README

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

Abstract

Supervised machine learning can be used to classify images with spatially correlated physical features. We demonstrate the concept by using the coordinate files generated from an off-lattice computer simulation of rodlike molecules confined in a square box as an example. Because of the geometric frustrations at high number density, the nematic director field develops an inhomogeneous pattern containing various topological defects as the main physical feature. We describe two machine learning procedures that can be used to effectively capture the correlation between the defect positions and the nematic directors around them, and hence classify the topological defects. First is a feedforward neural network, which requires the aid of pre-sorting the off-lattice simulation data in a coarse-grained fashion. Second is a recurrent neural network, which needs no such sorting and can be directly used for finding spatial correlations. The issues of when to pre-sort a simulation data file and how the network structures affect such a decision are addressed.

Acknowledgements

I would like to thank all the little people who made this thesis possible.

Dedication

This is dedicated to me.

Table of Contents

List of Tables

List of Figures

Introduction

In 1888 the Austrian botanist Friedrich Reinitzer remarked the curious melting behaviour of cholesteryl benzoate [56] (English translation: [57]). 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 [36]. Lehmann looked more closely at these samples and discovered the crystalline order present in the liquid. Publishing his findings in 1889 [40], 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/coallescence of droplets) and those of crystals (such as anisotropy of electromagnetic/optical properties and the periodic arrangement of molecules in spatial dimensions) [3].

LCs in nanoscience: controlled anisotropy of carbon nanotubes dispersed in liquid crystal The nanosciences have also picked up on the potential of LCs. Typically, it is desirable for carbon nanotubes (CNTs) to have a certain degree of alignment for their applications. By dispersion in a LC solvent, CNTs can be made to align with the LC nematic director [45, 19, 18].

LCs have also attracted interest in self-assembly nanomaterials research. Ref. [26], 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 nanopatricles into addressable arrangements can lead to materials with high processability, self-healing properties, and reversible control [26].

"A new era for liquid crystal research: applications of liquid crystals in soft matter nano-, bio-and microtechnology" [37]

Interesting emergent technology: liquid crystal elastomers offer interesting temperature dependent reactions such as dramatic volumetric changes or a transparency [67].

Liquid crystal 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 [27]. 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 [27].

In the production of spider dragline silk, an aqueous solution of the silk fibroin takes on a nematic phase as an orienting mechamisn at an intermediate stage of the process [35, 54, 33]. Cases of LC mesophases have been observed in certain DNA packings: inside the heads of bacteriophages [20], dinoflagellates [42], or plasmid DNA within bacteria [55]. Inside bacteriophages, the packing takes on a columnar shape within concentric rings [8]. 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 [55].

Nematic and chiral nematic (cholesteric) phases are the most common in such systems, but hexagonal columnar (such as in) [27]

In [41] they find nice results imaging a couple viruses in these confinements

Chapter 1

Motivation

Chapter 2

Literature Survey

2.1 Two-Dimensional Liquid Crystal in Box Confinement

- Information on Onsager can be found on pg 59 of de Gennes/Prost
- Include other studies of the box confined LC
- **Note** Page 31 of [10] mentions that the difference in the theoretic NI transition and experimental/simulated value is attributed to the higher virial terms contribute to free energy, which are absent in the Onsager model.
- See [10] page 24 for Onsager/hard rod stuff

In the following we look at the theoretic framework and computational execution for finding stable configurations of the two-dimensional liquid crystal under a square confinement based off the work of X. Yao, H. Zhang, and J. Z. Y. Chen in Ref. [73]. The configurations are produced as energy minima of an extended Onsager model.

Since its publishing in 1949, the model put forth by Lars Onsager [52] has been a mainstay in describing the interaction of anisotropic colloidal particles, hence its popularity in studying the liquid cyrstal systems of rod or disk shaped molecules. 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 (maybe) and** solving for stable configurations. Indeed this model was the launching point by which the rods-in-a-box configurations were solved for numerically in works [73, 9]. The following is a quick explanation of the methods

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

$$\beta F = \int \rho(\mathbf{r}, \mathbf{u}) \ln \left[L^2 \rho(\mathbf{r}, \mathbf{u})\right] d\mathbf{r} d\mathbf{u} + \frac{1}{2} \int \rho(\mathbf{r}, \mathbf{u}) w(\mathbf{r}, \mathbf{u}; \mathbf{r}', \mathbf{u}') \rho(\mathbf{r}', \mathbf{u}') d\mathbf{r} d\mathbf{u} d\mathbf{r}' d\mathbf{u}', \quad (2.1)$$

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,

$$w(\mathbf{r}, \mathbf{u}; \mathbf{r}'\mathbf{u}') = 1 - \exp\{-\beta \nu(\mathbf{r}, \mathbf{u}; \mathbf{r}', \mathbf{u}')\}, \quad (2.2)$$

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

$$\nu(\mathbf{r}, \mathbf{u}; \mathbf{r}', \mathbf{u}') = \begin{cases} +\infty & \text{if rods overlap} \\ 0 & \text{otherwise} \end{cases} \quad (2.3)$$

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 2.1 can be simplified for a spatially homogeneous domain (with thin rods $D \ll L$) to

$$\int w(\mathbf{r}, \mathbf{u}; \mathbf{r}', \mathbf{u}') dr' = L^2 |\mathbf{u} \times \mathbf{u}'|, \quad (2.4)$$

(in two dimensions) as visualized in Fig [FIGURE]. Now 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 [10] (pg. 25), truncating 2.1 at the second virial term is quite 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 D/L systems—but is less appropriate in two dimensions where the higher-order terms are closer to the second order magnitude. This does render this equation for the free energy *quantitatively* less accurate, but importantly the principal physics are still captured in this form. Indeed, this does underestimate the critical density of the I-N transition at $NL^2/A = 3\pi/2 \approx 4.71$, whereas the accepted value by experiment and simulation places it in the range $6 - 9$ [SOURCES].

In their work, Refs. [9, 73] determine numerical solutions from an 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 for box edge length a [10](in Ref. [73] a third is required, b/a, for rectangular width b). The ratio L/a is really a finite size scaling parameter, and the reduced density essentially determines the system topology of either isotropic or nematic, and the selection of (meta)stable states therein. Phase diagrams as functions of these three parameters are shown in Figure 2.1. 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.

2.1.1 Orientational order

Naturally, of great interest is a quantitative way to mark the whether a system is nematic of 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 an order parameter that vanishes to zero in the isotropic phase and approaches unity in the nematic phase. For example, in a ferromagnetic system the magnetization is conveniently readily an appropriate order parameter. However, conjuring one for liquid crystals is less trivial.

We proceed with our derivation in three dimensions and then consider the two dimensional case. Naively one may opt for a directional average but since rods have a center of symmetry the average of $\mathbf{u}$ will cancel out. The next invariant is then the tensor

$$Q_{\alpha\beta} = \frac{1}{N}\sum_i \left(u_\alpha^{(i)} u_\beta^{(i)} - \frac{1}{d}\delta_{\alpha\beta} \right), \tag{2.5}$$

in d dimensions, where α and β represent the Cartesian variables x, y and z, $\delta_{\alpha\beta}$ is the Kroenecker 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 $u_\alpha^{(i)} u_\beta^{(i)} = u_\beta^{(i)} u_\alpha^{(i)}$.

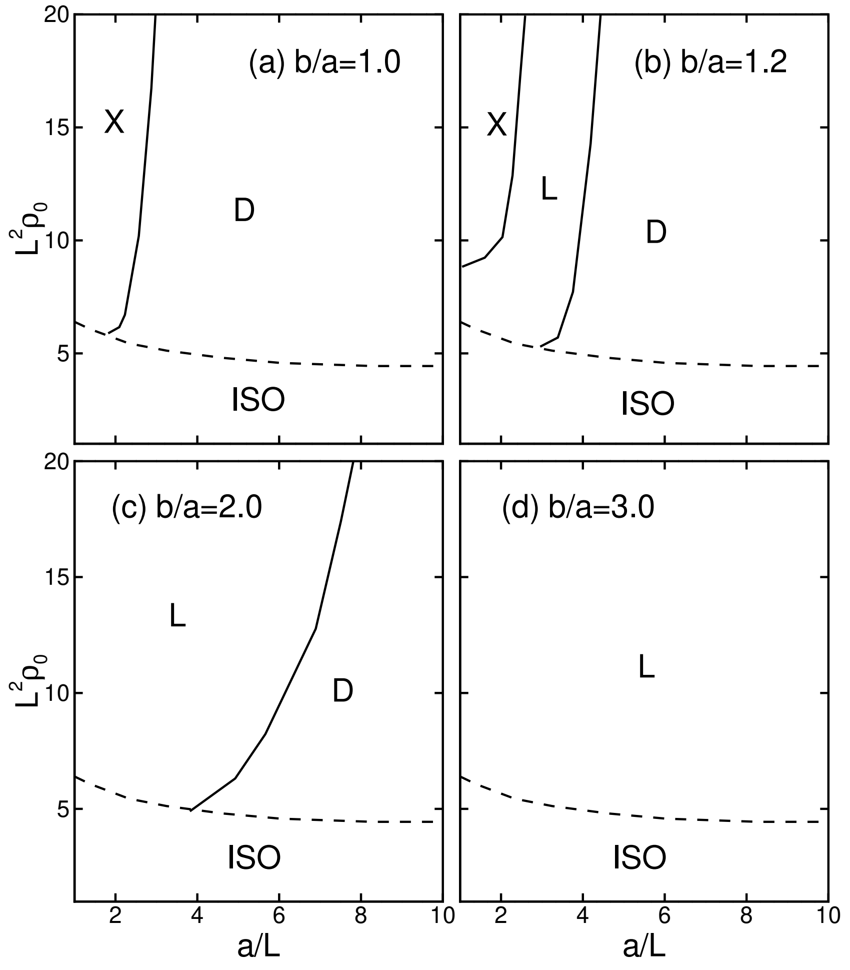


Figure 2.1: Phase diagrams of the rectangular confined rod system from work [73] [PERMISSION?]. Here, $\rho_0 = N/a^2$ and the dashed line indicates a second-order phase transition.

