Simulating carbon-based nanostructures with SAGESnet, a Scaling and Arbitrary Grid Encoding Simulation neural network for large-scale molecular dynamics system simulations

Jackson Meeks

Summary:

This paper proposes a novel method for predicting the nanostructure of carbon-based materials over large dimensional and temporal regimes via the use of a multi-scale neural network architecture, allowing for the generation of progressively larger scale simulations at each level up.

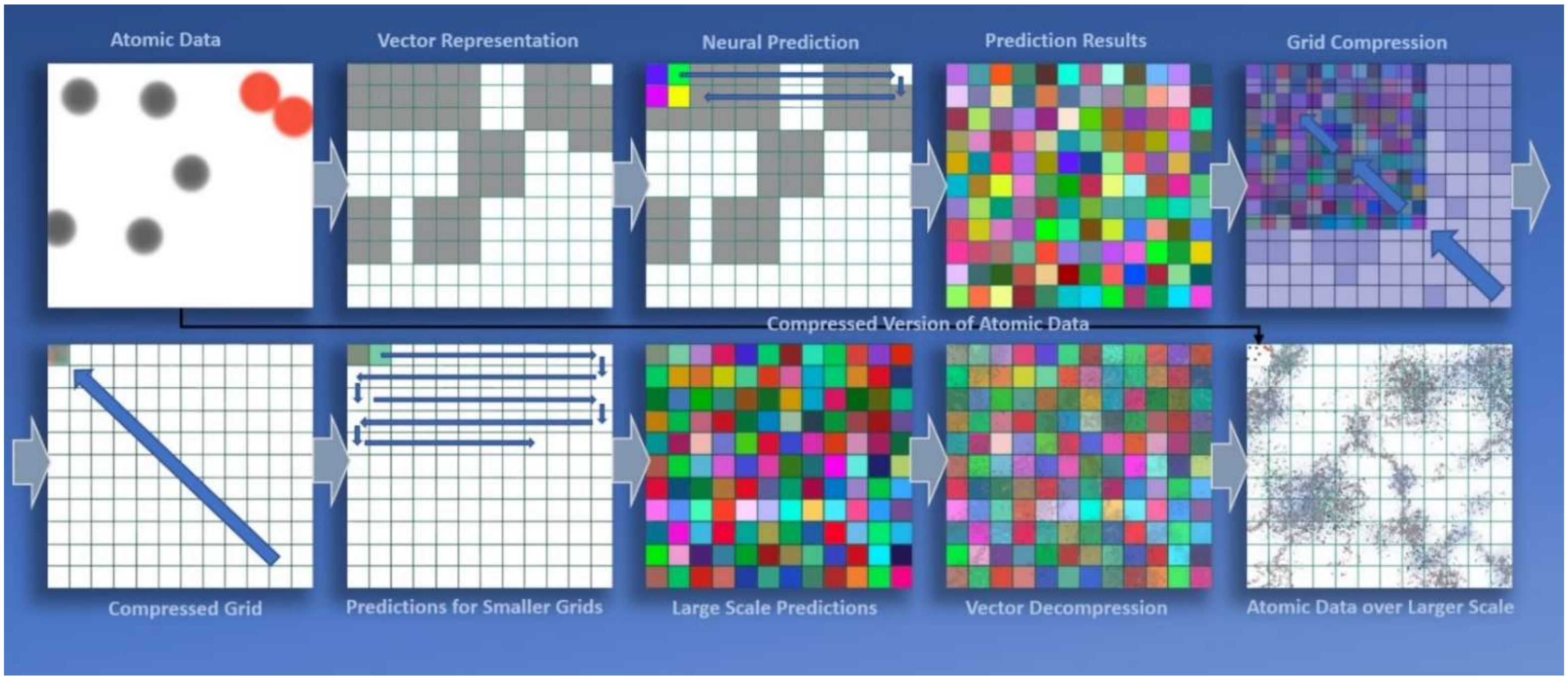

Figure 1: This figure represents the steps taken by the neural network architecture in order to perform molecular dynamics simulations. Each of these steps is described in greater detail in the Specific Architecture section of Objective 1.

Using such a method will permit the computation of large-scale simulations at previously unseen efficiency, while making relatively small sacrifices in terms of the functional predictability and reliability of these simulations. This method is based on the idea of computational reducibility (3, 4) – a concept that is employed broadly in science – that assumes that many complex systems can be reduced to simpler representations which still bestow predictability upon these systems. In order to establish effective goals for the accomplishment of this proposal, we present two overarching objectives:

1. The first objective is to build a neural network architecture that is engineered to receive atomic data in the form of atoms labeled with their element number, location in 3D space, and their trajectory vector, and to make predictions about how the locations and trajectories of these atoms will change over time in relation to one another – that is, this network takes in atomic data and trajectories, and returns simulated atomic data and trajectories for a later timestep. What is unique about this architecture is that it is set up in such a way as to allow for simulations over very large space and time scales in a computationally efficient manner. The procedures used to accomplish this efficiency will be discussed.

2. The second objective is to train and test the neural network architecture on physically accurate molecular dynamics simulations made with the high quality AIREBO potential (1), used for modelling reactive hydrocarbons in molecular dynamics simulations. Following this validation with molecular dynamics data, we will proceed to a comparison of our simulation results with real-world data, testing structural dynamics generated by our model for carbon based materials, such as graphene, carbon nanotubes, and hydrocarbons against research on the material properties of these structures documented in the literature. Following this validation against real-world data, we will discuss some ways in which neural networks can help us to better understand the laws of nature, and of how this trained model could be employed to help us accomplish this, and we will also discuss how this neural network could help us to reverse engineer materials with specific target properties that we specify.

Background and Introduction:

As mentioned, this proposal is based upon the concept of computational reducibility (3, 4). One assumption made with the notion of computational reducibility is that many complex systems can be reduced in such a way as to still provide valuable prediction of their states without needing to take every individual component into consideration. One such example of this type of reduction is Newton's laws of motion, which allow for the accurate prediction of the motion of objects without needing to predict the motion of all of the individual atoms within those objects.

