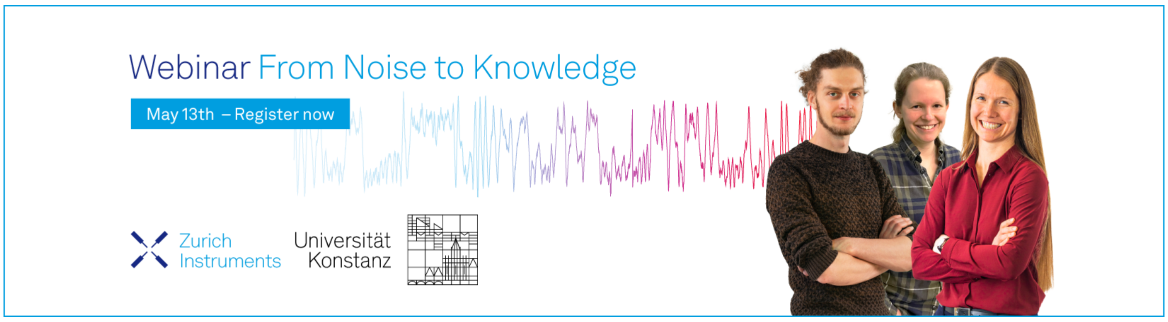
RESEARCH ARTICLE | SEPTEMBER 23 2020

Neural network model for structure factor of polymer systems

Jie Huang ; Shiben Li ; Xinghua Zhang ; Gang Huang

Check for updates

J. Chem. Phys. 153, 124902 (2020)
https://doi.org/10.1063/5.0022464



Neural network model for structure factor of polymer systems

Cite as: J. Chem. Phys. 153, 124902 (2020); doi: 10.1063/5.0022464
Submitted: 22 July 2020 • Accepted: 31 August 2020 •
Published Online: 23 September 2020

Jie Huang,[1] Shiben Li,[1,a)] Xinghua Zhang,[2,a)] and Gang Huang[3]

AFFILIATIONS

[1]Department of Physics, Wenzhou University, Wenzhou, Zhejiang 325035, China
[2]School of Science, Beijing Jiaotong University, Beijing 100044, China
[3]Institute of Theoretical Physics, Chinese Academy of Sciences, Beijing 100190, China

[a)]Authors to whom correspondence should be addressed: shibenli@wzu.edu.cn and zhangxh@bjtu.edu.cn

ABSTRACT

As an important physical quantity to understand the internal structure of polymer chains, the structure factor is being studied both in theory and experiment. Theoretically, the structure factor of Gaussian chains has been solved analytically, but for wormlike chains, numerical approaches are often used, such as Monte Carlo simulations, solving the modified diffusion equation. In these works, the structure factor needs to be calculated differently for different regions of the wave vector and chain rigidity, and some calculation processes are resource consuming. In this work, by training a deep neural network, we obtained an efficient model to calculate the structure factor of polymer chains, without considering different regions of wavenumber and chain rigidity. Furthermore, based on the trained neural network model, we predicted the contour and Kuhn lengths of some polymer chains by using scattering experimental data, and we found that our model can get pretty reasonable predictions. This work provides a method to obtain the structure factor for polymer chains, which is as good as previous and more computationally efficient. It also provides a potential way for the experimental researchers to measure the contour and Kuhn lengths of polymer chains.

I. INTRODUCTION

The structure factor of a polymer system defined as

$$S(\mathbf{k}) = \frac{1}{\rho}\int_V \langle\rho(\mathbf{r})\rho(0)\rangle \exp(i\mathbf{k}\cdot\mathbf{r})d\mathbf{r} \tag{1}$$

is a measurable physical property, which characterizes the density–density correlation of the system.[1] In theory, the structure factor can be used in field theory calculations. In the Gaussian fluctuation theory,[2,3] the structure factor of the interacting system is calculated using the structure factor of the ideal chain. Also, in the dynamic mean-field theory, the inter-chain correlation properties in the diffusion process[4–6] are also described by the structure factor of the ideal chain. Experimentally, the basic characteristics of polymers, such as the degree of polymerization, the rigidity, and the chirality, can be analyzed by fitting the scattering data with the structure factor. As for calculation, the structure factor can be predicted from a microscopic chain model. The well-known expression of the Gaussian chain model has a Debye function form[7] and can be used to analyze the experimentally determined structure factor for a θ-point dilute polymer solution in a moderate to small wavenumber range, $ka \leq 1$, where a is the Kuhn length.

In addition, there is a large class of semiflexible polymer chains, where the effects of finite rigidity are important, which cannot be described by the Gaussian chain model. The wormlike chain model is one of the best semiflexible chain models. In this model, the polymer is an inextensible thread subject to a linear-elastic bending energy.[8] The configuration of a wormlike chain of total length L is described by a smooth space curve with its coordinate specified by $\mathbf{R}(s)$, where s is an arc-variable continuously varying from one end ($s = 0$) to another ($s = 1$).[7,9,10] The Boltzmann weight for such a configuration is given by

$$\mathscr{W}[\mathbf{R}(s)] = \exp[-\beta H_0], \tag{2}$$

where

$$\beta H_0 = \frac{a}{4L}\int_0^1 \mathrm{d}s\left|\frac{\mathrm{d}\mathbf{u}(s)}{\mathrm{d}s}\right|^2 + \frac{L}{a}\int_0^1 \mathrm{d}s w[\mathbf{R}(s),\mathbf{u}(s)]. \quad (3)$$

The tangent vector $\mathbf{u}(s) \equiv (1/L)\mathrm{d}\mathbf{R}(s)/\mathrm{d}s$ specifies the local orientation of the polymer chain at location s. $u(s)$ is a unit vector, and $|u(s)| = 1$ due to the local inextensible constraint. The first term describes an energy penalty for a bent curve. Originally, a bending energy modulus $\beta\varepsilon$ was written as the coefficient;[9] upon identification of the free-space mean-square radius of gyration with that of a Gaussian chain in the large L/a limit, we can show that the prefactor can be written in the current form, where

$$a = 2\beta\varepsilon \quad (4)$$

for a three-dimensional system. The Kuhn length a is directly used here for comparison with results calculated from a Gaussian-chain model. $w(\mathbf{R},\mathbf{u})$ is an external field applied to the polymer chain. The wormlike chain model involves two characteristic length scales: the length of chain L and the effective Kuhn length a.

