Soft Matter

Machine Learning for Phase Behavior in Active Matter Systems

Journal:	*Soft Matter*
Manuscript ID	SM-ART-02-2021-000266.R1
Article Type:	Paper
Date Submitted by the Author:	21-Jun-2021
Complete List of Authors:	Dulaney, Austin; Caltech, Division of Chemistry and Chemical Engineering Brady, John; California Institute of Technology, Chemical Engineering

SCHOLARONE™
Manuscripts

Soft Matter

ARTICLE TYPE

Cite this: DOI: 00.0000/xxxxxxxxxx

Machine Learning for Phase Behavior in Active Matter Systems

Austin R. Dulaney,* and John F. Brady

Received Date
Accepted Date

DOI: 00.0000/xxxxxxxxxx

We demonstrate that deep learning techniques can be used to predict motility-induced phase separation (MIPS) in suspensions of active Brownian particles (ABPs) by creating a notion of phase at the particle level. Using a fully connected network in conjunction with a graph neural network we use individual particle features to predict to which phase a particle belongs. From this, we are able to compute the fraction of dilute particles to determine if the system is in the homogeneous dilute, dense, or coexistence region. Our predictions are compared against the MIPS binodal computed from simulation. The strong agreement between the two suggests that machine learning provides an effective way to determine the phase behavior of ABPs and could prove useful for determining more complex phase diagrams.

1 Introduction

Since its inception, the field of active matter has been dominated by studies of motility-induced phase separation (MIPS). The majority of these studies focus on developing a theoretical framework to describe clustering behavior and the accumulation of active particles at boundaries. Due to the striking similarities between classical and active phase behavior, the creation of thermodynamic-like frameworks has been of particular interest but continues to be a source of debate.[1–5] A key difficulty surrounding this approach is the lack of a well-defined notion of temperature and free energy—as these systems are far from equilibrium—which results from the intrinsic swimming motion of active particles.

Adhering to the structure of traditional thermodynamic frameworks has resulted in several definitions of a non-equilibrium chemical potential,[6–8] each of which predicts an active binodal but fails to predict the correct coexistence pressure measured inside the phase envelope from simulation. The shortcomings with the current chemical potential definitions do not preclude its existence but necessitate alternative measures for determining the phase boundaries. Large scale computer simulations provide a means to computing system pressure, which can provide insights into the phase behavior through the mechanical instability criterion. While this method is robust and has shown great success,[2,9,10] it inherently has a steep trade-off between accuracy and computational cost. Determination of the phase boundary requires the change in system pressure with volume fraction to be zero. To make such a judgment one either needs to finely sweep volume fraction space or rely on fitting functions to smoothly fit the pressure data. Both methods are highly dependent on the quality of the pressure data obtained at each point in phase space, and large fluctuations in active pressure make this a difficult task, especially deep in the coexistence region.[2,9]

To overcome these limitations we turn towards methods used to characterize other inherently complex materials. Recently, there has been a surge of interest in using machine learning algorithms to aid in material characterization.[11–14] While early studies were predominantly interested in materials containing explicit symmetries or those confined to two-dimensional lattices,[11,12] there has been some development in classifying amorphous materials.[14]

In this study, we leverage the developments in machine learning to aid in characterizing the observed phase behavior in suspensions of active Brownian particles (ABPs). ABPs are an important minimal model system for determining the behavior of self-propelled colloids, bacteria, and other living organisms. The key feature that distinguishes an active colloid from a passive one is the driven and persistent nature of its motion. This distinct characteristic of its dynamics gives rise to a wealth of interesting behaviors including self-assembly,[15] clustering,[16,17] and motility-induced phase separation.[1–5] As such, active materials have garnered interest from the chemical and material science communities for novel drug delivery methods, remediation strategies, and material design methods at the microscale.[15,18,19]

Due to the nonequilibrium nature of these systems, it is difficult to develop analytic theories that can accurately predict the more complex collective behaviors. Thus, we look towards machine learning to aid in this endeavor, as it has been used to infer

Division of Chemistry and Chemical Engineering, California Institute of Technology. Pasadena, CA, 91125. E-mail: jfbrady@caltech.edu

dissipation from static images of active systems[20] and is of interest to the active matter community at large.[21] Machine learning algorithms are capable of discerning difficult—and potentially nonintuitive—nonlinear relationships among system variables, which would otherwise go unnoticed. These algorithms also have the benefit of readily handling multi-body correlations, which are exceptionally taxing or intractable through traditional analytic means.

Using a combination of deep learning and large-scale simulation, we focus on characterizing the phase behavior of particles in a suspension of active Brownian disks. We use machine learning to predict particle phase at a per particle level for simulations conducted at different regions in phase space. We then use these phase labels to get an estimate for the fraction of particles in each phase present at a given point in phase space. This fraction is then used to determine the system phase behavior. The manuscript is outlined as follows. In section 2.1 we define the implementation of the active Brownian particle model used in our simulations. We then outline the datasets generated for use in our machine learning model in section 2.2. Here we also discuss the feature selection used for our machine learning model. In section 2.3, we describe the machine learning model architecture used in this work and provide details on the training procedures. In section 2.4, we discuss the input features used for our model. In section 3, we discuss the representation of our simulation snapshots as graphs. In section 4, we present our results in the form of predictions of the phase behavior for suspensions of ABPs at different regions in the phase diagram. Finally, in section 5 we discuss the implications of this work and future directions.