Such examples fill the scientific literature. The principle of being able to reduce components of a system to a predictable formula forms one of the most basic cornerstones of physics and all the sciences. However, there are many regimes in which many of these reductional formulas have not yet been discovered. One of the regimes whose phenomena have not yet been reduced to accurate analytical formulas is that of the nanoscale. This realm of complexity and emergence exists between that of quantum mechanics with its formulas that, while not simple to solve, tend to give very accurate results, and that of Newtonian dynamics – the formulas of which have been used in computing everything from the mechanics of cars to the trajectories of spaceships.

One may consider how the equations of motion, as incredibly simple as they are, are capable of describing systems composed of trillions upon trillions of atoms. One day we may have equations that accurately and efficiently describe matter at the nanoscale, but this is not currently the case. This is unfortunate, as the nanoscale holds the keys to some of the most important questions in science – including those related to the functioning of biological cells and their molecular building blocks (2) as well as possible treatments for diseases that affect the human body at this scale. Discovering general formulas capable of efficiently predicting nanoscale phenomena could signal the beginning of a significant leap forwards in our understanding of biology and other sciences, as well as the beginning of a significant leap forwards in our ability to manipulate matter. Currently, however, we are often limited to using computationally intensive numerical simulations to make accurate predictions in this realm.

There are various traditional methods used by scientists in an attempt to predict the behavior of matter at this scale, which methods are generally classified by the time and space scale at which they perform their simulations. These methods include ab-initio, density-functional theory (DFT), molecular dynamics, coarse-grained molecular dynamics and continuum simulations (5). Each of these methods offers different strengths and weaknesses. Ab-initio simulations, based on the Schrodinger equation, while being extremely accurate are also extremely computationally intensive, scaling at the roughly the value of N^4, where N is the number of bodies (such as electrons) in the system and, thus, their simulation scale is very limited. DFT is slightly less accurate than ab-initio simulations but can simulate larger systems. It is still, however, limited in the number of atoms it can simulate.

Molecular dynamics allows for relatively large-scale simulations of tens of thousands to even billions of atoms when supercomputers are employed (6). Molecular dynamics (MD), as opposed to ab-initio simulations, performs simulations that are largely based off of Newtons equations for motion, while employing interatomic potentials (that can often be based off of ab-initio simulations) that attempt to describe the interactions between atoms and/or molecules based on their field potentials. MD uses various algorithms for the numerical integration of Newton's equations of motion that attempt to more accurately describe the interactions between the particles in a given simulation while also minimizing the error and computational workload (16). For this proposal, we will be using data generated by molecular dynamics due to its ability to compute large scale simulations at a relatively low computational cost compare to ab-initio methods. The MD scale for training data is more computationally approachable in terms of the objective of this project for performing larger nanoscale simulations we are proposing to accomplish with our neural network model. Based on the success of this approach, our model should easily adapt to ab-initio training data, due to its flexible scaling features. We will go into further discussion on the usage of molecular dynamics in our proposal.

Other methods used to simulate materials at an even larger scale can be used, but tradeoffs are often made as scientists attempt to simulate larger and larger scale systems with limited computational power.

One approach that seeks to address some of the computational limits of other forms of simulation is multiscale modelling. Multiscale modelling seeks to simulate materials over multiple time and space scales, attempting to retain the accuracy of smaller scale models while maximizing the system size capable of being simulated. Multiple variations of this technique have been reported in the literature (7, 9). Some of the more recent attempts have involved the use of machine learning to help increase computational efficiency, while maintaining accuracy. This proposal seeks to build on some of this work in machine learning for multiscale system modelling.

Other studies have been performed that seek to reduce the computational overhead of simulating atomic trajectories and interactions via the help of neural networks. Neural networks are a computational framework based loosely on the principles by which biological neurons operate. A biological neuron is capable of integrating complex signals from multiple sources into meaningful representations. One proposed method for how biological neurons operate is called Hebbian learning (13). Hebbian learning has often been explained as the principle that "neurons that fire together, wire together." What this implies is that connected neurons that fire in quick succession to one another will have their connection strengthened, while those that fire out of sync with one another will have their connection weakened. Much work has been done to try and better understand this process, whose explanation has been vastly simplified in this work (14).

Artificial neural networks attempt to apply the principles of Hebbian learning in a more systematic and mathematical function. They consist of "neurons" that receive weighted inputs from multiple other neurons. The neuron sums the weighted input from other neurons

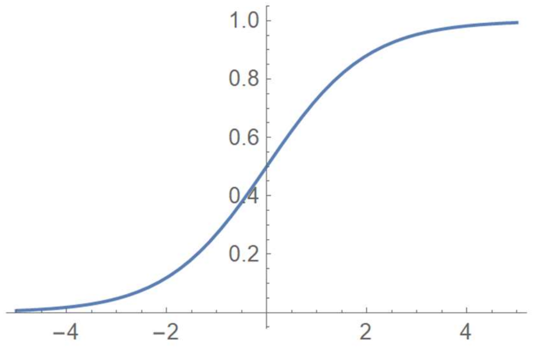

Figure 2: This image represents a sigmoid function - one that is often used to introduce nonlinearities into neural networks.

and performs a nonlinear operation on this sum, where the nonlinear operation can be something as simple as passing this sum through a sigmoid function. A neural network consists of multiple "layers" of neurons each connected sequentially with latter layers. Information (often encoded in the form of a vector) that is assumed to be useful in making a desired prediction is fed into the neurons of the first layer of the neural network and is "fed forward" (15), layer by layer (where each neuron in these layers performs the operation of summation and nonlinearization previously described), until the now transformed information reaches the output layer of the network. The output layer of the neural network is special in that the information sent out by its neurons is treated as a prediction and compared to a prediction output provided by the user, which shows the network what prediction it *should* have made. A training process ensues that iteratively helps the whole neural network to learn to relate its input information to a desired output, by adjusting the connection weights between individual neurons of different layers. This can just be thought of as a tool for function approximation for functions whose analytical representation is either unknown or too difficult to compute efficiently without further representational reduction. Neural networks have been used for various classification and regression tasks – from image captioning and classification to housing price predictions.

As mentioned, neural networks are receiving increased attention in the chemistry and materials science communities, due to their ability to efficiently represent complex processes and relationships Some of these attempts include using a neural networks to act as efficient translators for information transfer between finite molecular data and continuum simulations (8), as well as many works that have attempted to use neural networks for the purpose of learning interatomic potentials based off of ab-initio data (9), that can then be used in molecular dynamics simulations. Two models worth particular mention are deep tensor neural networks (10), and VAMPnets (11).