The key to calculating the structure factor is the calculation of the Green's function (with $w = 0$) in Eq. (3). As it turns out, no analytic expression of the Green's function is available for the wormlike chain model, as indicated earlier by Stepanow,[11,12] Spakowitz and Wang,[13] and Zhang.[3,14]

Kholodenko exploited the similarity between the Green's function of the semiflexible polymer model and the propagator of Dirac's fermion, in the rigid and flexible limits.[15,16] The limits for Gaussian-chain and rod expressions can be reproducible from the formula. It is by far the simplest, in comparison with the approximations proposed earlier by Yoshizaki and Yamakawa[17] and later by Pedersen and Schurtenberger.[18]

Pedersen and Schurtenberger performed a series of Monte Carlo simulations of such a chain, with and without the excluded-volume interaction between monomers. The structure factor can then be obtained numerically from the simulations.[18] They have provided an empirical formula to represent their simulation data. More recently, Hsu and co-workers calculated the structure factor of a semiflexible chain model on a simple cubic lattice using Monte Carlo simulations.[19,20]

Spakowitz and Wang proposed an alternative approach; calculating the problem of constrained one-dimensional random walk, they obtained the Green's function of a wormlike chain formally.[13] They re-grouped the random walk trajectories according to the number of loops in a loop expansion of the problem. Based on this consideration, the moment expansion can be expressed as an infinite continued fraction. The calculation of the continued fraction problem is equivalent to inverting a matrix that has the same format as the matrix used in Stepanow's work. To find the structure factor, however, one must go back to the numerical treatment of the formalism; in particular, an inverse numerical Laplace transformation is needed.[13,21]

In our previous work,[3] a numerical method to obtain the structure factor of a homogeneous wormlike polymer solution based on the standard wormlike chain model was obtained. We calculated the s-dependent Green's function, utilizing a formal solution to the modified diffuse equation (MDE)[10,22] that the Green's function in Fourier space satisfies and propagating the solution as s increases. This method was numerically more straightforward than some other approaches suggested recently. The solution captured the correct physical behavior of the structure factor in the entire parameter space of L/a and ka.

The motivation of this work is twofold. First, we attempt to find a more efficient formulation of the structure factor for wormlike chains in the entire parameter space of L/a and ka, but the more direct way where we do not need to do any heavy calculation like Monte Carlo simulations or solve partial differential equations. Second, we want to build a possible measurement tool for the scattering experiments of polymer chains. If scattering intensity data are given, the contour length L and Kuhn length a can be easily obtained.

Recently, neural networks (NNs), as an important branch of machine learning (ML), are widely used in polymer physics, such as classifying phases of matter,[23] solving nonlinear partial differential equations (PDE),[24] predicting the structure of macromolecules,[25] and polymer conformation classification.[26] Since the NN has been proven to be able to approximate almost any functions,[27] we do not need to find the structure factor from the perspective of guessing the analytic formula but only need to use a NN to replace it. To ensure the high accuracy of a NN model, a sufficient dataset is needed to train this NN. Fortunately, we can get numerous exact structure factor data with different ka and L/a by using the method in Ref. 3. By training with a dataset, we can get a trained NN model to calculate the structure factor easily.

The outline of the rest of this paper is as follows: We first demonstrate how to apply the NN to structure factor fitting in Sec. II, including the basic introduction of NNs, the training task, the training process, and the influence of NN architecture. Following this, we build a method to predict the contour length L and Kuhn length a of polymer chains in Sec. III by using the trained NN.

II. THE NEURAL NETWORK MODEL FOR STRUCTURE FACTOR

A. A brief introduction to neural networks

To begin with, we introduce some basic concepts of neural networks. NNs, also called artificial neural networks (ANNs), are computing systems, which can learn to perform different tasks by considering example generally without being programmed with task-specific rules. A NN is based on a collection of connected nodes or units called *neurons*. As in Fig. 1(a), a link from neuron i to neuron j serves to propagate the activation a_i from i to j. Each link also has a numeric weight of $w_{i,j}$ associated with it, which determines the strength and sign of the connection. Each neuron has a dummy input $a_0 = 1$ with an associated weight $w_{0,j}$. Each neuron j first computes a weighted sum of its inputs,

$$in_j = \sum_{i=0}^{n} w_{i,j}a_i.$$

Then, it applies an activation function g to this sum to derive the output,[28]

$$a_j = g(in_j) = g\left(\sum_{i=0}^{n} w_{i,j}a_i\right). \quad (5)$$

The nonlinear activation function is the key to the power of NN that allows it to approximate almost any function. In this work, the

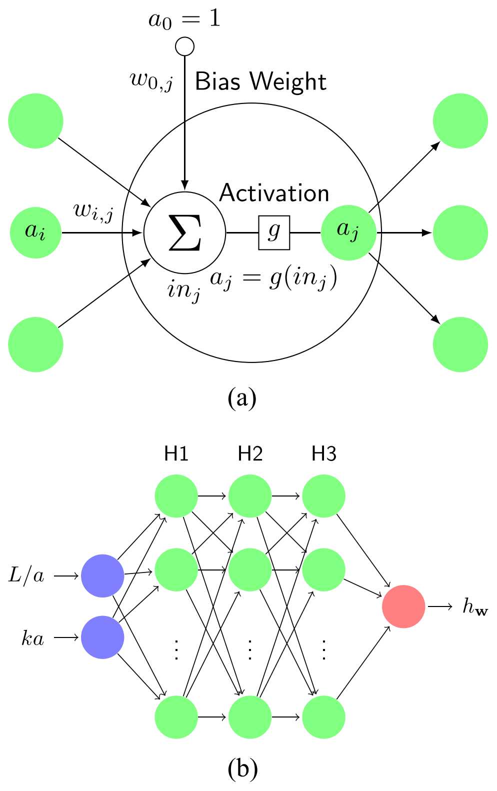


FIG. 1. (a) The mathematical model for a neuron. The unit's output activation is $a_j = g(\sum_{i=0}^{n} w_{i,j} a_i)$, where x_i is the output of unit i and $w_{i,j}$ is the weight on the link from unit i to this unit. (b) A fully connected NN with three hidden layers.

sigmoid function was used,

$$g(z) = \frac{1}{1 + e^{-z}}.$$

Having decided on the mathematical model for the individual neurons, then a fully connected NN can be obtained by connecting them. As shown in Fig. 1(b), a fully connected NN is arranged in layers, which can be divided into an input layer, many hidden layers, and an output layer. Only the input layer does not participate in the calculation in Eq. (5). There can be many neurons on each layer. Any neuron in the hidden layer and output layer is connected to all neurons in the previous layer.