2 Methods

2.1 Simulation Details

Suspensions of monodispersed, purely active particles are modeled using the active Brownian particle (ABP) model. The active motion is characterized by an intrinsic swim velocity $\mathbf{U_0} = U_0\mathbf{q}$—where $\mathbf{q}$ is the particle orientation—which reorients on a timescale τ_R. Particles of radius a interact through a Weeks-Chandler-Anderson (WCA) potential with cutoff radius $r_{cut} = (2a)2^{1/6}$ and depth $\varepsilon = 200F^{swim}a$, where $F^{swim} = \zeta U_0$ is the magnitude of the force resulting from the product of the translational drag ζ and swim velocity. Here we assume particles reorient via a stochastic torque $\mathbf{L}^R$ governed by zero-mean white noise statistics with variance $2\zeta_R^2\delta(t)/\tau_R$, where ζ_R is the rotational drag coefficient. Particle positions and orientations can be evolved in time using overdamped Langevin dynamics

$$0 = -\zeta\mathbf{U}_i + \mathbf{F}_i^{swim} + \sum_{i\neq j}\mathbf{F}_{ij}^P, \tag{1}$$

$$0 = -\zeta_R\Omega_i + \mathbf{L}_i^R, \tag{2}$$

where $\mathbf{F}_i^{swim} = \zeta U_0\mathbf{q}_i$ is the swim force of particle i, $\mathbf{F}_{ij}^P$ is the interparticle force between pair i,j, $\mathbf{U}_i$ is the velocity, Ω_i is the angular velocity, and ζ and ζ_R are the translational and rotational drags, respectively. The angular velocity is related to the particle orientation by $\partial\mathbf{q}_i/\partial t = \Omega_i \times \mathbf{q}_i$. Normalizing position and time by a and τ_R, respectively, results in the dimensionless reorientation Péclet number $Pe_R \equiv a/l$, which is the ratio of a particle's size to its persistence length $l = U_0\tau_R$—the distance traveled between reorientation events.[2]

We performed independent simulations of 40,000 particles for $10{,}000\tau_R$ for various combinations of the two governing nondimensional parameters: the packing fraction ϕ and Pe_R. To avoid introducing an additional force scale Pe_R was varied by changing τ_R at a fixed value of U_0. All simulations were conducted using the HOOMD-Blue software package.[22,23] Hydrodynamic interactions have been neglected.

2.2 Datasets

Our machine learning model is structured to predict phase identity at a per particle level, similar to what was done by Ha *et al.*[24] This results in a binary classification task in which particles can be members of the gas phase or the dense phase. For simplicity, we ignore the second-order hexatic transition present in two-dimensional hard disk systems and treat the hexatic phase as part of the dense phase.

We use the simulations outlined in section 2.1 to produce datasets for each point in phase space represented by a (ϕ, Pe_R) pair. For each of these phase points, we look at 6 snapshots spaced $1{,}000\tau_R$ apart from the last $5{,}000\tau_R$ timesteps. We do not use any data in the first half of the trajectory to allow the system to reach steady state. From these snapshots, we construct a feature set for each phase point which consists of 240,000 entries. Predictions of the phase behavior at each phase point are averaged across each of the 6 time points to reduce bias from a single configuration.

2.3 Learning Framework

Here we give a brief overview of neural networks and describe the architecture and training routine used in this work.

Neural networks have shown great potential for predicting particle phase for both two-state and amorphous phase-separated systems.[13,14] The most common neural network employed is the fully connected feedforward network. Feedforward networks are composed of layers of transformations modified by nonlinear functions. These layers can be stacked resulting in the output of one layer acting as the input of the following layer. The basic form for a layer f is $f(x) = g(Wx+b)$, where g is the nonlinear activation function, x is the vector input data, W is the weight matrix, and b is a vector of biases. When constructing a fully connected network the activation functions g for each layer need not be the same, and additional regularization terms can be added to prevent overfitting to training data. Some common activation functions are the sigmoid, hyperbolic tangent, and rectified linear unit (ReLU), defined as $g(x) = max(0,x)$. Once constructed, a network is given an objective, or loss, function to minimize and updates the weight and bias terms through either gradient descent† or a more sophisticated algorithm like Adam.[25] Here we

† Gradient descent is an iterative method for locating the local minima of a function by determining the steepest gradient of the function with respect to its independent variables and updating them so as to move towards the optima.

are interested in a binary classification and thus use binary cross entropy to compute loss

$$L = -(y\log(p) + (1-y)\log(1-p)), \tag{3}$$

where L is the loss, y is the binary indicator of whether the positive class is the correct label for a given observation, and p is the probability that an observation is of the positive class.

Recent advances in machine learning have resulted in the adoption of graph convolutional neural networks (GNNs), which utilize graph theory to add information on the spatial proximity of training data.[26–28] These are similar to traditional convolutional neural networks (CNNs), which rely on convolution and connected layers to make predictions. The primary uses for CNNs have been been in the areas of computer vision and natural language processing, due to the inherent structure of image and text data. Similarly, GNNs use convolutions and the inherent structure of the data, but adjacent training points need not be distributed on a rectilinear grid like an image or sequentially like in text.[28] In both architectures, the input matrix is convolved with a set of matrices—the convolution layer—to produce output matrices. These convolution layers are equivariant under translation and rotation, making them highly effective at learning abstract features of an image or graph while simultaneously reducing the number of parameters.

The amorphous configurations found in particle-based phase-separated systems can benefit from traditional CNNs,[14] but this requires spatial discretization of the system which may vary with particles of different sizes. We avoid this when looking at MIPS in active disks by using a GNN to provide information on the local structure to the network.

Our training and model architecture is as follows. We first train a supervised deep neural network (DNN) on data in the single-phase region above the critical point. After the supervised network is trained, we predict particle labels for a simulation of interest. These predictions are then taken and those that predict the phase with a >90% confidence are used as seed labels in a semi-supervised GNN. We then take the simulation snapshot and represent it as a graph, which we will discuss in more detail in section 3. We then train a GNN for each graph. The training in this step is structured as a transductive, or semi-supervised, learning problem. For each graph we use the seeded particle (node) labels to propagate labels to the remainder of the graph. We use the same features from the DNN, but instead of learning a very general problem, we are using confidently labeled particles to influence the labels given to their neighbors. In this work, we use the graph attention network (GAT) architecture[27] for our GNNs implemented using the DGL software package.[29] The resulting prediction from the GNN is then weighted against the prediction provided by the DNN in the first step.

