CS230

Analyzing Phase Transitions through Deep Learning

Anjli Patel
Department of Chemical Engineering
Stanford University
apatel6@stanford.edu

Stephanie Tietz
Department of Electrical Engineering
Stanford University
stietz@stanford.edu

Shaughnessy Brennan Brown
Department of Mechanical Engineering
Stanford University
sbbrown@stanford.edu

Abstract

This project classifies silicon atoms in crystalline or interface states based on geometric parameters calculated during molecular dynamics simulations. Classification is of critical importance when determining the mechanism(s) by which atoms transition between the two states. A 10 layer neural network with 100 neurons per layer showed good performance, with accuracy exceeding that of previously-implemented SVMs. The architecture and weights trained on the silicon data were applied to water data, which shares an underlying bonding structure. Initializing the neural network with weights from the silicon training and then fully retraining all layers on water data yielded the best results while saving runtime. However, retraining with frozen layers saw a steep drop in the model performance metrics. Future work will test the successful transfer learning methodology on solid materials with similar underlying structures (i.e. tin or germanium) to see if the results extend beyond those presented herein.

1 Introduction

One of the grand challenges identified by the US Department of Energy in 2018 asks "How do we design and perfect atom- and energy-efficient synthesis of revolutionary new forms of matter with tailored properties?" To address this question, a diverse community of scientists utilize molecular dynamics (MD) simulations to model complex material interactions. These programs track the motion of individual atoms in a system under a variety of thermodynamic and non-equilibrium conditions, outputting relevant structural geometry and bond information. However, in order to accurately replicate real-life interactions, MD simulations must track hundreds of thousands of atoms, each with tens or hundreds of associated parameters at each time step. This not only makes the programs extremely computationally-intensive, but also makes data synthesis and interpretation increasingly difficult.

This project focuses on one subset of materials interactions - the phase transition. During a phase transition, a material undergoes a change in its crystalline structure. Here, we analyze silicon transitioning between its ambient "crystalline" phase (labeled y=1) to an "interface phase" (labeled y=0) as it melts (Figure 1). This delineation is set purely geometrically, assigning a label based on the number and arrangement of bonds on a per-atom basis.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

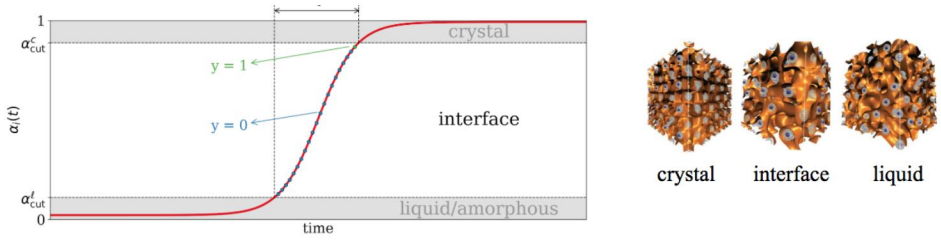

Figure 1: Definition of crystal, interface, and melt structures with labeling scheme

Using deep learning, this project seeks to assign these labels without specifying the geometric basis of classification. This enables analysis of a broader range of complex materials for which little experimental data exists. The model is trained on the results of a silicon MD simulation with geometrically-specified labels and applied to the results of a water MD simulation. Previous works employing shallow neural network models applied to phase transitions provide a starting point for designing model architecture and setting metrics for success.

2 Related work

This work was inspired by a postdoctoral scholar named Rodrigo Freitas in Evan Reed's research group who trained a support vector machine to recognize when an atom was about to crystallize using data from MD simulations. His model was trained with the goal of weeding out interface atoms from the large amount of data MD simulations yield so further study can be completed just on the interface atoms.

Outside of Rodrigo's work, a review by Suchsland et al compares various shallow neural network models to predict phases for a few simple phase transition models [2]. There have also been reports that employ neural networks to predict the melting point of ionic liquids [3], relative crystallinity of zeolites [4], and material phases from Monte Carlo simulations [5]. While the goal of this project is distinct from that of these reports, they provided guidance for choosing appropriate neural networks and descriptors to represent atomic structures despite the inability to directly compare results.