A NN can be viewed as a mapping h from its input $\mathbf{x}$ to its output $h_\mathbf{w}(\mathbf{x})$, where $\mathbf{w}$ is the collection of all the weights in this NN. By changing $\mathbf{w}$, different h's can be obtained. The process of tuning $\mathbf{w}$ to approximate another function f is called *training* the NN. We train the NN by showing lots of input–output pairs repeatedly to the NN so that it can gradually *learn* the mapping from the input to output by tuning the $\mathbf{w}$. The input–output pairs constitute a *training set*, and this type of learning is called *supervised* learning.

To be more specific, the training task can be described as follows. Given a training set of N example input–output pairs $(\mathbf{x}_1, y_1)$, …, $(\mathbf{x}_j, y_j)$, …, $(\mathbf{x}_N, y_N)$, where $\mathbf{x}_j = ((L/a)_j, (ka)_j)^T$ and y_j was generated by the method in Ref. 3,

$$y = f(\mathbf{x}) = (L/a)(ka)^2 S(L/a, ka), \tag{6}$$

find a function h that approximates the function f. The way to train the NN is as follows: First, we defined a loss function,

$$Loss(\mathbf{w}) = \frac{1}{N}\sum_x \|f(x) - h_\mathbf{w}(x)\|^2, \tag{7}$$

which indicates how far away the h is from the objective function f. By training the NN, we would like to find the weights $\mathbf{w}^*$ so that the loss function over the examples could be minimized, i.e.,

$$\mathbf{w}^* = \arg\min_\mathbf{w} Loss(\mathbf{w}). \tag{8}$$

There are many optimization algorithms, also called *optimizers*, to find $\mathbf{w}^*$, but the basic idea can be expressed as follows. The weight w_i is updated by

$$w_i \leftarrow w_i - \alpha\frac{\partial Loss(\mathbf{w})}{\partial w_i}, \tag{9}$$

where α is the learning rate. In this work, we use an adaptive learning rate optimizer called Adam[29] that is designed specifically for training NNs. To increase training efficiency, the training set is usually divided into many mini-batches. Each time a mini-batch is used to update $\mathbf{w}$. Besides, the training set is used to update $\mathbf{w}$ many times. The process of updating $\mathbf{w}$ using a training set once is also called an *episode*.

The training set for the structure factor of this work comes from the numerical method in Ref. 3, which obtained an excellent agreement with the structure factor computed from the method of

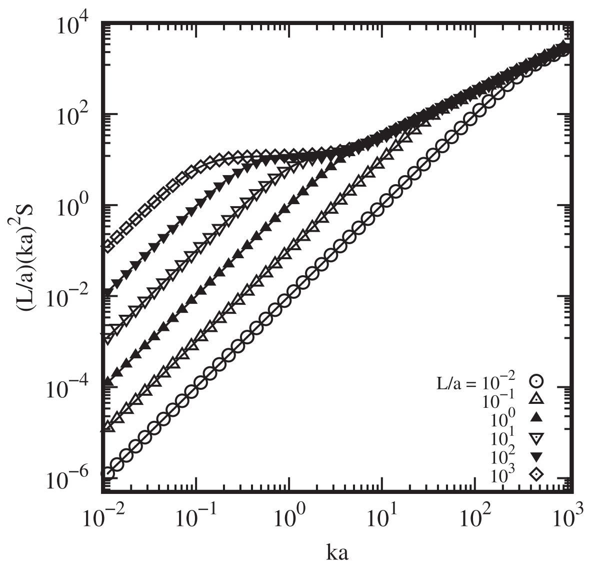


FIG. 2. Structure factor comparison between the target values (circles, triangles, etc.) and trained NN predictions (lines) in logarithmic coordinates with 4 hidden layers and 25 nodes on each hidden layer with $Loss = 6.92 \times 10^{-7}$.

infinite continuous fractions by Spakowitz and Wang.[13] Compared with other methods such as the Dirac propagator approach[15] or Monte Carlo simulations,[18] this method gives rise to the precise determination of the structure factor in the entire L/a–ka space (L/a, $ka \in [10^{-2}, 10^{3}]$), especially, in low and large L/a regimes so that it laid the foundation for a reliable training set.

In Ref. 3, the polynomial $(L/a)(ka)^2S$ is a function with L/a and ka as arguments. We consider the $\mathbf{x} = (L/a, ka)^T$ and the corresponding $(L/a)(ka)^2S$ as a training sample. For each L/a, 100 points for ka in $[10^{-2}, 10^{3}]$ are uniformly sampled on $\ln(ka)$. Similarly, 100 points for L/a in $[10^{-2}, 10^{3}]$ are uniformly sampled on $\ln(L/a)$. Therefore, 10 000 training samples were obtained, which covers the entire domain of ka and L/a described by the wormlike chain model.

B. Training results of the NN model

By using TensorFlow, we chose an Adam optimizer with a fixed learning rate of 10^{-5}, 4 hidden layers, and 25 neurons per layer. After 6×10^{5} epochs of training, the loss value L was reduced to 6.9×10^{-7}.

To demonstrate the effectiveness of the structure factor in the form of NN, in Fig. 2, we have plotted the NN model predictions of five different rigidities with $L/a = 10^{-1}$, 10^{0}, 10^{1}, 10^{2}, and 10^{3}. For comparison, we also made the solution of the structure factor given in Ref. 3, denoted by circles, triangles, etc. The NN can well represent the structure factor for different rigidities for the entire k range, consistent with the exact result obtained by the method proposed in Ref. 3.

To further verify the fitting results of the trained NN, we made comparisons at different scales in linear coordinates. Figures 3(a)–3(d) correspond to different ka regions $[10^{-2}, 10^{-0}]$, $[10^{0}, 10^{1}]$, $[10^{1}, 10^{2}]$, and $[10^{2}, 10^{3}]$, respectively, where the solid lines are the values given by the NN model, and the circles, triangles, etc., are the solutions from Ref. 3. It can be concluded that the NN model can give highly accurate structure factor values in the *entire* L/a–ka space. Therefore, our model also has a good description of rigid and semi-rigid polymer chains, which is of practical significance, such as fluctuation theory and scattering experiment.

What is more important is that our trained NN model can provide a continuous function of the structure factor in the entire L/a–ka space through the limited discrete training samples. As shown in Fig. 4, a continuous $(L/a)(ka)^2S$ plane in logarithmic coordinates is obtained. This result indicates that given *any* L/a–ka pair, the structure factor can be predicted directly. In addition, we can easily calculate the values of *multiple* structure factors *simultaneously*, so the NN model greatly improves the calculation efficiency.