A flowchart of our learning process is outlined in Fig. 1. The purple, orange, and teal lines of the GAT convolution represent the different attention heads for the layer. Each attention head serves as a means to create feature abstractions.[27] The coefficients $\alpha_{i,j}$ are learned weight parameters which determine the weighted importance of neighbor j on particle i. The attention

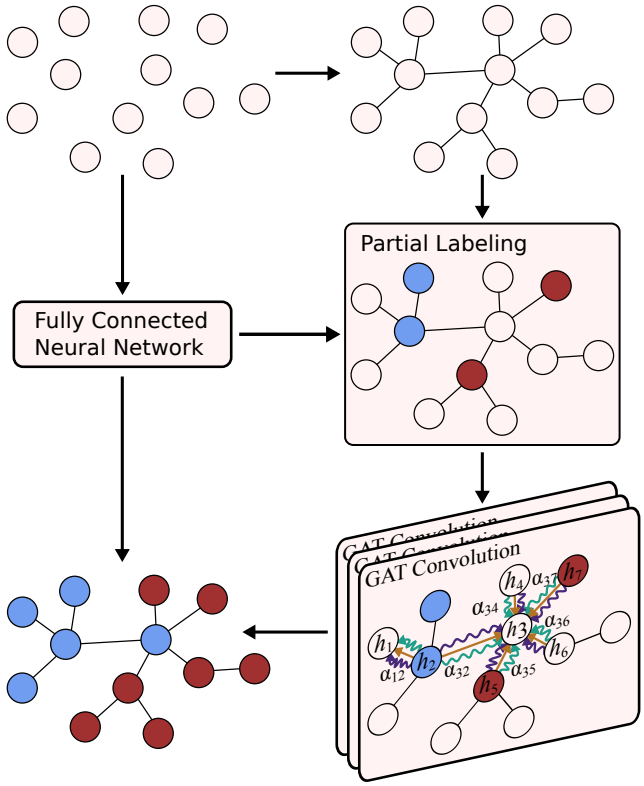

Fig. 1 The learning architecture used in this work to predict particle phase labels. First, a particle feature matrix is fed into a fully connected DNN. Simultaneously particles are connected to form a graph structure. The graph is partially labeled using the most confident (>90%) labels from the DNN and is then used with the feature matrix as inputs into a GNN consisting of three GAT convolution layers with a final softmax activation function. The resulting label probabilities from the GNN are then averaged with the label probabilities output from the DNN to achieve the final label probabilities. Each particle is then given the most probable label.

heads from each node are then concatenated or averaged to produce the layer output, which may be a label probability or feature abstraction. Details of the GAT implementation can be found in the work by Veličković *et al.*[27] Further details of the model architecture used in this work are presented in appendix B.

2.4 Feature Selection

In order to label individual particles, our feature space is limited to quantities that can be computed on a per-particle basis. This includes Voronoi volume, the number of first shell Voronoi neighbors, and the average of first shell Voronoi volumes. We repeat this averaging process for the second and third shell neighbors as well to incorporate information about the local environment. We also include the hexatic and translational order parameters defined as $\psi_6(i) = 1/n\sum_j^n e^{i6\theta_{ij}}$ and $G_6(\mathbf{r}_{ij}) = \sum_j^n \psi_6(i)\cdot\psi_6^*(j)$, respectively, where n is the number of Voronoi neighbors, $\mathbf{r}_{ij}$ is the vector connecting pair ij, θ_{ij} is the angle between $\mathbf{r}_{ij}$ and the reference vector $(0,1)$, and $\psi_6^*(i)$ is the complex conjugate of the hexatic order parameter. The Voronoi volumes and the hexatic and translational order parameters were computed using the Freud analysis software.[30] In order to account for some of the

