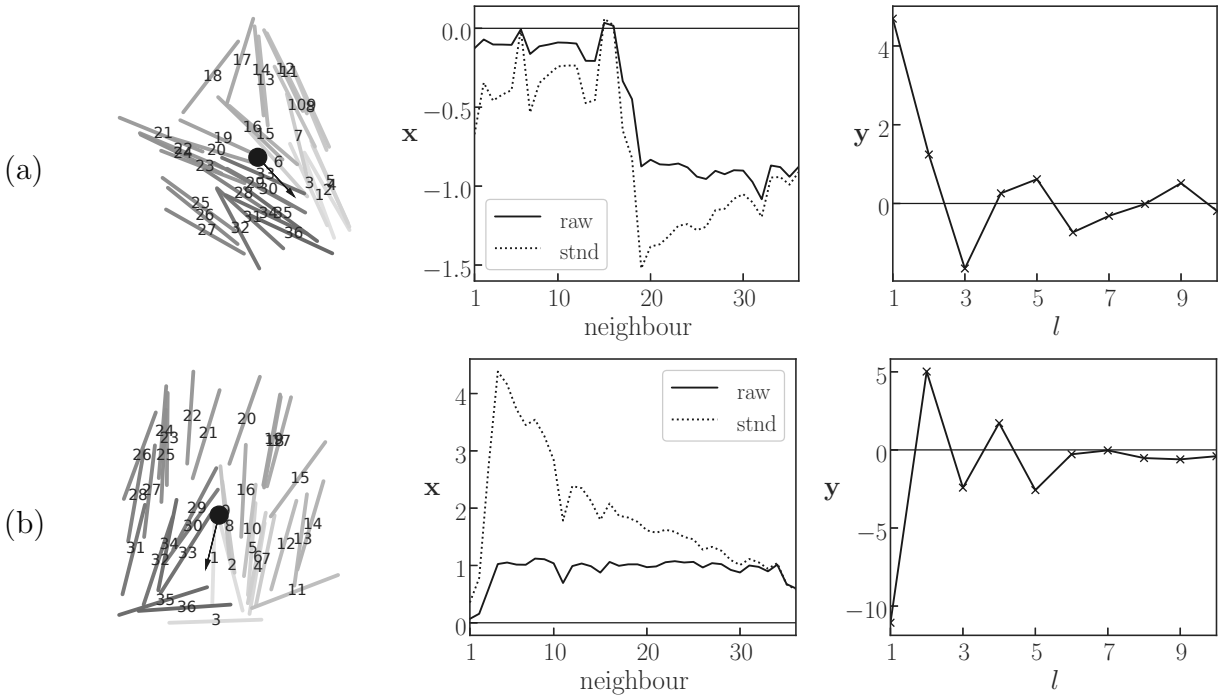


Figure 6.7: Feature vectors (middle column) in units of π and their PCA-basis response (right column) for two cases of potential false rotations. Dotted lines are the feature vector plots are after standardization.

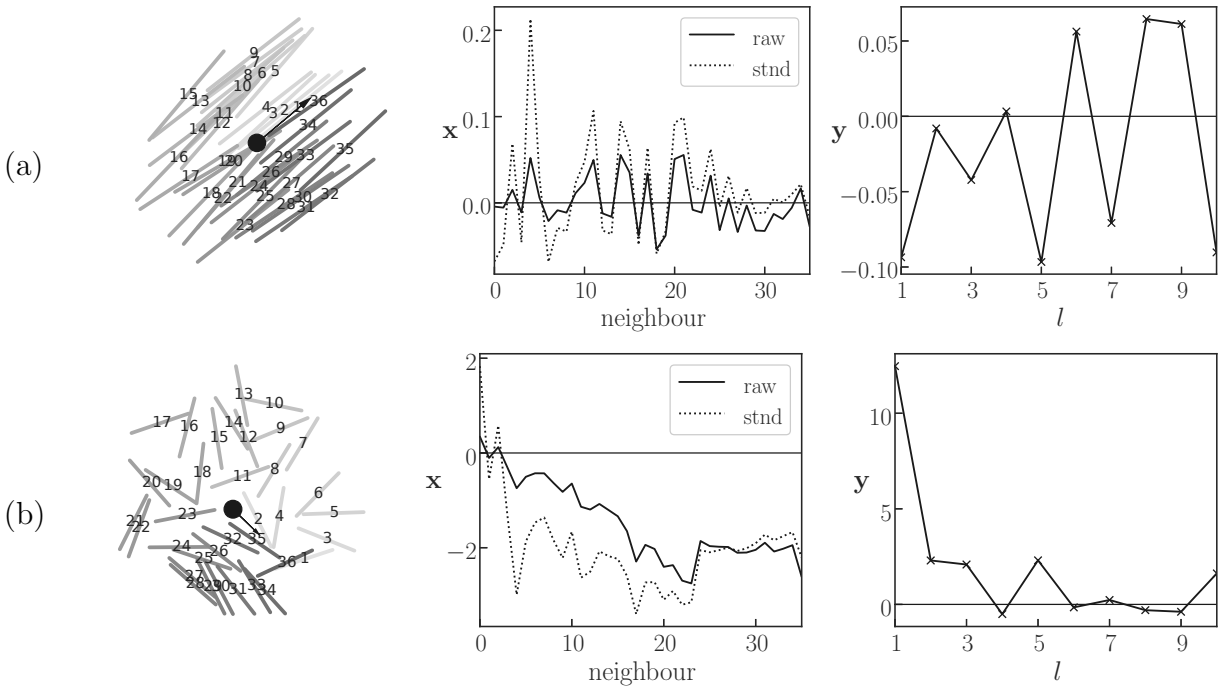


Figure 6.8: Feature vectors (middle column) in units of π and their PCA-basis response (right column) for nematic (a) and isotropic (b) samples under the WD method. Neighbour numbers are overlain on top of the rods and the nematic director is indicated with a black arrow. Dotted lines are the feature vector plots are after standardization.

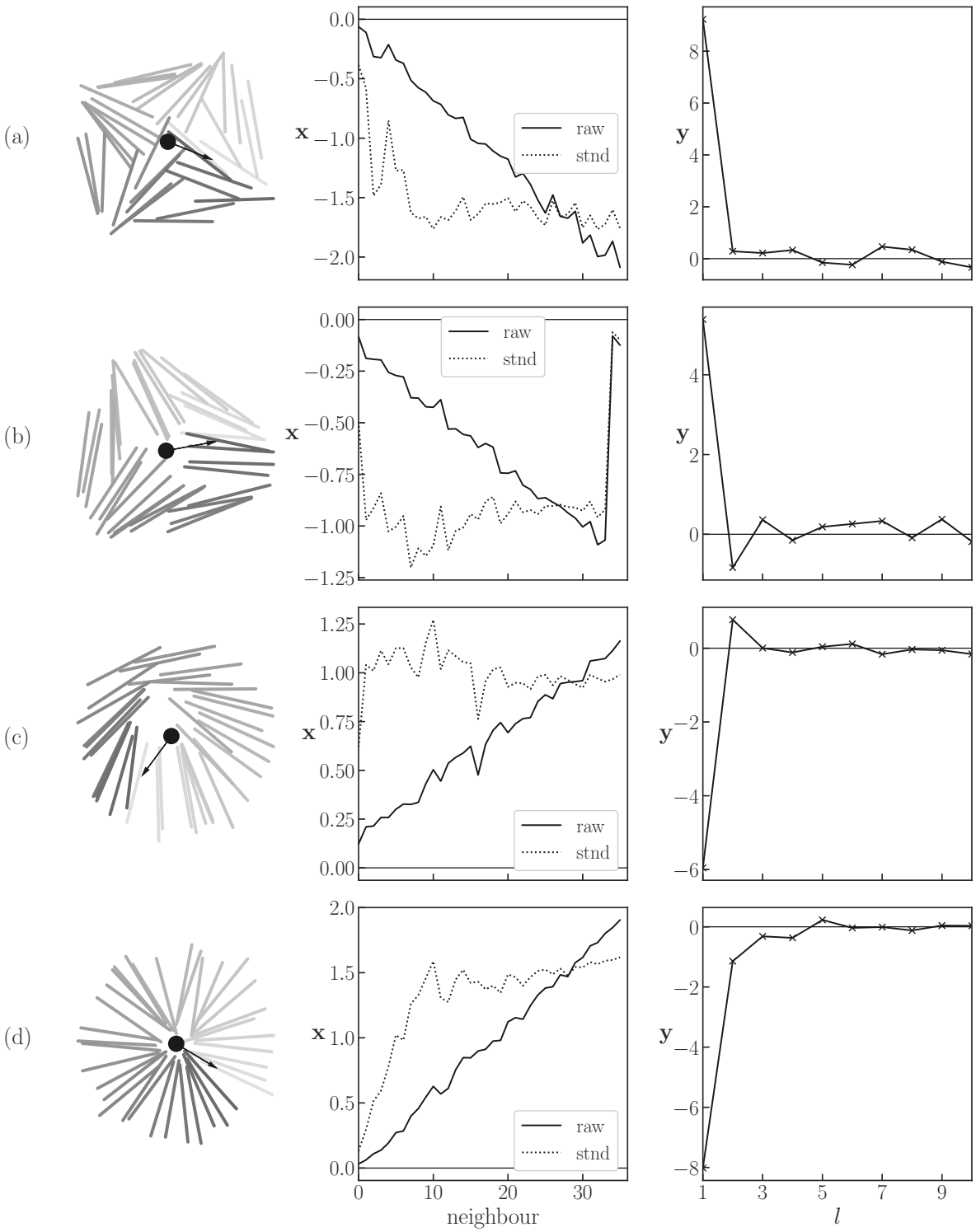


Figure 6.9: Feature vectors for the studied defect states in the WD method. The values in $\mathbf{x}$ are in units of π. Going from light to dark, rod colouring indicates neighbour index. Winding starts from the black arrow nematic director.