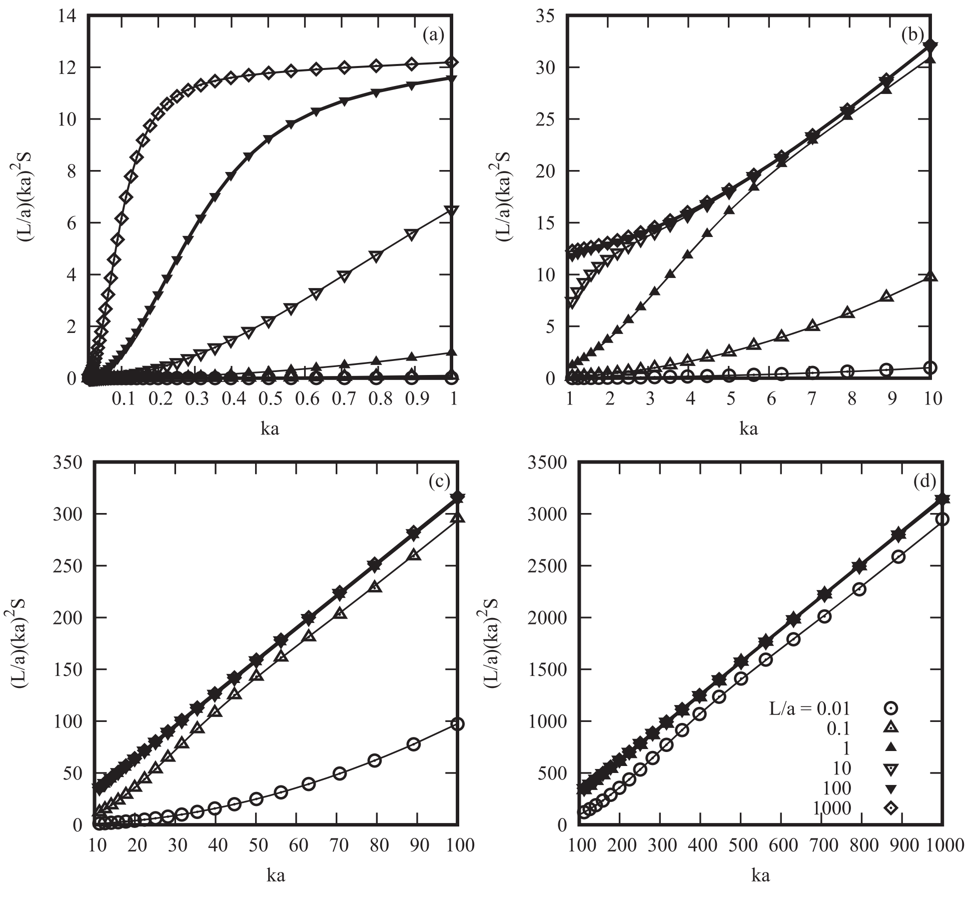


FIG. 3. Structure factor comparison between the target values (circles, triangles, etc.) and trained NN predictions (lines) in linear coordinates: (a) $ka < 1$, (b) $ka \in [1, 10]$, (c) $ka \in [10, 100]$, and (d) $ka \in [100, 1000]$.

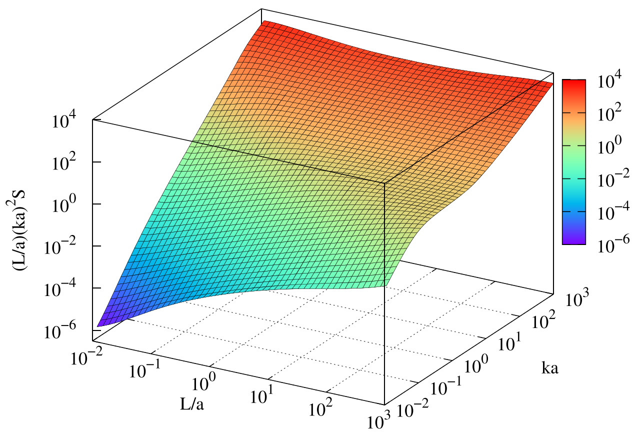


FIG. 4. The continuous surface of $(L/a)(ka)^2S$ generated by the trained NN model.

C. The effect of network structure

Hyperparameters, such as the number of neurons, the number of layers, optimizer, and learning rate, are also important in the training. In this work, we focused on the effect of the number H of hidden layers and the number N of neurons in each hidden layer.

To simplify the parameter adjustment process, we have made the number of nodes on each hidden layer the same. To study the effect of H on the fitting results, we fixed N, then we used four values of H (1, 2, 4, 8) to get four network structures. Finally, we have separately trained the NNs.

Figure 5 shows the change of *Loss* in four independent pieces of training when the total number N of hidden layer nodes is fixed at 128. At the end of training, episode ~ 900 000, the *Loss* for $H = 2$, 4 is less than for $H = 1$, 8. This result indicates that when N is fixed, too many or too few hidden layers will lower the performance of the trained NN model. Therefore, choosing the appropriate H can help make the model converge faster.

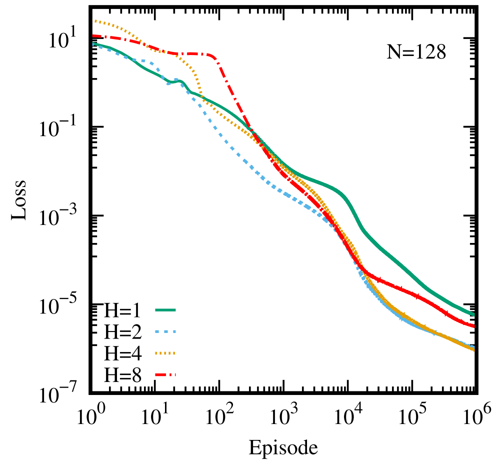


FIG. 5. Time dependence of the loss function for different numbers H of hidden layers ($N = 128$).

TABLE I. The comparison of *Loss* for different N and H after 9×10^5 episodes of training.

	$N = 32$	$N = 64$	$N = 128$	$N = 256$
$H = 1$	1.531×10^{-5}	8.059×10^{-6}	5.607×10^{-6}	7.129×10^{-6}
$H = 2$	1.423×10^{-6}	6.197×10^{-7}	1.024×10^{-6}	1.644×10^{-6}
$H = 4$	3.847×10^{-6}	1.680×10^{-6}	9.017×10^{-7}	5.411×10^{-7}
$H = 8$	2.597×10^{-4}	1.918×10^{-5}	3.109×10^{-6}	2.005×10^{-6}

To further test the effect of N, we trained the NN with $N = 32$, 64, 128, and 256. In Appendix B, the corresponding episode dependence of the *Loss* for different N is shown. In Table I, we listed the *Loss* values at the end of training when *episode* = 900 000, with different N and H. From this table, we noticed that for all N, the *Loss* for $H = 2$, 4 is smaller than for $H = 1$, 8. Therefore, there must be an optimal H value between $[H_{\min}, H_{\max}]$, which is [1,8] in our case. In addition, as N increases, the loss value decreases more, which means that the training processes can converge faster and can get more accurate prediction results.