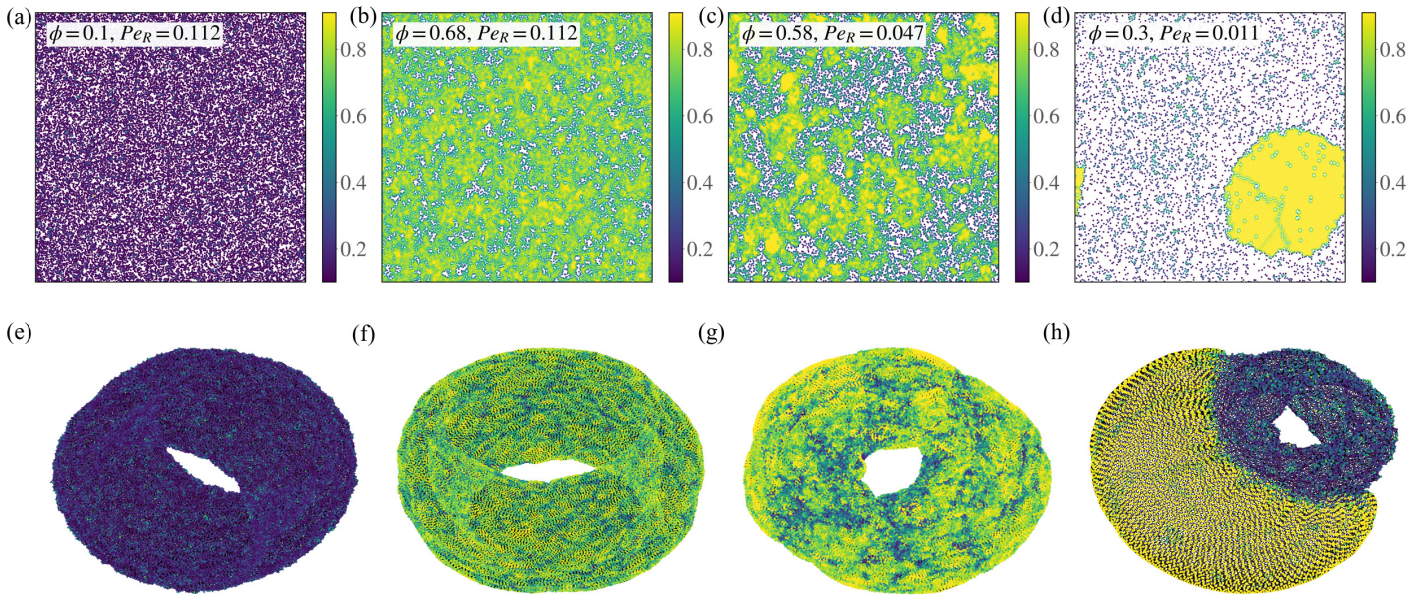

Fig. 2 Simulation snapshots and respective graph structures for different regions of phase space colored by particle Voronoi volume. We look at the weakly active ($Pe_R \sim 0.11$) (a),(e) dilute and (b),(f) dense regions, (c),(g) the region near the critical point ($Pe_R \sim 0.047$), and (d),(h) deep within the coexistence region ($Pe_R \sim 0.011$). For each simulation, the snapshot is of the entire simulation box.

dynamics we include the force-orientation autocorrelation $\mathbf{F}_i \cdot \mathbf{q}_i$ and the particle speed $\mathbf{U}_i = \mathbf{F}/\zeta$.

Our initial set of features is paired down using a boosted random forest to remove highly collinear features in order of importance. The final feature set is comprised of the Voronoi volume, number of third shell neighbors, hexatic order parameter, translational order parameter, and the force-orientation autocorrelation in order of importance. The process of removing collinear features is discussed further in appendix A and the correlation matrices for the full and final feature sets are shown in Fig. 5. It is interesting to note that the number of third shell neighbors is ranked highly in importance because the model might be learning the order-disorder hexatic transition. Lastly, we take each of our features and average them across all first shell neighbors to create an additional set of aggregate features. This aggregation step improved the performance and training stability of our DNN in the first step of our model.

3 Graph Representation

The MIPS transition is markedly similar to the liquid-vapor transition seen in traditional thermodynamic fluids with the two coexisting phases both being disordered. In thermodynamic fluids one could measure local density and use spatial density discontinuities to distinguish between the coexisting phases. However, this is difficult to do in practice as we are constrained to finite systems in simulations. Regions close to the critical point are subject to large density fluctuations which make it difficult to observe persistent macroscopic phase domains. Therefore, we need an alternative way to gather this similar type of local structure in the system. We do this by representing the system as a graph.

For each simulation snapshot, we represent the system as a graph where each particle is a node in the graph and connections are made between first shell Voronoi neighbors, resulting in a fully-connected graph. Since the boundaries are periodic, one can imagine the simulation plane being wrapped such that the top and bottom edges connect to form a cylinder and the left and right edges connect forming a three-dimensional, toroidal shape as depicted in Fig. 2(e)–(h). While the shape of the graph is three-dimensional, each node is constrained to the surface of the toroidal object.

In Fig. 2, we present simulation snapshots at different points of the phase diagram with their corresponding graph representations. Each particle and corresponding graph node are colored based on the Voronoi volume fraction of that particle. The graphs in each region of the phase diagram possess unique morphologies and characteristics. The gas phase [see Fig. 2(a),(e)] is marked by a uniform graph with a rough surface. The disorder in the phase prevents a smooth surface from forming and any structure present is short-range. If we next look at a primarily dense system [Fig. 2(b),(f)], we see that the graph representation still has bumps on the surface, but they are not as sharp. The increased system density causes jamming and reduces the magnitude of fluctuations, which results in longer-range morphological features. When we approach the critical point [see Fig. 2(c),(g)], the graph starts to form "lumps" which are connected by coarse sections of the surface. This results from mixing regions with dense and dilute phase features. We can think of the connecting coarse regions as articulation points in the graph surface, which become less pronounced with lower activity. As we go deep into the coexistence region, the dense region is made up of a single large crystal providing clear spatial domains for the two phases, as shown in Fig. 2(d). The distinctive regions manifest as a coarse mesh for the dilute phase—similar to Fig. 2(e)—and a smooth surface for the dense phase with perturbations resulting from crystalline defects [see Fig. 2(h)].

The clear distinction in graph structure in the different phase

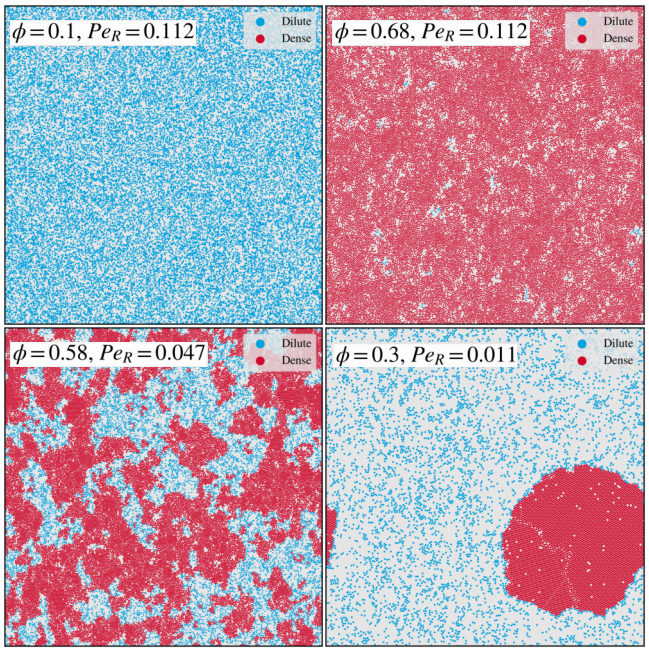

Fig. 3 Simulation snapshots for different regions of the phase diagram with particles colored based on their predicted phase. Each snapshot depicts the entire simulation box.