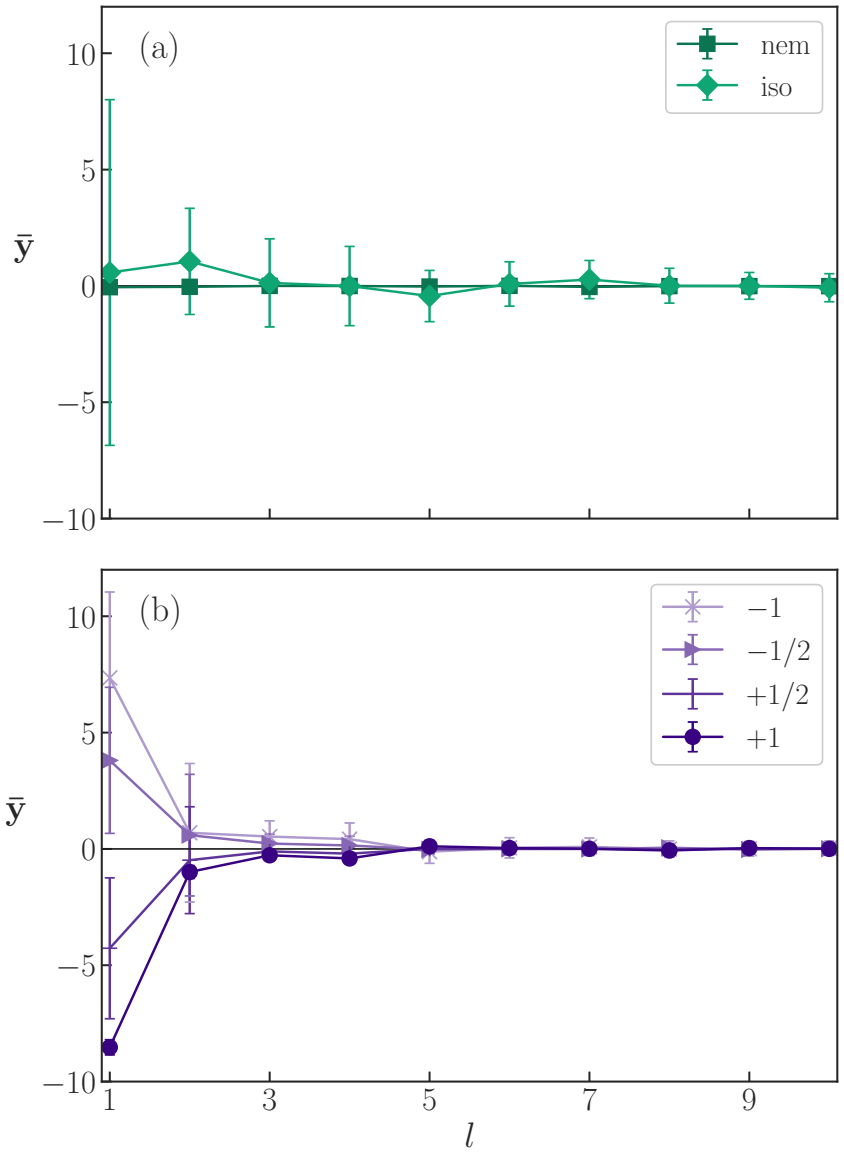


Figure 6.10: Response curves from the WA method on (a) nematic and isotropic samples, and (b) defect state samples. Each point is an average of 800 samples, with error bars measuring the standard deviation.

WA method

Reduced dimension samples from the WA method are plotted in Figure 6.11. Recall each neighbour j feature $x_j^{(i)} = \cos(2\theta_{j\alpha^{(i)}})$. The first component EVR captures $\approx 30\%$ of the variance, with the subsequent three sharing $\approx 15\%$ each. The 5^{th} component captures $\approx 10\%$ before a sharp drop to $\approx 1\%$ for the remaining components.

To understand the forms of Figure 6.11 we consider the eigenvectors in Figure 6.1, and the I-N and defect feature vector and responses of Figures 6.12 & 6.13. First off, similar to the other models, $\mathbf{w_1}$ (Figure 6.1) captures the average alignment to α, with a couple extra dips at the ends and middle. The dips indicate, for uncertain reasons, those neighbours are slightly less significant for signaling what $\mathbf{w_1}$ is trying to capture. The dips at the endpoints are likely because a large portion of defect sample feature vectors start and end near 1 (see examples in Figure 6.13). Nematic and other well-aligned states will also be ≈ 1 (Figure 6.12), so the neighbours near the start and end are not as useful in discerning defects from nematic samples. Again, ± 1 defect vectors return to 1 in their centers and so the center information is less useful as well.

The next four eigenvectors show a pronounced sinusoidal form. These forms respond to the sinusoidal forms of defect samples, as can be seen in Figure 6.13. Notice that in the eigenvectors in Figure 6.1, $\mathbf{w_4}$ and $\mathbf{w_5}$ pair with $\mathbf{w_2}$ and $\mathbf{w_3}$, respectively, by a phase difference. This would arise from different starting points, α, in the procession. Being $\pi/2$ out of phase with their pair, $\mathbf{w_4}$ and $\mathbf{w_5}$ capture the maximum number of defects missed by $\mathbf{w_2}$ and $\mathbf{w_3}$. For the sake of a more effective defect detector, we may add the responses from these pairs in quadrature. That is

$$\widetilde{\mathbf{y}}_{\mathbf{2}}^{(i)} = \sqrt{\mathbf{y_2}^{(i)} + \mathbf{y_4}^{(i)}},$$
$$\widetilde{\mathbf{y}}_{\mathbf{3}}^{(i)} = \sqrt{\mathbf{y_3}^{(i)} + \mathbf{y_5}^{(i)}}.$$

In these forms, $\widetilde{\mathbf{y}}_{\mathbf{2}}$ is an effective $\pm 1/2$ defect detector, and $\widetilde{\mathbf{y}}_{\mathbf{3}}$ for ± 1 defects.

In Figure 6.14, responses of for I-N and defect samples are shown. $\mathbf{w_1}$ is certainly effective at distinguishing nematic samples from others, and even the defects are distinguished from the isotropic states by a clear margin. However, $\mathbf{w_1}$ cannot discern defect type whatsoever. In the quadrature components, $\widetilde{\mathbf{w}}_{\mathbf{2}}$ has a clear response to $\pm 1/2$ defects, and to the same degree $\widetilde{\mathbf{w}}_{\mathbf{3}}$ responds to ± 1 defects.

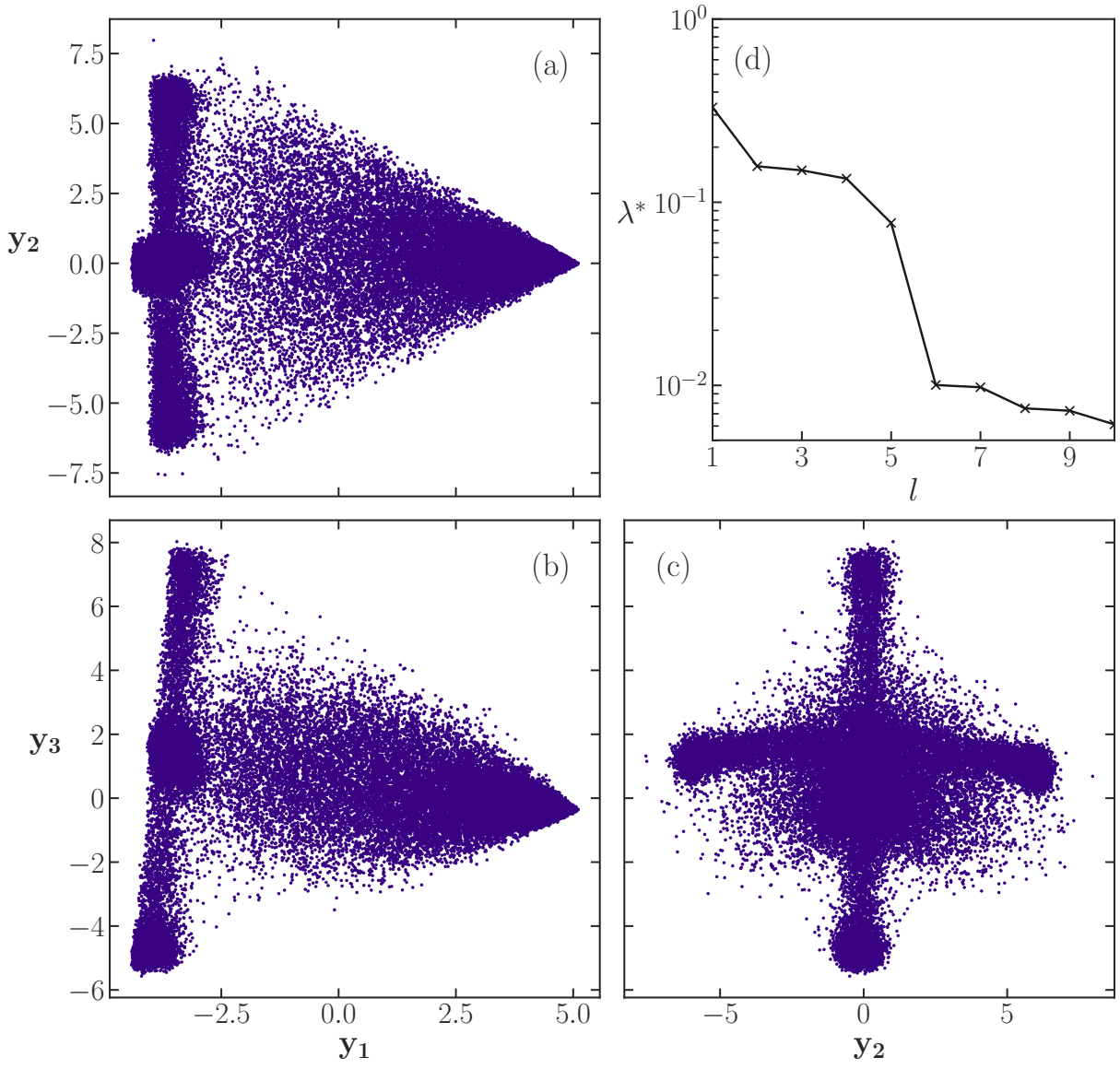


Figure 6.11: Samples in the PCA reduced space for the first three components, and the EVRs for the first 10 components, using the WA method. Every fifth data point is plotted to reduce saturation.

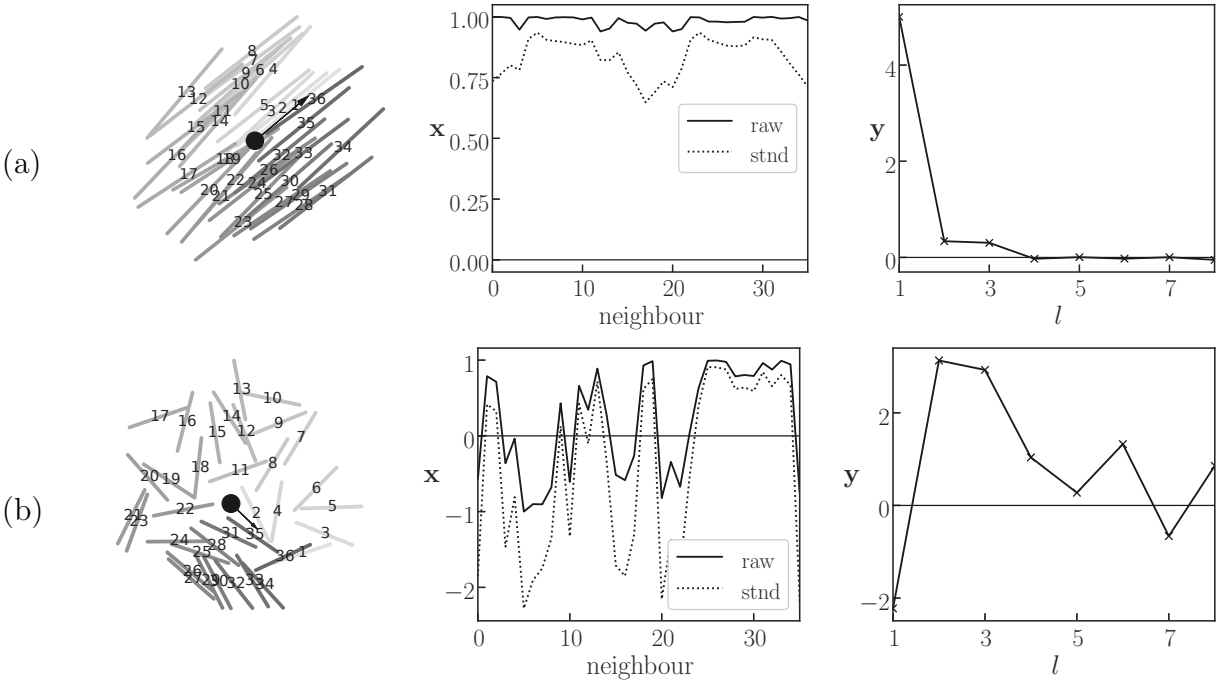


Figure 6.12: Feature vectors (middle column) and their PCA-basis response (right column) for nematic (a) and isotropic (b) samples under the WA method. Neighbour numbers are overlain on top of the rods —and the nematic director is indicated with a black arrow. Dotted lines are the feature vector plots are after standardization.