Deep tensor neural networks are network architectures that use multiscale features of molecular data in order to perform chemical calculations on these molecules, including calculating information about their trajectories as well as the energies present in the system. This type of network has led to state-of-the-art predictions of molecular properties at a near ab-initio level of accuracy. The architecture of this neural network is related to convolutional neural networks that we will further discuss.

VAMPnets are neural networks that take atomic data in the form of molecular configurations and classify this data into informational kinetic models representing the given data. What is unique about this approach is that this neural network architecture attempts to take a traditional pipeline for developing Markov states to describe chemical data, that usually require a lot of expertise on the part of the individual designing the model in order to avoid large prediction errors, and automate this arduous pipeline with one neural network architecture that attempts to encode the same information based on machine learning statistics – thus, removing some of the potential for human error. The Markov states generated by this network architecture are comparable and, sometimes, superior to those generated by the hand-crafted pipelines' results.

These studies listed have provided incredible breakthroughs in data science for chemical predictions. This current work proposes to take some of the concepts employed in these studies and combine them into a more streamlined end-to-end prediction network for small to very large scale molecular dynamics simulations. Having such a streamlined model that connects a lot of the previously successful advances in machine learning for chemical predictions is something that is currently being heavily researched (12).

Among the advances incorporated into our model, there are two that stand out. The first is that this structure allows for simulation over arbitrary grid sizes determined by the user – meaning that this method can simulate either small or large scale molecular dynamics atomic trajectories, as opposed to many other types of neural networks that only take fixed input sizes. The second, strongly related to the first, is that this network is capable of simulating over very large space and time regimes, due to its multiscale nature and the use of neural network autoencoders that help to provide compressed and informationally dense representations of atomic data over multiple timesteps, allowing for meaningful information transfer between multiple simulation scales. Autoencoders will also be further discussed in the Proposed Work section. This network's predictive abilities are also due to the use of convolutional neural network operations that help to minimize the amount of training data needed to generate accurate predictions by reducing the number of parameters of the neural network (NN) architecture. Having such a model that combines all of these previously used features, as well as some new ones, will allow for the simulation of larger scale systems than were previously computationally accessible, as well as allow those with smaller computational resources to be capable of performing large scale simulations, from which they were previously excluded. To the author's knowledge, no other NN models have attempted to combine all of these computational architectures into a single neural network for the purpose of simulating molecular dynamics.

Proposed Work

Objective 1:

General Overview: The purpose of the first objective is to define a neural network structure that is specifically suited to interacting with and predicting 3-dimensional chemical data and its trajectories across multiple time and space scales. This model seeks to accomplish functional accuracy – meaning that some of the more specific chemical information will be lost as information is passed up to larger space and time scales, but the emergent properties and structures based on the underpinning chemical makeup of the system will be maintained. Effectively, this model should be able to predict how a material functions at large scales based off of the interplay between its lower level components. It will not, however, have the goal of being able to perfectly recreate exact atomic trajectories at smaller scales from larger scale data. An analogy for this principle would be comparing the differences between solutions in two different beakers containing the same chemical solutes. Looking at the solutions and measuring their macroscopic properties, these two systems will appear very similar. However, if we were to examine these solutions much more closely, we would be able to see that the individual positions of their atoms are very different from one another. Thus, when we state that we want our model to be functionally accurate, we mean that we want our model to be capable of predicting the emergent properties of materials at different scales, without necessarily being capable of perfectly recreating conditions at smaller scales. This represents a more probabilistic approach to modelling the chemical and material properties of a system at various scales.

This method employs a neural network that takes in atomic coordinates and trajectories and uses this information to create large scale chemical and structural simulations with the help of an autoencoder neural network structure that compresses data to smaller and denser representations than the original input (17, 18). The neural architecture proposed in this paper is composed of multiple smaller scale neural networks working in conjunction with one another, each making predictions at larger time and dimensional scales.

In order to accomplish this, we will build a model that is capable of:

1. Receiving chemical data in the form of atoms labeled with their element and initial trajectory values, as well as their positions in 3D space.
2. Converting this chemical information into a form that is readable by the neural network.
3. Transferring data, in an automated fashion, between multiple time and space scales.
4. Performing functionally accurate material simulations on this data, including simulating the mass trajectories and material properties, at these various scales.

With these properties in mind, we propose SAGESnet, a Scaling and Arbitrary Grid Encoding Simulation neural network for large scale molecular dynamics system simulations, whose specific architecture will be described, below. Before proceeding to the specific architecture of this network, we first have to have a basic understanding of a couple of concepts used in neural networks. The first of these concepts that will be given an important overview is the autoencoder neural network.

An autoencoder neural network is a neural network that, through its training process, learns dense fixed length representations of higher dimensional data. So, for example, some autoencoder neural networks are trained on images of faces, where the input image could be 500*500 pixels, giving an input vector to the autoencoder network of a length of 250,000 numbers. This is obviously a lot of data that the neural network needs to process – and this is only considering grayscale images. This neural network, however, may learn a way to compress this image data into a shorter length vector representation (maybe to a vector of a length of 1000 scalars, for example). The way this network accomplishes this is a function of its structure and training. The architecture of this network has a bottleneck in the center of the network. So, for example, the layers of the network may have the dimensions, layer by layer, of 10,000, 5,000 and 1,000 neurons, followed by layers of 5,000, and 10,000 neurons, respectively. Then, the training process for this network is simple. The input given to the network is a picture of one of the faces, and the intended output from the network is also a face – the exact same face as was given for the input. In this manner, this network is given a large dimensional vector representing an individual's face, and must find a way to compress that information efficiently enough to be able to pass this information through the middle layer as a vector of the length of only 1,000 scalars, and then use this length 1,000 vector to recreate the original image as faithfully as possible, in order to minimize the penalty it receives when it does not give a prediction that is close in features to the original image. For this specific example of training an autoencoder network to learn facial representations, the autoencoder will be forced to be as efficient as possible and will, over time, learn similarities between different faces that tend to repeat, and will store information about these repetitive features within its weights. Once these representations are learned, we can then chop the neural network in half, so as to obtain the length 1,000 vectoral representation of faces instead of images of a whole face, and we could then perform further compression on this representation (with PCA, for example, or other common dimension reduction methods), to use this representation for clustering images of people with similar faces together. This is useful for chemical data because, thankfully, there are chemical arrangements that tend to repeat on a normal basis. For example, oxygen tends to bind well to hydrogens to form a molecule that is relatively stable and constant. We can utilize autoencoders to represent often recurring spatial and temporal chemical patterns, allowing us to have lower dimensional representations of the emergent chemical and structural properties of grouped elements in a system.