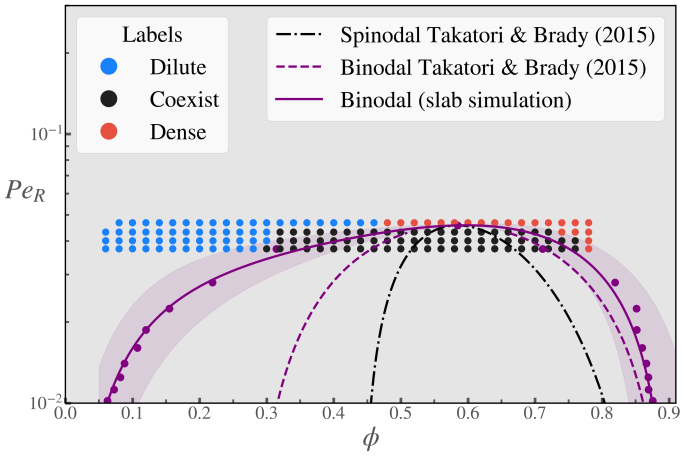

Fig. 4 The Pe_R–ϕ phase diagram for purely active Brownian particles. We show the spinodal (black dash-dotted line) and binodal (purple dashed line) predicted by Takatori and Brady[2] along with the binodal computed from slab simulations (purple points). A fourth-order polynomial fit (solid purple line) is used to give a more complete picture of the computed binodal. The shaded region represents one standard deviation above and below the predicted fitting parameters. The remaining points on the graph are colored based on their predicted region of phase space from our machine learning model. We use a cutoff of >95% gas fraction to be considered gas (blue) and <5% gas fraction to be considered in the dense phase (red). Every value for gas fraction between those values is considered within the coexistence envelope. Here we show Pe_R values in the range 0.0468–0.0374.

regions lends support for the use of graph neural networks to aid in predicting particle phase. The use of local structure also serves to help make decisions for particles near phase interfaces and regions which may be marked by large density fluctuations.

4 Results

Here we present the results from our machine learning model. Our model was trained on very dilute ($\phi < 0.2$) or very dense ($\phi > 0.7$) phase points above the critical point and deep within the coexistence region. Training points from deep within the coexistence region were labeled by inspection and particles near phase interfaces were not used for training. The model was then used to predict particle phase below the critical point ($Pe_R^{crit} \sim 0.047$). Figure 3 presents the snapshots of particles shown in Fig. 2(a)–(d) colored by their predicted phase labels. It can be seen that our model is highly capable of distinguishing particle phase in the homogeneous phase and deep within the coexistence region. The most challenging region is near the critical point as these particles are more difficult to readily distinguish from inspection.

Therefore, to evaluate performance in this region we take the predicted particle labels for each phase point and compute the fraction of dilute particles F_g present and average this across all $T = 6$ time points for a given (Pe_R,ϕ) pair. This is represented by

$$F_g = \frac{1}{T}\sum_{t=0}^{T}\left(1 - \frac{1}{N}\sum_{j}^{N} y_j(t)\right), \tag{4}$$

where y_j is the predicted label of particle j and N is the total number of particles. In our model the positive case is the dense phase ($y_j = 1$) and the null case is the dilute phase ($y_j = 0$). To account for small fluctuations in prediction we consider a point to be in the dilute region if F_g >95% and to be in the dense region if F_g <5%. Any other value of F_g is labeled as coexisting as we are only considering points below the critical point ($Pe_R^{crit} \sim 0.047$). The rationale for our choice of cutoff values for F_g is presented in appendix C.

Figure 4 presents the MIPS phase diagram with points colored based on which phase the system is predicted to be in using our machine learning model. We compare the predicted phase against the binodal predicted by Takatori and Brady[2] (purple dashed line) and treat the binodal computed from slab simulations (purple points) as the ground truth we are trying to achieve. The solid purple line is a fourth-order polynomial fit of the computed binodal. The spinodal predicted by Takatori and Brady (black dash-dotted line) is shown for completeness. We find remarkable agreement between the binodal obtained from simulations and our machine learning predictions.

Please note that in this instance we are comparing how close our predicted results are to the simulation binodal and how close the results presented by Takatori and Brady[2] are to the simulation binodal. The binodal computed from simulation can serve as a ground truth because we can measure the pressure in the system directly. Getting an accurate measure of the change in the system pressure with volume fraction becomes increasingly difficult near the critical point, which is why we use the polynomial fit to get a sense of where the binodal lies. It is clear that our new machine learning-based model much more accurately captures the dilute branch of the binodal than the analytic theory presented by Takatori and Brady.[2]

From Fig. 4 it is clear that predicting dilute particles is more challenging than predicting dense particles. We suspect this diffi-

culty arises from the large tail in the distribution of Voronoi densities for particles in the dilute phase. This ambiguity in the densities becomes increasingly prominent the closer you are to the critical point and is why one could not simply use a naive approach like selecting a cutoff for Voronoi densities to predict the phase fraction. Ha *et al.* observed a similar type of overlap for particle density distributions when studying the phase behavior of a Lennard-Jones fluid.[24]

5 Conclusions

We have created a machine learning model to predict the phase identity of individual active Brownian particles. Our results indicate that single-particle parameters are sufficient for learning particle phase when some amount of structure is included in the system. We have also shown that the MIPS phase transition can be predicted using this machine learning model. From our model optimization and feature analysis, we conclude that kinematic features—such as particle speed or the force-orientation correlation—are important for distinguishing the phases present in the MIPS transition (see appendix A and Fig. 6), unlike the traditional liquid-vapor transition present in thermodynamic fluids. The directed motion present in active systems results in a stronger separation for particle speeds and longer correlation lengths than would be seen in traditional systems when considering phase identity near the critical point. Ha *et al.* were able to successfully characterize particle phase of a Lennard-Jones fluid with high accuracy using a convolutional neural network and only three structural features,[24] but we find that our model performance steeply drops off near the critical point if we do not include at least one of the kinematic features mentioned above.

