Stream-AI-MD: Streaming AI-driven Adaptive Molecular Simulations Targeting the SARS-CoV-2 NSP16/NSP10 Complex

Alexander Brace[1†], Michael Salim[2†], Vishal Subbiah[3†], HengMa[1], Murali Emani[2], Anda Trifan[1,4], Corey Adams[2], Thomas Uram[2], Hyunseung Yoo[1], Andrew Hock[3], Jessica Liu[3], Venkatram Vishwanath[2*], Arvind Ramanathan[1*]

[1] Data Science and Learning, [2] Argonne Leadership Computing Facility, Argonne National Laboratory, [3] Cerebras Systems Inc., [4] University of Illinois at Urbana Champaign,

[†] Joint first authors, [*] Contact Author(s)

ABSTRACT

We demonstrate the efficacy of a novel heterogeneous HPC system for AI-driven MD simulations to better inform our understanding of the nsp16/nsp10 PPI within the SARS-CoV-2 proteome. To realize this vision, we propose Stream-AI-MD, a novel instance of applying DL/AI methods to drive adaptive MD simulation campaigns in a streaming manner. We leverage the ability to run simultaneously run an MD simulations on GPU clusters. The data from atomistic MD simulations are streamed continuously to DL/AI approaches that guide the conformational search in a biophysically meaningful manner on a wafer-scale AI engine. The streamed data is continually resolved into a latent space representation learned from the simulation to quantitatively track the conformational states that have been sampled and can be used to guide the sampling process and result in an improved time to solution to understand complex biological phenomena.

KEYWORDS

Deep learning, accelerators, molecular biophysics, adaptive simulations, protein-protein interactions, streaming data analytics

ACM Reference Format:
Alexander Brace[1†], Michael Salim[2†], Vishal Subbiah[3†], HengMa[1], Murali Emani[2], Anda Trifan[1,4], Corey Adams[2], Thomas Uram[2], Hyunseung Yoo[1], Andrew Hock[3], Jessica Liu[3], Venkatram Vishwanath[2*], Arvind Ramanathan[1*] . 2020. Stream-AI-MD: Streaming AI-driven Adaptive Molecular Simulations Targeting the SARS-CoV-2 NSP16/NSP10 Complex. In *Supercomputing '20: IEEE/ACM International Conference for High Performance Computing, Network, and Storage Analysis, Nov 09–19, 2020, Atlanta, GA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/1122445.1122456

1 JUSTIFICATION

Leveraging a *wafer-scale artificial intelligence (AI) engine*, we implement Stream-AI-MD – a *streaming* AI-driven adaptive ensemble molecular dynamics (MD) simulation toolkit. By effectively interleaving MD simulations with AI-methods, we demonstrate *XX orders of magnitude gain* in sampling the formation of the SARS-CoV-2 2'-O-methyltransferase protein complex, an important COVID-19 drug target. We highlight the benefits of de-coupling AI-based training from HPC clusters and demonstrate significantly higher throughput for such workflows.

2 PERFORMANCE ATTRIBUTES

Table 1: Performance Attributes

Performance Attribute	Our Submission
Category of achievement	Scalability, time-to-solution
Type of method used	Explicit, Deep Learning
Results reported on the basis of	Whole application including I/O
Precision reported	Double Precision (Simulation) Reduced Precision (AI/Learning)
System scale	Measured on full system
Measurement mechanism	Application Timers

3 PROBLEM OVERVIEW

3.1 Science use case

The ongoing coronavirus disease (COVID-19) pandemic has resulted in a devastating human toll resulting in over a million deaths and 36 million infections worldwide [1]. Given the wide ranging implications of the pandemic, there is an urgent need for developing effective therapeutics that can target the causative agent for COVID-19, namely the novel Severe Acute Respiratory Syndrome coronavirus-2 (SARS-CoV-2) [36]. SARS-CoV-2 expresses a rather large (30 kb) single stranded RNA genome which consists of 27 proteins, including 16 non-structural proteins (nsp) that assemble to form the replication-transcription complex (RTC) [46]. These proteins are essential for viral replication, playing an important role in the virus' life cycle and therefore are highly attractive as drug targets [6].