3 Dataset and Features

This project utilizes two 500,000-atom datasets from previously-run molecular dynamics (MD) simulations on silicon (Si) and water. Each atom has a defined local structure based on spherical harmonics, which relates its position and classification (crystal y=1 or interface y=0) to surrounding atoms. A family of 21 features is used based on radial structure functions. The data comes courtesy of Rodrigo Freitas, mentioned above, and each atom is labeled with the correct classification based on geometric constraints at each timestep of the MD simulation (referred to as the instantaneous dataset). This enables supervised learning. Preliminary analysis showed a significant imbalance between the label distribution with 80% "amorphous" (y=0) examples and 20% "crystalline" (y=1) examples. However, this imbalance is representative of the expected results. Consequently, the train/dev/test sets maintain this label distribution while ensuring that each set is large enough to include sufficient "crystalline" (y=1) examples to learn meaningful low-level features. Due to the size of the datasets, the train/dev/test split is approximately 90/5/5.

4 Methods

The Si and water datasets were preprocessed by normalizing the features to yield a mean of 0 and variance of 1. The datasets were randomly shuffled prior to creating the training, dev, and test sets. The primary metrics used to evaluate each model included the f1 score, precision, recall, and accuracy. A linear support vector machine (SVM) was initially developed as a baseline model, and several fully connected neural network architectures were subsequently explored, as illustrated in Figure 2. To account for the imbalance in crystalline (20% of dataset) versus amorphous samples, the crystalline

samples were weighed (w) more heavily in the binary cross-entropy loss function (equation 1) to optimize the f1 score.

$$Loss = y * -log(\hat{y}) * w + (1 - y) * -log(1 - \hat{(}y) \tag{1}$$

The rectified linear unit (ReLU) activation function was used in each hidden layer of the neural network, and the output layer used sigmoid activation. The models were trained using Adam optimization with minibatches (batch size = 1024).

5 Experiments/Results/Discussion

5.1 Model Architecture and Hyperparameter Tuning

To explore the neural network architecture, the number of layers and neurons were varied to achieve the maximum f1 score without overfitting the data. Figure 3 shows the evaluation metrics for a subset of models that were explored. The initial architecture consisted of 1 hidden layer containing 100 units, and the number of layers were incrementally increased up to 10 layers. The 10-layer network achieved high training set performance but overfitted the data, so dropout regularization was added. Additional layers did not further improve the f1 score of the model.

In addition to establishing the model architecture, hyperparameters were tuned to maximize model performance. Specifically, the loss function weights of the positive examples, alternative activation functions (leaky ReLU), dropout rate, and learning rate were varied. The positive examples were also bootstrapped to augment the training set and counteract the imbalance in the data. However, this led to overfitting the training data with limited improvement in the test performance. The f1 scores and precision vs. recall curves of the 10-layer model with dropout are shown in Figure 4.

5.2 Instantaneous data

The linear SVM on the instantaneous Si dataset resulted in the following performance metrics: precision 0.42, recall 0.80, and accuracy 74%. The 10-layer (100 neurons + dropout) neural network slightly improved the f1 precision, accuracy, and f1 score. The neural network had a runtime of approximately 20 minutes for full training on AWS.

5.3 Time-averaged data

Because the feature values fluctuate widely at each given timestep, scientists often elect to take the time average of the labels during analysis. The model architecture optimized on the instantaneous data was applied to the time-averaged data. Note that time-averaging results in significantly fewer positive examples (1.2%) compared to the instantaneous dataset. As a result, the optimal loss function weight ratio for the neural network was 25:1 for positive to negative examples.

As shown in Figure 5, the time-averaged data hit a ceiling of 0.11 on the achievable f1 score. Because the data labels were assigned on an average basis while the feature remained instantaneous, the neural network could not find a generalizable pattern comparable to that found by using instantaneous labels.