III. PREDICT THE CONTOUR LENGTH AND KUHN LENGTH OF POLYMER CHAINS

Small-Angle Neutron Scattering (SANS) is a widely used technique to study the structure of polymers. In SANS, the scattering intensity I is measured as a function of the length of the scattering vector q. The structure factor of wormlike chains in NN formation developed in the present work is an exact formation. Any previous approximation formations used to analyze the scattering experiments can be replaced by the NN formation directly. In this part, we used some public scattering intensity data of polymers from the SANS experiment as examples to demonstrate the uses of the trained NN model and then predicted the two important parameters of polymer chains, the contour length L and Kuhn length a.

A. Method

The scattering intensity of a polymer chain can be fitted using the following equation:[18,30]

$$I_p(q) = cP(q)S(q), \tag{10}$$

where c is a scaling factor and $S(q)$ is the structure factor obtained by the NN model, and

$$P(q) = \left[\frac{2J_1(Rq)}{Rq}\right]^2 \tag{11}$$

is the form factor, where $J_1(x)$ is the first order Bessel function and R is the cross-sectional radius. We approximated the finite cross section of the polymer chains into a cylinder with a radius of R and a length of a.[18] The structure factor S is determined by the parameter L and a, and the form factor P is determined by R. Thus, there are four fitting parameters: contour length L, Kuhn length a, radius R, and the scaling factor c.

We can use Eq. (10) to fit the SANS data by changing the parameters L, a, R, c. Define

$$\epsilon(a,L,R,c) = \frac{1}{N}\sum_{i=1}^{N}\left(I_p^i(a,L,R,c) - I^i\right)^2 \tag{12}$$

as the optimizing target, where I is the scattering intensity of a polymer chain obtained by the SANS and I_p is the scattering intensity predicted by our NN model. The parameters a, L, R, c need to be adjusted to make the predicted scattering intensity $I_p(q)$ and the measured one $I(q)$ as close as possible. When ϵ is small enough, the optimal parameters L^*, a^*, R^*, c^* can be obtained. Therefore, we predicted the contour length (L^*) and Kuhn length (a^*) of the polymer chain.

There are multiple fitting parameters in the formula [Eq. (12)], which will bring difficulties in fitting the experimental data, especially, in scattering data with fluctuations. The approximated structure factor of the chain in solution [Eq. (12)] is approximated by modifying the ideal chain structure factor. To describe the effects of chain thickness and the solvent–monomer and monomer–monomer interaction, some additional parameters have to be introduced in the formulation. There are many approximated formulations used in the scattering experiments in the literature.[18,31,32] The prerequisite of these formulations is the structure factor of the ideal wormlike chain model. The major purpose of the present work is to develop an accurate structure factor formulation of the ideal wormlike chain model.

B. Discussions

Using the method in Sec. III A, given the SANS intensity data of polymer chains, we can determine the contour lengths and Kuhn lengths. Two examples are given below.

1. Polystyrene

The scattering intensity data of atactic polystyrene (PS) in carbon disulfide (CS2) with different selective deuteration of the polymer have been determined by Rawiso, Duplessix, and Picot[33] using SANS. As shown in Fig. 6, we have used two sets of scattering intensity datasets for the phenyl ring deuterated (Exp1: circle dots) and fully deuterated (Exp2: triangle dots) PS. In Table II, the Kuhn lengths a are determined to be 22.38 Å and 22.17 Å, which are in good agreement with the previous determinations for SANS data (22 Å–27 Å). The contour lengths L are 1300 Å and 1574 Å for Exp1 and Exp2, respectively. These results are also in good agreement with the values 1360 and 1810 Å in Ref. 18.

2. Poly[3-(2′-ethyl)hexylthiophene]

Another set of scattering intensity data of polymer chain P3EHT4 is from the experiments of Bryan McCulloch *et al.*[32] Note that they proposed a very novel model of SANS intensity, the polydispersity-corrected wormlike chain model,

$$I(q) = K\int_{n=0}^{n=\infty} w_i g(u_{ni}) n_i \mathrm{d}n_i + I_{\mathrm{inc}}, \tag{13}$$

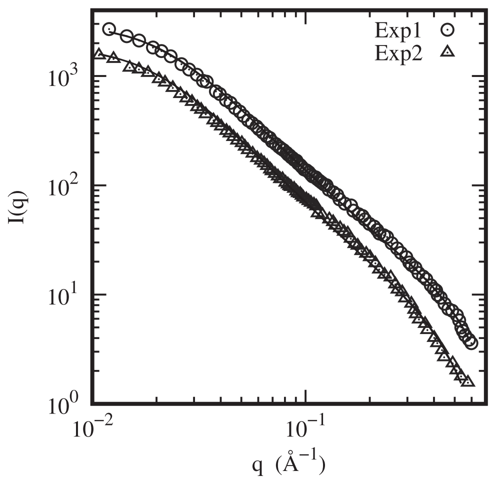


FIG. 6. Scattering intensity comparison between the SANS data and the trained NN model prediction for PS with a molecular weight M_w of 50 000 in CS$_2$. Exp1 is for the phenyl ring deuterated PS, and Exp2 is for the fully deuterated PS.

where the structure factor is denoted by g in Ref. 32,

$$g(u) = \frac{2}{u^2}\left(u - 1 + \mathrm{e}^{-u}\right) + \frac{2}{5q^2L^2}\left[4u - 11u\mathrm{e}^{-u} + 7\left(1 - \mathrm{e}^{-u}\right)\right], \tag{14}$$

w_i is the weight fraction at a particular molecular weight, and

$$u = q^2R_g^2 = q^2\left[\frac{Ll_p}{3} - l_p + \frac{2l_p^3}{L}\left(1 - \frac{l_p}{L} + \frac{l_p}{L}\mathrm{e}^{-L/l_p}\right)\right],$$

where l_p is the persistence length.