We have demonstrated that the local structure plays an important role in determining the phase behavior of active systems. Our graph representations of the system possess unique characteristics specific to their region of the phase diagram—which can be learned using a general graph neural network framework with attention. This matches the results from Swanson *et al.* and Ha *et al.* who included structure via a message-passing network and convolutional neural network, respectively, to characterize amorphous materials.[14,24] It is important to remember that while the simulation and graph representations are transferable, the graph representation creates local connections that can be used by the machine learning algorithm to improve the prediction.

We believe machine learning can be used for more challenging classification problems. It would be straightforward to extend our framework to also distinguish between the hexatic crystalline phase and the disordered dense phase to produce a more complete phase diagram. Our model is already capable of learning the importance of the third shell average Voronoi volumes, which act as a surrogate for the third peak in the radial distribution function. This peak provides a way to distinguish between liquid and solid phases. We also feel a more specific model could be created to directly predict which region of the phase diagram the system is in by performing classification at the graph-level instead of the node-level (as was done in this work). A graph-level classifier can then be readily generalized using an unsupervised learning scheme, where the model is learning distinctions between the present phases. Classification at this level can also serve to characterize micro phase separation. In our node-level classification we predict phase labels at the particle level and do not account for macroscopic spatial correlations, but our model could serve as a baseline to find regions where the dense and dilute particles exist before using a graph-level classifier to determine if micro phase separation is present or not.

The learning architecture used here should also readily generalize to active systems with thermal noise, polydispersity, or higher dimensionality. These deviations from the problem focused on in this work would result in different distributions for feature values, but should still maintain similar relationships between features. The graph network can be further extended to include edge features, which would allow for more complicated or varied interparticle interactions and should prove to be a useful tool in the characterization of other amorphous systems. Extending our current model to predict phase behavior in three dimensions can be done by taking various planes of a three-dimensional simulation and treating each as a snapshot of a two-dimensional system. Averaging over several slices of the simulation would result in the final prediction for the phase fraction of the specific phase point at that moment in time. This can then be averaged over instances in time to create a phase diagram like that presented in Fig. 4.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

A.R.D. would like to thank Yisong Yue for thoughtful discussions pertaining to graph neural networks. We gratefully thank S.A. Mallory for providing a majority of the simulation data used in this work. J.F.B. acknowledges support by the National Science Foundation under Grant No. CBET-1803662. We gratefully acknowledge the support of the NVIDIA Corporation for the donation of the Titan V GPU used to carry out this work.

A Feature Correlation and Importance

The correlation matrix for our full initial feature set is presented at the top of Fig. 5. In the figure we have used a shorthand notation where ϕ is the Voronoi volume fraction, ϕ_i is the Voronoi volume fraction averaged over the i^{th} shell neighbors, N_i is the number of neighbors in shell i, U is the particle speed, $\mathbf{F}\cdot\mathbf{q}$ is the force-orientation correlation, ψ_6 is the hexatic order parameter, and G_6 is the translational order parameter. The hexatic and translational order parameters are broken into their real part, imaginary part, magnitude, and angular components represented by $\Re(\cdot)$, $\Im(\cdot)$, $|\cdot|$, and $(\cdot)_{6,\theta}$, respectively. Features were dropped in order of the strength of the measured collinearity with other features. When considering a pair of collinear features, the feature that contributes the least to the total importance is removed.

We use a simple boosted random forest to compute feature importance. Our random forest classifier is made up of 1000 estimators, with a max decision tree depth of 8, and trained for 30 epochs with early stopping. Our boosted random forest is implemented in XGBoost. This classifier is then used to compute the

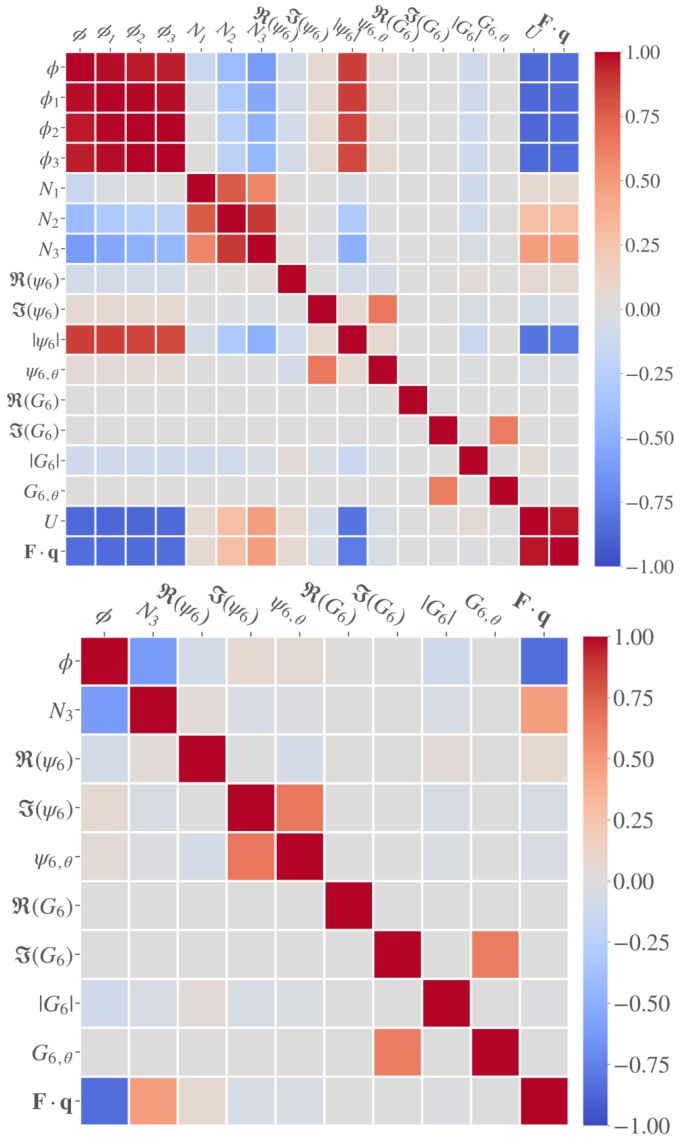

Fig. 5 The correlation matrix for the (**top**) full and (**bottom**) reduced feature sets. Strong positively (red) and negatively (blue) correlated features are removed in the reduced feature set.