As humans, we also perform similar processes in language. We learn the definitions for new words that contain more information per character about the same concept than the original definition, in an effort to communicate more efficiently, and would tend to spare ourselves of learning words that are rarely used, instead focusing on those that are most useful in our day to day communications with others.

This discussion on autoencoders brings up one of the major scientific advantages of this proposal – as autoencoders can be used to cluster images of similar human faces together, the autoencoders in this network architecture may also be used to classify often recurring chemical patterns. This could help us to get a better understanding of the reaction kinetics of our chemical system, as well as material properties – such as what elements tend to clump together. And, because the architecture described in this proposal is for the purpose of multiscale modelling over various space and time regimes, this means that we could utilize the autoencoders at each simulation scale in our network to classify chemical phenomena at greater time and space scales, allowing for researchers to have a better method for classifying the chemical systems with which they are working. This principle of using neural networks for understanding physical principles will be further discussed in the second aim. We will see that neural networks are not only useful as "tools" for making predictions – they can also help us to gain new understanding of physical processes with which we are not yet as familiar, and help us to find patterns in scientific data and make discoveries that would have otherwise taken a much longer time to discover.

For the specific architecture of our neural network, autoencoders are used to transfer chemical information from one layer – one simulation scale – to the next. This allows our multiscale NN to devote the most computational resources to the patterns that occur most often.

One other subject that must be discussed, briefly, before proceeding to a specific overview of the neural architecture we use for our model, is related to convolutional neural network layers.

Convolutional NNs, inspired by processes in the human visual cortex, are, in general terms, small scale neural networks that range over data presented to them, piece by piece, and perform computations on that data, often performing filtering on the data in such a way as to augment certain characteristic or repeating features in the data (19).

One important aspect about convolutional neural networks (CNN) is that, unlike more traditional neural networks, they have spatial awareness encoded into them. This means that a CNN can detect when certain features are in close proximity to one another, which is incredibly useful for image detection or, more importantly for the field of nanoscience, recognizing relations in 3-dimensional chemical data. For these purposes, CNN based machine learning has been researched intensively for its utility in making chemical predictions, as well as for predicting or discovering nanoscale properties and structures (20, 21).

In the context of images, convolutional kernels scan small filtering windows over an image, multiplying these portions by an image filter that has a certain pattern that matches most strongly only with certain features. For a simplified example, the filter might be in the shape of an eye and, if this convolutional kernel with an eye shaped filter scans over a segment of an image where an eye is present, that filter will return a higher output value than otherwise – it will send out a stronger signal in response to a feature that matches its filter. SAGESnet uses a 3-dimensional variant of a convolutional kernel to raster over chemical data present in our simulation grid to aid in the prediction of how atomic trajectories and material properties change over time as a function of their location in the grid.

We now have an adequate understanding of some of the most crucial neural network architectures employed in our model. We will now discuss the specific architecture of our model that makes these chemical predictions over a large space and time scale possible.

Specific Architecture: In SAGESnet, the first step of our prediction process is the conversion of a single molecular dynamics timestep of atomic data into a form that is more easily handled by neural networks. This occurs when atoms with their informational labels (including trajectory, locational, and elemental information) are placed in a quantized 3-dimensional grid as a preparatory step prior to being converted by a low level autoencoder network into a format readable by the CNN based prediction network. This conversion step takes the atomic data at each point within the 3D (human readable) grid and converts it into a machine-readable form (that is not necessarily readable by humans, but is informationally useful to the neural networks), *placing it into a new 3-dimensional grid that we will call the simulation grid*. At this step, all of the atomic data vectors are converted to vectors that are not necessarily the same length as the original human readable atomic labels, but whose length will be a hyperparameter – a user defined variable that is tuned by the individual performing the NN network training (an example of a hyperparameter would be the number of neurons in a neural network layer, or the scale at which we inform the program to perform the simulation – neither of these examples are learned during training but, rather, are specified by the individual training the neural network).

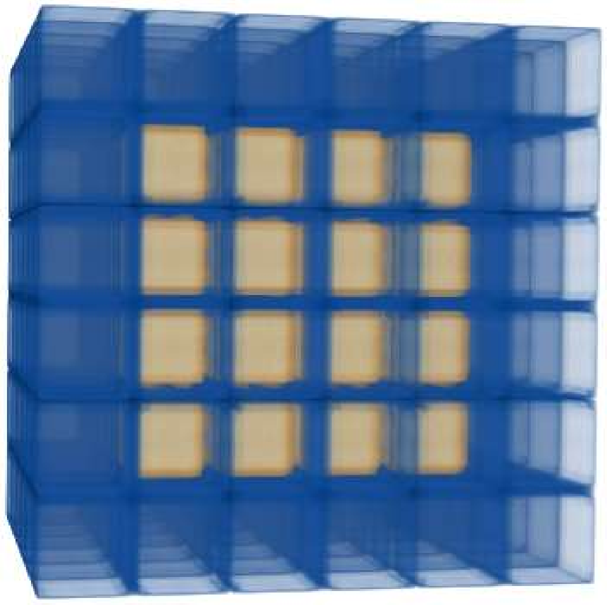
Figure 3: This represents the setup of a 3-dimensional grid in this architecture, where the yellow cubes are the cells, and the blue cubes are the virtual cells, used to transfer information to other grids.

This transformation of data between the human readable and machine readable grid is much like how a translator helps people who speak different languages to communicate – and is a result of the network training. The exact transformation previously performed on this data and the form of its current representation is determined by the neural network training process performed on molecular dynamics data. As we learned about autoencoders, this data is highly representational of information pertinent to the simulation. What matters is that this data representation can be converted back into coordinate and trajectory data again, and that it is trained to represent atomic coordinates, element types, and trajectories in such a way that maximizes the ability of the CNN layer to perform accurate predictions.