In principle, we can directly use the structure factor S obtained by our trained NN model to replace g in Eq. (13) and then fit the contour length and Kuhn length of P3EHT4. However, due to the lack of the original absolute molecular weight distribution of P3ETH4, we did not use Eq. (13) to fit the SANS intensity data in this work. Alternatively, we have made a comparison chart of $(L/a)(ka)^2S$ and $(L/a)(ka)^2g$ with the Kuhn length $a = 10$ to compare the structure factor S from the NN model and g in Eq. (14). We found that when

TABLE II. The predictions of L and a (Å) for PS (Fig. 6).

	Upper	Lower
Reasonable target value L[a]	1360	1810
NN L	1300.03	1573.76
Reasonable target value a	22–27	22–27
NN a	22.38	22.17
ϵ[b]	0.000 68	0.0013

[a]These values were calculated from molecular weights and the structure taking into account the polydispersity.[18]
[b]ϵ is fitting error defined in Sec. III A.

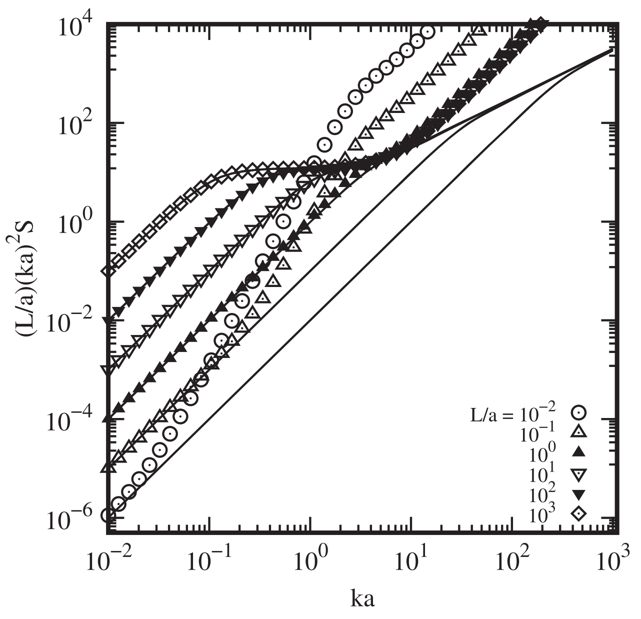


FIG. 7. The structure factor comparison between the trained NN model (lines) and the wormlike chain model g (circles, triangles, etc.).[32]

$L/a > 1$, $ka < 10$, the S of the NN model and the g in Eq. (14) matched well as shown in Fig. 7. Specifically, because g is deduced from the formula of the flexible chain model and uses an approximate form at large k. Therefore, for flexible polymer chains ($L/a = 10^1, 10^2, 10^3$), g gives a good description of the structure factor when $ka < 10^0$. However, for large ka ($ka > 10^1$), g describes the chain not very well. In addition, for semi-rigid polymer chains ($L/a = 10^0$), g is good when k is small, but it is not good enough when $k > 10^0$. In fact, the semi-rigid chain is the key in many cases. Besides, g cannot describe the structure factor of rigid chains ($L/a < 10^{-1}$) because the approximation for the rigid chains is not good enough. The origin of g here and the structure factor put by Pedersen[18] and Kholodenko[31] are very similar. However, their structure factor models can provide a better description in the rigid limit and the large k limit.

As described in Sec. II, our NN model can give precise predictions of S in the entire L/a–ka space. Therefore, we expect the same fitting results as in Ref. 32, if g in Eq. (13) is replaced by S. Nevertheless, the Kuhn length a of P3EHT4 determined by the intensity model we used in Eq. (10) is 5.013 Å with ϵ minimized.

As shown in Fig. 8, the intensity $I_p(q)$ calculated from our NN model fits very well with the SANS intensity $I(q)$.

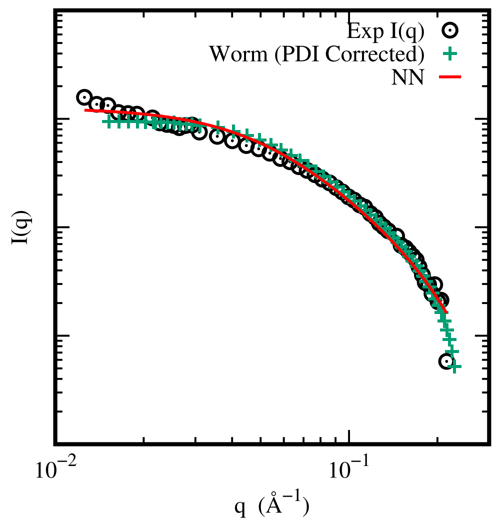


FIG. 8. Scattering intensity comparison between the SANS data and the prediction of different models including Debye, wormlike chain (PDI corrected),[32] and NN models for P3EHT-4.

IV. SUMMARY

We have developed an efficient model for the structure factor of a wormlike chain polymer by training a fully connected NN. Our NN model is of the following characters: (a) High-precision, continuous numerical solutions in the entire L/a–ka space can be obtained easily. (b) It is highly consistent with the calculations in previous numerical and analytical methods.[3] Besides, we also proposed one application of the model. Combining SANS intensity data, we can determine the contour length and Kuhn length of polymer chains. Therefore, our NN model may provide a potential tool for exploring the properties of polymer chains for experimental researchers.

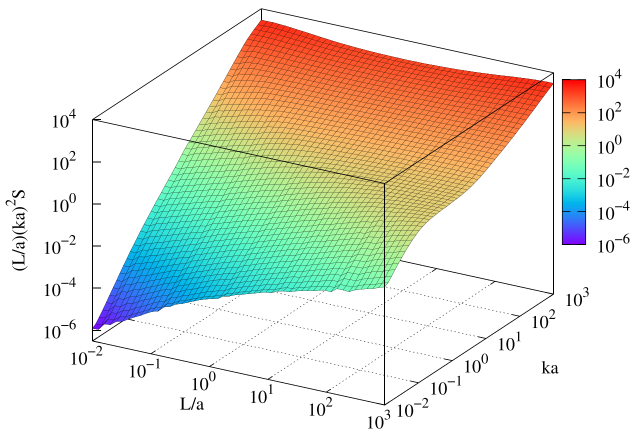
ACKNOWLEDGMENTS

This research was supported by the National Natural Science Foundation of China (NSFC) (Grant Nos. 21973070, 21774013, and 21574011) and the Beijing Natural Science Foundation (Grant No. 2182057). The authors also wish to express their appreciation for Jeff Z. Y. Chen for his valuable suggestions. J.H. thanks Ying Jiang for his helpful guidance and discussion.