Targeting protein-protein interactions (PPI). Proteins are highly dynamic – they do not function in isolation and their interactions with other proteins/other biomolecules are intrinsic to their functions within a cell [22]. PPIs are thus critical for physiological and

[1] https://www.who.int/emergencies/diseases/novel-coronavirus-2019

Supercomputing '20, Nov 09–19, 2020, Atlanta, GA

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

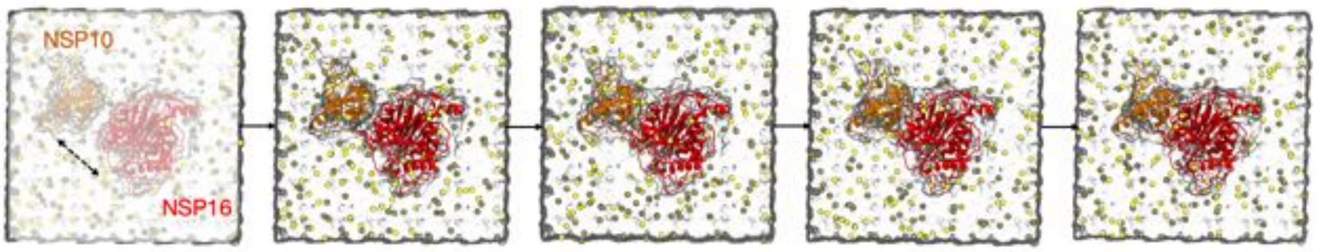

Figure 1: Summary of nsp16/nsp10 systems built as part of the Stream-AI-MD simulation campaign. We start out with fully separated nsp16 and nsp10 systems in solution and allow for successive windows to let nsp16/nsp10 to form a complex. An unsupervised deep learning approach is used to identify potential intermediates in the complex formation process, thus adaptively progressing along these intermediates.

pathological cellular pathways and represent a highly interesting class of therapeutic targets [5]. Although PPIs are recognized as viable drug targets, they are also deemed challenging for optimizing ligand binding sites since PPIs possess relatively "flat" binding pockets with large hydrophobic regions [49]. Further, PPIs are highly flexible, meaning that many druggable pockets may be transient and may not be resolved using traditional structure-determination techniques (such as X-ray crystallography) [45]. Thus, in spite of our knowledge about $O(500{,}000)$ PPIs in a typical cell, only about 27 PPI-based drugs are currently in various stages of clinical trials [26].

Within virus-host interactions, PPIs are an under-represented category of drug-targets [20]. Despite advances in our knowledge about virus-host PPIs as well as viral PPIs (i.e., proteins that interact within the viral proteome itself), targeting these PPIs pose additional challenges including the intrinsic variations that viral proteins tend to undergo and the bias associated within documented PPI databases [11, 19, 23]. *Thus, a grand challenge for the drug-discovery community is in targeting PPIs by uncovering the mechanistic principles of how these interactions are mediated.* We posit that such a principled approach can lead to better therapeutics that are also specific to the viral targets of interest. This assumes immediate importance in the context of SARS-CoV-2, where there are several known virus-host and viral proteome PPIs and targeting these PPIs may offer a potential route to therapeutic discovery.