After the atomic information is converted by the autoencoding network and is distilled in the simulation grid one step must be performed before predictions are made via the CNN network. This step allows for greater control of the system environment in the simulation. The way this is accomplished is via a blanket input vector that is mapped over all cell vectors via a simple convolutional kernel. The purpose of this operation is to map information about environmental factors, such as the temperature and pressure of the system or information about electromagnetic fields present in the system, into each cell location. This parameter will also allow the user to specify the presence of force vectors in locations of the 3D cellular grid. Such parameters – especially for temperature and pressure – are available, as a general rule, in molecular dynamics software to allow for simulations in different environmental conditions, and these considerations must also be taken into account by this simulation model.

The CNN prediction network can then proceed with predicting timestep t+1, from the data present in the current timestep t (the current information stored in the simulation grid). The actual shape of the CNN is a hyperparameter specified by the user, but the general principle of its operation is that it takes input from a center cell, performs a transformation on the information from that cell, and maps this transformation onto surrounding cells, taking into account the positions of each cell relative to the center cell. This procedure emulates a force field type of procedure, where the center cell causes the field, and the effects of this field are summed into the surrounding cells. The transformation performed by the CNN is determined the training of the neural network, and this transformation is a function of both the values within the center cell, as well as the relative position of the output cell with regards to the center cell.

The goal of this CNN is to predict the next molecular dynamics timestep. The CNN rasterizes over all cells in the network, performing the transformation described previously. After the CNN is finished scanning over the 3D grid, all the cells in the grid will have received multiple summation inputs resulting from the CNN interacting with all of their neighboring cells and sending the results of those interactions into the considered cell. One could imagine this process as the noise levels in an apartment – the noise level in one apartment is a function of the noise level of all the surrounding apartments, transformed by the sound vibrations' interactions with the walls before entering into the apartment in question. It is important to keep in mind that values sent out from the CNN with each interaction may also be negative. Another important aspect of the output from the CNN with each grid interaction is that its input is the machine-readable vector representation of a cell, and its output is a vector of the same length. Therefore, the state of each cell (any of the numbers within the cell's vector representation) can be completely altered by the output from the CNN transformation sent into that cell. This is how the states of cells in the grid are altered.

The way the accuracy of this prediction is assessed is important. First, the autoencoder network must decode the prediction made by the CNN back to a human readable form. Then these predictions for the next MD timestep are compared with the actual next MD timestep provided by the training data. This includes comparison between the actual and predicted atomic coordinates, trajectories, and element types. This comparison constitutes the *loss* of the NN prediction. The lower the loss, the better the prediction of the network – that is, the closer it was to the actual simulated molecular dynamics data. The loss will be computed by the MSE (mean squared error) loss function. This loss function is useful for regression type prediction problems, such as predicting MD trajectory data. It is commonly used in the literature. The equation for the MSE loss is:

$$\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$$

Where *n* is the number of samples, Y_i is the actual target value and $\hat{Y}_i$ is the predicted value. The utility of the MSE loss function is that it penalizes predicted values that are further from their target values much more so than more accurate predictions, helping to accelerate the training process and prevent the network from being over adjusted even when its predictions are relatively close to the target value.

Another very important part of this prediction process is that cells along the outer layer of the grid send output outside the scope of the grid. This is the mechanism that allows the user to specify arbitrary grid sizes, because larger scale grids have a means for communicating with one another. The simulation process just described can be repeated as many times as specified to simulate multiple timesteps.

The last network that can act on this simulation grid is a temporal and spatial scale transforming autoencoder and is the key to the ability of this model to perform large-scale simulations. This autoencoder is trained to be able to convert multiple cells in a grid structure over multiple timesteps into a single cell representation that can be used in a larger space and time scale grid. The training process for this conversion is similar to the one performed when training an autoencoder on image data – the user assigns a hyperparameter for the number of timesteps and cells they want to be represented in a larger scale cell, and trains the autoencoder to pass this information as optimally as possible through a bottleneck layer of the same dimensions as the larger scale representational cell, while attempting to reconstruct the input data in the output prediction.

The encoder is then chopped in half, and the output of the middle layer is used to represent the larger scale cell. This larger scale representational cell is represented by a vector of the same length as the lengths of the vectors representing smaller cells. Thus, the computational intensity of simulating larger time and space scales is equal to that of simulating smaller layers. Therefore, the training of this autoencoder network, that also rasterizes over the entire smaller scale grid (this time, however, skipping steps so every cell from the smaller scale grid is only contained in a single larger scale cell) is incredibly important to the multiscale simulation ability of this proposed network. Extra measures will be taken to ensure the best training of this aspect of the network, including extra training on data that is not as well represented by this autoencoding operation. This autoencoder will also, afterwards, be trained in conjunction with the larger scale prediction layer (where the MSE has already been used in the autoencoder training and is used in this training process as well) in order to tune it not only for representing data at lower levels, but also for providing data that is useful for prediction purposes to the larger scale grid.

As mentioned, this autoencoding step is the feature of our model that allows for large scale simulations over space and time. This is possible because this network has learned, through its training process, to only pass the most vital data from smaller scale simulations up to the larger simulation grids. Using these network architectures in conjunction – one for prediction and the other for data compression and decompression – allows for simulations over very large scales. This process can be repeated over multiple scales to simulate very large temporal and spatial regimes.

Feasibility: Many other researchers have addressed parts of what this work proposes. For example, researchers at Argonne National Laboratory have used machine learning to develop intermolecular potentials for simulating water (9), and other researchers have used neural networks as interfaces for passing information between small (molecular dynamics) and larger scale (continuum) simulations between their respective boundaries (8), while other researchers have even used autoencoders to aid in atomic level simulations over large time scales (26). All of these efforts have aided in the computational efficiency of the simulation models, allowing for much larger scale models with very little trade-off in terms of accuracy. This model expands upon some of these accomplishments by allowing for automated simulation scaling over multiple levels.