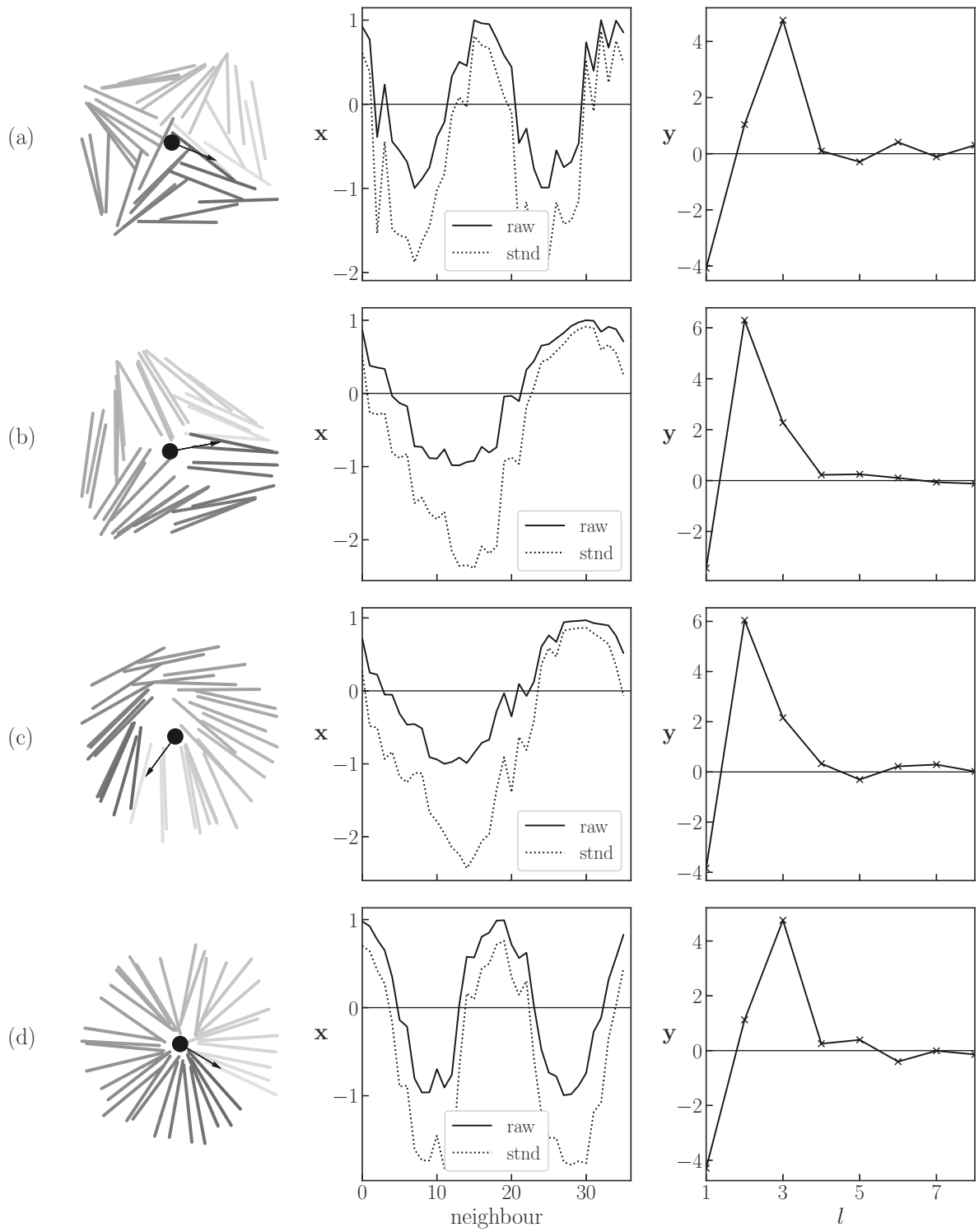


Figure 6.13: Feature vectors for the studied defect states in the WA method. Going from light to dark, rod colouring indicates neighbour index. Winding starts from the black arrow nematic director.

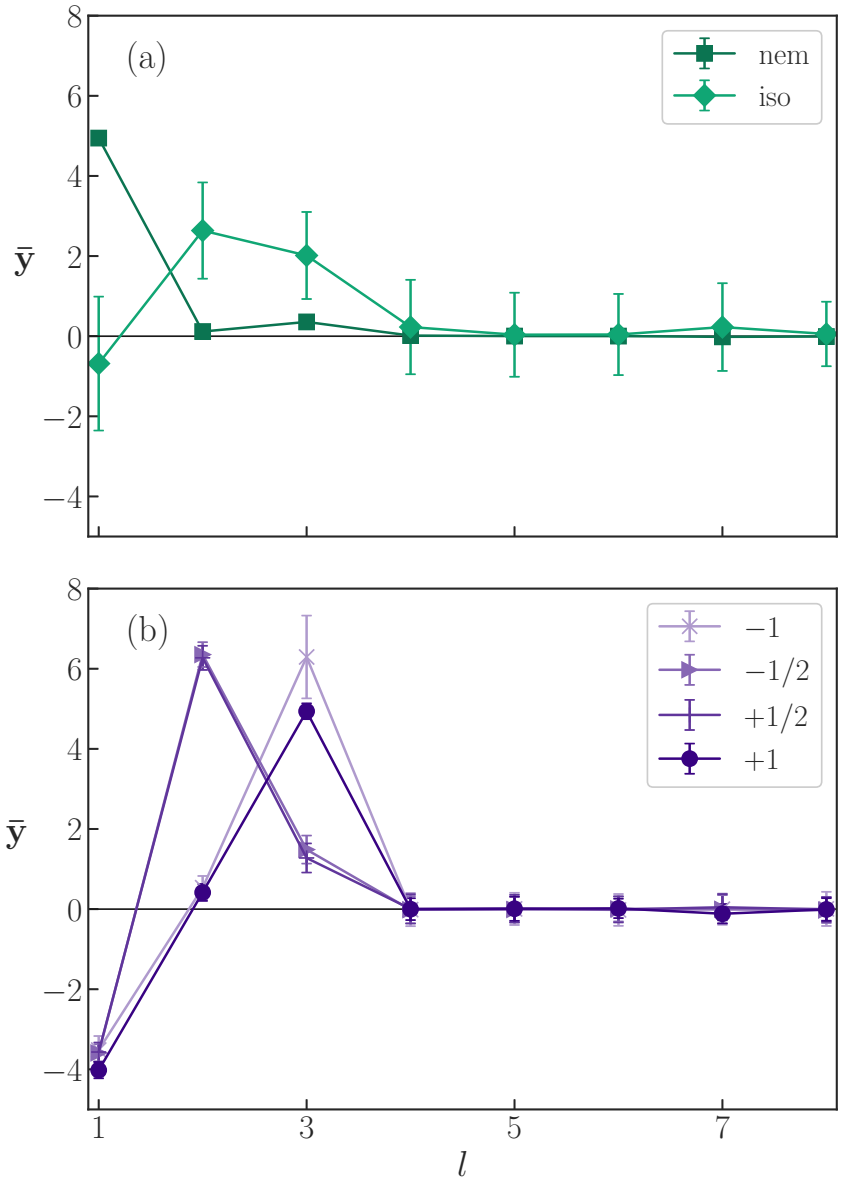


Figure 6.14: Response curves from the WA method on (a) nematic and isotropic samples, and (b) defect state samples. Each point is an average of 800 samples, with error bars measuring the standard deviation.

PCA on full domains

To see how these methods perform on full LC domains, we may cast a 20×20 lattice of probes on the DTUX topologies, and a large system of 3600 rods at a nematic density of $\rho = 16$. This large system was randomly initialized and allowed to equilibrate for 100 000 MCSs. This stopping point was selected because it had a suitable number of nematic grains and defect points.

First, in Figure 6.15 (a)-(d) the WD probe responses to DTUX states are overlain the LC snapshot, and the same responses are overlain the training dataset in (e)-(h). We can see that the proper regions (those predicted from theory and highlighted in Figure 2.2) are excited. Of note, the center -1 defect of X is captured by the points in the -1 band of (e), and none of the other states have points in this band. In every case, the $-1/2$ defects are well captured.

Looking at Figure 6.16, WD captures almost all areas that could be argued as defects. There are a couple spots (bottom-left, near-top-right) that look missed. However, this could easily be a colouring issue along with the fact that the transition from nematic to defect is not sharp. The top and bottom of the nematic band connects with the $\pm 1/2$ bands. The scatter plot of Figure 6.16 shows multiple points in this critical region, along with some others that hover on the colour dropoff zone.

The next few Figures, 6.17, 6.18, 6.19, and 6.20 show the WA method performance on the same DTUX states and the large 3600 rod system. The quadrature components $\tilde{\mathbf{y}}_\mathbf{2}$ and $\tilde{\mathbf{y}}_\mathbf{3}$ are used for the 2nd and 3rd components (except for in Figure 6.15 since colouring is based on $\mathbf{y_1}$). The DTUX responses are accurate and also cleaner than the WD method. As a plain defect vs. nematic detector, this provides a great tool. Looking at the probe responses to $\widetilde{\mathbf{w}}_\mathbf{2}$, we should expect something similar the $\mathbf{y_1}$ except for a weakening on defects closer to ± 1 winding. Indeed, we can see the response does weaken near the center of the X state. Otherwise, it delivers the appropriate responses, minus a bit of patchiness in the U state. The $\widetilde{\mathbf{w}}_\mathbf{3}$ response in Figure 6.19 is not too clean. We can see from plots (e)-(h) that the samples in these DTUX states do not have the strong ± 1 defects the model was trained on. The patchiness shows that $\widetilde{\mathbf{w}}_\mathbf{3}$ is not well generalized. Finally, in Figure 6.20, all three components responses are coloured on the large system snapshot. Each method picks up all the defects, except for in (c) $\widetilde{\mathbf{w}}_\mathbf{3}$ ignores the $+1/2$ strong defect on the right wall. The WA method more completely covers the defect areas than the WD method.

In summary, these methods form an intriguing study on learning topological features with PCA. The Random method is only useful for detecting nematic regions. The WD method is the most useful tool for differentiating defect types and nematic regions. Its

response could possibly be further cleaned up with some clustering on the defect and nematic bands, but the gradual transition from nematic to $\pm 1/2$ to ± 1 cannot be avoided. The highly variable responses to isotropic samples is a non-issue since defect detection only applies to nematic systems. The WA method sacrifices winding polarity for a more robust defect detector. The efficacy of the $\widetilde{\mathbf{w}}_{\mathbf{2}}$ and $\widetilde{\mathbf{w}}_{\mathbf{3}}$ components is likely hindered from their bias towards the eigenvector forms $\mathbf{w_2}, \mathbf{w_3}, \mathbf{w_4}$, and $\mathbf{w_5}$ which have weaker responses as samples deviate from their forms.