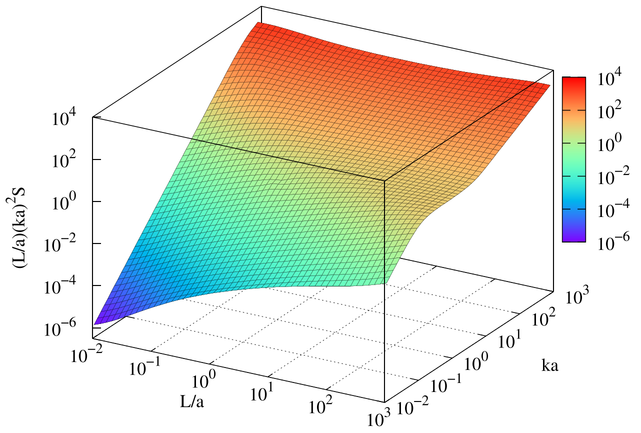
APPENDIX A: STRUCTURE FACTOR OBTAINED BY INTERPOLATION

Due to the monotonicity of the $(L/a)(ka)^2S$-surface, the structure factor may also be obtained by using suitable interpolation algorithms. In addition to using the NN, we also use two interpolation algorithms to accelerate the computation of the structure factor. In these two interpolation algorithms, we use the *same* data points as described in Sec. II A.

We found that interpolation algorithms approximately give the numerical solution of the structure factor in the entire ka, L/a space. In Fig. 9, the $(L/a)(ka)^2S$-surface is obtained by (a) the nearest-neighbor method and (b) the cubic-spline method. The $(L/a)(ka)^2S$-surface obtained by the nearest-neighbor method in Fig. 9(a) is less smooth than by the cubic-spline method in Fig. 9(b). The structure factor surface obtained by the cubic-spline method is closer to the solution of the MDE and the NN model. The interpolation works well at the small and large k limits where the fractal dimension of



(a)



(b)

FIG. 9. The surface of $(L/a)(ka)^2S$ obtained by interpolation: (a) nearest-neighbor method and (b) cubic-spline method.

wormlike chains can be well determined. For the medium range k condition and semiflexible chain condition, S varies rapidly. More data are required for the interpolation.

APPENDIX B: LOSS FUNCTION FOR DIFFERENT *N*

Figure 10 shows how the loss function changes for different N.

APPENDIX C: STRUCTURE FACTOR

In this appendix, we list some analytical expressions of the structure factor in different methods.

1. Kholodenko

In Ref. 15, the structure factor S is obtained, which correctly reproduces the rigid-rod and random-coil limits and is given analytically by

$$S(k) = \frac{2}{x}\left[I_{(1)}(x) - \frac{1}{x}I_{(2)}(x)\right], \tag{C1}$$

where $I_{(n)}(x) = \int_0^x f(z)z^{n-1}\mathrm{d}z$, $n = 1, 2$, $x = 3L/a$,

$$f(z) = \begin{cases} \frac{1}{E}\frac{\sinh(Ez)}{\sinh z} & (k \le 3/2\mathrm{a}) \\ \frac{1}{\hat{E}}\frac{\sin(\hat{E}z)}{\sinh z} & (k > 3/2\mathrm{a}) \end{cases}$$

and

$$E = \left[1 - \left(\frac{2}{3}ak\right)^2\right]^{1/2}, \hat{E} = \left[\left(\frac{2}{3}ak\right)^2 - 1\right]^{1/2}.$$

2. Pederson and Schurtenberger

In Ref. 18, the structure factor of a semiflexible chain is given by

$$S = S_{\mathrm{SB}}P + S_{\mathrm{loc}}(1-P), \tag{C2}$$

where

$$S_{\mathrm{loc}} = \frac{c_1}{Laq^2} + \frac{\pi}{Lq} \tag{C3}$$

is the approximate scattering function at high q suggested by Burchard and Kajiwara,[34]

$$S_{\mathrm{SB}} = S_{\mathrm{Debye}} + \frac{c_2 a}{L}\left[\frac{4}{15} + \frac{7}{15x} - \left(\frac{11}{15} + \frac{7}{15x}\right)\exp(-x)\right] \tag{C4}$$

is the scattering function calculated for the Daniels approximation by Sharp and Bloomfield,[35] and

$$P = \exp\left[-\left(\frac{qa}{q_1}\right)^{p_1}\right],$$

where q_1 and p_1 are empirical constants. In Eq. (C4),

$$S_{\mathrm{Debye}}(x) = \frac{2}{x^2}[\exp(-x) + x - 1]$$

is the scattering function given by the Debye function,[36] with $x \equiv R_g^2q^2$, and

$$R_g^2 = \frac{La}{6}\left\{1 - \frac{3a}{2L} + \frac{3a^2}{2L^2} - \frac{3a^3}{4L^3}\left[1 - \exp\left(-\frac{2L}{a}\right)\right]\right\}.$$

The parameters depend on the L/a. For $L/a > 2$, $c_1 = 1$, $c_2 = 1, p_1 = 5.33, q_1 = 5.53, R_g^2 = La/6$. For $L/a \le 2$, $c_1 = 0.0625$, $c_2 = 0$, $p1 = 3.95$, $q_1 = 11.7a/L$.