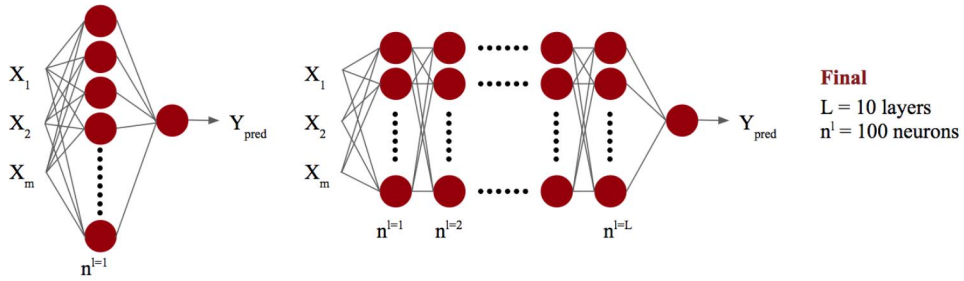

Figure 2: Beginning, iterated, and final neural network architectures

Model – Silicon, instantaneous	Loss Function Weight Ratio (y = 1 : y = 0)	Precision	Recall	f1	Accuracy
Linear SVM	1:1	0.0	0.0	--	80%
Linear SVM	3:1	0.42	0.80	0.55	74%
NN, 1 layer (100 neurons)	1:1	0.46	0.74	0.57	78%
NN, 10 layers (100 neurons)	1:1	0.47	0.70	0.56	78%
NN, 10 layers (100 neurons) + dropout	1:1	0.50	0.68	0.57	80%

Figure 3: Test set performance metrics of various architectures on instantaneous silicon data

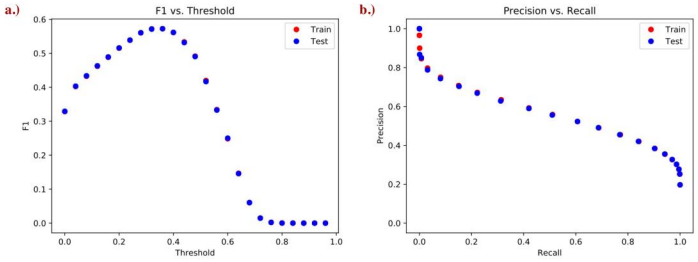

Figure 4: a.) f1 score vs. sigmoid cutoff threshold; b.) precision vs. recall curve for best-performing 10-layer/100 neuron neural network with dropout regularization

This indicates that the standard practice of utilizing time-averaged labels will constrain neural network performance. **Future experiments should exclusively use the instantaneous labeling scheme.**

5.4 Transfer learning to water

Though Si is a generally a solid at standard temperature and pressure, it shares highly-correlated bonding geometry with water, typically a liquid under the same conditions. The phase transformation mechanisms of both materials are still the subject of active research. This section explores the possibility of transferring the learning done on Si to water. Figure 6 (left) shows the similar bonding structures. All the neural networks tested during transfer learning had the same architecture as the 10-layer, 100 neurons with dropout model trained on Si, as shown in Figure 2 (right).

Model – Silicon, time-averaged	Loss Function Weight Ratio (y = 1 : y = 0)	Precision	Recall	f1	Accuracy
Linear SVM	83.3:1 (balanced)	0.03	0.86	0.06	65%
Linear SVM	25:1 (NN optimum)	0.00	0.00	--	99%
NN, 10 layers (100 neurons)	25:1	0.07	0.35	0.11	95%

Figure 5: Test set performance metrics from testing various architectures on time-averaged silicon data

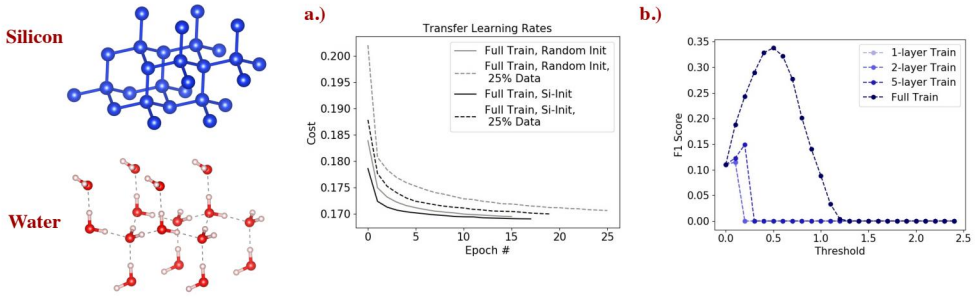

Figure 6: Structures of silicon and water showing similar bond geometry; a.) cost as a function of epoch showing faster learning rate on water data using weights initialized from silicon training; b.) f1 score vs. threshold for models trained using various numbers of frozen layers