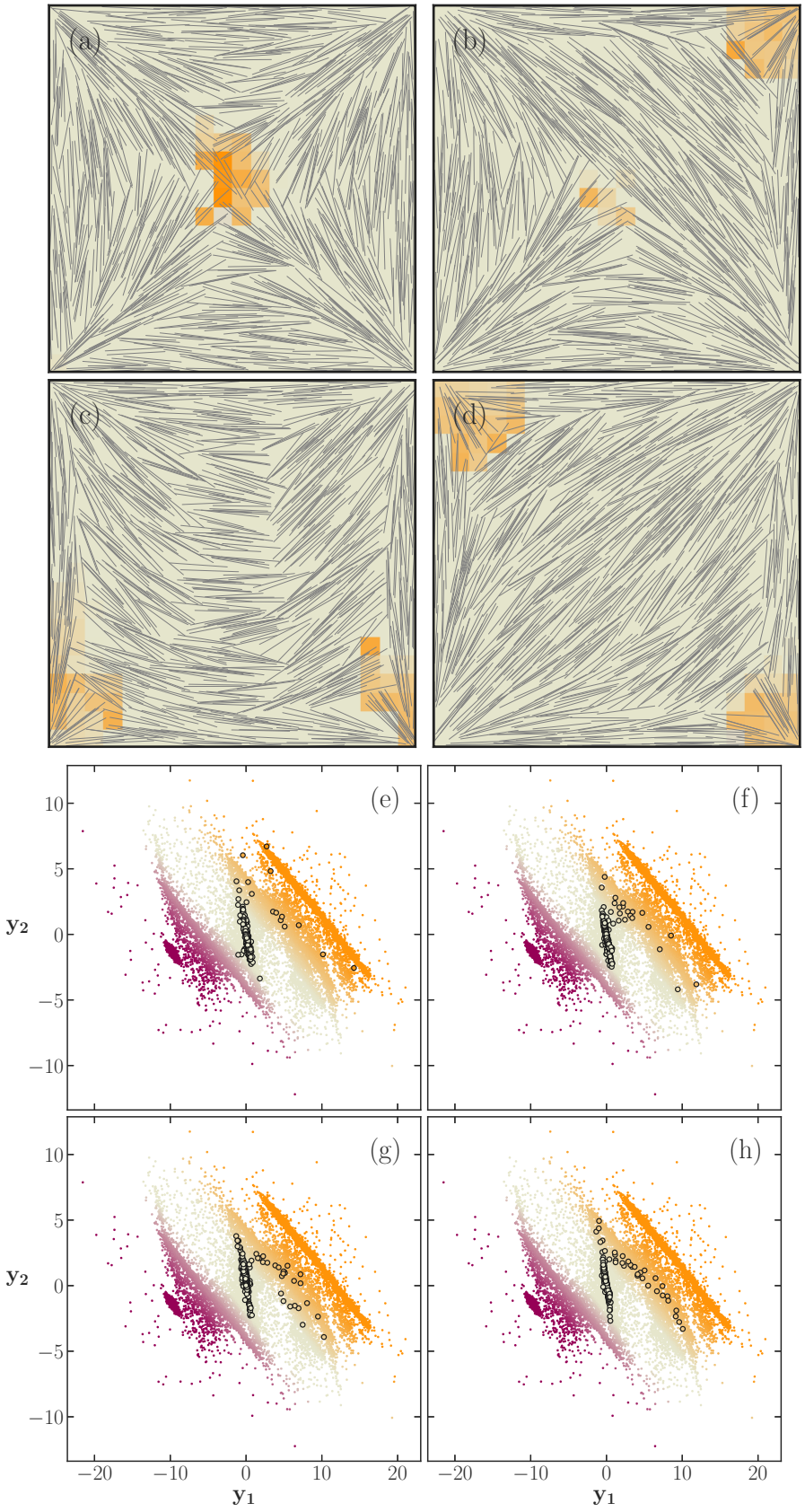


Figure 6.15: WD method on the X (a,e), T (b,f), U (c,g) and D (d,h) topologies. Probe colours are overlaid with the rod plots in (a)-(d), and over the training set in (e)-(h).

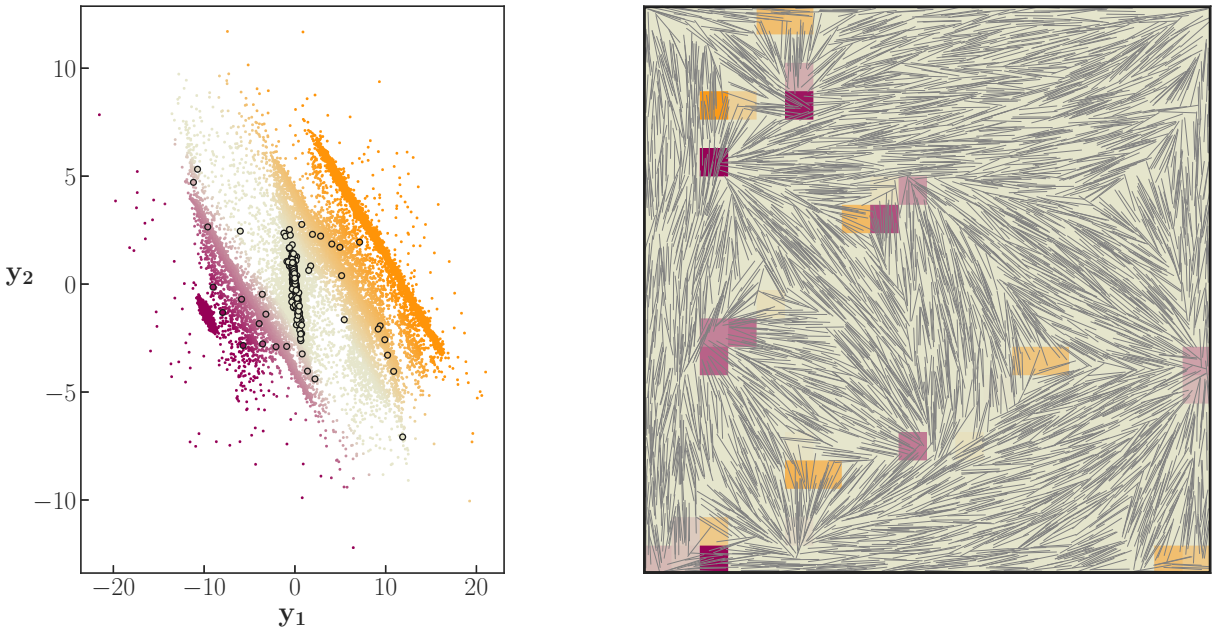


Figure 6.16: WD method on a large liquid crystal domain. Purple (orange) colouring corresponds to positive (negative) defect winding. Probe samples are overlaid with the training set (left) and the corresponding rod-molecule plot (right).

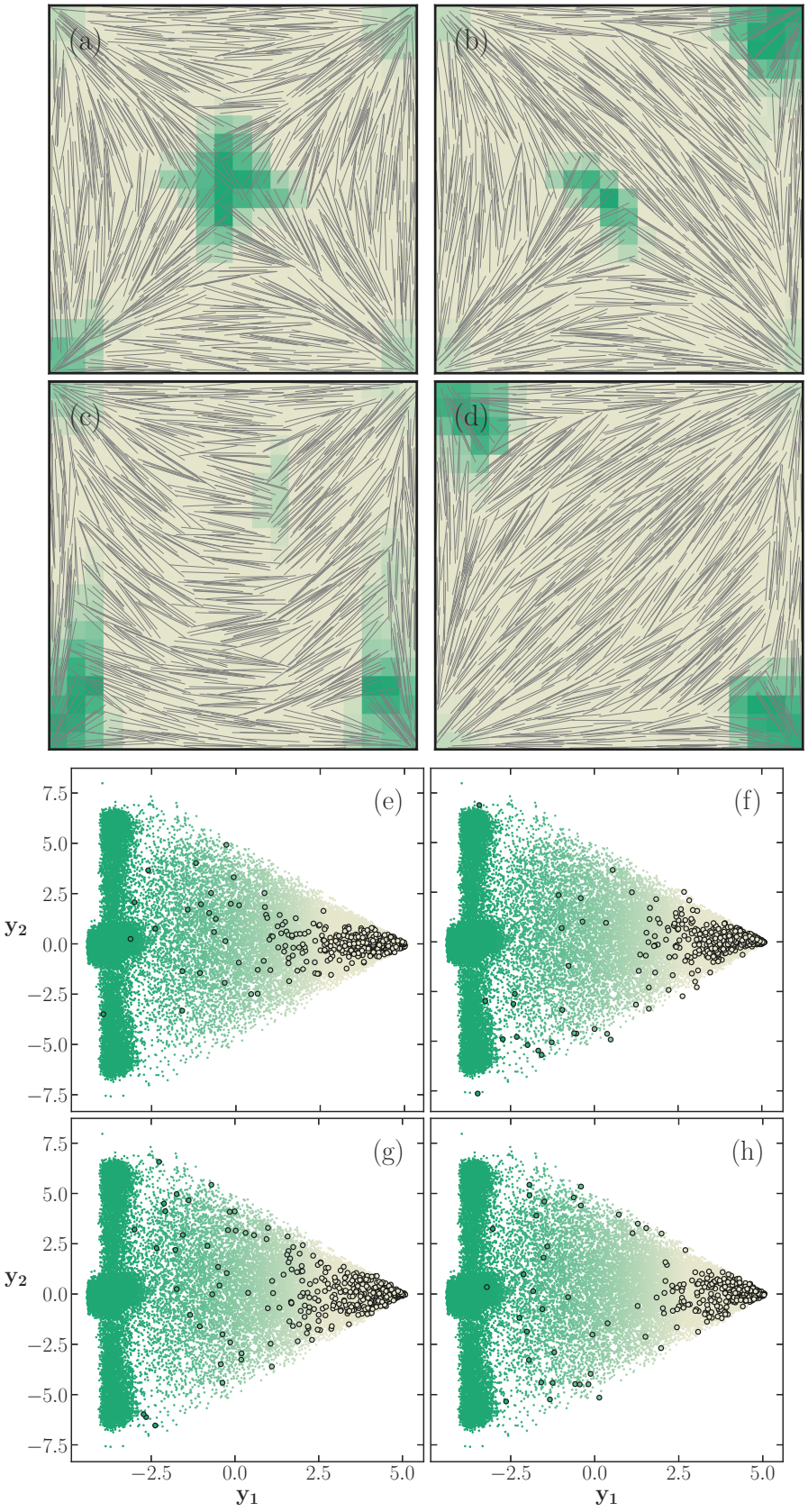


Figure 6.17: WA $\mathbf{w_1}$ response to X (a,e), T (b,f), U (c,g) and D (d,h) topologies. Probe colours are overlaid with the rod plots in (a)-(d), and over the training set in (e)-(h).