- 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

$$Q_{\alpha\beta} = \int_0^{2\pi} f(\theta) \left(u_\alpha u_\beta - \frac{1}{2}\delta_{\alpha\beta} \right) d\theta, \tag{2.6}$$

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

$$Q_{\alpha\beta}(x, y) = \int_0^{2\pi} f(x, y, \theta) \left(u_\alpha u_\beta - \frac{1}{2}\delta_{\alpha\beta} \right) d\theta, \tag{2.7}$$

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

$$Q = \frac{1}{2} \begin{pmatrix} S(x, y) & T(x, y) \\ T(x, y) & -S(x, y). \end{pmatrix} \tag{2.8}$$

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

$$\begin{aligned} S(x, y) = -2Q_{yy} = 2Q_{xx} &= 2\int_0^{2\pi} f(x, y, \theta) \left(\cos^2\theta - \frac{1}{2} \right) d\theta \\ &= \int_0^{2\pi} f(x, y, \theta) \cos(2\theta) d\theta \end{aligned} \tag{2.9}$$

and

$$\begin{aligned} T(x, y) = 2Q_{xy} = 2Q_{yx} &= 2\int_0^{2\pi} f(x, y, \theta) \left(\cos\theta \sin\theta \right) d\theta \\ &= \int_0^{2\pi} f(x, y, \theta) \sin(2\theta) d\theta. \end{aligned} \tag{2.10}$$

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 eigenvalue of Q pertains to the main order parameter measured from a local nematic director (Allen and Tildesly provide another reference, Zannoni 1979, on page 93 for eigenvalue and Q etc.),

$$\Lambda(x, y) \equiv \sqrt{S^2(x, y) + T^2(x, y)}. \tag{2.11}$$

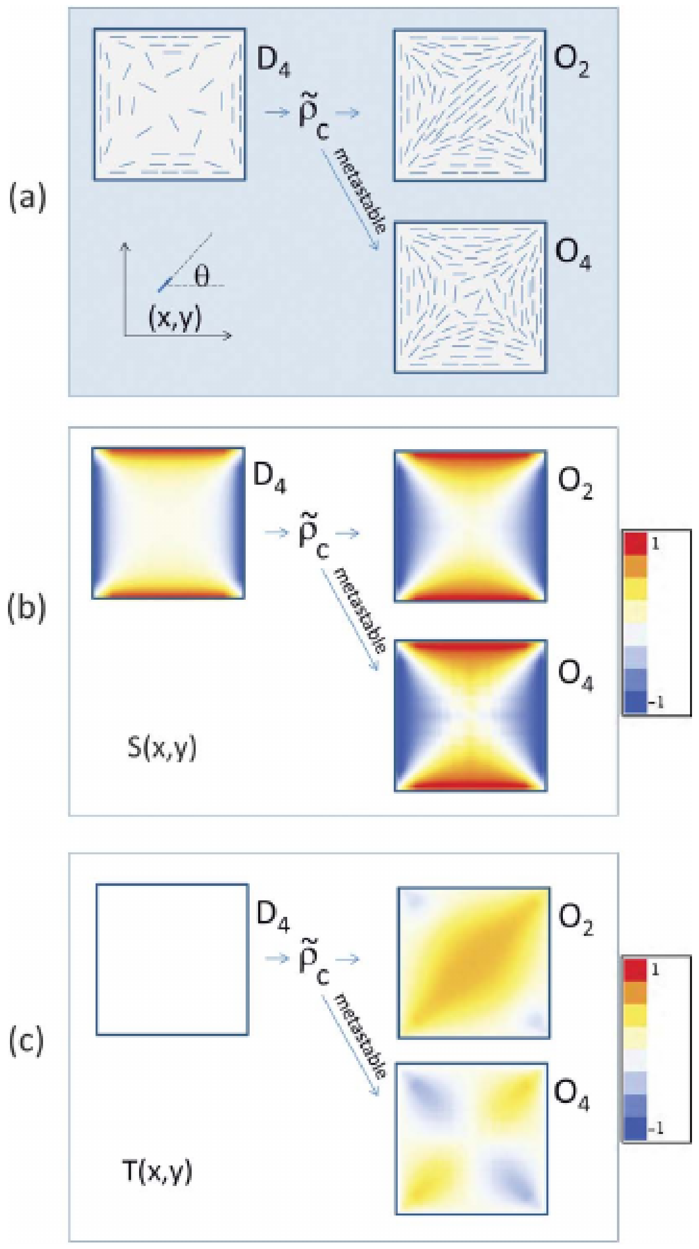


Figure 2.2: Stable D and metastable X solutions found in Ref. [9]. In their work the D state is labeled O_2 and the X state is O_4, with D_4 labeling the isotropic state. $\tilde{\rho}_c$ indicates a transition past the critical transition density. Order parameters S and T are also given. T presents itself as an effective way to distinguish these two topologies.

An intensity map of T and S on the stable diagonal D and metastable X solutions in Figure 2.2 show how these parameters can be used to characterize different topologies.

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

$$\overline{S}(x, y) = \int_{-a/2}^{a/2} \int_{-a/2}^{a/2} \frac{S(x, y)}{a^2} dxdy, \tag{2.12}$$

$$\overline{T}(x, y) = \int_{-a/2}^{a/2} \int_{-a/2}^{a/2} \frac{T(x, y)}{a^2} dxdy, \tag{2.13}$$

for box edge length a. In both cases, boundary effects will be annulled since rods order themselves parallel to wall edges [53]. A bulk order parameter

$$\overline{\Lambda} \equiv \sqrt{\overline{S}^2 + \overline{T}^2} \tag{2.14}$$

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

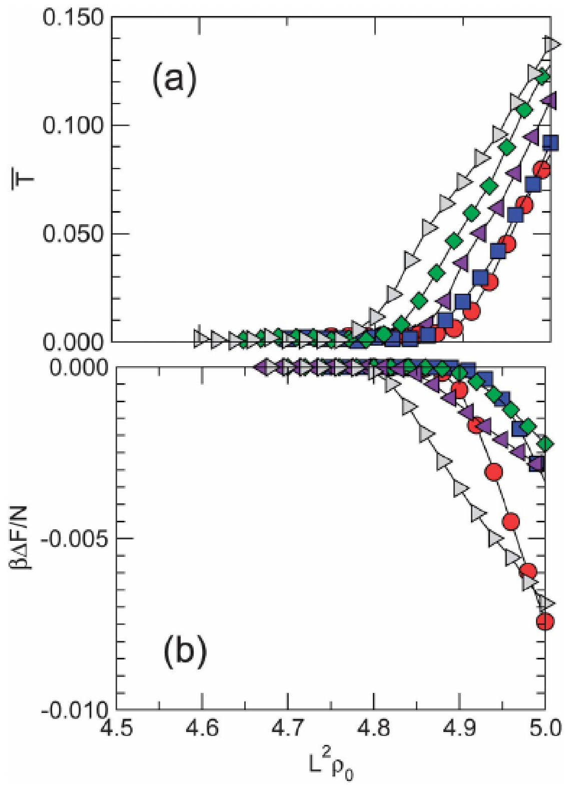


Figure 2.3: $\overline{T}$ as function...from [9]

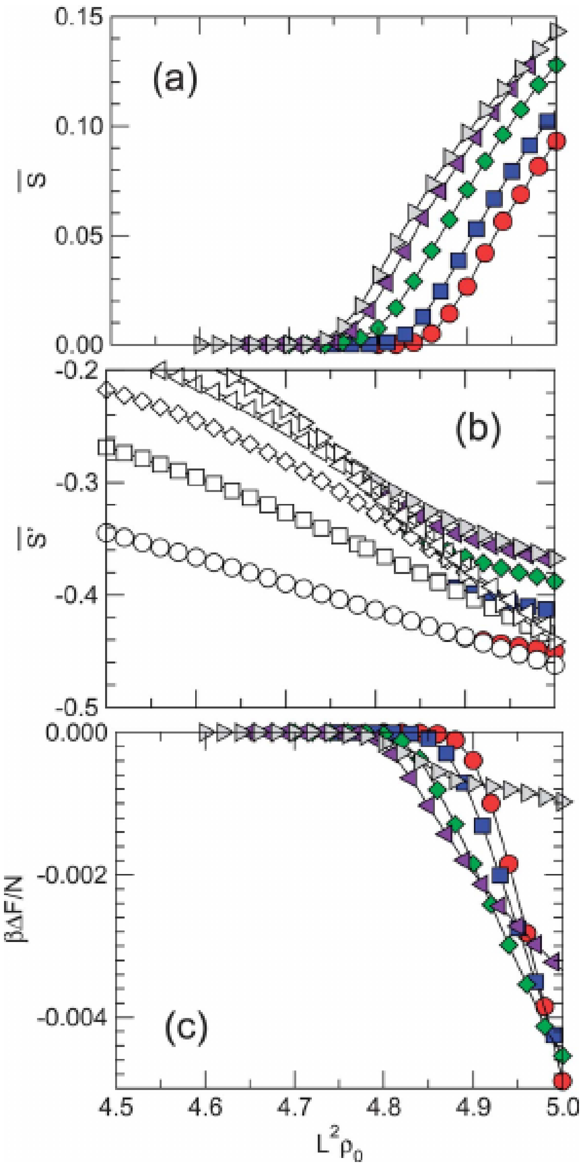


Figure 2.4: $\overline{S}$ as function...[9]

Chapter 3

Computational Methods

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 1940's 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.* [47].

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 Steps" (MCSs), where one MCS attempts N random movements of molecules, either by iterating over the molecule list or randomly selecting N molecules, though Ref. [28] 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

$$\mathbf{r}_i(t+1) = \mathbf{r}_i(t) + \delta\mathbf{r} \tag{3.1}$$

$$\theta_i(t+1) = \theta_i(t) + \delta\theta \tag{3.2}$$

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 flat

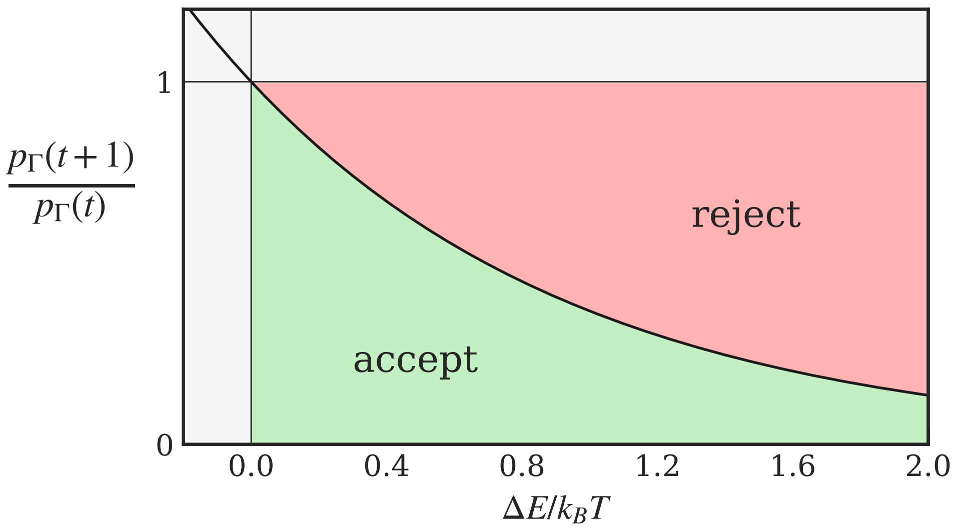


Figure 3.1: Probability ratio of Eq. 3.4 showing chances of acceptance or rejection upon generating ε for a given ΔE.

out reject this. Instead we consider the Boltzmann factor, that is the probability of the system being in state $\mathbf{\Gamma}(t+1)$:

$$p_\Gamma(t+1) = \frac{1}{Z} e^{-E(t+1)/k_B T} \tag{3.3}$$

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

$$\begin{aligned}\frac{p_\Gamma(t+1)}{p_\Gamma(t)} &= \frac{e^{-E(t+1)/k_B T}}{e^{-E(t)/k_B T}} \\ &= e^{-(E(t+1)-E(t))/k_B T} = e^{-\Delta E/k_B T}.\end{aligned} \tag{3.4}$$

We then base the decision of acceptance/rejection off this relative probability. To do so, a random number ε is generated in the range $0 \leq \varepsilon \leq 1$, 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 3.1 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 $\Delta E/k_BT$, 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 with the proposed move. As described in Ref. [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,

$$\Delta E = \sum_j \epsilon_{ij}(t+1) - \sum_j \epsilon_{ij}(t). \tag{3.5}$$

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. 3.2 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 Ref. [72] 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.