Potential Challenges and Alternative Approaches: The first challenge is related to data acquisition. Neural networks need a lot of data in order to provide accurate predictions, and the amount of data necessary scales exponentially with the number of parameters in the neural network, if one wants to prevent a neural network from overfitting – meaning it only learns the data it is presented but doesn't generalize will to unseen data. For these reasons, we have decided to use molecular dynamics data for our training process. Molecular dynamics data is computationally intensive to generate, but not so much as ab-initio atomic simulations, which is much more accurate. Our hopes are that, if our model can perform well on molecular dynamics data, this model could also be trained on ab-initio data in future works, as the structure of the network is relatively malleable to different scales of chemical data. Another effort that we will make to reduce the cost of training SAGESnet is to first test the ability of this network to predict cellular automata that, like atoms, can exhibit emergent behaviors at larger space and time scales, in an effort to focus first on optimizing the structure of the network (such as the hyperparameters). Cellular automata data is cheap to generate and, therefore, could prove very valuable in helping to validate the concept of taking advantage of computational reducibility with the help of neural networks, as well as provide an insight into our network architecture's ability to perform these reductions

With this reduction to molecular dynamics data, as opposed to using ab-initio data, another concern with this model is how valid its predictions will be in real world scenarios. In an effort to mitigate some of the disadvantage of using MD data, we have chosen to use the high quality AIREBO potential, which isn't the most efficient molecular dynamics potential, but provides accurate representations of hydrocarbon simulations, as it is fitted to ab-initio generated data for hydrocarbons. This potential also takes bond formation and breaking into account, so our model will still be able to model some aspects of chemical reactivity, and mechanical responses to large forces.

As mentioned, because we have chosen to work with MD data, we have chosen to use the AIREBO potential. This should suit our needs, as we have also chosen to limit the types of conditions that our model can simulate, restricting our current simulations to hydrocarbons and carbon-based structures, to be more specific to the experiments we are performing.

Another technique to help decrease the amount of data we will need is to train the lowest level of our network until it achieves a very high prediction accuracy in our simulations, and then use this lower level network to generate more training data in a more computationally efficient manner, which can then be used for training the larger scale neural networks.

Another challenge this work presents is related to some of the technical aspects of neural network architecture and training. Neural networks use a type of programming methodology called "fuzzy logic", meaning they are more general than the traditional form of programming. Because of this they are better at working with more abstract data – such as photographs, for example – but this also means that they are not as proficient at information storage, and this means that it could be possible for this neural network model to "lose" atoms, or the mass could change slightly over the course of the simulations. This is obviously a problem. A solution we propose to this possibility is to record the amount of mass at the beginning of the simulation (or the number of each atom type) and scale the model output at each round of training to maintain this number constant. This can also help us to ensure that the distribution of atoms of each type does not change over the course of the simulation. Something related to this challenge is the limited scope of influence that every cell in the grid has over its neighbors. Due to computational limits, we cannot have every cell in a grid fully aware of (connected to) every other cell in the grid, and must limit the distance of influence that each cell has over another grid cell. One method that could help to overcome this challenge is by implementing a feedback loop between scales in the network where, instead of simulating just one scale at a time, we could simulate multiple scales and have larger scales feed information from their simulations down to lower levels, thus increasing the scope of "awareness" each cell has to its neighbors.

Another step we are taking, not explained fully in this proposal for the sake of space, is to use yet another neural network that has the sole task of viewing the inputs to every single cell and predicting the type of loss their prediction will incur. By this method, we can get a real time prediction, for every cell, of the likely confidence bands for the prediction given by the cell in question. This could be used, in future works, to create a network model that scales certain collections of cells to a scale at which they are capable of making the highest accuracy prediction. This assumes that some space and time scales may provide better or worse predictability for certain phenomena (such as the idea that the flow of water at a larger scale is less challenging than predicting the movement of individual water molecules within that flow of water). A similar network will also be trained to help ensure the accuracy of compressions performed by the autoencoder network. We could also use the concept of a "sensitivity grid" that probes a simulation grid, making slight alterations to different values within cells and recording the overall effect on the system from these small changes. This is basically akin to computing the derivative of system volatility as a function of small changes to cell values. Using this grid, we could set up mechanisms to force the architecture to pay extra attention to these larger impact cells and their values.

There are likely other challenges that this work may present, and these challenges can be addressed as more time and writing space present themselves.

Objective 2:

General Overview: Designing and describing a neural network model that is capable of simulating atomic data is an incredibly challenging task. Attempting to concisely describe all of the ins and outs of that network design may be just as challenging. For the sake of documenting what is original about the SAGESnet architecture, as well as to provide enough working knowledge to a multidisciplinary audience in an effort to make this proposal (hopefully) accessible, a large portion of this proposal has been devoted to a description of the first objective. To support this decision, the author argues that neural networks, themselves, can act as sources for scientific knowledge and understanding, once one is able to decipher their properties. For a long time, neural networks have been thought of as black boxes – but there has been a recent trend among physicists and other scientists to break open neural networks in an effort to understand how these machines are capable of solving some incredibly complex problems in chemistry and the physical sciences (22). The thought is that by better understanding the inner workings of these systems, we may be able to glean insight out of them that helps us to understand more about the inner workings of the laws of nature. First, however, before we discuss the types of insights that we may gain from opening up the wiring of SAGESnet, we must train it, and validate its performance against other similar attempts.

As previously mentioned, this model will be trained on simulation data derived from molecular dynamics simulations. While the computational power of desktop computers is greatly increasing with the continual advances in graphics processing units technology (a fact that has spurred much of the growth in machine learning), many types of simulations are still too computationally expensive to perform without the use of larger scale computational resources. This computational demand for such simulations is one of the main reasons we have chosen to currently use less computationally intensive methods that still provide predictively accurate results. For the purposes of this proposal, we will focus on simulating graphene systems, which have been heavily studied and documented in the literature, thus providing a solid test bed for validating the accuracy of our model. We will give a brief overview of the merit of graphene, as well as explain a method to verify the practicality of our model for simulating graphene.

Brief Overview of Graphene: Graphene is a material which has great potential in terms of its versatility for various applications, from wearables to electronics, to even helping to produce the basic necessities of life – but it's costly to mass produce (23). Obtaining a greater understanding of how graphene behaves, as well as the chemical reactions that result in its formation, could lead to significant advances in areas such as water desalination, energy production and storage, and materials science. It is for these reasons that our last aim is to use the SAGESnet model to better understand the mechanisms behind graphene properties in an effort to aid in the design of cost-effective and scalable chemical syntheses and modifications of graphene-based structures, allowing for better production costs and versatility of structures.