Understanding the mechanism of SARS-CoV-2 NSP16/NSP10 viral PPI for drug discovery. The SARS-CoV-2 nsp16/nsp10 2'-O-methyltransferase complex (referred to nsp16/nsp10 throughout the paper), methylates Cap-0 viral mRNAs to improve viral protein translation and thus avoids host (human) immune detection [42]. The nsp16/nsp10 complex (Fig. 1) is a heterodimer consisting of nsp16 and nsp10; nsp16 is comprised of a Rossmann-like β-sheet sandwiched between 11 α-helices and several flexible loop structures and nsp10 is a zinc (Zn) binding protein composed of three β strands and a fairly large *disordered* loop that forms the interface between the two proteins in the complex. The nsp16 uses S-adenosyl-L-methionine (SAM) as its methyl group donor for methylation at the ribose 2'-O position of the first nucleotide of the RNA bound to it and is activated when nsp10 binds to it. The binding of nsp10 to nsp16 is posited to stabilize the SAM binding site, allowing the enzyme to catalyze methylation process. In addition, nsp10 binds to the bifunctional enzyme, nsp14 and triggers its exonuclease activity [43]. Given the importance of nsp16/nsp10 in the SARS-CoV-2 lifecycle, we wanted to understand how the interface between nsp16/nsp10 is mediated and whether the intrinsic flexibility of the nsp10 binding loop interacts with and stabilizes the SAM binding site.

An allied goal of this study is to elucidate transient binding pockets within nsp16/nsp10 that may be targeted for small molecule design. Although our goal is not to design a small molecule, we quantitatively probe if any of the potential intermediates identified from the nsp16/nsp10 complex formation process can be targeted for drug design. These intermediates can either be a consequence of the interactions between the flexible loop regions in nsp16/nsp10 or through other routes by which pockets are formed. We posit that the insights gained can inform design of novel small molecules or peptidomimetics (and/or other biologics) to prevent the activation of nsp16, thus preventing mRNA maturation [4, 15, 17, 29].

Current challenges in sampling PPI conformational landscapes. A long standing challenge in molecular biophysics is in understanding how PPIs are mediated; in other words, the association process of two (or more) proteins. Although a number of PPI mechanisms have been proposed, characterizing the constitutive steps and visualizing the formation of various PPI intermediates remains elusive for both structure determination techniques as well as computational simulations, such as molecular dynamics (MD) or Monte Carlo methods [13, 31, 52?]. This is mostly a consequence of the complexity in the conformational landscapes of PPI interactions. Given that even small proteins have $3 \times N$ coordinates (referred to a conformation in MD simulations), assuming a modest size system of $N = 1000$ atoms, implies that the exploration of conformational states is an extremely high dimensional search problem [16, 35]. Given that most atomistic MD simulations use a time-step of $10^{-15}s$ (femtoseconds) and many PPIs are mediated at timescales $> 10^{-6}$-10^{0}s, spanning 9-15 orders of magnitude using traditional simulation approaches can be extremely challenging. Few known successful examples of such exhaustive sampling methods exist; as a consequence current approaches utilize adaptive or enhanced sampling methods. Even then, accurately capturing PPIs and characterizing the intermediates in the association process is challenging. Hence, there is a need for novel methods that can enable atomistic MD simulations to characterize PPI landscapes.

3.2 Methods

Stream-AI-MD Solution. To overcome these challenges, we propose *Stream-AI-MD*, a novel instance of applying DL/AI methods to drive adaptive MD simulation campaigns in a *streaming manner* (see Fig. 2). We leverage the ability to run simultaneously run an ensemble of K ($E = \{E_1, E_2, \ldots E_K\}$) MD simulations on GPU clusters. Ensemble methods have generally been more successful than traditional sampling approaches for sampling complex biological systems and hence we leverage this method for our application. The data (coordinates, contact maps, or other features) from E atomistic MD simulations are *streamed* continuously to DL/AI approaches that guide the conformational search in a biophysically meaningful manner. The streamed data is continually resolved into a latent space representation (Φ_L) learned from the simulation to quantitatively track the conformational states that have been sampled. Φ_L can be conceived as a L-dimensional latent space that clusters the conformers from the ensemble runs into a small number of *states* that share structural and energetic similarities (see Sec. 4). This latent space in turn can be used to guide the conformational sampling process by selecting a small number of “interesting” conformers from which simulations can be seeded next. Simulations that are not producing any interesting data can then be terminated, followed by such new runs getting initiated. Thus, the DL/AI approaches address two aspects: (1) to build biophysically meaningful low-dimensional representations in an unsupervised manner; and (2) to identify *events* within the trajectory that may signify biologically relevant conformational changes (e.g., protein folding/ misfolding) from which additional sampling can be started.