Chapter 4

Machine Learning with Neural Networks

- Let's think about the universal approximation theorem, proved by Hornik etal [30]. A multilayer perceptron is a universal approximator of a function. Question though: can our problem be represented as a function? Let's think about a simple example of the input neurons being for x and y dimensions, and then the output being a singular neuron for $\cos(x) + \sin(y)$. That seems pretty sensible. I'm mostly concerned with how our input function is constructed so strangely. For example, if it's random whether x or y will be input to the first neuron, then this model breaks down. I do think the nature of the unreliability of this input vector means the approximation theorem doesn't apply.

4.1 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 [30], 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 due to insufficient number of hidden neurons *or lack of a deterministic relationship between input and output.*

We emphasize the last point because it's believed this is the root cause of the failure of the FNN in learning the XTUD topologies. There is not 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.

4.2 PCA detection of defects

4.2.1 Principal Component Analysis

In this section we explore Principal Component Analysis (PCA) as a technique for detecting defect regions. PCA is a useful statistical method for transforming a dataset into a form that better distinguishes features in the data, and it may also be used as a dimension reduction tool on a dataset. It also falls into the category of unsupervised learning since it reveals features in the data without any label-directed goals. PCA is well-known, and we'll follow the explanation offered by Johnathan Shlens [60].

We let the $m \times n$ matrix $\mathbf{X}$ represent our dataset of m observations on 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. The usefulness of this technique does rely on the assumption that a high variance along an axis indicates an important feature of the data, but this generally the case.

To begin, we note the covariance matrix of the $\mathbf{X}$ variables (adjusted to zero-mean to simplify calculations)

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

The diagonal elements of $\mathbf{C_X}$ are the variances of individual variables, and the off-diagonal elements are the covariances of different variables. 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 $\mathbf{W}$:

$$\begin{aligned}\mathbf{C_Y} &= \frac{1}{m}\mathbf{Y^T Y} \\ &= \frac{1}{m}(\mathbf{W^T X^T})(\mathbf{XW}) \\ &= \mathbf{W^T}\left(\frac{1}{m}\mathbf{X^T X}\right)\mathbf{W} \\ &= \mathbf{W^T C_X W}. \end{aligned} \tag{4.1}$$

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

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

where we also use the orthonormality of $\mathbf{W}$. The eigenvalues, λ, are also called the "explained variance" in this context. Their normalized values, $\lambda_i^* = \lambda_i / \sum \lambda_j$, or explained variance ratios, are used as a metric as to the importance of each axis, values close to 1 indicate more significance and those closer to 0 being less significant. Generally, one expects only some 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. 4.1, the ith diagonalized variance of $\mathbf{C_Y}$ is the variance of $\mathbf{X}$ along the ith principal component.

- $\mathbf{W}$ indeed diagonalizes $\mathbf{C_Y}$.

4.2.2 PCA learning order parameters for phase transitions

Recently, PCA has seen some fascinating use on condensed matter systems. In 2016, L. Wang (Ref. [69]), 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 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. Championing 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, authors R.B. Jadrich *et al.* (Ref. [31]) 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 $\mathbf{X}$ 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.

4.2.3 PCA learning disclinations

We now put forth the question: Can PCA use nearest neighbour information of our LC system to identify nematic vs. disclination/defect states of the variety shown in Fig, or beyond that which *type* of defect? The following discusses this question, with results on the endeavor presented in Sec. 5.

The results of Ref. [31] are certainly encouraging, though their objective was somewhat different. PCA detected a long-range angular correlation appeared signaling the nematic phase, and for this they used every 10th nearest neighbour. Here we are concerned with detailed local features. In theory, the nematic vs. non-nematic states should be differentiable. For a given probe i (either a rod or a point in space), we may construct a feature vector $f_i = [x_1^{(i)}, x_2^{(i)}, \cdots, x_{nn}^{(i)}]$ of some physical quantity 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 would to take neighbouring rod orientations with respect to a local nematic director α isntead.

A random ordering of neighbours may work in detecting nematic vs. defect states, but would 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 clockwise procession around the defect center should be a more effective approach.

We adopt an approach where we first identify a nematic director α from neighbours. We use a discretized version of the method outlined in Section 2.1.1 that produces the Q-matrix in equation 2.8:

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

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

$$S_{nn} = \frac{1}{N_{nn}}\sum_i \cos(2\theta_i)$$

$$T_{nn} = \frac{1}{N_{nn}}\sum_i \sin(2\theta_i)$$

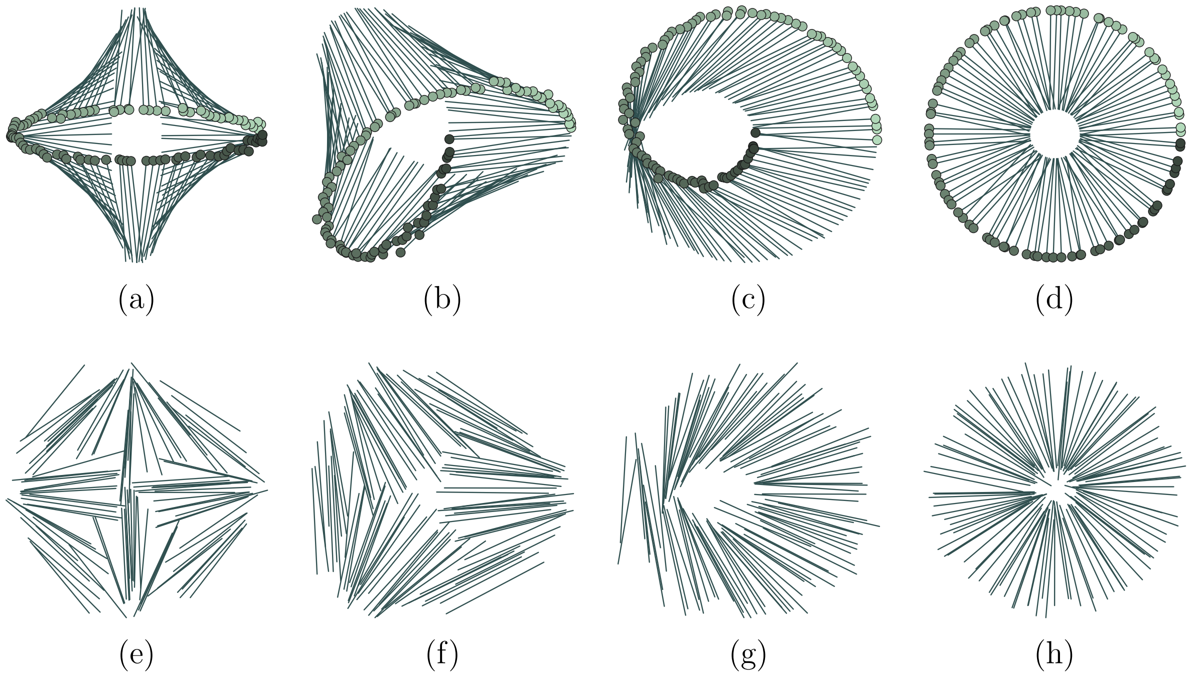

(a) (b) (c) (d)
(e) (f) (g) (h)

Figure 4.1: (a)-(d) Examples of initialized disclinations with winding numbers -1, $-1/2$, $+1/2$, and $+1$ respectively, as a visual aid for understanding the procession around the defect center. In proceeding 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 uncrossed samples made up the defect PCA dataset.

Chapter 5

Results

5.1 Introduction

The use of neural networks as a tool in condensed matter research has seen a growth in popularity. 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 [65, 7, 70, 71, 64, 51, 4].

One of the simplest neural networks is the feedforward neural network (FNN). A powerful tool, FNNs are able to accomplish a variety of image recognition tasks [59], a frequent benchmark test being the MNIST hand-written digit data set [5]. Figure 5.1(a) shows one such digit from the MNIST set. Increasing the size and depth of a network boosts its ability to learn more complex patterns and features contained in images and then recognize these learned patterns in a new image. An extension of the FNN, the convoluted neural network (CNN), has the network structure organized in such a way that local features of a pattern are dissected [39]. CNNs have thus been a natural choice for condensed matter research, in particular problems on lattices such as the Ising model [7], the XY model [4], correlated fermions [6, 12], and other quantum systems.

In condensed matter physics we often deal with ordered states, where certain physical features display spatial correlations in long range. Two typical examples are shown in Fig.

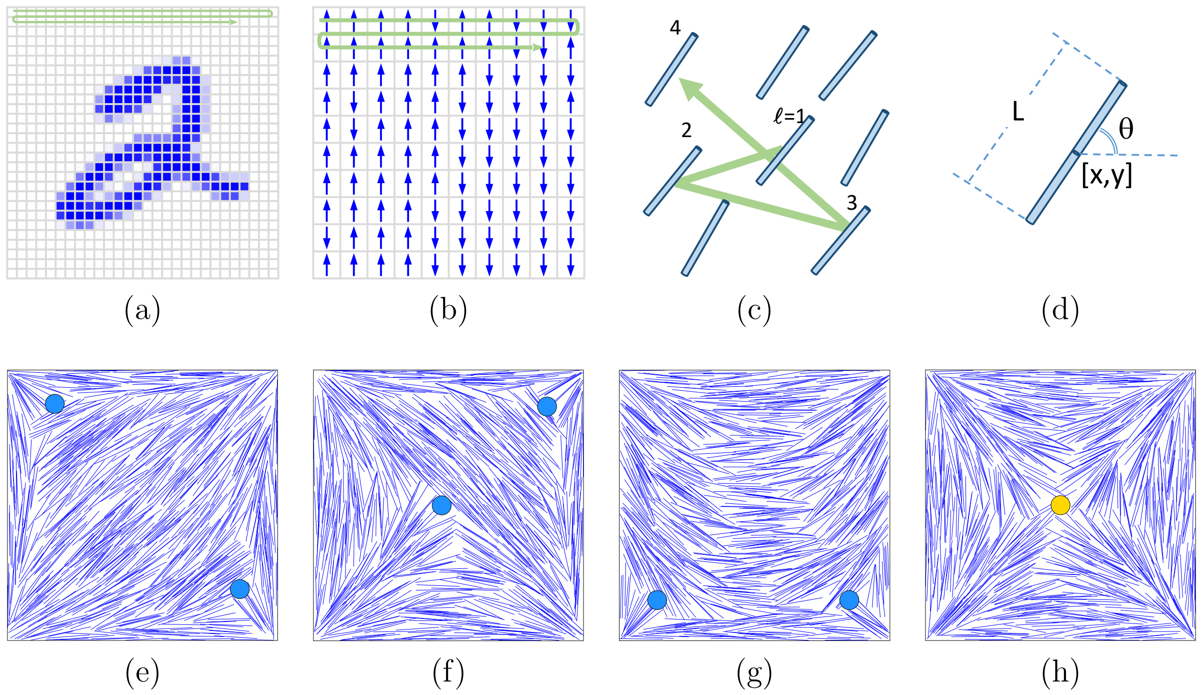


Figure 5.1: (a)-(d) Illustration of a two-dimensional hand-written image, Ising model, nematic state, and coordinates for a rod-like molecule, as well as (e)-(h) example configurations of different defect states generated from Monte Carlo simulations for a confined rod-like nematic fluid. The parameters 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.