Graphene Molecular Dynamics and Neural Network Simulations: The study we cite for as a comparison for our work is one that has been previously mentioned in this proposal (21), performed by Hanakata et al., titled "Accelerated search and design of stretchable graphene kirigami using machine learning". In this paper, the researchers blended machine learning and molecular dynamics simulations together into a single pipeline that they used to search for certain graphene cutout patterns that yield highly stretchable graphene sheets. The applications for graphene cutouts are numerous, where some of the most interesting of these include applications in robotics and wearable devices.

The method followed by the researchers in this study was a type of search utilizing molecular dynamics and machine learning to find high strain yield graphene sheet cutout configurations. The researchers had roughly 30,000 possible configurations, and it was computationally infeasible for them to perform molecular dynamics simulations to test the strain of each of them, so they developed a pipeline where 100 configurations from this set were randomly chosen and simulated in MD. Then, the resultant strain information of the sheets was obtained from the MD simulation. An image representation of the cutout configuration was generated, and this was fed into a convolutional neural network model that learned to predict the strain of graphene cutouts given their appearances. Next, the trained neural network was used to perform accelerated predictions on all 30,000 configurations at a much higher computational efficiency, and the top 100 predicted configurations for high strain were then simulated in MD. These results were then used to further train the neural network's predictive abilities, and the cycle was repeated until convergence occurred – that is, the training was continued until the NN predictions began giving consistent material predictions round after round, and the top 100 candidate graphene cutouts were obtained.

As we know of no previous works that have performed multiscale simulations completely via neural networks, it is difficult to compare our model with others in a strictly numerical fashion. We can, however, attempt to validate our model by following a similar procedure to that cited. The researchers in this study used the AIREBO molecular dynamics potential previously mentioned, as well as the open source MD software LAMMPS, maintained by Sandia National Laboratory (24), and also described the parameters they used within their MD simulations. We will perform classical MD simulations, using the parameters defined in this study, on graphene cutouts with stretch forces applied to their edges as was performed in the study and feed the results of these simulations to our model so it can learn to simulate similar conditions. Performing training on these simulations will allow us to gain an understanding of what hyperparameter settings are best to use in our network for this type of simulation with graphene.

After our model has been trained with the best hyperparameters we could find, as well as after the average training loss has bottomed out (indicating that further training would be unlikely to yield better results), instead of continuing to perform classical molecular dynamics simulations, we will switch to using solely SAGESnet to simulate the molecular dynamics of randomly chosen graphene configurations that have not yet been seen by the NN model. Instead of predicting a scalar strain yield, however, we will be able to obtain the strain/stretch properties of the graphene sheet configurations directly from the atomic scale simulations performed by our network.

Due to the computational efficiency of neural networks over more traditional MD approaches, we should be able to perform a larger number of simulations via the NN architecture than the researchers in this paper were able to perform with traditional MD. Due to the efficiency of our neural network model, a good test of the performance of our network will be to determine if, with these computational reductions afforded by the network, it is now computationally feasible to study all 30,000 random configurations of graphene cutouts, in an effort to discover the cutout that affords the highest strain yield. Once this cutout has been discovered, the same simulation will be performed in classical MD to verify the validity of the simulation, and the classical MD derived strain yield will be compared with the best strain yield discovered by the method of the cited work. It would be considered an indicator of the success of our model to be able to find a graphene cutout configuration with a higher yield strain, or one equal to that discovered in the paper, by simulating all of the configurations in a computationally feasible amount of time. In the study, one simulation of graphene took about 1 hour to compute with 4 CPU cores available – so computing all 30,000 MD simulations on such a device would take about 3.5 years! The results of this proposal could be considered a great success if SAGESnet could perform all of these 30,000 computations while obtaining similar strain yields to those discovered in the paper, on a similarly configured device in a computationally feasible amount of time (maybe, for example, indicated by a 50-60% reduction in computational time while still achieving similar RMSE values).

After some of these simulations have been performed a random sample of simulations generated by our model will be chosen and classical MD simulations will be performed on the same initial graphene cutout structures as were fed to the SAGESnet model. The trajectories and strain yield of these models will be compared to gain an understanding of the divergence between the SAGESnet simulations and traditional MD simulations. In the cited paper, the researchers were able to get a root mean squared error (RMSE) of less than 0.6 for the strain parameters predicted by their NN compared to those predicted by MD. We would consider it to be a good indicator for the validity of our network to obtain a RMSE either equal to or lower than this value on randomly sampled configurations.

The Use of Neural Networks for Gaining Understanding About Reality: A recent interactive paper (25) studied the use of a neural network type cellular automata grid that was capable of generating self-healing images that grew from a single starting cell. What is remarkable about this paper is the attempt that was made by the authors to understand a process common to biology – that is, how it is that such large creatures are able to originate from a such a small single embryo. The authors in this paper did not tell the initial cell exactly how it was supposed to accomplish this task – they merely placed it in its environment and applied a loss function that rewarded the cell for growing into a shape that more closely approximated its target shape given by the authors. Studies like this that investigate the principles of emergence can help us to gain a greater understanding of biology, and what processes might be driving forces behind it. And, as we discussed, earlier, the autoencoders used in SAGESnet could be used for the purpose of classifying different patterns that arise in chemical systems, patterns of which we have not yet become aware – and this type of discovery could lead to breakthroughs in our understanding of these complex systems, and could help to lead us to new discoveries in chemical nanoscience. An understanding of these types of patterns could lead us to a greater understanding of why nanostructures can be so hard to predict, and could help us to create frameworks for better understanding what patterns or structures are most likely to emerge in certain nanoscale systems. And, using this framework, we could even gain greater understanding as to what types of nanoscale systems we could create that might demonstrate target emergent patterns based on our specifications, ushering in a new level of control for matter at the nanoscale.

Concluding Statement:

We have seen that the SAGESnet architecture offers many possibilities, some of them representing longer term goals, and some of them more directly applicable. From the simulation of chemical data and nanoscale systems over multiple temporal and spatial domains, to the classification of chemical systems in an effort to better understand the underpinning mechanisms behind them – it is fascinating to consider what types of new findings we could make given these new tools provided by artificial intelligence, and of how, when dealing with neural networks, a study of the tools themselves can lead to greater scientific insights. This proposal will present a lot of work in terms of further refinement and training of the SAGESnet architecture to be performant in real world nanomaterial prediction, but it also opens up the possibility of potentially being able to simulate matter in a computationally efficient manner on a large scale than previously possible, while still capturing material properties dependent on smaller scale phenomena – and these possibilities make this process of discovery well worth it.