For Stream-AI-MD, we leverage a variational autoencoder (VAE) with convolutional filters (CVAE) that we previously developed to analyze protein folding and PPI simulations (see Sec. 4 [7]. The CVAE builds a latent representation (Φ_L) for the simulation data and uses a number of model parameters including (i) L, describing the number of latent dimensions; (ii) number of convolutional filters as well as the stride parameter for these filters that capture a protein's secondary and tertiary structure changes; (iii) the optimizer used for training; and (iv) the dense layer used to compress the information from the simulation dataset. We have explored these parameters and how they affect the overall performance of the model in our previous work and extended it to support adaptive sampling methods (Sec. 4). However, much of this analysis was carried out *offline*, meaning that the CVAE was trained based on available data before sampling the conformational landscape. In this work, we propose to utilize the CVAE in a *online or streaming* manner, where simulation data is streamed to the deep learning approach to provide continuous feedback for adaptively sampling the conformational landscape.

Although conceptually Stream-AI-MD looks straightforward, everal intrinsic challenges need to be addressed. Even with small biomolecular systems, the size of the datasets generated by the MD simulations can be quite large. Hence data input/output (I/O) throughputs can be quite limiting when implementing such workflows. Further, DL/AI models can take a significant time to train, which means that by the time a trained model becomes available, the simulations could have progressed to sample additional areas of the landscape that would make the currently learned model *obsolete*. Thus, in order to match up with the intrinsic impedance between the data generation throughput of MD simulations and data consumption of the DL/AI, there is a need to optimally enable training runs that are balanced with simulation runs. Further, accelerators such as GPUs are optimized typically for smaller sized images (such as 256 × 256 or 512 × 512). However, with bio-molecular simulations, these input sizes (considering contact maps as inputs from MD simulations), can vary significantly and may not necessarily be accommodated on GPUs. Although optimizations are available to accommodate larger image sizes, they can still pose additional challenges since unlike natural images, contact maps have a particular context associated with their primary sequence of the protein and hence patch-based training approaches may not be entirely suitable.

A number of hardware accelerators are currently available for training DL/AI approaches. These accelerators provide the ability to train relatively complex models (i.e., number of trainable parameters) with larger batch sizes, achieving faster training/inference time – which offers a tremendous advantage in streaming data analysis.

To our knowledge, Stream-AI-MD represents the first application of DL/AI-driven adaptive MD simulation deployed on emerging heterogeneous hardware. We exploit the ability to train fairly large DL/neural network models *on-the-fly using a novel wafer-scale deep learning accelerator*. Similarly, we also *exploit the latest graphics processing units (GPU) architectures namely, the NVIDIA-A100 cards for running MD simulations.* In addition, we developed an integrated software environment that allows for efficient *off loading* of training/inference/ simulation runs across the heterogeneous hardware.

For our science application, we examine whether Stream-AI-MD simulations can characterize the challenging problem of characterizing the various intermediates in the formation of the nsp16/nsp10 PPI, and specifically, how the intrinsic flexibility of nsp16 and nsp10 may facilitate the formation and stabilization of the SAM binding site. Finally, we also analyze the benefits and caveats of Stream-AI-MD, where coupling AI/ML workflows with HPC simulations can have far reaching impacts in AI for science applications.

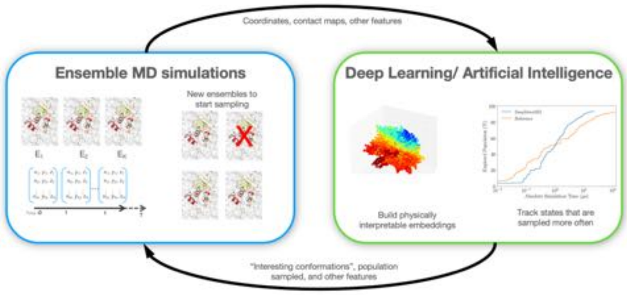
Coordinates, contact maps, other features
Ensemble MD simulations
New ensembles to start sampling
E_1
E_2
E_K
Deep Learning/ Artificial Intelligence
Build physically interpretable embeddings
Track states that are sampled more often
"Interesting conformations", population sampled, and other features

Figure 2: Stream-AI-MD conceptual overview.