5.1. In the ferromagnetic state, within an ordered domain spins align in one direction, as illustrated in Fig 5.1(b). In the nematic liquid crystal state, within an ordered domain molecular directions are all aligned towards a common angle [Fig 5.1(c)]. The images produced in these examples come from Monte Carlo simulations (see Appendix A for the current liquid-crystal system). Here, an "image" used for network learning 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).

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. Developing a characterization procedure to categorize these defects is a challenging task. A neural network (NN), then, becomes an ideal tool to identify these topological defects. In studying the Kosterlitz-Thouless transition of the XY model [4], a multilayered network was trained on raw orientational configurations to learn the vortex unbinding that marks the transition. Accomplishing this requires the NN to understand topological features not too unlike the liquid crystal topologies in Fig. 5.1(e)-(h). It would seem then that feeding the liquid crystal configurational data in a similar fashion should enable the identification of defects in ordered states.

However, we show here that such network structures are not readily appropriate for studying defect types of *off-lattice* problems, such as the liquid-crystal defects shown in Fig. 5.1((e)-(h)). 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 x- and y- location coordinates and angular orientation [see Fig 5.1(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 Fig 5.1(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 Fig. 5.1(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. In this paper we discuss different ways of incorporating existing NNs for this purpose. As it turns out, an FNN (and conceivably a 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 Sect. 5.3.

Can we re-connect a relationship between l and x_l, y_l that can be easily used by an FNN for an off-lattice dataset? In Sect. 5.3 we develop and discuss a coarse-graining method which cuts the original simulation box into $m \times m$ equally sized cells where $m = 1, 2, 4, 8\ldots$ 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.

However, we present another 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 [49], or predicting upcoming words in a sentence based on previous words [48, 13]). In Sect. 5.4, we propose a scheme to feed x, y, and θ data 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.

We selected the topological defects appearing in the system of N rod-like molecules confined by a square boundary in two dimensions (2D) as a vehicle to deliver the concepts in this paper. This system has been the focus of recent theoretical and experimental studies due to its practical relevance [23, 61, **?**, 15] and interesting theoretical aspects [66, 44, **?**, 25, **?**, **?**, 73]. For example, the possible defect states were recently studied in-depth in terms of the continuum Landau-de Gennes model [**?**] and the Onsager model [73], producing a comparable ensemble of stable and metastable states as the result of minimizing the free energies in these different models. Our aim here is not a detailed study of this particular liquid crystal system, to which we direct the readers to the above references for more details on their significance and origin. Rather, we selected four major defect states, D, T, U, and X, shown in Fig. 5.1(e)-(h), for use in our machine learning study on the molecular data files. For this purpose, we developed a Monte Carlo algorithm to produce the necessary data. The computer simulation procedure, as well as the system's isotropic-nematic transition properties, are explained in Appendices 5.7 and 5.8.

The location of the nematic defect points are indicated in Fig. 5.1(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.

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 5.10, we re-enforce some of our concepts mentioned above, by treating the digit recognition problem as a defect state identification problem.

This work is motivated by the question of whether machine learning can be used to identify topological defects, which in this case are those of confined two-dimensional liquid crystals. We clearly make recommendations on data handling and the choice of neural network, suitable for defect identification; these can form useful steps for more general problems in studying other condensed matter systems, in particular, data generated from off-lattice models.

5.2 Preparation of the training and testing data

The physical system studied here is the defect states generated from the MC simulation of a two-dimensional off-lattice liquid-crystal model. 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. Monte Carlo details can be found in Appendix 5.7.

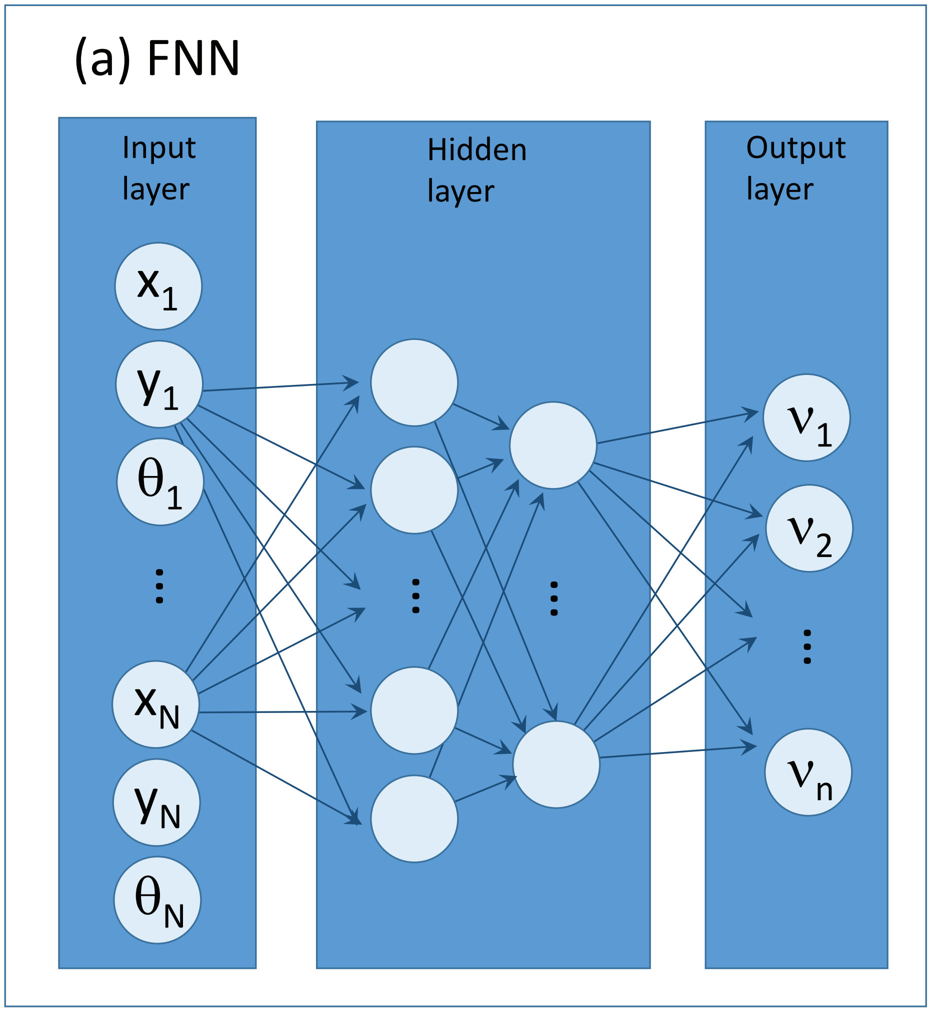
One can show that the parameter that drives the phase transition is the reduced density,

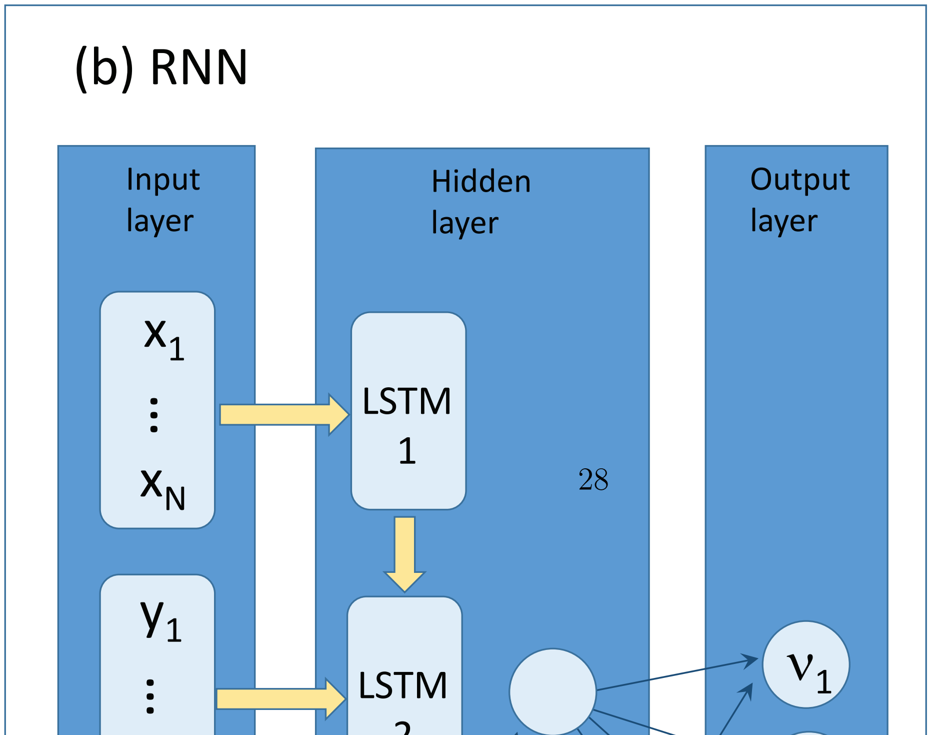
$$\rho \equiv \frac{NL^2}{a^2}. \tag{5.1}$$

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). The critical density $\rho^* \simeq 6.71$ can be estimated from a typical machine learning application, described in Appendix 5.8.

Our main concern here is identifying the defect states, not the isotropic-to-nematic phase transition itself. 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 Refs. [23, 61, **?**, 15, 66, 44, **?**, 25]. 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 Appendix 5.7]. Of these, 400 snapshots were reserved for the testing dataset.

For each defect type in Figs. 5.1(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


(a) FNN
Input layer
Hidden layer
Output layer
x_1
y_1
θ_1
x_N
y_N
θ_N
ν_1
ν_2
ν_n


(b) RNN
Input layer
Hidden layer
Output layer
x_1
x_N
y_1
LSTM 1
LSTM 2
ν_1

from MC simulations, they already contain all different orientations of the defect pattern naturally. To ensure that all four orientations of the square boundary are indeed equally treated, we further rotate these 4000 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 Fig. 5.1(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.

5.3 Application of FNN

First we review a few basic neural network concepts. 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). In our case we look to classify system states through an FNN, sketched in Fig. 5.2(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 [58]. 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 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.

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 [14, 1], and an early stopping technique determined sufficient training time [50]. Dropout was used at a 50% drop rate to reduce overfitting the training [63]. The Adam algorithm was used for optimization [34], and Softmax was applied to the output neurons to normalize the output set [see Appendix 5.9]. 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 Appendix 5.9.