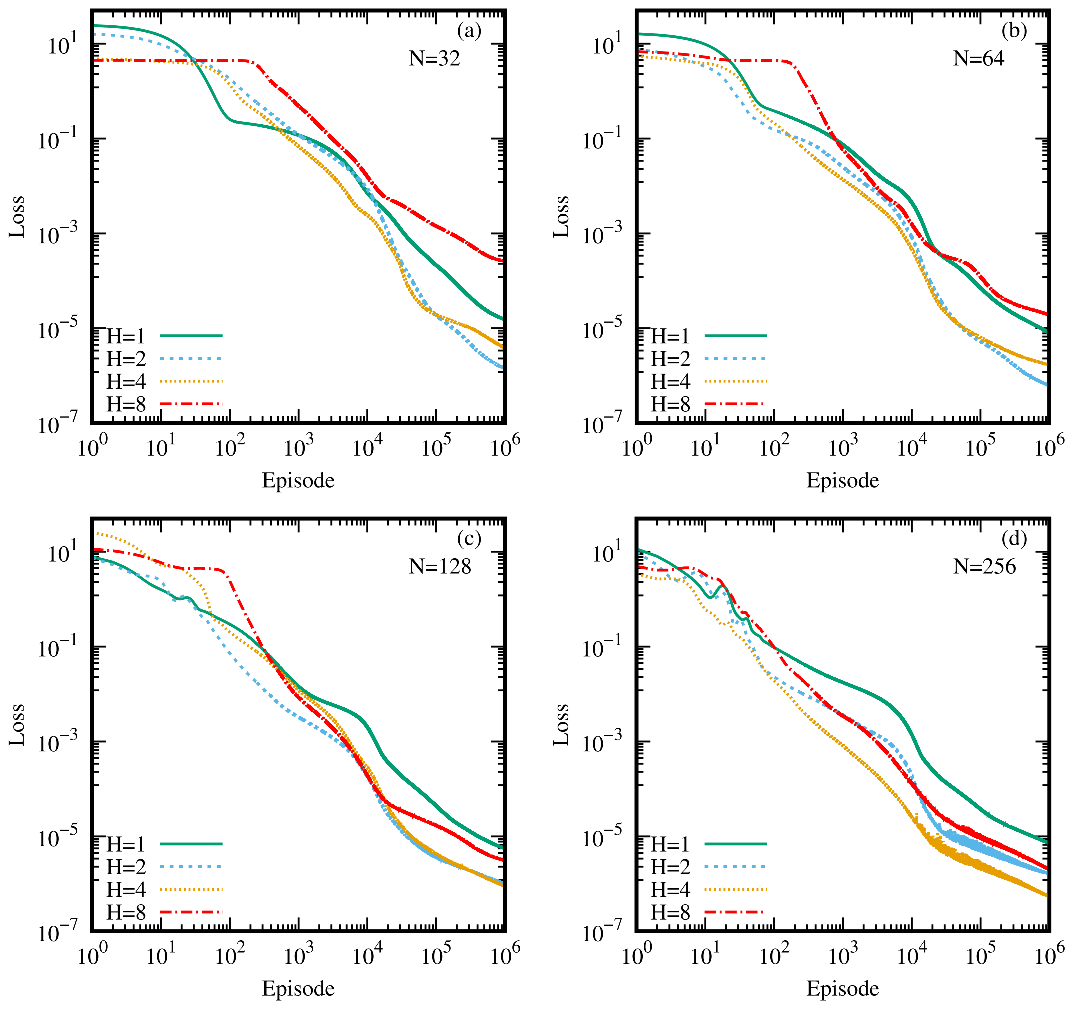


FIG. 10. Time dependence of the loss function for different H and N of the NN: (a) $N = 32$, (b) $N = 64$, (c) $N = 128$, and (d) $N = 256$.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] D. I. Svergun, in *Structure Analysis by Small-Angle X-Ray and Neutron Scattering*, edited by G. W. Taylor (Plenum, 1987).
[2] X. Bu and X. Zhang, Polymers **8**, 301 (2016).
[3] X. Zhang, Y. Jiang, B. Miao, Y. Chen, D. Yan, and J. Z. Y. Chen, Soft Matter **10**, 5405 (2014).
[4] X. Chen, S. Qi, X. Zhang, and D. Yan, ACS Omega **5**, 7593 (2020).
[5] S. Qi, X. Zhang, and D. Yan, J. Chem. Phys. **132**, 064903 (2010).
[6] X. Zhang, S. Qi, and D. Yan, J. Chem. Phys. **137**, 184903 (2012).
[7] M. Doi and S. F. Edwards, *The Theory of Polymer Dynamics* (Oxford University Press, New York, 1986).
[8] L. D. Landau, E. M. Lifshitz, J. B. Sykes, W. H. Reid, and E. H. Dill, *Physics Today* (Pergamon, New York, 1986).
[9] N. Saitô, K. Takahashi, and Y. Yunoki, J. Phys. Soc. Jpn. **22**, 219 (1967).
[10] K. F. Freed, Adv. Chem. Phys. **22**, 1 (1972).
[11] S. Stepanow, Eur. Phys. J. B **39**, 499 (2004).
[12] S. Stepanow, J. Phys.: Condens. Matter **17**, 1799 (2005).
[13] A. J. Spakowitz and Z.-G. Wang, Macromolecules **37**, 5814 (2004).
[14] Y. Yang, X.-y. Bu, and X.-h. Zhang, "Structure factor based on the wormlike-chain model of single semiflexible polymer," Acta Polym. Sin. **8**, 1002–1010 (2016).
[15] A. L. Kholodenko, Macromolecules **26**, 4179 (1993).
[16] A. L. Kholodenko, Ann. Phys. **202**, 186 (1990).
[17] T. Yoshizaki and H. Yamakawa, Macromolecules **13**, 633 (1980).
[18] J. S. Pedersen and P. Schurtenberger, Macromolecules **29**, 7602 (1996).
[19] H.-P. Hsu, W. Paul, and K. Binder, J. Chem. Phys. **137**, 174902 (2012).
[20] H.-P. Hsu, W. Paul, and K. Binder, Polym. Sci., Ser. C **55**, 39 (2013); arXiv:1303.2543.
[21] S. Mehraeen, B. Sudhanshu, E. F. Koslover, and A. J. Spakowitz, Phys. Rev. E **77**, 061803 (2008).
[22] Q. Liang, J. Li, P. Zhang, and J. Z. Y. Chen, J. Chem. Phys. **138**, 244910 (2013).

[23]E. P. L. Van Nieuwenburg, Y.-H. Liu, and S. D. Huber, Nat. Phys. **13**, 435 (2017).
[24]M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics informed deep learning (Part I): Data-driven solutions of nonlinear partial differential equations," arXiv:1711.10561v1.
[25]J. Li, H. Zhang, and J. Z. Chen, Phys. Rev. Lett. **123**, 108002 (2019).
[26]O. Vandans, K. Yang, Z. Wu, and L. Dai, Phys. Rev. E **101**, 1 (2020).
[27]G. Cybenko, Math. Control, Signals, Syst. **2**, 303 (1989).
[28]P. Norvig and S. J. Russell, *Artificial Intelligence: A Modern Approach*, 3rd ed. (Pearson Education, USA, 2010).
[29]D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv:1412.6980.
[30]S. Hansen, J. Appl. Crystallogr. **46**, 1008 (2013).
[31]A. L. Kholodenko, J. Chem. Phys. **96**, 700 (1992).
[32]B. McCulloch, V. Ho, M. Hoarfrost, C. Stanley, C. Do, W. T. Heller, and R. A. Segalman, Macromolecules **46**, 1899 (2013).
[33]M. Rawiso, R. Duplessix, and C. Picot, Macromolecules **20**, 630 (1987).
[34]W. Burchard and K. Kajiwara, Proc. R. Soc. London, Ser. A **316**, 185 (1970).
[35]P. Sharp and V. A. Bloomfield, Biopolymers **6**, 1201 (1968).
[36]P. Debye, J. Phys. Colloid Chem. **51**, 18 (1947).