4 STATE OF ART

Simulation based approaches for studying PPI mechanisms. Current methods for studying PPIs have leveraged bioinformatics based methods to predict if two proteins can interact with each other, or identify potential *hotspots* or amino acid residues that mediate PPIs, and other predictive features of how mutations may affect the stability of PPIs. While these methods provide important insights into which PPIs may be *druggable*, they do not necessarily provide information regarding the mechanism of the PPI formation [49]. To understand the biophysical mechanisms by which PPIs are mediated experimental techniques (such as structure determination methods, biochemical methods, etc.) are used in conjunction with multiscale MD simulations [10, 48]. Given the wide range of length- and time-scales at which PPIs are mediated, they generally pose a significant challenge for even integrated experimental and computational approaches to probe these biologically relevant events. Simulation-based approaches usually rely on MD-based sampling to understand PPI formation [18, 25]. Thanks to advances in computing hardware and infrastructure, the ability to use long time-scale simulations needed for simulating PPI has been significantly accelerated. Many simulations are now routinely able to access $O(\mu s)$ timescale simulations (some even millisecond timescales) providing the ability to sample the conformational intermediates involved in the formation of PPIs [35]. However, the length scales accessed by these methods is still somewhat limited (involving only small proteins) making it a challenge to study other processes involved.

Machine learning (ML) and artificial intelligence (AI) techniques for analyzing conformational landscapes of bio-molecular systems. A number of techniques have been developed to analyze MD simulation datasets [32, 34]. These methods utilize linear, non-linear, and hybrid methods (mostly in an unsupervised manner) to build a latent manifold/embedding from which clustering approaches can be used to identify conformational states sampled [38].

Recently, our group leveraged autoencoders (AEs) to capture key representational information from MD simulations within a low-dimensional latent space in an unsupervised fashion [7]. Autoencoders typically have an hourglass-shaped architecture in which data is compressed into a low-dimensional latent space in the early layers and then reconstructed back in later layers. Therefore, the latent space learns to capture the most essential information required for reconstruction. Variational autoencoders (VAEs) require an additional optimization constraint that enforces the latent space to be normally distributed [14]. By forcing the latent space to be normally distributed, we force the network to fully utilize the latent space so that information is distributed more evenly; this allows us to sample from any point in the latent space to generate new results that reflect the patterns in the original dataset.

Rather than using regular feedforward layers in our VAE, we apply convolutional layers because they utilize sliding filter maps that can better recognize local patterns independent of its position in the data. In contact map representations, the state of the protein depends on the local interactions between a few atoms rather than on the global position of all atoms in the protein. Because these local interactions do not always appear in the exact same place in the protein, convolutional layers are better suited to recognize these local patterns independent of their position compared to feedforward networks. The architecture for the convolutional autoencoder (CVAE) used in our experiments is illustrated in Fig 3.

Few examples of streaming (or online) data analytics for MD simulations exist [9, 24, 39, 40, 51, 56]. These approaches have been used in the context of *outlier detection*, where the goal is to detect a particular conformational change or some dynamic event. In ML/AI, this is also referred to change-point detection [53]. Most of these approaches use traditional machine learning (such as principal component analysis, or other its variants). However, since deep learning algorithms take at least several hours to train, applications requiring near-realtime feedback rarely use such methods in spite of their superior performance (in terms of prediction, classification, etc.).