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 [see Appendix 5.9]. This approach has seen success in learning phase transitions of Ising systems [7], polymer systems [70], and the XY model to a degree [4]. However, this method showed a pronounced failure in the current application. As we show in Fig. 5.3(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 [see Appendix 5.7], 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 [Fig. 5.1(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 (e.g. θ) correlation is destroyed. This unseen property, and its essential importance can also be demonstrated via the digit recognition problem in Appendix 5.10.

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 liquid crystal defects shown in Fig. 5.1.

The confinement box (Fig. 5.1) is divided into $m \times m$ cells, where m is an integer. The cells are labeled $M = 1, 2, ...m \times m$ horizontally, row by row. 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. Even if a rod extends into multiple cells, the fixed input vector allows it only to appear once. The center of mass of the rod makes 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 works well with an FNN. Figures 5.3(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. 5.1 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 single 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.

5.4 Application of RNN

A typical structure of the recurrent neutral network (RNN) is represented in Fig. 5.2(b). The crux of this RNN is the long short-term memory (LSTM) cell module [29]. 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

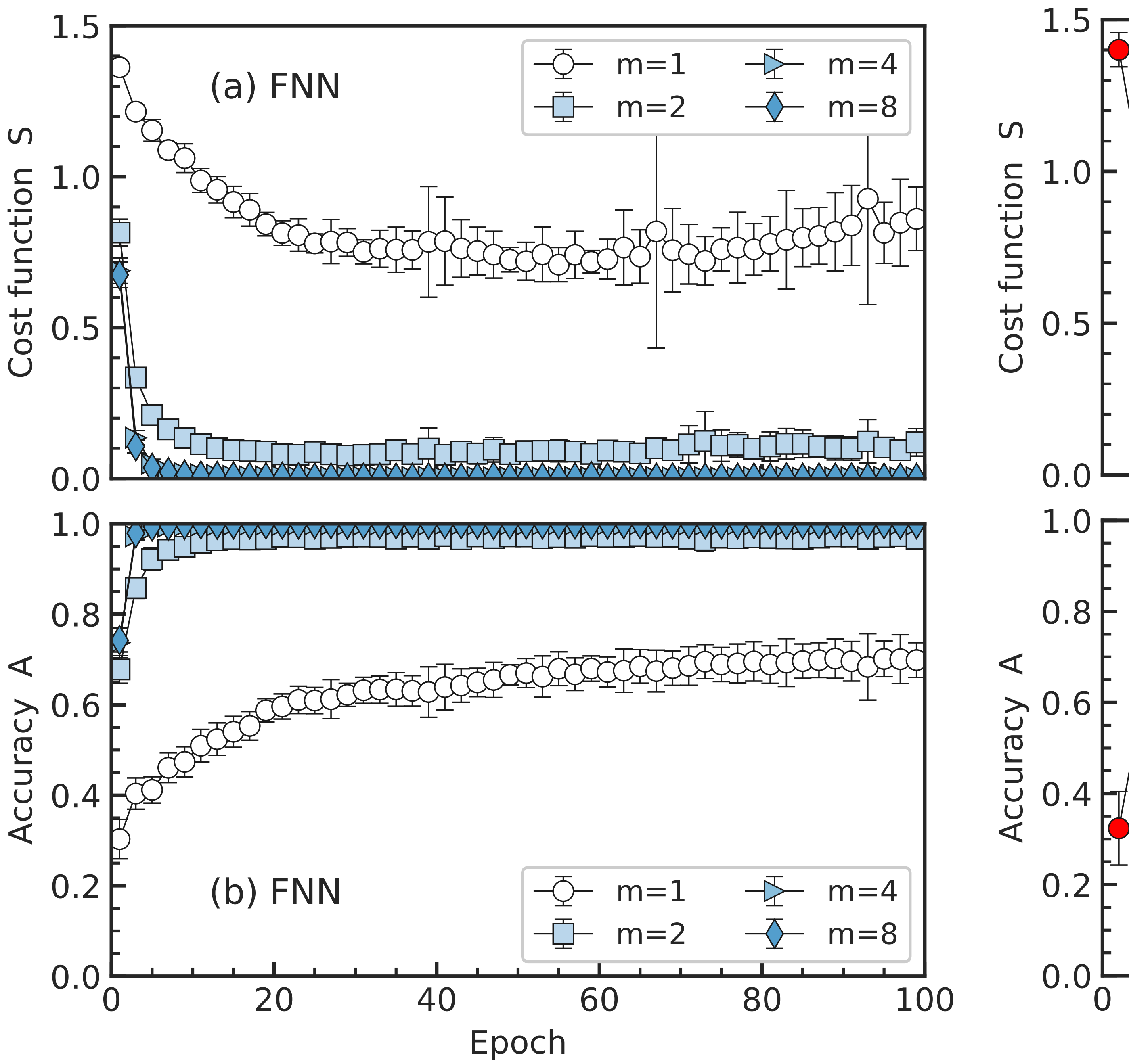


Figure 5.3: Cost function S and accuracy A, defined on the training and test datasets respectively, monitored on an FNN [(a) and (b)] and RNN [(c) and (d)] as functions of epoch step. Symbols in the plots represent the averaged S and A produced from 20 repeated training runs, from which errorbars are 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. 5.1(e)-(h) is divided into $m \times m$ cells [see Section 5.3]. 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., $m = 1$).

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

RNNs initially gained popularity for time series applications. An LSTM cell can take a complete data file and at each iteration of input, consider the new raw data 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 libraries efficiently.

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. 5.2(b) to find the correlation between the spatial coordinates x, y and the orientational coordinate θ. The raw Monte Carlo data has a line-by-line format $[l, x_l, y_l, \theta_l]$, where $l = 1, ..., N$. 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 [63, 50, 34].

The performance of this RNN is exceptionally good in identifying the defect states. Figures 5.3(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. 5.3(a) and (b), we used the raw, unsorted data as the input on our RNN experiment.

Another use of keen interest would be detecting which types of nematic defects (usually represented by a winding number) are present and their locations in a larger nematic image. The current approach of using the entire configuration data in RNN is not ready for this task but we could certainly extend our method for this purpose. We can divide the large nematic image into appropriately small cells, and use the data in each cell as the input for RNN. Using supervised learning in much the same way as this study, the RNN could be trained to recognize a defect cell versus a normal cell; then, in an application stage, the trained RNN can be used to sweep through the entire image and identify the type and location of a defect cell. Future study in this direction is needed.

Without other analysis tools it is difficult to say which *exact* topological features the RNN is learning to distinguish the XTUD image set. One could argue that the location of a defect in a larger system is a point-like object, but the XTUD topologies here, complete in their square confinement, are compositions of bulky nematic patterns interrupted at defect points [Fig. 5.1(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.

5.5 Summary

In this paper, we examined the ability of an FNN and RNN in learning and identifying the topological-defect configurations produced from Monte Carlo simulations of a liquid-crystal model. The predominating physical characteristics of the defect pattern is the positioning of defects and its coupling with the nematic pattern around the defects. Our main conclusions include: for effective learning with an FNN the simulation data must be presorted to restore the data ordering of spatial information, and with an RNN no such presorting is needed.

Exploiting the RNN capability to correlate spatial features (such as the topological defects) in their sequential inputs is a novel use here. 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 spin labels implicitly representing spatial locations.

5.6 Acknowledgement

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

5.7 Monte Carlo Simulation

In order to generate the data pool to train our neural networks, we adopt the Monte Carlo simulation method that generates configurations of our molecular-level model. The simulated system contains 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 [see Fig. 5.1(d)]. 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 L/a.

A typical MC attempt of a randomly selected molecule consists of a translational move by changing $[x, y]$ and a rotational move by changing θ about the rod center of mass. These moves are realized by adding a coordinate shift within the range $[-\Delta, \Delta]$ (for the former) or an angular shift $[-\delta, \delta]$ (for the latter). Any moves that violate the excluded-area constraint and the boundary conditions are rejected. The magnitudes of Δ and δ are determined by trial and error to maintain acceptance rates close to 50% [38], independently for the translational and rotational moves, to allow sufficient system evolution. An MC step (MCS) pertains to making N MC attempts (one for each rod on average) described here. To speed up the simulation, the cell-index technique [**?**] was incorporated in the algorithm.

All the 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 $3\pi/2$.

5.8 The isotropic-nematic phase transition and FNN

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 Fig. 5.1, 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? As discussed in Refs. [7, 70] 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 5.3 but with no pre-sorting and two states instead of four. Additionally for this study, configurations produced at different

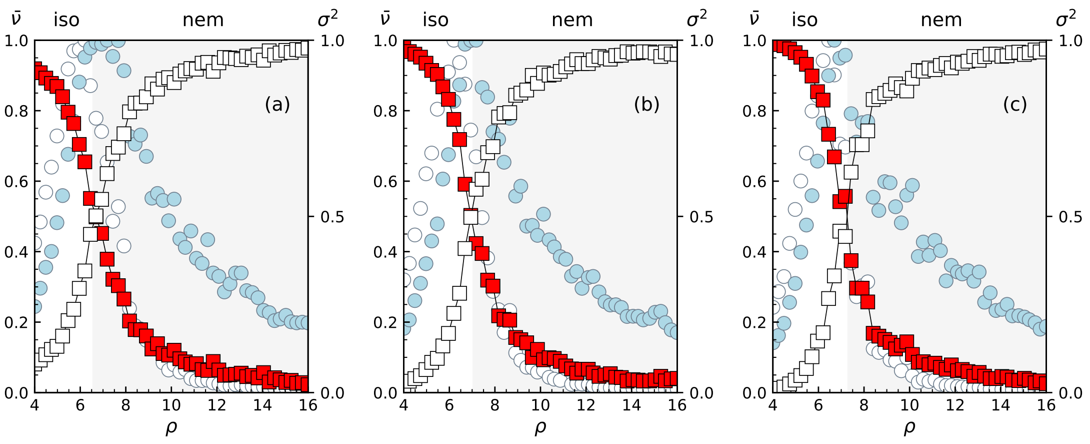


Figure 5.4: Identification of the I-N transition point. An FNN is trained by using training files at low and high densities ($\rho = 4$ and 16 for the I and N states, respectively). Then, the trained FNN is used to calculate the average outputs $\bar{\nu_1}$ and $\bar{\nu_2}$ at each given ρ. These characteristic 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.

values of ρ are required. At a fixed N, a series of configurations are obtained with varying ρ by adjusting a/L [see Eq. (5.1)]; this is achieved by a Monte Carlo procedure described in Appendix 5.7.

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 [14, 50, 34, 63]. The FNN easily learned the difference between the I and 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 Fig. 5.4. 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 [70, 7, 68], 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 [21], 6.5 [16], and $6 - 9$ [24]; the mean-field theory calculation of $\rho^* = 3\pi/2L^2 \approx 4.71$ based on the Onsager theory is a known under-estimate [32, 11].

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

$$\Lambda \equiv \sqrt{C^2 + S^2} \tag{5.2}$$

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 Fig. 5.4. 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 [53, **?**, **?**]. The definition in (5.2) 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 Fig. 5.4 show the same transition density within the numerical error.

5.9 Supervised training and accuracy testing

This paper suggests two types of networks, FNN and RNN, for identifying defect states in liquid-crystal systems. As shown in Fig. 5.2, 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.