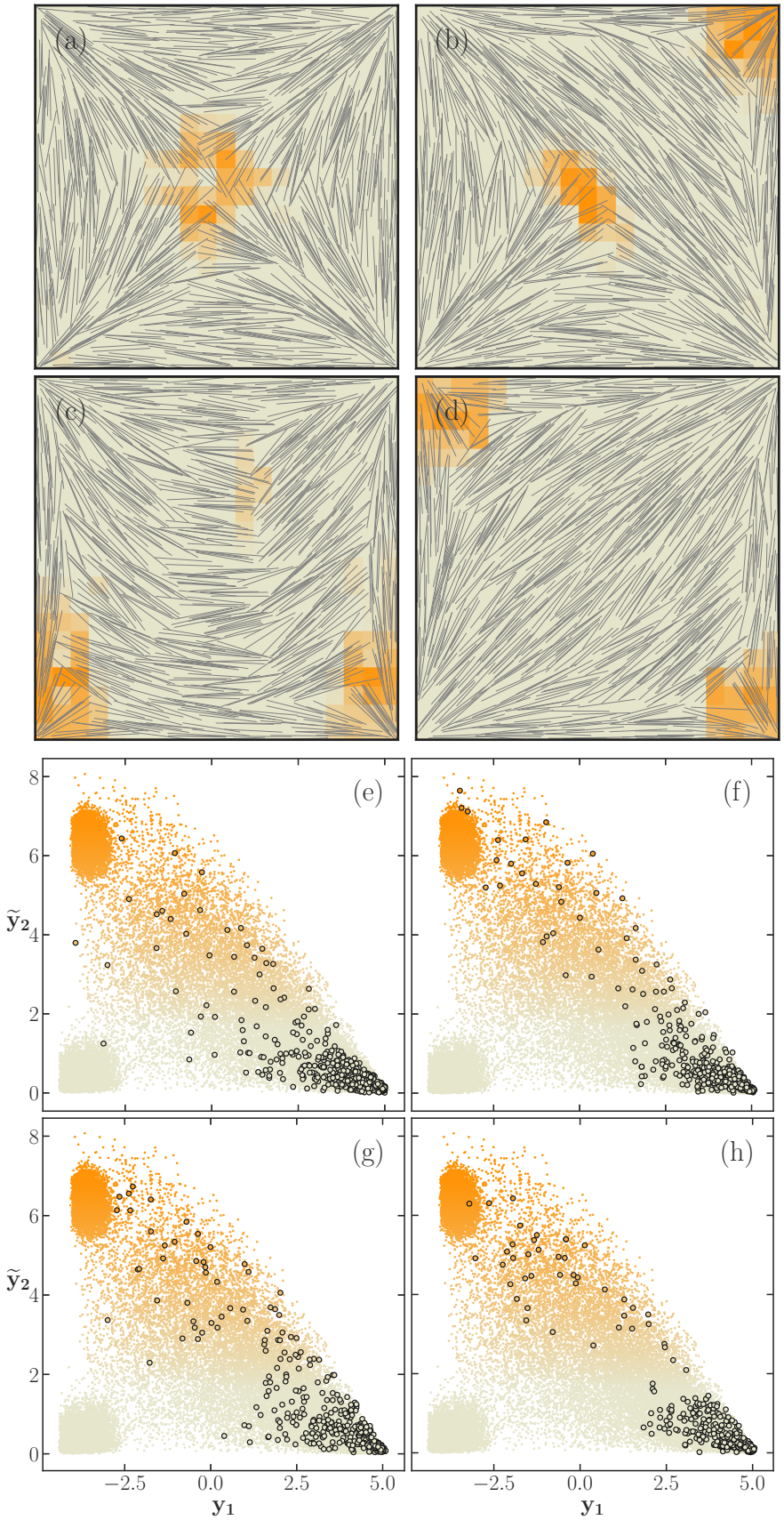


Figure 6.18: WA $\widetilde{\mathbf{w}}_2$ response to X (a,e), T (b,f), U (c,g) and D (d,h) topologies. Probe colours are overlaid with the rod plots in (a)-(d), and over the training set in (e)-(h).

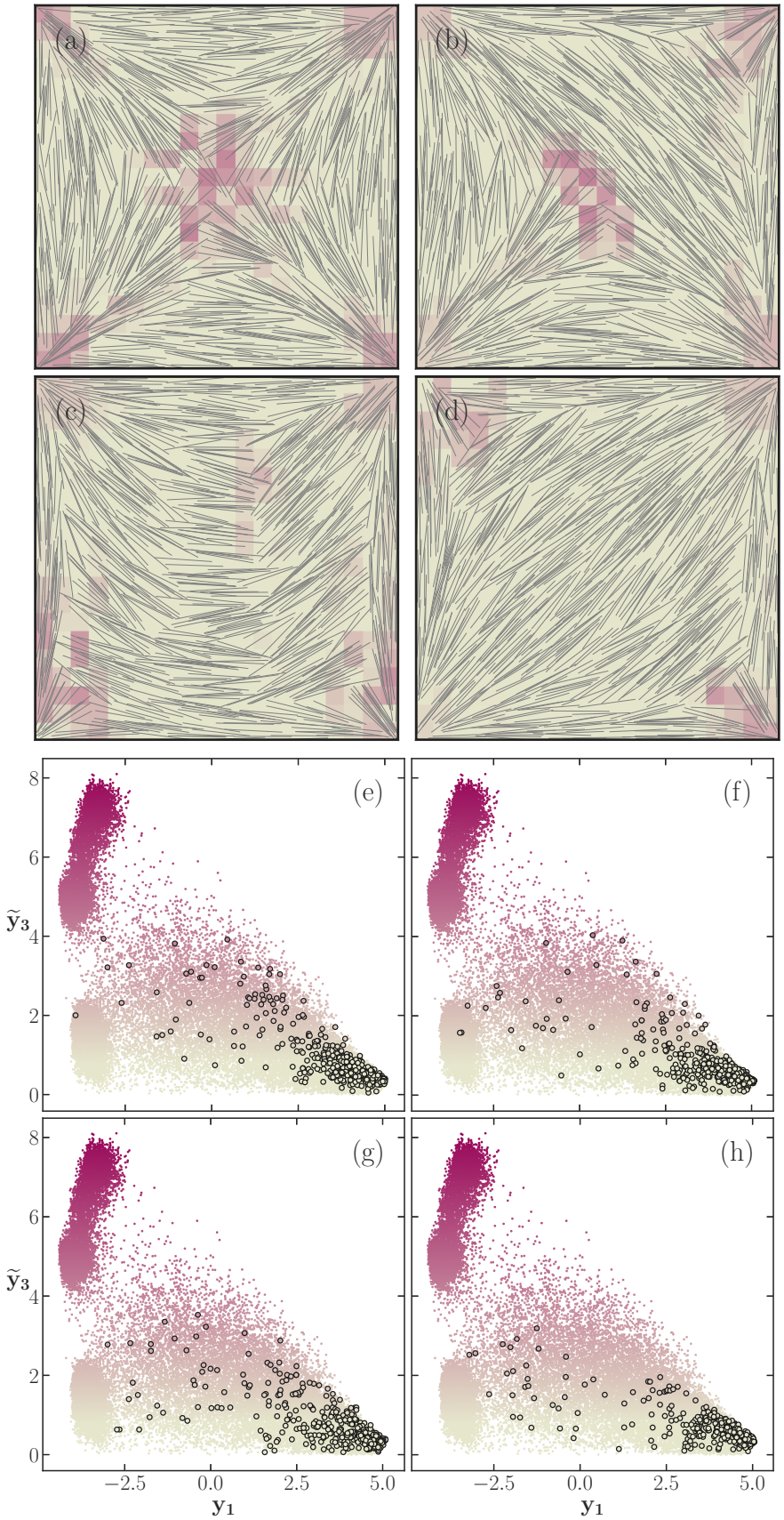


Figure 6.19: WA $\widetilde{\mathbf{w}}_3$ response to X (a,e), T (b,f), U (c,g) and D (d,h) topologies. Probe colours are overlaid with the rod plots in (a)-(d), and over the training set in (e)-(h).

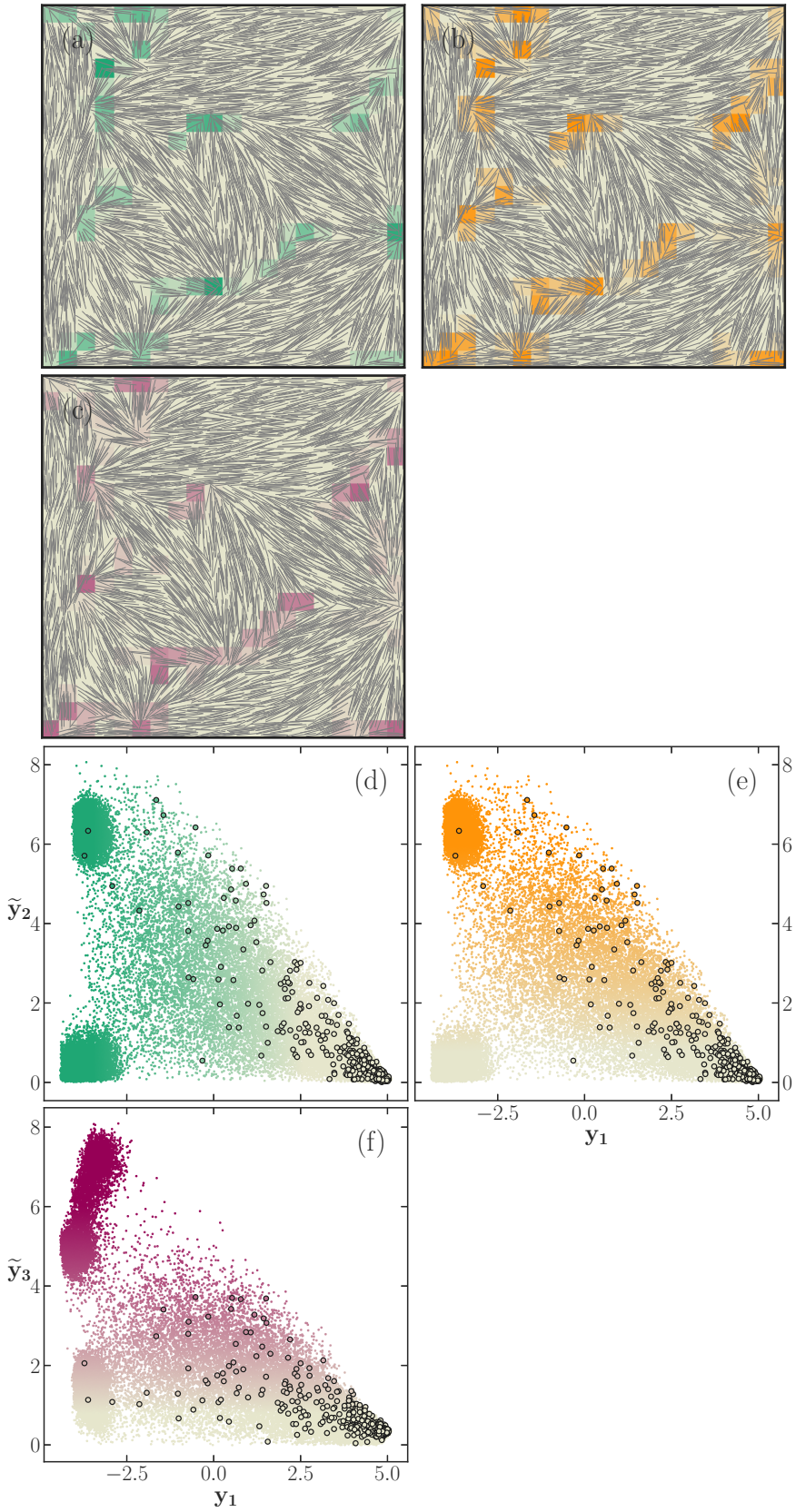


Figure 6.20: WA method on a large liquid crystal domain. Green, orange, and purple colours correspond to $\mathbf{w_1}$, $\widetilde{\mathbf{w}}_\mathbf{2}$, and $\widetilde{\mathbf{w}}_\mathbf{3}$ responses, respectively. Probe samples are overlaid with the corresponding rod-molecule plot (a)-(c) and training set (d)-(f).

Chapter 7

Summary & Outlook

From this study, we may walk away with the following conclusions. Although CNNs and FNNs are effective for a large set of problems in condensed matter and many-body physics, their architecture is inappropriate for off-lattice systems since there is no bijection between spacial coordinates and input index. We illustrated this dependence with a coarse graining method that added a level of positional sorting to the input. With this aid, the FNN could then learn the DTUX states.