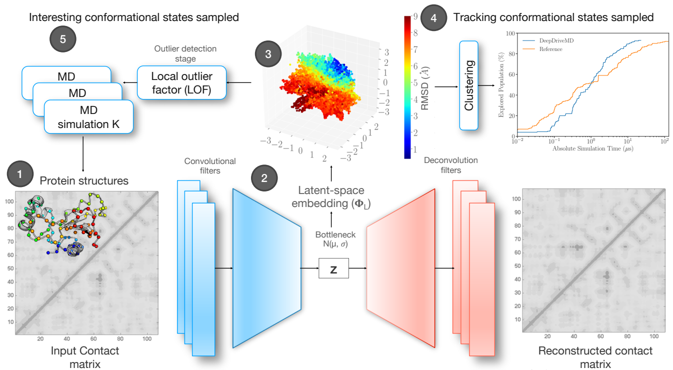

Figure 3: CVAE driven adaptive simulations. (1) Protein trajectories are fed as contact map representations which are passed through (2) convolutional filters and then compressed using a dense layer to obtain (3) a latent representation of the protein conformational landscape. Reconstruction of the contact maps is constructed via deconvolution filters. The latent representation is then used to (4) track the conformational states sampled thus far, and (5) outlier conformations detected from these states using LOF are fed back to run additional MD simulations.

Using AI/ML methods for adaptive sampling in MD ensembles. An attractive option to overcome the intrinsic sampling limitations posed by MD simulations is to leverage AI/ML methods to *steer* them. In particular, AI methods such as the CVAE can be used to *learn* reaction coordinates (RCs) from simulations that correspond to significant conformational changes along some latent (low) dimensions and use these RCs to guide the direction of sampling in the original high dimensional space [8, 12, 21, 30, 33, 41, 41, 47, 55, 57]. These adaptive sampling approaches have two advantages in accelerating the sampling of complex biological events: (1) the low dimensional RCs learned from the simulations correspond to biophysically meaningful/measurable values (such as radius of gyration, R_g) and (2) the RCs organize the conformational landscape into a small number of clusters that can point to interesting parts of the landscape that have been sampled less frequently – naturally pointing out regions of the PPI landscape that are not fully explored. We thus posit that such *guided search* of the PPI conformational landscape can elucidate essential features (i.e., possible intermediates, important amino-acid residues, etc.) that play a role in the function of PPIs.

Previously, our group had studied similar questions in the context of understanding transient unfolding and long-range interactions within the murine γ -herpesvirus 68 vBCL2 (M11) protein when it binds to the disordered BCL2 homology 3 domain from BCL2-interacting coiled-coiled protein (BECN1). Using equilibrium MD, μs timescale simulations and ML approaches, we identified a small number of structural intermediates and a group of residues that mediated the M11-BECN1 interaction. However, it is notable that this study did not *guide* the conformational search; rather, the goal was to understand which residues in M11 mediated the BECN1-PPI [37].

5 INNOVATIONS REALIZED

5.1 Wafer-scale AI Engine

This is the first time that a cluster of graphics processing units (GPUs) has been teamed with a novel wafer-scale, deep learning accelerator to achieve a new performance level for AI-driven, adaptive molecular dynamics. The wafer-scale, deep learning accelerator deployed here is the Cerebras CS-1. This is the first commercial wafer-scale computer [50]; it has performance and scale advantages for DL/AI generally and this application specifically.

Processing, memory, and communication in CS-1 reside in the Cerebras Wafer-Scale Engine (WSE), a 462 cm^2 silicon wafer with approximately 400,000 processor cores. Each core has 48 KB of dedicated SRAM memory (for a total of 18GB on-chip), and all cores are connected to one another over a high bandwidth, low latency, two-dimensional interconnect mesh.

The primary innovation of this device is full wafer-scale integration. Whereas traditional computer chips are made by cutting a wafer into a smaller die (which are then individually packaged on smaller boards and often reconnected using copper or fiber interconnect in a traditional HPC cluster) the WSE retains all die on the wafer and connects them by extending interconnect over silicon across the "scribe lines," where a traditional wafer would have been cut. This novel process yields a single, full-wafer chip with orders of magnitude greater compute resources, memory bandwidth, and low-latency interconnect bandwidth than typical small-scale, general-purpose processors.