5.4.1 All layers unfrozen

Testing the neural network on water data after unfreezing all the layers of the model pre-trained on Si is equivalent to weight initialization from Si. Figure 6.a. shows the cost as a function of epoch for several tests run with this Si-initialization and random initialization. The dotted lines indicate retraining the whole model with only 25% of the training data. The plot shows that the tests run on models fully retrained using Si-initialization had lower costs than those run with random initialization. Additionally, retraining the model on only 25% of the training data resulted in slower learning rates than the trials with 100% of the data, but the cost values equilibrated to similar values. **As a takeaway, initializing the model to Si weight can be used to speed up learning on datasets of different, new materials.**

5.4.2 N layers unfrozen

While running the model on water features after unfreezing all layer of the model pre-trained on Si results in reduced runtimes, a further reduction in training resources could come from "freezing" N model layers with Si-initialization. Figure 6.b. shows how the f1 score varies as a function of threshold on the final sigmoid output layer for Si-trained models re-trained on water data with 1, 2, and 5 layers unfrozen. The fully-retrained model (no frozen layers) clearly performs the best, and the shape of the resulting dark purple curve mirrors that observed in the initial Si-training of the model. Training with 2 layers unfrozen and 5 layers unfrozen drops the f1 score significantly across the majority of thresholds. Consequently, transfer learning does not yield the same model performance metrics with N frozen layers. **Fully-retraining a network initialized with transferred weights yields best results.**

6 Conclusion/Future Work

In conclusion, a 10-layer, 100-neuron neural network with dropout achieved the highest performance metrics (f1 score = 0.57) for the Si phase classification problem and outperformed the baseline linear SVM. The results presented herein are the result of a coarse hyperparameter search limited by the time constraints of the course. Further model iteration and tuning on the instantaneous Si data could boost the performance.

One of the primary benefits of transfer learning is the reduction in the quantity of data necessary to train the model. Our results found reasonable performance emerges when between 25 and 100 % of the data water set was applied to a model pre-trained on the Si dataset. Future work will refine this interval further to better specify the amount of data necessary in this scientific context for transfer learning. Additional directions to explore include extending the model to other material systems and incorporating time-dependent features to make time-averaged predictions.

7 Contributions

Anjli pre-processed the datasets and ran model training and hyperparameter tuning experiments on AWS. Shaughnessy worked on the initial AWS and compiled material into the milestone, final report, and poster. Stephanie developed code using Python, TensorFlow, Keras, and scikit-learn for the support vector machine as well as sections of code for the neural network and transfer learning runs.

References

[1] Fletcher, L. B., et al. "Ultrabright X-ray laser scattering for dynamic warm dense matter physics." Nature Photonics 9.4 (2015): 274.

[2] Suchsland, Philippe, and Stefan Wessel. "Parameter diagnostics of phases and phase transition learning by neural networks." Physical Review B 97.17 (2018): 174435.

[3] Valderrama, Jose O., Claudio A. Faundez, and Vilma J. Vicencio. "Artificial neural networks and the melting temperature of ionic liquids." Industrial Engineering Chemistry Research 53.25 (2014): 10504-10511.

[4] Ghanbari, Shahram, and Behzad Vaferi. "Prediction of degree of crystallinity for the LTA zeolite using artificial neural networks." Materials Science-Poland 35.3 (2017): 486-495.

[5] Carrasquilla, Juan, and Roger G. Melko. "Machine learning phases of matter." Nature Physics 13.5 (2017): 431.

Coding Language: Python Libraries used: TensorFlow, Keras, scikit-learn, numpy, pandas
Model Training and Analysis code can be found at: https://github.com/tortoisehare/cs230-project