To train the network to recognize the isotropic (I) and nematic (N) states, only two output nodes ($n = 2$) are used [Appendix 5.8]. When training to learn the DTUX states, four output nodes ($n = 4$) are needed (Sects. 5.3 and 5.4).

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 for the I-N states, for example, α can be I or N, and

$$(\nu_1^I, \nu_2^I) = (1, 0), \tag{5.3}$$

$$(\nu_1^N, \nu_2^N) = (0, 1). \tag{5.4}$$

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

$$(\nu_1^D, \nu_2^D, \nu_3^D, \nu_4^D) = (1, 0, 0, 0), \tag{5.5}$$

$$(\nu_1^T, \nu_2^T, \nu_3^T, \nu_4^T) = (0, 1, 0, 0), \tag{5.6}$$

$$(\nu_1^U, \nu_2^U, \nu_3^U, \nu_4^U) = (0, 0, 1, 0), \tag{5.7}$$

$$(\nu_1^X, \nu_2^X, \nu_3^X, \nu_4^X) = (0, 0, 0, 1). \tag{5.8}$$

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,

$$S = -\frac{1}{K}\sum_{j=1}^{K}\sum_{i=1}^{n} \nu_i^{\alpha_j} \log \nu_i^{(j)}, \tag{5.9}$$

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 of the network 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 maximum output node of $(\nu_1^{\alpha_j}, \nu_2^{\alpha_j}, ..., \nu_n^{\alpha_j})$. 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

$$A = \frac{1}{k}\sum_{j=1}^{k} A_j. \tag{5.10}$$

5.10 FNN and RNN applications of recognizing the MNIST data

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. 5.1(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% [59]. The output layer shown in Fig. 5.2 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 Figs. 5.5(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 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. 5.2(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. 5.5 by squares. The RNN manages to effectively learn the correlation between i, j and P_{ij} and successfully identifies all 10 numerical digits.

5.11 PCA

Figure 5.7 shows explained variance ratios and the first five componentes of the normalized comprehensive dataset. By inspecting these eigen waveforms we can discern what PCA is looking for. w_1 responds to the nematic-isotropic character of neighbours. Since it is nearly a constant form, isotropic feature vectors $\mathbf{f_I}$ will give $\mathbf{f_I w_1} \approx \sum -0.5 w_{1i} \approx 5$, and nematic $\mathbf{f_N}$ will find $\mathbf{f_N w_1} \approx \sum 0.6 w_{1i} \approx -5$, as is observed in the distribution of Figure 5.6. MENTION THE VALUES OF FIGURE 4.4. The waveforms above w_1 have an interesting pattern. They seem to represent higher frequencies of rotation winding around the defect point, and also may be grouped in pairs. Consider the feature vector waveforms in Figures 5.9 & 5.10. The 1/2 and 1 winding number vectors approximate

$$f_{1/2}(n_i) = \sin(2\pi n_i / N_{nn} + \phi) \tag{5.11}$$

$$f_1(n_i) = \sin(\pi n_i / N_{nn} + \phi), \tag{5.12}$$

for neighbour n_i and random phase ϕ since in practice

DISCUSS Go into the topic surrounding the difficulty of avoiding false rotations. This seems to be the main problem with using just th_jalpha information (polar2) as opposed to our best mode of cos(2*th_jalpha) (polar). Heat map images show that polar2 can be very sketchy (along w0 at least). File `edge_3_11.47` nblskip 1, px,py = 15,15 and px,py = 15,16 provide two good examples of how different things can be.

Let's map out this discussion on methods. Three modes to consider: random, winding alignment "WA" (polar), winding difference "WD" (polar2). I think we will need a suite of plots here:

- scatter plots of w_1/w_2 probably don't need to go to higher comparisons, except it may be interesting on the WA plot
- The eigenvalue and vector plots
- Probe plots
  - smalldefects with all three modes. These could maybe be combined across modes. Otherwise it may feel like relegating to appendix, however random mode colors the rods differently.
  - iso-nem for WA at least. Does WD capture this? Looks like no. Especially near the phase transition where some skew rods will really throw things off and produce false rotations. It performs fine in nematic regions. I just looked at

the X-state with WD and 0,1,2, dimensions and it hardly responds to different regions, and also flipflops near the center. false rotations causing issues no doubt.

- Many heat map plots
  - possibly some of the smalldefects in random mode
  - run edge_3_11.47 showing how WD and WA behave differently
  - some XTUD images. We should do higher order colouring here. For example, the 3rd vs 2nd dimension colouring should be interesting, though 2nd will colour over 3rd. See note above wrt WD and X-state
  - Heat maps of smalldefects for WA and WD probably overkill
-

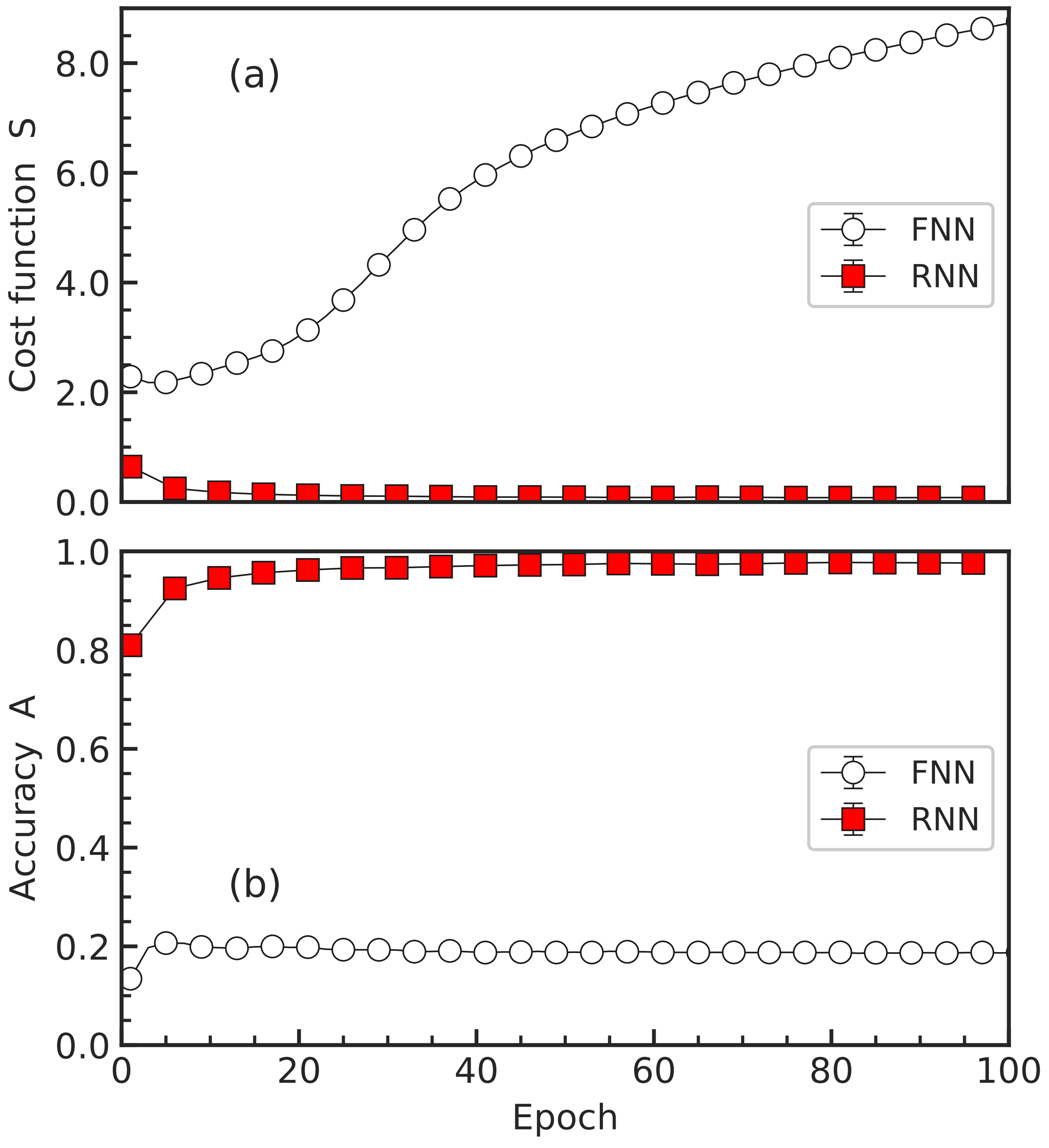


Figure 5.5: Demonstration of the importance of pixel ordering for MNIST recognition when FNN is used (circles) and the automatic pixel-position and feature correlation for MNIST recognition when RNN is used (squares). When the pixel ordering [green arrows in Fig. 5.1(a)] is scrambled, an FNN cannot be 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 Fig. 5.2(a). On the other hand, when the same data are used with an RNN [Fig. 5.2(b)] the network can successfully identify the 10 different digits, as shown by the minimal cost entropy and near-perfect test set accuracy.