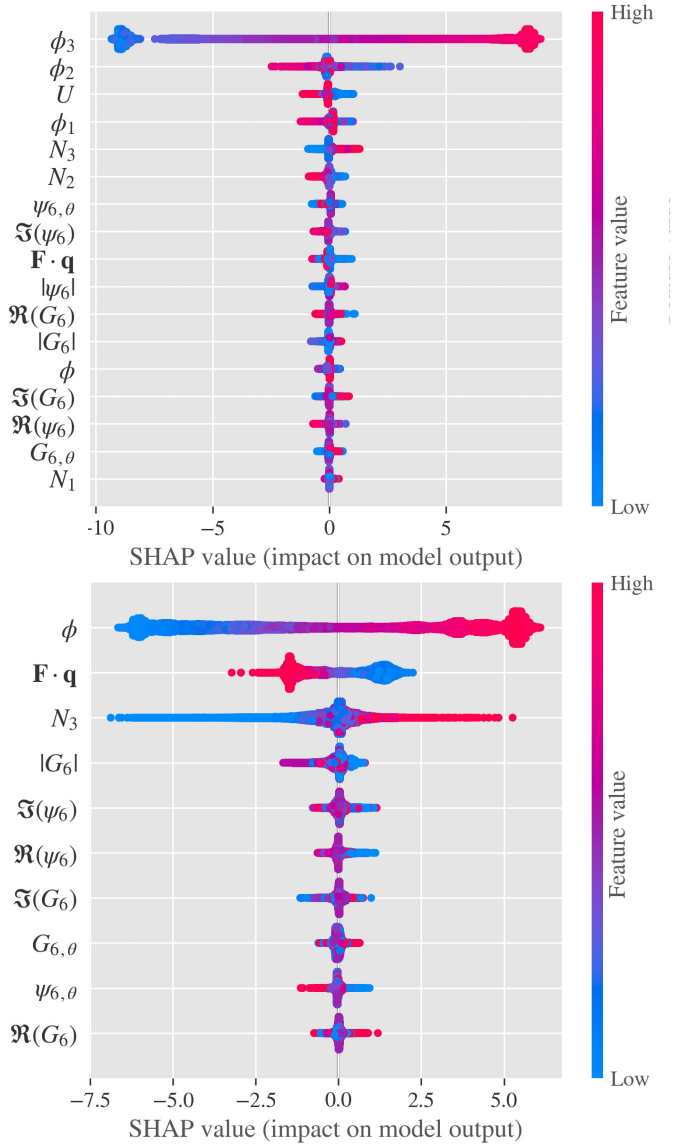

Fig. 6 Feature importance for the (**top**) full and (**bottom**) reduced feature sets computed using SHAP. The color corresponds to the magnitude of a given feature. The SHAP value presents how important a feature is at predicting the positive class.

SHAP feature importance (see Fig. 6).[31] The color in Fig. 6 indicates the value of the feature in the line, and the actual SHAP value indicates how important a feature value was for predicting the positive (dense) case. As an example, from Fig. 6(top) we see that ϕ_3 is the most indicative feature, and large values of this feature strongly indicate that the particle is dense, whereas very low values indicate that the particle in question is likely dilute. The SHAP analysis for the full feature set is not very insightful due to the presence of strong collinearity, but it can still be used to determine which feature to drop from a pair of highly collinear features. After removing a feature the importance is recalculated as this can change as the feature set changes. The final correlation matrix for the features used in this work is shown at the bottom of Fig. 5, and the final feature SHAP values are shown in the bottom of Fig. 6. There is greater diversity in the SHAP values obtained, and now the volume fraction as the most important feature, which is in line with our physical intuition.

B Model and Training Details

The trained DNN used in this work is 5 layers with batch normalization and dropout on some of the layers for regularization. The number of layers in the network, size of each layer, batch normalization, and dropout values were determined from 1,500 rounds of hyperparameter optimization with the Hyperopt package.[32] The hyperparameter optimization was performed in three stages, each of which was 500 rounds. We first optimize the learning rate to speed up future training as much as possible. The optimal learning rate $lr = 3 \times 10^{-3}$ was used for the remaining optimization rounds with a batch size of 32. The next round of optimization focuses on the number of neurons in the network, the number of layers, and the activation function used for the layer

Table 1 The specific model architecture of the trained deep neural network used for the results presented in this work

Layer	Size	Activation	Batch Norm	Dropout
1	128	ReLU	–	–
2	128	LeakyReLU [a]	True	0.69
3	128	LeakyReLU [a]	True	0.35
4	64	LeakyReLU [a]	–	0.75
5	2	SoftMax	–	–

[a] LeakyReLU activation function has negative slope $\alpha = 0.1$

Table 2 The architecture of the graph network portion of our model

Layer	Size	Activation	Attention Heads
1	8	LeakyReLU [a]	2
2	8	LeakyReLU [a]	2
3	8	LeakyReLU [a]	2
4[b]	2	SoftMax	–

[a] LeakyReLU activation function has negative slope $\alpha = 0.2$
[b] This is a fully-connected layer used to get the final prediction

(between ReLU and LeakyReLU). The final optimization round is focused on regularization and tunes the batch normalization and dropout for each layer in the network. Our final chosen parameters are presented in Table 1.

The GNN model architecture used in this work was explored manually. The graph network is intentionally kept small as this was shown by Veličković *et al.* to be effective at transductive learning.[27] The parameters of our GNN are shown in Table 2. Each layer in the network is a GAT convolution layer except for the last one, which is a fully-connected layer to give the outputs.