A more elegant solution to this was to correlate spatial and angular information with an RNN. This method is also generalizable since it requires no feature engineering. Tested on the DTUX states and the MNIST digits (Appendix B), the RNN mastered both. Exploiting the RNN capability to correlate spatial features in their sequential inputs is a novel use as far as we know. Our study opens up opportunities for many other off-lattice applications of neural networks. By iteratively feeding an RNN with features to be correlated, it is able to correlate arbitrary numbers of interested features of an off-lattice problem. This is particularly important as off-lattice simulations usually produce datasets with no embedded spatial ordering, whereas lattice simulations automatically have labels implicitly representing spatial locations.

On the unsupervised front, PCA brought us close to the goal of a defect identifier and locater. The Random and WA methods were highly effective at capturing the degree of alignment of samples. In identifying types, the loss of winding polarity in the WA mode made it not as effective as WD which maintains polarity. At the same time, the WD method is not as robust as may be desired on MC samples, and does not produce the smooth response intensity with distance from defect centers, as was displayed with the WA method. However, it is important to remember that certain samples are ambiguous

even to the human eye as to what type of defect is shown, or even if one is present at all.

Looking forward, there are a few enticing directions for further research. A significant portion of the condensed matter with ML research is on quantum or lattice systems, often using CNNs or FNNs. Finding a universal way to "connect" off-lattice systems with those architectures would be extremely useful. The solution, you guessed it, may even be a special NN layer at the beginning that interprets and transforms the off-lattice input into something usable by lattice architectures. Applying the RNN architecture in the way used here on other classical systems would be highly interesting. It would also be fascinating to see how it performs on already-established NN classification benchmarks. Its dominating success shows we did not push it to its potential. Along with this, testing its limits with number of features to correlate should be investigated.

On the unsupervised side of things, although theoretically sensible, in practice the nematic director is not too consistent. As displayed in Figures 6.6 and 6.1, the second WD component seems to account for this α variation. Perhaps it is too variant to be useful. Also, more advanced PCA methods exist, and could help improve the abilities of learning defect samples. Our results here are good proof of concept, but for instance, the WD method is not robust or reliable enough for a serious usage.

Recently, Huembeli et al. (2018 [43], 2019 [44]), used a relatively new UL architecture called a "domain adversarial neural network" (DANN) [33] to extract the most important features for phase learning. The architecture is quite elegant. There are three networks: a feature extractor $\mathcal{F}$, a domain discriminator $\mathcal{D}$, and label predictor $\mathcal{P}$. The original feature vectors $\mathcal{X}$ (raw spins, etc.) are piped into $\mathcal{F}$, and then the output of $\mathcal{F}$ is fed into $\mathcal{D}$ and $\mathcal{P}$. In a phase learning example, the dataset has some data with known labels from deep in the phase space, and others with no labels, importantly, those near the unknown critical point. The predictor $\mathcal{P}$ is trained to guess the correct labels from the output of $\mathcal{F}$, but only sees the labeled training set. The discriminator's goal is to tell if an image comes from the labeled or the unlabeled set. The training proceeds with the goal that $\mathcal{F}$ can output something that $\mathcal{P}$ will know the label of, but $\mathcal{D}$ will not know which set it was from. This trains $\mathcal{F}$ to extract the most important features that signal the phase changes so that those on one side of the phase boundary *look the same* as the labeled data deep in that phase. The product of this training is a high-quality feature extractor.

This method was used to learn the phases of the 2D Ising model, the Bose-Hubbard model, the Su-Schrieffer-Heeger (SSH) model [43], and to detect characteristic features of phase transitions in many-body localization [44]. What makes this a potentially very powerful phase learner is that it uses machine learning *to create* the feature vectors. In

our PCA analysis, the main bottleneck was choosing the optimal feature vector. Perhaps a DANN could produce a much better feature vector. Relating to the issue raised above of connecting off-lattice and lattice systems, maybe here too a DANN could help.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. TensorFlow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] Michael P. Allen and Dominic J. Tildesley. *Computer simulation of liquids*. Oxford University Press, 2017.

[3] Michael P. Allen and Mark R. Wilson. Computer simulation of liquid crystals. *Journal of Computer-Aided Molecular Design*, 3(4):335–353, 1989.

[4] Denis Andrienko. Introduction to liquid crystals. *Journal of Molecular Liquids*, 267:520–541, 2018.

[5] R. Barberi, J. J. Bonvent, M. Giocondo, M. Iovane, and A. L. Alexe-Ionescu. Bistable nematic azimuthal alignment induced by anchoring competition. *Journal of Applied Physics*, 84(3):1321–1324, 1998.

[6] Matthew J. S. Beach, Anna Golubeva, and Roger G. Melko. Machine learning vortices at the Kosterlitz-Thouless transition. *Physical Review B*, 97:045207, 2018.

[7] Léon Bottou, Corinna Cortes, John S. Denker, Harris Drucker, Isabelle Guyon, Lawrence D. Jackel, Yann LeCun, Urs A. Muller, Edward Sackinger, Patrice Simard, et al. Comparison of classifier methods: a case study in handwritten digit recognition. In *Pattern Recognition, 1994. Vol. 2-Conference B: Computer Vision & Image Processing., Proceedings of the 12th IAPR International. Conference on*, volume 2, pages 77–82. IEEE, 1994.

[8] Peter Broecker, Juan Carrasquilla, Roger G. Melko, and Simon Trebst. Machine learning quantum phases of matter beyond the fermion sign problem. *Scientific Reports*, 7(1):8823, 2017.

[9] Juan Carrasquilla and Roger G. Melko. Machine learning phases of matter. *Nature Physics*, 13(5):431, 2017.

[10] D. Carvalho, Noel A. García-Martínez, Jose L. Lado, and Joaquín Fernández-Rossier. Real-space mapping of topological invariants using artificial neural networks. *Physical Review B*, 97(11):115453, 2018.

[11] Mario E. Cerritelli, Naiqian Cheng, Alan H. Rosenberg, Catherine E. McPherson, Frank P. Booy, and Alasdair C. Steven. Encapsidated conformation of bacteriophage T7 DNA. *Cell*, 91(2):271–280, 1997.

[12] Chuang Chen, Xiao Yan Xu, Junwei Liu, George Batrouni, Richard Scalettar, and Zi Yang Meng. Symmetry-enforced self-learning Monte Carlo method applied to the Holstein model. *Physical Review B*, 98:041102, 2018.

[13] Jeff Z. Y. Chen. Structure of two-dimensional rods confined by a line boundary. *Soft Matter*, 9(45):10921, 2013.

[14] Jeff Z. Y. Chen. Theory of wormlike polymer chains in confinement. *Progress in Polymer Science*, 54:3–46, 2016.

[15] Jeff Z. Y. Chen, D. E. Sullivan, and Xiangqun Yuan. Surface-Induced Liquid Crystal Transitions of Wormlike Polymers Confined in a Narrow Slit. A Mean-Field Theory. *Macromolecules*, 40(4):1187–1195, 2007.

[16] Zheng Yu Chen. Continuous isotropic-nematic transition of partially flexible polymers in two dimensions. *Physical Review Letters*, 71:93, 1993.

[17] Zheng Yu Chen and Shi-Min Cui. Orientational wetting layer of semiflexible polymers near a hard wall. *Physical Review E*, 52:3876–3880, 1995.

[18] Kelvin Ch'ng, Juan Carrasquilla, Roger G. Melko, and Ehsan Khatami. Machine learning phases of strongly correlated fermions. *Physical Review X*, 7(3):031038, 2017.

[19] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[20] Kenny Choo, Giuseppe Carleo, Nicolas Regnault, and Titus Neupert. Symmetries and many-body excitations with neural-network quantum states. *Physical Review Letters*, 121:167204, 2018.

[21] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (ELUs). *arXiv preprint arXiv:1511.07289*, 2015.

[22] Louis B. G. Cortes, Yongxiang Gao, Roel P. A. Dullens, and Dirk G. A. L. Aarts. Colloidal liquid crystals in square confinement: isotropic, nematic and smectic phases. *Journal of Physics: Condensed Matter*, 29(6):064003, 2017.

[23] Marco Cosentino Lagomarsino, Marileen Dogterom, and Marjolein Dijkstra. Isotropic-nematic transition of long, thin, hard spherocylinders confined in a quasi-two-dimensional planar geometry. *Journal of Chemical Physics*, 119(6):3535–3540, 2003.

[24] P. G. de Gennes and J. Prost. *The Physics of Liquid Crystals*. International Series of Monographs on Physics. Clarendon Press, 1995.

[25] L. Deng, Michael L. Seltzer, Dong Yu, Alex Acero, Abdel-rahman Mohamed, and G. Hinton. Binary coding of speech spectrograms using a deep auto-encoder. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[26] Ingo Dierking, Giusy Scalia, and Piero Morales. Liquid crystal–carbon nanotube dispersions. *Journal of Applied Physics*, 97(4):044309, 2005.

[27] Ingo Dierking, Giusy Scalia, Piero Morales, and Darren LeClere. Aligning and reorienting carbon nanotubes with nematic liquid crystals. *Advanced Materials*, 16(11):865–869, 2004.

[28] I. Dozov, M. Nobili, and G. Durand. Fast bistable nematic display using monostable surface switching. *Applied Physics Letters*, 70(9):1179–1181, 1997.

[29] William C. Earnshaw and Sherwood R. Casjens. DNA packaging by the double-stranded DNA bacteriophages. *Cell*, 21(2):319–331, 1980.

[30] D. Frenkel and R. Eppenga. Evidence for algebraic orientational order in a two-dimensional hard-core nematic. *Physical Review A*, 31(3):1776, 1985.

[31] Jennifer Galanis, Daniel Harries, Dan L. Sackett, Wolfgang Losert, and Ralph Nossal. Spontaneous patterning of confined granular rods. *Physical Review Letters*, 96:028002, 2006.

[32] Jennifer Galanis, Daniel Harries, Dan L. Sackett, Wolfgang Losert, and Ralph Nossal. Spontaneous patterning of confined granular rods. *Physical Review Letters*, 96(2):028002, 2006.

[33] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[34] Ioana C. Garlea and Bela M. Mulder. Defect structures mediate the isotropic-nematic transition in strongly confined liquid crystals. *Soft Matter*, 11:608–614, 2015.

[35] J. G. Gay and B. J. Berne. Modification of the overlap potential to mimic a linear site–site potential. *The Journal of Chemical Physics*, 74(6):3316–3319, 1981.

[36] Sebastian Gurevich, Ezequiel Soule, Alejandro Rey, Linda Reven, and Nikolas Provatas. Self-assembly via branching morphologies in nematic liquid-crystal nanocomposites. *Physical Review E*, 90(2):020501, 2014.

[37] Ian William Hamley. Liquid crystal phase formation by biopolymers. *Soft Matter*, 6(9):1863–1871, 2010.

[38] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[39] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[40] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[41] Wenjian Hu, Rajiv R. P. Singh, and Richard T. Scalettar. Discovering phases, phase transitions, and crossovers through unsupervised machine learning: A critical examination. *Physical Review E*, 95(6):062122, 2017.

[42] Li Huang and Lei Wang. Accelerated Monte Carlo simulations with restricted Boltzmann machines. *Physical Review B*, 95:035105, 2017.

[43] Patrick Huembeli, Alexandre Dauphin, and Peter Wittek. Identifying quantum phase transitions with adversarial neural networks. *Physical Review B*, 97(13):134109, 2018.