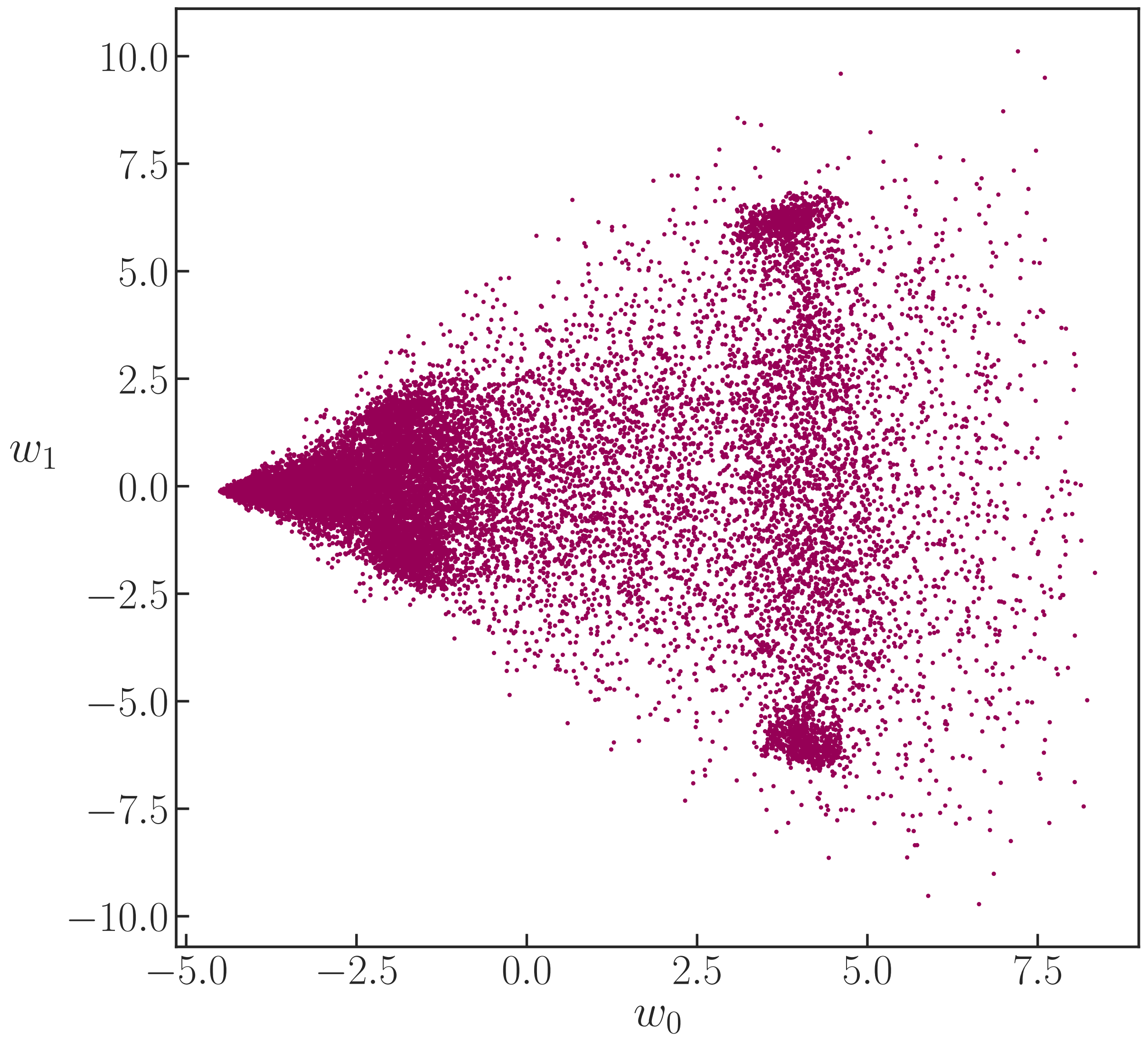


Figure 5.6: (a) Observations from the comprehensive dataset along the first two principal components. The concentrations around $w_0 = 4$ correspond to corner samples.

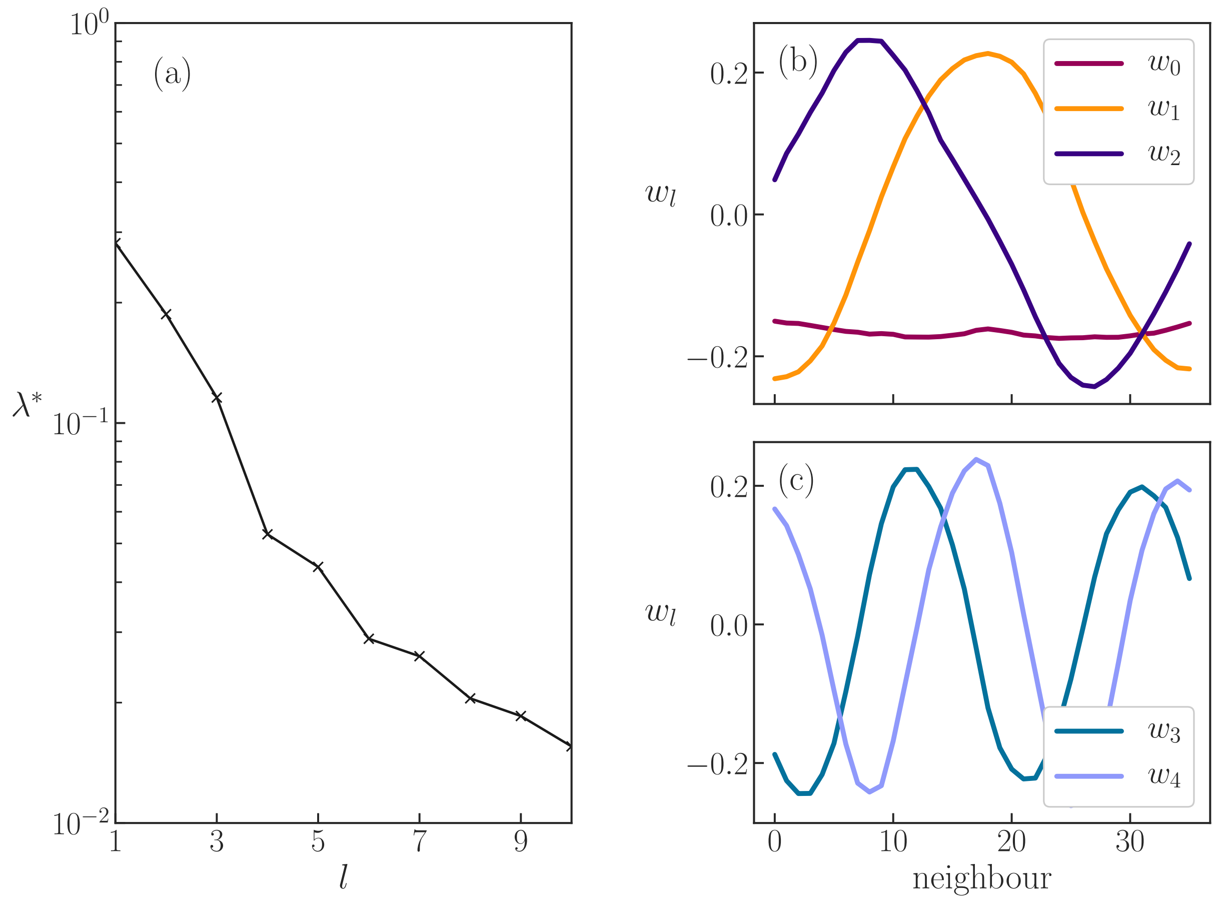


Figure 5.7: (a) Explained variance ratios for the normalized comprehensive dataset and (b)-(c) the first five components. See text for discussion.

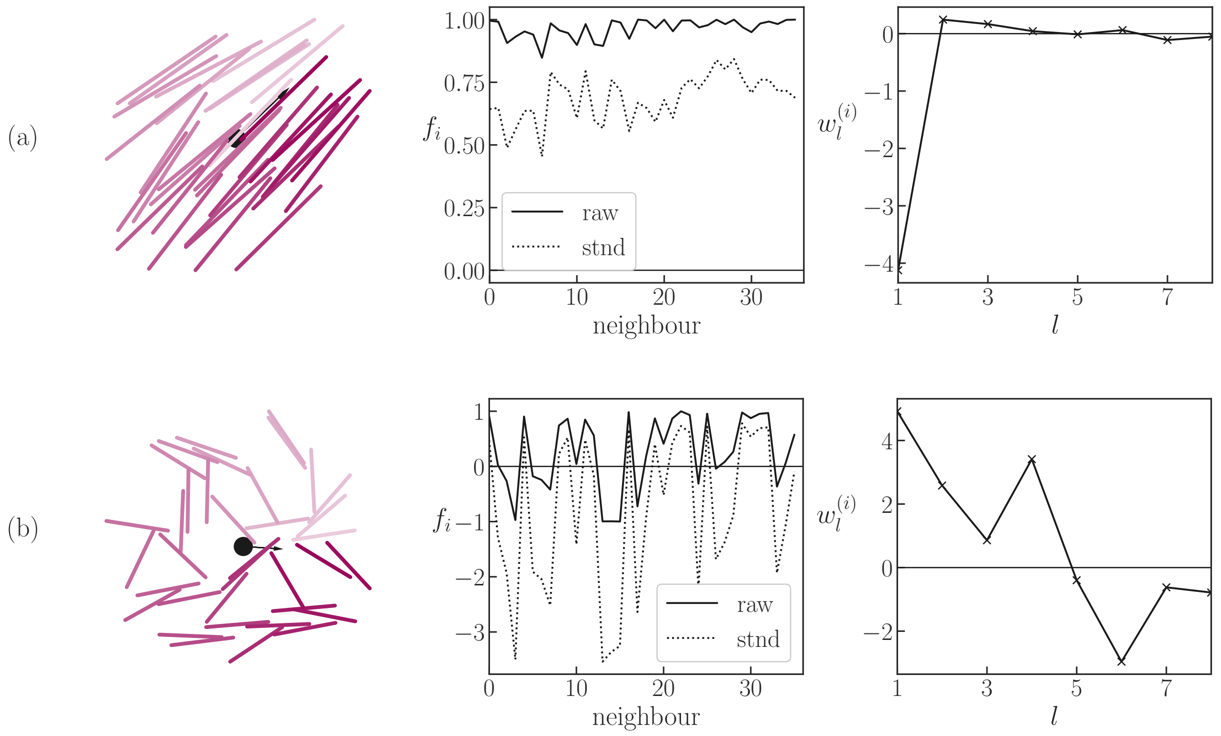


Figure 5.8: Feature vectors (middle column) and their modified PCA components (right column) for isotropic and nematic neighbours. We can see that w_1 captures isotropic and nematic information by the opposing strong responses.

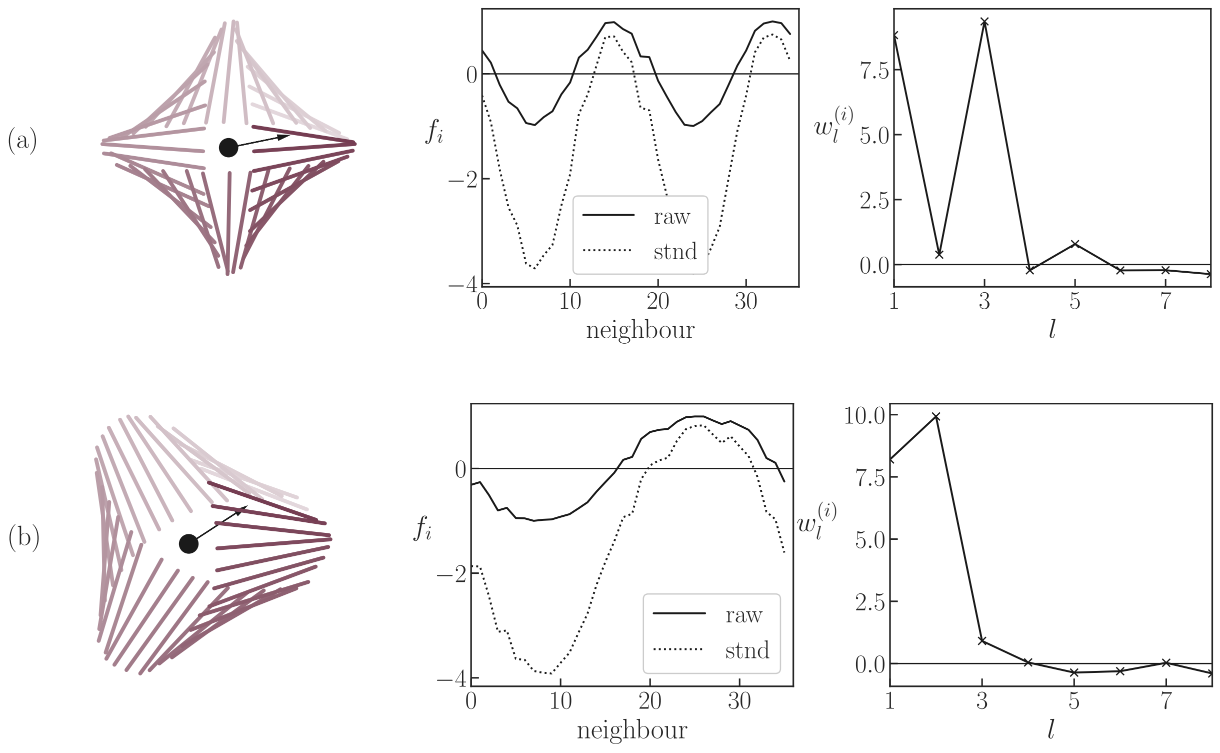


Figure 5.9: Feature vectors (middle column) and their modified PCA components (right column) for -1 (a) and $-1/2$ (b) winding numbers. The -1 defect shows a high response in the 3rd component, with the $-1/2$ defect giving a high response in the 2nd component. Both are still highly nematic and give a strong 1st component response.