References:

1. Stuart, S., Tutein, A. & Harrison, J. (2000). A reactive potential for hydrocarbons with intermolecular interactions. *The Journal of Chemical Physics*, 112(14).
2. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., & Bourne, P.E. (2000). *The Protein Data Bank Nucleic Acids Research* (28) [Data set]. **https://www.rcsb.org/**
3. Israeli, N., & Goldenfeld, N. (2004). Computational irreducibility and the predictability of complex physical systems. *Physical review letters*, *92*(7). **https://doi.org/10.1103/PhysRevLett.92.074105**
4. Lewis, F.D. (1976). On computational reducibility. *Journal of Computer and System Sciences*, 12. Retrieved May 8, 2020, from **Google Scholar**
5. Paggi, M., & Hills, D. (2020). *Modeling and simulation of tribological problems in technology*, 97. Springer.
6. Shibuta, Y., Sakane, S., Miyoshi, E., Takaki, T., & Ohno, M. (2019). Micrometer-scale molecular dynamics simulation of microstructure formation linked with multi-phase-field simulation in same space scale. *Modelling and Simulation in Materials Science and Engineering*, 27(5). **https://doi.org/10.1088/1361-651X/ab1d28**
7. Xiao, S., Hu, R., Li, Z., Attarian, S., Björk, K-M., & Lendasse, A. (2019). A machine-learning-enhanced hierarchical multiscale method for bridging from molecular dynamics to continua. *Neural Computing and Applications*, **https://doi.org/10.1007/s00521-019-04480-7**
8. Asproulis, N., & Drikakis, D. (2013). An artificial neural network-based multiscale method for hybrid atomistic-continuum simulations. *Microfluidics and Nanofluidics*. 15(4), 559–574. **https://doi.org/10.1007/s10404-013-1154-4**
9. Chan, H., Cherukara, M.J., Narayanan, B., Loeffler, T.D., Benmore, C., Gray, S.K., & Sankaranarayanan, S.K.R.S. (2019). Machine learning coarse grained models for water. *Nature Communications*, 10(1). **https://doi.org/10.1038/s41467-018-08222-6**
10. Schütt, K.T., Arbabzadah, F., Chmiela, S., Müller, K.R. & Tkatchenko, A. (2017). Quantum-chemical insights from deep tensor neural networks. *Nature Communications*, 8(13890). **https://doi.org/10.1038/ncomms13890**
11. Mardt, A., Pasquali, L., Wu, H. & Noé, F. (2018). VAMPnets for deep learning of molecular kinetics. *Nature Communications*, 9(5). **https://doi.org/10.1038/s41467-017-02388-1**
12. Noé, F., Tkatchenko, A., Müller, K. & Clementi, C. (2020). Machine learning for molecular simulation. *Annual Review of Physical Chemistry*, 71. **https://doi.org/10.1146/annurev-physchem-042018-052331**
13. Hebb, D.O. (1949). *The organization of behavior*. New York: Wiley & Sons.
14. Levitan, I.B., & Kaczmarek, L.K. (2015). The neuron: Cell and molecular biology (4th ed.). Oxford University Press.

15. Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow* (2nd ed.). O'Reilly Media, Inc.

16. Rougier, E., Munjiza, A., & John, N. W. M. (2004). Numerical comparison of some explicit time integration schemes used in DEM, FEM/DEM and molecular dynamics. *International Journal for Numerical Methods in Engineering*, *61*(6), 856–879. **https://doi.org/10.1002/nme.1092**

17. Wang, L., You, Z.-H., Chen, X., Xia, S.-X., Liu, F., Yan, X., Zhou, Y., & Song, K.-J. (2018). A computational-based method for predicting drug–target interactions by using stacked autoencoder deep neural network. *Journal of Computational Biology*, 25(3), 361–373. **https://doi.org/10.1089/cmb.2017.0135**

18. Wang, Y.-B., You, Z.-H., Li, X., Jiang, T.-H., Chen, X., Zhou, X., & Wang, L. (2017). Predicting protein–protein interactions from protein sequences by a stacked sparse autoencoder deep neural network. *Molecular BioSystems* 13(7), 1336–1344. **https://doi.org/10.1039/C7MB00188F**

19. Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. **https://doi.org/10.1109/5.726791**

20. Kajita, S., Ohba, N., Jinnouchi, R., & Asahi, R. (2017). A universal 3D voxel descriptor for solid-state material informatics with deep convolutional neural networks. *Scientific Reports*, 7(16991). **https://doi.org/10.1038/s41598-017-17299-w**

21. Hanakata, P.Z., Cubuk, E.D., Campbell, D.K., & Park, H.S. (2018). Accelerated search and design of stretchable graphene kirigami using machine learning. *Phys. Rev. Lett.* 121(255304). **https://doi.org/10.1103/PhysRevLett.121.255304**

22. Iten, R., Metger, T., Wilming, H., del Rio, L., & Renner, R. (2020). Discovering physical concepts with neural networks. Phys. Rev. Lett. 124(010508). **https://doi.org/10.1103/PhysRevLett.124.010508**

23. Boretti, A., Al-Zubaidy, S., Vaclavikova, M., Al-Abri, M., Castelletto, S., & Mikhalovsky, S. (2018). Outlook for graphene-based desalination membranes. npj Clean Water, 1(1). **https://doi.org/10.1038/s41545-018-0004-z**

24. LAMMPS Molecular Dynamics Simulator. (2020, May 8). LAMMPS Molecular Dynamics Simulator. **https://lammps.sandia.gov/**

25. Distill. (2020, May 8). Growing Neural Cellular Automata: Differentiable Model of Morphogenesis. **https://distill.pub/2020/growing-ca/**

26. Wehmeyer, C.; Noé, F. Time-Lagged Autoencoders: Deep Learning of Slow Collective Variables for Molecular Kinetics. *The Journal of Chemical Physics* **2018**, *148* (24).