C Gas Fraction Cutoff Values

Here, we discuss the rationale for the gas fraction cutoff values (>95% for gas and <5% for dilute) used in Fig. 4 to determine whether a simulation has coexisting phases or homogeneous dilute or dense phase present. Since the prediction of the behavior of each phase point results from the individual particle predictions at that point, it would be intuitive to use the testing accuracy of our machine learning model to determine our cutoff values, but this requires there be ground truth labels to compare our predictions against. Fluctuations near the critical point make it impossible to confidently label particles in this region, requiring the labeled test data to come from the homogeneous phases on either side of the binodal or deep within the coexistence region where the phases are strongly separated. This results in no "difficult" testing data—as we had to provide confident ground truth labels ourselves—which results in very high test accuracy (>99%) for our supervised model. This issue is compounded by the fact that our graph neural network uses a semi-supervised learning scheme, meaning the majority of the particles have no ground truth data to compare against. We are then left with a measure of accuracy that is greater than the true accuracy.

Ultimately, we want to compare the phase behavior at a point in phase space to the binodal computed from slab simulations (purple circles in Fig. 4). Our fourth-order fit of the slab simulation data (purple line) is provided with the 95% confidence interval (purple shaded region) to create a full picture of the binodal. Since we are assuming 5% error in the binodal computed by our slab simulation data, we use the same error as a heuristic for the final cutoff values from the machine learning prediction.

Notes and references

1 Y. Fily and M. C. Marchetti, *Physical Review Letters*, 2012, **108**, 235702.
2 S. C. Takatori and J. F. Brady, *Physical Review E*, 2015, **91**, 032117.
3 D. Levis, J. Codina and I. Pagonabarraga, *Soft Matter*, 2017, **13**, 8113–8119.
4 J. U. Klamser, S. C. Kapfer and W. Krauth, *Nature Communications*, 2018, **9**, 5045.
5 A. P. Solon, J. Stenhammar, M. E. Cates, Y. Kafri and J. Tailleur, *New Journal of Physics*, 2018, **20**, 075001.
6 S. C. Takatori, W. Yan and J. F. Brady, *Physical Review Letters*, 2014, **113**, 028103.
7 S. Chakraborti, S. Mishra and P. Pradhan, *Physical Review E*, 2016, **93**, 052606.
8 S. Paliwal, J. Rodenburg, R. van Roij and M. Dijkstra, *New Journal of Physics*, 2018, **20**, 015003.
9 Y. Fily, Y. Kafri, A. P. Solon, J. Tailleur and A. Turner, *Journal of Physics A: Mathematical and Theoretical*, 2018, **51**, 044003.
10 A. Patch, D. M. Sussman, D. Yllanes and M. C. Marchetti, *Soft Matter*, 2018, **14**, 7435–7445.
11 J. Carrasquilla and R. G. Melko, *Nature Physics*, 2017, **13**, 431–434.
12 E. P. L. van Nieuwenburg, Y.-H. Liu and S. D. Huber, *Nature Physics*, 2017, **13**, 435–439.
13 P. Suchsland and S. Wessel, *Physical Review B*, 2018, **97**, 174435.
14 K. Swanson, S. Trivedi, J. Lequieu, K. Swanson and R. Kondor, *Soft Matter*, 2020, **16**, 435–446.
15 S. A. Mallory, C. Valeriani and A. Cacciuto, *Annual Review of Physical Chemistry*, 2018, **69**, 59–79.
16 J. Palacci, S. Sacanna, A. P. Steinberg, D. J. Pine and P. M. Chaikin, *Science*, 2013, **339**, 936–940.
17 C. Bechinger, R. Di Leonardo, H. Löwen, C. Reichhardt, G. Volpe and G. Volpe, *Reviews of Modern Physics*, 2016, **88**, 045006.
18 W. Gao and J. Wang, *ACS Nano*, 2014, **8**, 3170–3180.
19 S. Ebbens, *Current Opinion in Colloid & Interface Science*, 2016, **21**, 14–23.
20 L. Tociu, G. Rassolov, E. Fodor and S. Vaikuntanathan, 2020.
21 F. Cichos, K. Gustavsson, B. Mehlig and G. Volpe, *Nature Machine Intelligence*, 2020, **2**, 94–103.
22 J. A. Anderson, C. D. Lorenz and A. Travesset, *Journal of Computational Physics*, 2008, **227**, 5342–5359.
23 J. Glaser, T. D. Nguyen, J. A. Anderson, P. Lui, F. Spiga, J. A. Millan, D. C. Morse and S. C. Glotzer, *Computer Physics Communications*, 2015, **192**, 97–107.
24 M. Y. Ha, T. J. Yoon, T. Tlusty, Y. Jho and W. B. Lee, *Journal of Physical Chemistry Letters*, 2018, **9**, 1734–1738.
25 D. P. Kingma and J. L. Ba, 3rd International Conference on

Learning Representations, ICLR 2015 - Conference Track Proceedings, 2015.

26 T. N. Kipf and M. Welling, 2016.

27 P. Veličković, A. Casanova, P. Liò, G. Cucurull, A. Romero and Y. Bengio, 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings, 2018.

28 Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang and P. S. Yu, *IEEE Transactions on Neural Networks and Learning Systems*, 2021, **32**, 4–24.

29 M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, T. Xiao, T. He, G. Karypis, J. Li and Z. Zhang, 2019.

30 V. Ramasubramani, B. D. Dice, E. S. Harper, M. P. Spellings, J. A. Anderson and S. C. Glotzer, *Computer Physics Communications*, 2020, **254**, 107275.

31 S. Lundberg and S.-I. Lee, 2017, 4765–4774.

32 J. Bergstra, D. Yamins and D. D. Cox, 30th International Conference on Machine Learning, ICML 2013, 2013, pp. 115–123.