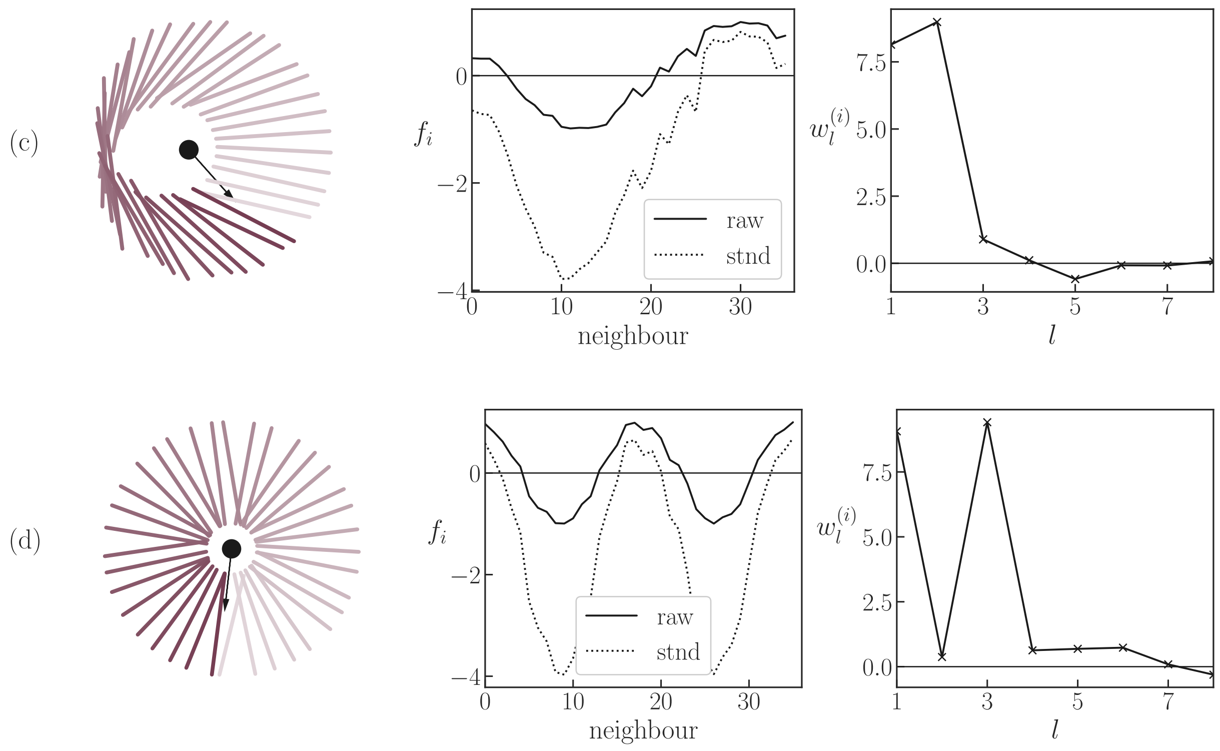
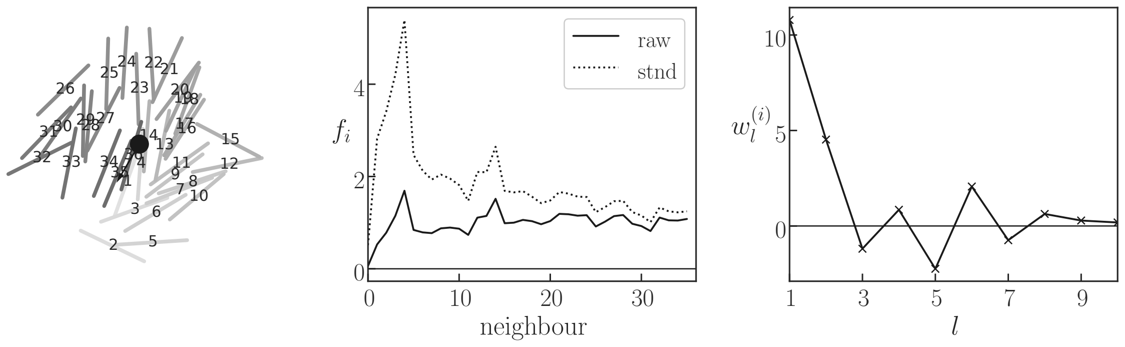


Figure 5.10: Feature vectors (middle column) and their modified PCA components (right column) for +1/2 (a) and +1 (b) winding numbers. The +1/2 defect shows a high response in the 2$^{\text{nd}}$ component, with the +1 defect giving a high response in the 3$^{\text{rd}}$ component. Both are still highly nematic and give a strong 1$^{\text{st}}$ component response.

(a)



(b)

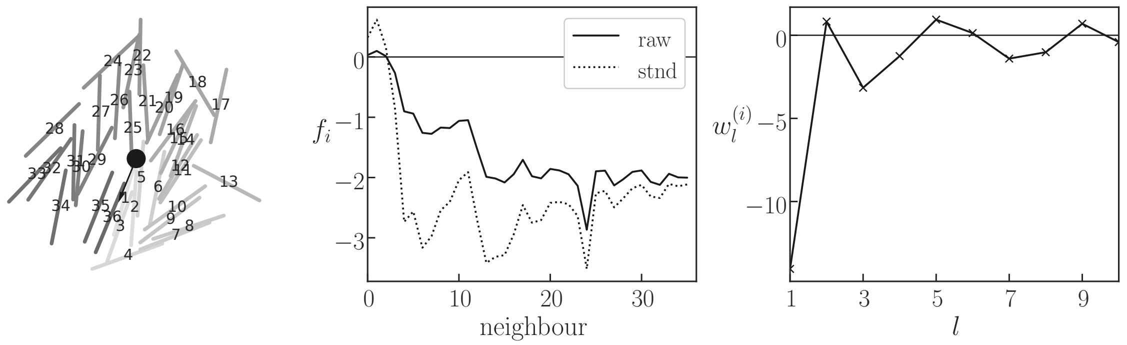


Figure 5.11: Feature vectors (middle column) and their modified PCA components (right column) for +1/2 (a) and +1 (b) winding numbers. The +1/2 defect shows a high response in the 2nd component, with the +1 defect giving a high response in the 3rd component. Both are still highly nematic and give a strong 1st component response.

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

[4] Matthew J. S. Beach, Anna Golubeva, and Roger G. Melko. Machine learning vortices at the kosterlitz-thouless transition. *Phys. Rev. B*, 97:045207, Jan 2018.

[5] 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.

[6] 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.

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

[8] 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.

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

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

[11] Zheng Yu Chen. Continuous isotropic-nematic transition of partially flexible polymers in two dimensions. *Phys. Rev. Lett.*, 71:93, 1993.

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

[13] 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.

[14] 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.

[15] 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. *J. Phys.: Condens. Matter*, 29(6):064003, 2017.

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

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

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

[19] 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.

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

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

[22] Jean-Christophe P Gabriel, Franck Camerel, Bruno J Lemaire, Hervé Desvaux, Patrick Davidson, and Patrick Batail. Swollen liquid-crystalline lamellar phase based on extended solid-like sheets. *Nature*, 413(6855):504, 2001.

[23] Jennifer Galanis, Daniel Harries, Dan L. Sackett, Wolfgang Losert, and Ralph Nossal. Spontaneous patterning of confined granular rods. *Phys. Rev. Lett.*, 96:028002, Jan 2006.

[24] Jennifer Galanis, Daniel Harries, Dan L Sackett, Wolfgang Losert, and Ralph Nossal. Spontaneous patterning of confined granular rods. *Phys. Rev. Lett.*, 96(2):028002, 2006.

[25] 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.

[26] 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.

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

[28] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. 1970.

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

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

[31] RB Jadrich, BA Lindquist, and TM Truskett. Unsupervised machine learning for detection of phase transitions in off-lattice systems. i. foundations. *The Journal of chemical physics*, 149(19):194109, 2018.

[32] Richard F Kayser and Harold J Raveché. Bifurcation in onsager's model of the isotropic-nematic transition. *Phys. Rev. A*, 17:2067–2072, Jun 1978.

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

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

[35] 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.

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

[37] Jan PF Lagerwall and Giusy Scalia. A new era for liquid crystal research: applications of liquid crystals in soft matter nano-, bio-and microtechnology. *Current Applied Physics*, 12(6):1387–1412, 2012.

[38] David Landau and Kurt Binder. *A Guide to Monte Carlo Simulations in Statistical Physics*. Cambridge University Press, New York, NY, USA, 2005.

[39] 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.

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

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

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

[43] Françoise Livolant. Ordered phases of dna in vivo and in vitro. *Physica A: Statistical Mechanics and its Applications*, 176(1):117–137, 1991.

[44] Chong Luo, Apala Majumdar, and Radek Erban. Multistability in planar liquid crystal wells. *Phys. Rev. E*, 85:061702, Jun 2012.

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

[46] Apala Majumdar. The landau-de gennes theory of nematic liquid crystals: Uniaxiality versus biaxiality. *Communications in Pure and Applied Analysis*, 2010.

[47] 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.

[48] 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.

[49] 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.

[50] Grégoire Montavon, Geneviève B Orr, and Klaus-Robert Müller. *Neural networks: tricks of the trade*. Springer, 2003.

[51] Alan Morningstar and Roger G Melko. Deep learning the ising model near criticality. *Journal of Machine Learning Research*, 18(163):1–17, 2018.

[52] 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.

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

[54] 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.

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

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

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

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

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

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

[61] Marina Soares e Silva, Jose 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.

[62] Andrés M Somoza, Celeste Sagui, and Christopher Roland. Liquid-crystal phases of capped carbon nanotubes. *Physical Review B*, 63(8):081403, 2001.

[63] 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.

[64] 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.

[65] Giacomo Torlai and Roger G. Melko. Learning thermodynamics with boltzmann machines. *Phys. Rev. B*, 94:165134, Oct 2016.

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

[67] 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.

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

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

[70] Qianshi Wei, Roger G. Melko, and Jeff Z. Y. Chen. Identifying polymer states by machine learning. *Phys. Rev. E*, 95:032504, Mar 2017.

[71] Sebastian J Wetzel and Manuel Scherzer. Machine learning of explicit order parameters: From the ising model to su (2) lattice gauge theory. *Phys. Rev. B*, 96(18):184410, 2017.

[72] WW Wood and JD 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.

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

APPENDICES