The Cerebras WSE has a unique dataflow architecture optimized for deep learning. All aspects of compute, memory, and interconnect are designed for high throughput numerical computation. The instruction set of the core includes a full complement of general-purpose instructions as well as a set of ML-specific extensions to efficiently handle the tensor-based sparse linear algebra operations that are common to neural network computing for deep learning and artificial intelligence. Execution at the core level is triggered by incoming data; output is routed to adjacent cores and across the wafer by the WSE's 2D interconnect mesh.

The WSE's large scale programmable fabric enables pipelined layer-parallel execution for neural network training and inference workloads [54]. Unlike with traditional processor architectures, in this execution model, memory and computation for each layer of the neural network is assigned to a distinct subset of the WSE 400,000 core array. On-chip routing is configured to enable layer-layer communication according to the neural network graph topology. Training data is streamed into the system from external host CPU systems over ethernet and through the CS-1 I/O subsystem onto the wafer; computation for all layers of a neural network training or inference task occurs in parallel as activations flow across the fabric from layer to layer.

For the current Stream-AI-MD application, this architecture enables high utilization and throughput at a variety of batch sizes for rapid wall-clock model training to achieve better impedance matching with adjacent GPU systems used for MD simulations. By providing focused AI acceleration for the neural network portion of this application, the CS-1 also allows us to fully dedicate our specialized GPU resources towards the more graphics-oriented task of running the simulations as well.

The CS-1 is programmed using common, open-source ML frameworks like TensorFlow or PyTorch. Cerebras' software compiler runs entirely on standard, x86 CPU machines. This compiler translates a user's neural network from a framework representation into an optimized executable for the WSE. In this way, any deep learning model, including the one used for the Stream-MD-AI application, can be automatically mapped to the WSE to take advantage of its AI-optimized architecture, large computational resources, and massive bandwidth.

5.2 Enabling near real-time analysis of MD simulation datasets using CVAE

Given the large amounts of time it takes for deep learning algorithms to train, methods for near real-time data ingestion and analysis have been rarely used for MD simulation datasets. Although examples of *in situ* learning do exist [27], the sheer volume of data streamed from such simulations is often the bottleneck for enabling near-real-time feedback. One of the key advantages that is enabled by the WSE is that its fast interconnects can ingest large amounts of data simultaneously while training on larger batch sizes than what a single GPU can handle. Thus, when simulation datasets are generated on the GPUs (especially in an ensemble mode), the WSE is able to keep up with the data rates. We posit that such an architecture could enable near real-time training of the CVAE from running MD simulations. To our knowledge, this also represents a first-time application of *online/streaming* analysis of MD simulation datasets using deep learning approaches.

5.3 Optimized coupling of Simulations and AI Engines

To facilitate Stream-AI-MD, we developed a workflow orchestrator in Python to flexibly configure, launch, and stream data among the three core services: MD simulation, outlier detection (inference), and CVAE training (learning). A key design goal of the orchestrator is to facilitate the detailed configuration of each of these service components using a single YAML file. This YAML file also allows one to describe the system where each component will execute and its resource needs. For configurations in which all components reside on a single resource, the orchestrator provisions a single aggregate partition and maps the components appropriately onto these. In the case where the resources span multiple administrative domains, the orchestrator communicates with each resource manager involved and facilitates the coupling and co-scheduling of the various components. On the ThetaGPU system, the orchestrator uses the Balsam workflow engine [44] to map OpenMM simulations onto A100 GPUs while allowing inference or learning tasks to scale across the remaining resources. In order to move data among the various components, the orchestrator supports multiple transfer mechanisms in an extensible manner, including file-based transfers using globus URL-copy, scp, rsync, memory-based coupling, among others - the underlying mechanism used depends on available transport mechanisms between the resources where components execute.

It provides the ability to stage data to improve I/O time by exploiting node-local storage and/or burst buffers which are characteristic of current and future supercomputing systems; this mitigates the challenges associated with reading and writing of data on shared file systems - a critical bottleneck for AI. The orchestrator also provides for "teeing" transfers from intermediate storage to multiple end-points; this enables storing the data streams over to the filesystem for later analysis and visualization while also facilitating the coupling of the data for other down-stream consumers. Thus, a key innovation here is that the orchestrator enables one to leverage distributed AI accelerator engines to couple simulation, learning, and inference tasks in near-real-time.