[44] Patrick Huembeli, Alexandre Dauphin, Peter Wittek, and Christian Gogolin. Automated discovery of characteristic features of phase transitions in many-body localization. *Physical Review B*, 99(10):104106, 2019.

[45] R. B. Jadrich, B. A. Lindquist, and T. M. Truskett. Unsupervised machine learning for detection of phase transitions in off-lattice systems. I. foundations. *The Journal of Chemical Physics*, 149(19):194109, 2018.

[46] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[47] Richard F. Kayser and Harold J. Raveché. Bifurcation in Onsager's model of the isotropic-nematic transition. *Physical Review A*, 17:2067–2072, 1978.

[48] Keven Kerkam, Christopher Viney, David Kaplan, and Stephen Lombardi. Liquid crystallinity of natural silk secretions. *Nature*, 349(6310):596, 1991.

[49] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[50] Alexei Yu Kitaev. Unpaired Majorana fermions in quantum wires. *Physics-Uspekhi*, 44(10S):131–136, 2001.

[51] Stephen Kitson and Adrian Geisow. Controllable alignment of nematic liquid crystals around microscopic posts: Stabilization of multiple states. *Applied Physics Letters*, 80(19):3635–3637, 2002.

[52] D. P. Knight and F. Vollrath. Liquid crystals and flow elongation in a spider's silk production line. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 266(1418):519–523, 1999.

[53] P. M. Knoll and H. Kelker. *Otto Lehmann*, pages 51–56. Books on Demand, 2010.

[54] John Michael Kosterlitz and David James Thouless. Ordering, metastability and phase transitions in two-dimensional systems. *Journal of Physics C: Solid State Physics*, 6(7):1181, 1973.

[55] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[56] Otto Lehmann. Über fliessende krystalle. *Zeitschrift für physikalische Chemie*, 4(1):462–472, 1889.

[57] Alexander H. Lewis, Ioana Garlea, José Alvarado, Oliver J. Dammone, Peter D. Howell, Apala Majumdar, Bela M. Mulder, M. P. Lettinga, Gijsje H. Koenderink, and Dirk G. A. L. Aarts. Colloidal liquid crystals in rectangular confinement: theory and experiment. *Soft Matter*, 10(39):7865–7873, 2014.

[58] Junwei Liu, Yang Qi, Zi Yang Meng, and Liang Fu. Self-learning Monte Carlo method. *Physical Review B*, 95:041101, 2017.

[59] Junwei Liu, Huitao Shen, Yang Qi, Zi Yang Meng, and Liang Fu. Self-learning Monte Carlo method and cumulative update in fermion systems. *Physical Review B*, 95:241104, 2017.

[60] F. Livolant and Y. Bouligand. New observations on the twisted arrangement of dinoflagellate chromosomes. *Chromosoma*, 68(1):21–44, 1978.

[61] Chong Luo, Apala Majumdar, and Radek Erban. Multistability in planar liquid crystal wells. *Physical Review E*, 85:061702, 2012.

[62] Michael D. Lynch and David L. Patrick. Organizing carbon nanotubes with liquid crystals. *Nano Letters*, 2(11):1197–1201, 2002.

[63] Tomohiro Mano and Tomi Ohtsuki. Phase diagrams of three-dimensional Anderson and quantum percolation models using deep three-dimensional convolutional neural network. *Journal of the Physical Society of Japan*, 86(11):113704, 2017.

[64] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[65] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, 2013.

[66] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems 27*, pages 2204–2212. Curran Associates, Inc., 2014.

[67] Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller. *Neural networks: tricks of the trade*. Springer, 2013.

[68] Yuki Nagai, Huitao Shen, Yang Qi, Junwei Liu, and Liang Fu. Self-learning Monte Carlo method: Continuous-time algorithm. *Physical Review B*, 96:161102, 2017.

[69] Lars Onsager. The effects of shape on the interaction of colloidal particles. *Annals of the New York Academy of Sciences*, 51(4):627–659, 1949.

[70] 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.

[71] A. Poniewierski. Ordering of hard needles at a hard wall. *Physical Review E*, 47(5):3396, 1993.

[72] D. Porter, F. Vollrath, and Z. Shao. Predicting the mechanical properties of spider silk as a model nanostructured polymer. *The European Physical Journal E*, 16(2):199–206, 2005.

[73] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[74] Ziv Reich, Ellen J. Wachtel, and Abraham Minsky. Liquid-crystalline mesophases of plasmid DNA in bacteria. *Science*, 264(5164):1460–1463, 1994.

[75] Friedrich Reinitzer. Beiträge zur kenntniss des cholesterins. *Monatshefte für Chemie/Chemical Monthly*, 9(1):421–441, 1888.

[76] Friedrich Reinitzer. Contributions to the knowledge of cholesterol. *Liquid Crystals*, 5(1):7–18, 1989.

[77] Martin Robinson, Chong Luo, Patrick E. Farrell, Radek Erban, and Apala Majumdar. From molecular to continuum modelling of bistable liquid crystal devices. *Liquid Crystals*, 44(14-15):2267–2284, 2017.

[78] Andrea Rocchetto, Edward Grant, Sergii Strelchuk, Giuseppe Carleo, and Simone Severini. Learning hard quantum distributions with variational autoencoders. *npj Quantum Information*, 4(1):28, 2018.

[79] Joaquin F. Rodriguez-Nieva and Mathias S. Scheurer. Identifying topological order via unsupervised machine learning. *arXiv preprint arXiv:1805.05961*, 2018.

[80] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386, 1958.

[81] Subir Sachdev. Topological order, emergent gauge fields, and Fermi surface reconstruction. *Reports on Progress in Physics*, 82(1):014001, 2018.

[82] Hiroki Saito and Masaya Kato. Machine learning technique to find quantum many-body ground states of bosons on a lattice. *Journal of the Physical Society of Japan*, 87(1):014001, 2018.

[83] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[84] Or Sharir, Yoav Levine, Noam Wies, Giuseppe Carleo, and Amnon Shashua. Deep autoregressive models for the efficient variational simulation of many-body quantum systems. *arXiv preprint arXiv:1902.04057*, 2019.

[85] Jonathon Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.

[86] Marina Soares e Silva, José Alvarado, Jeanette Nguyen, Nefeli Georgoulia, Bela M. Mulder, and Gijsje H. Koenderink. Self-organized patterns of actin filaments in cell-sized confinement. *Soft Matter*, 7:10631–10641, 2011.

[87] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[88] W. P. Su, J. R. Schrieffer, and A. J. Heeger. Solitons in polyacetylene. *Physical Review Letters*, 42(25):1698, 1979.

[89] Giacomo Torlai, Guglielmo Mazzola, Juan Carrasquilla, Matthias Troyer, Roger G. Melko, and Giuseppe Carleo. Neural-network quantum state tomography. *Nature Physics*, 14(5):447, 2018.

[90] Giacomo Torlai and Roger G. Melko. Learning thermodynamics with Boltzmann machines. *Physical Review B*, 94:165134, 2016.

[91] C. Tsakonas, A. J. Davidson, C. V. Brown, and N. J. Mottram. Multistable alignment states in nematic liquid crystal filled wells. *Applied Physics Letters*, 90(11):111913, 2007.

[92] Sabina W. Ula, Nicholas A. Traugutt, Ross H. Volpe, Ravi R. Patel, Kai Yu, and Christopher M. Yakacki. Liquid crystal elastomers: an introduction and review of emerging technologies. *Liquid Crystals Reviews*, 6(1):78–107, 2018.

[93] Evert P. L. van Nieuwenburg, Ye-Hua Liu, and Sebastian D. Huber. Learning phase transitions by confusion. *Nature Physics*, 13(5):435, 2017.

[94] Michael Walters, Qianshi Wei, and Jeff Z. Y. Chen. Machine learning topological defects of confined liquid crystals in two dimensions. *Physical Review E*, 99:062701, 2019.

[95] Ce Wang and Hui Zhai. Machine learning of frustrated classical spin models. I. principal component analysis. *Physical Review B*, 96(14):144432, 2017.

[96] Ce Wang and Hui Zhai. Machine learning of frustrated classical spin models (II): Kernel principal component analysis. *Frontiers of Physics*, 13(5):130507, 2018.

[97] Lei Wang. Discovering phase transitions with unsupervised learning. *Physical Review B*, 94(19):195105, 2016.

[98] Qianshi Wei, Roger G. Melko, and Jeff Z. Y. Chen. Identifying polymer states by machine learning. *Physical Review E*, 95:032504, 2017.

[99] Sebastian J. Wetzel and Manuel Scherzer. Machine learning of explicit order parameters: From the Ising model to SU (2) lattice gauge theory. *Physical Review B*, 96(18):184410, 2017.

[100] W. W. Wood and J. D. Jacobson. Monte Carlo calculations in statistical mechanics. In *Papers presented at the the March 3-5, 1959, western joint computer conference*, pages 261–269. ACM, 1959.

[101] Dian Wu, Lei Wang, and Pan Zhang. Solving statistical mechanics using variational autoregressive networks. *Physical Review Letters*, 122(8):080602, 2019.

[102] Deng-Ke Yang. *Fundamentals of liquid crystal devices*. John Wiley & Sons, 2014.

[103] Xiaomei Yao, Hui Zhang, and Jeff Z. Y. Chen. Topological defects in two-dimensional liquid crystals confined by a box. *Physical Review E*, 97(5):052707, 2018.

[104] Pengfei Zhang, Huitao Shen, and Hui Zhai. Machine learning topological invariants with neural networks. *Physical Review Letters*, 120(6):066401, 2018.

[105] Wanzhou Zhang, Jiayu Liu, and Tzu-Chieh Wei. Machine learning of phase transitions in the percolation and XY models. *Physical Review E*, 99(3):032142, 2019.

[106] Yi Zhang and Eun-Ah Kim. Quantum loop topography for machine learning. *Physical Review Letters*, 118(21):216401, 2017.

Appendix A

Learning the Isotropic-Nematic Phase Transition

In this work, we used the off-lattice model of rod-like molecules confined in a square box as an example of a system containing topological defects in its nematic state. The dominating physical parameter is ρ. Above a critical value ρ^*, the system is in the nematic (N) state showing a defect pattern such as those in Figure 2.2, and below ρ^* the isotropic (I) state where the rod-like molecules, away from the confinement walls, have random orientations.

Although our main focus of the current work is detecting topological defects in the nematic state, a highly related question is: Can an FNN detect the I-N transition of the system? An effective way to train an FNN to learn the molecular configurations produced from a simulation is via a supervised learning approach with classified images. Here, two classifications, I and N, correspond to nodes (ν_1, ν_2) respectively. This is the same method used in Section 5.2 but with no pre-sorting and two states instead of four. Additionally for this study, configurations produced at different values of ρ are required. At a fixed N, a series of configurations are obtained with varying ρ by adjusting a/L; this is achieved by a Monte Carlo procedure described in Section 5.1.

The FNN training session takes the molecular configurations at two densities, $\rho = 4$ (when the system is in an I state) and $\rho = 16$ (when the system is in a deep N state) so that it can learn the difference between these two states. For this identification we use the D defect state. Only the angles θ_l $(l = 1, N)$ recorded in an MC snapshot are needed here, hence the input layer contains N nodes. The FNN had two hidden layers of size 128 and 32. Expontential linear units were used, along with early stopping, Adam optimization, and 50% dropout [21, 49, 67, 87]. The FNN easily learned the difference between the I and

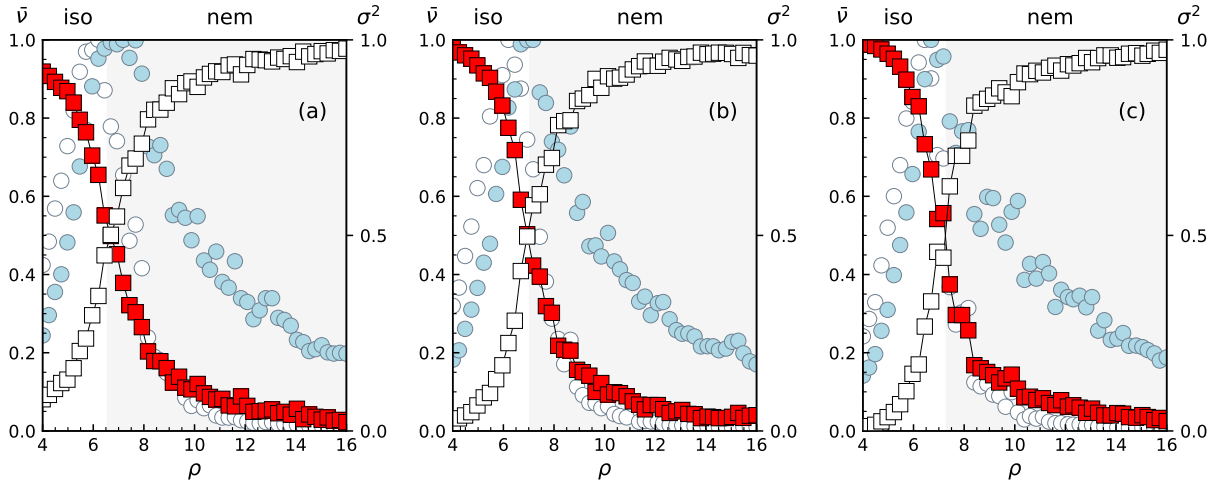


Figure A.1: Identification of the I-N transition point. An FNN is trained by using training files at low and high densities ($\rho = 4$ and 16 for the I and N states, respectively). Then, the trained FNN is used to calculate the average outputs $\bar{\nu_1}$ and $\bar{\nu_2}$ at each given ρ. These characteristic measures are to indicate if the system is in an I or N state, plotted by red and white squares, respectively. Error bars of these data points are smaller than the symbols sizes. The crossing point is defined as the I-N transition point. In the background, represented by circles (to the right scale), an independent estimate of the variance of the orientational order parameter σ^2 (normalized), shows a peak at the transition point. Blue and white circles represent the statistics from rods in the entire box and half-sized center region, respectively. The three plots, (a), (b), and (c), are produced with system sizes $N = 20^2$, 24^2, and 28^2, respectively.

N states, achieving approximately 99% accuracy within 100 training epochs, tested on 600 independent images not used in the training.

Having trained the FNN to identify the ideal I and N states, we then query the network output on unidentified snapshot data taken from MC simulations at various values of ρ in the domain $\rho = [4, 16]$. A total of $k = 5000$ MC snapshots were obtained for each value of ρ and fed into the input layer of the trained FNN. The network is looking for when any order in angular values emerges. When isotropic, the θ input will approach a uniform input (less the box edge alignment). Once the system inclines towards the nematic state, the network will pick up on deviations from this uniform input, signaling the phase transition.

For a given value of ρ with k images, an average

$$\bar{\nu}_1 = \frac{1}{k} \sum_{i=1}^{k} \nu_1^{(i)}$$

is used to identify how strongly the network believes that this density gives an I state; $\bar{\nu_2}$ for state N is also calculated in the same fashion. These averages are plotted in Figure A.1. Near the I-N transition point, due to the large fluctuations of the molecular configurations, the network can only identify the overall states by a percentage certainty. In recent literatures [9, 98], it is customary to use the $\bar{\nu}_1 = \bar{\nu}_2 = 1/2$ crossing position as the flag to identify the critical point ρ^*. For system sizes of $N = 20^2$, $N = 24^2$, and $N = 28^2$, our results suggest $\rho^* = 6.71 \pm 0.25$, 6.95 ± 0.25, and 7.08 ± 0.25, respectively. These values are comparable to those from previous numerical and experimental studies, which suggest values of ρ^* of ≈ 7.0 [30], 6.5 [23], and $6 - 9$ [32]; the mean-field theory calculation of $\rho^* = 3\pi/2L^2 \approx 4.71$ based on the Onsager theory is a known under-estimate [16, 47].

While ρ^* were estimated from the FNN study, we provide here independent estimates from statistical physics for comparison. We start by calculating a nematic order parameter Λ for every configuration file. In 2D, we use the definition

$$\Lambda \equiv \sqrt{C^2 + S^2} \tag{A.1}$$

where

$$C = \langle \cos 2\theta \rangle,$$
$$S = \langle \sin 2\theta \rangle,$$

are averages over the N rod-like molecules within a configuration file. Note that in an ideal I state $\Lambda = 0$, and in a uniformly aligned N state $\Lambda = 1$. Then, we analyze the statistics of

Λ calculated from the 5000 configurations used for each ρ. According to statistical physics, the variance $\sigma^2 = \langle \Lambda^2 \rangle' - \langle \Lambda \rangle'^2$, plotted as a function of ρ, displays a peak at the phase transition density ρ^*. Here $\langle ... \rangle'$ represents an algebraic average of the 5000 data points, not to be confused with a box average $\langle ... \rangle$. One can find the σ^2 data presented by blue circles in Figure A.1. Indeed, this independent analysis produces peak positions closely matching the FNN determination of ρ^*.

In the system sizes studied, the boundary effects can permeate into a non-negligible portion of the domain. Measurements of the order along the box edge is then likely to produce values higher than the bulk in many cases from rod-wall alignment [15, 17, 71]. The definition in Eq. (A.1) is not affected by these boundary effects if vertical and horizontal boundaries make equal contributions to U and T. As an independent check, we also used configurations of rods belonging to a half-sized, centered box to evaluate σ^2. The resultant white circles in Figure A.1 show the same transition density within the numerical error.

Appendix B

FNN and RNN Digit Recognition

The MNIST dataset includes 6×10^4 training images and 10^4 testing images of handwritten numerical digits. One of the images is shown in Fig. 2.2(a). An industrial standard of machine learning is the correct identification of the 10 numerical digits in the MNIST set. It is well-known that deep FNN is an effective tool for this purpose, reaching accuracies above 99% [83]. The output layer shown in Fig. 2.1 now has $n = 10$ nodes, each characterizing a digit, from $0, 1, 2, ..., 9$.

A typical data set to represent an MNIST image has the data structure P_{ij}, where $i = 1, ...28$ is the row number of the pixels, and $j = 1, ...28$ the column number, and P the grey scale "ink" intensity of the writing. Implicitly, when the data is stored according to the order of i and j, spatial location (that is, the pixel position specified by i and j) follows this order. The configuration file of a 2D Ising model would have a similar data structure where P has only two values, up and down.

To compare with the off-lattice output data, we rewrite a data point P_{ij} by a line of four numbers $[l, i_l, j_l, P_{ij}]$ where $l = 1, 2, \ldots, 28 \times 28$ is a sequential number that labels i_l and j_l. This has the same data structure as the configurational record obtained from an off-lattice molecular simulation where a typical line reads $[l, x_l, y_l, \theta_l]$. However, there is one important difference: the MNIST data has a correlation between l and (i_l, j_l) in an orderly fashion, whereas in the off-lattice coordinate record, there is no correlation between l and x_l, y_l.

If we decorrelate l from (i_l, j_l) but use $(i_l, j_l, P_{i_l j_l})$ as the input, can an FNN still identify the numerical digits? The decorrelation is done by scrambling the line ordering of (i_l, j_l, P_{ij}) data for each digit image. As demonstrated by the circles in Figures B.1(a) and (b), the FNN now *fails* to learn the correlation between i, j and $P_{i_l j_l}$. The FNN network looks for

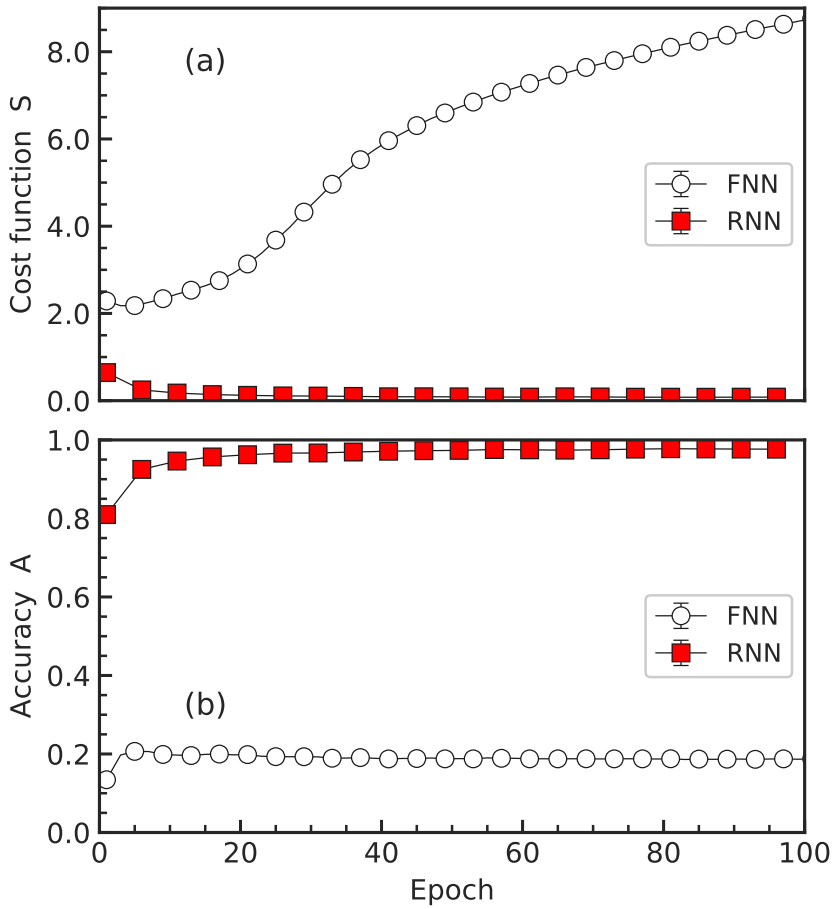


Figure B.1: Demonstration of the importance of pixel ordering for MNIST recognition when FNN is used (circles) and the automatic pixel-position and feature correlation for MNIST recognition when RNN is used (squares). When the pixel ordering (green arrows in Fig. 2.2(a)) is scrambled, an FNN cannot be trained adequately to recognize the MNIST images, shown by the rising cost and low accuracy on test data, even when the position-feature data are used together in the input to Figure 2.1(a). On the other hand, when the same data are used with an RNN (Figure 2.1(b)) the network can successfully identify the 10 different digits, as shown by the minimal cost entropy and near-perfect test set accuracy.

the correlation between features (that is, i, j and P_{ij}) and the data ordering. This ordering, of course, is destroyed in scrambling the line ordering. We can also demonstrate that the coarse-graining method mentioned in the text helps to reestablish the position-intensity correlation; the plots are omitted here.

A strong contrast is the use of RNN. The three inputs to LSTM blocks in Fig. 2.1(b) are now i, j, P_{ij}, in a random order because of the intentional data scrambling. The cost function S and test set accuracy A are shown in Fig. B.1 by squares. The RNN manages to effectively learn the correlation between i, j and P_{ij} and successfully identifies all 10 numerical digits.