6 PERFORMANCE MEASUREMENT

The performance of this experiment is measured across two dimensions, namely, scientific and computational performance.

6.1 Measuring scientific performance

The scientific performance is evaluated using the time to solution metric. First, we tested Stream-AI-MD in folding a fast folding protein, namely FSD-EY (also referred to as BBA in the paper). The BBA protein consists of a classic $\beta\beta\alpha$ fold and is engineered to specifically fold at fast timescales ($O(\mu s)$) [28]. Previously, we used this protein as a test case to validate if our adaptive ensemble MD protocol, DeepDriveMD could correctly fold the protein to its native state. We found that DeepDriveMD could provide at least an order of magnitude faster folding times for the simulations based on the outliers identified. Since the simulation ensembles are guided with the native-state BBA structure, we prioritized the outliers with a lower RMSD are prioritized when starting a new simulation. To measure the effectiveness of such a strategy and efficacy of ML/AI methods to drive simulations, we chose to quantify the distribution of outliers RMSD is plotted at each inference stage. The trend of these outliers shows stage-wise which direction the inference node is driving the MD simulations.

Secondly, we also used Stream-AI-MD to sample the binding pathway of nsp10/nsp16 complex. The ML/AI intervention is implemented to drive the simulation ensembles towards under-sampled conformational states, especially related to places where the two proteins encounter each other in order to sample the bound states. Here, the ML/AI driven ensemble is expected to generate more encounter complexes. To verify this hypothesis, the reaction coordinate distribution of Stream-AI-MD conformers is generated and compared with an equilibrium sampling of the ensemble.

6.2 Measuring computational performance

The computational performance is measured by evaluating the end-to-end performance of the Stream-AI-MD pipeline via application timers as well as profiling the performance of each stage. The performance measures the entire application including I/O and network transfers.

To characterize the performance of the simulation on ThetaGPU, we use the Nvidia NSight Compute 2020.1.2 profiler [1] which extracts several hundreds of metrics at runtime. These metrics are organized within different sections where each section focuses on a specific part of kernel analyses. Nsight organizes metrics in 9 sections namely, *Compute Workload Analysis*, *Instruction Statistics,Launch Statistics*, *Memory Workload Analysis*, *Occupancy*, *Scheduler Statistics*, *Source Counters*, *GPU Speed Of Light* and *Warp State Statistics*. The section speed of light provides a comparison against the best possible behavior across gpu, memory, dram utilization.

For training on the CS-1, we measure training throughput using the standard CS-1 reporting mechanism which monitors for the number of samples per second returned from the WSE.

6.3 Systems and Environment

6.3.1 *Medulla CS-1.*

Medulla CS-1 consists of a single Cerebras CS-1 hosted by eight x86 CPU servers. The CS-1 system has a single 400,000 core Cerebras Wafer-Scale Engine (WSE) processor with 18 GB on-chip memory. The WSE supports operations in IEEE fp16 half-precision and IEEE fp32 single-precision. The CS-1 is connected to host servers and adjacent infrastructure over 12x 100 Gbps Ethernet connections for an aggregate system I/O up to 1.2 Tbps.

The Medulla CS-1 host CPU servers stage, pre-process, and stream input data to the CS-1 system for AI model training. Each host server has two Intel 20-core Xeon Gold CPUs, 192 GB DDR memory, 8 TB storage via five 1.6 TB Intel SSDs with PCI 3.1 x4 NVMe interface, and 200 Gbps network interconnect via two MellanoxConnectX-5 100 Gbps Ethernet NICs.

Software Environment: Each of the CS-1 host CPU servers is installed with CentOS Linux 7, Slurm 19.05.3-2, and Singularity version 3.4.1-1.el7. We compiled the CVAE neural network portion of the Stream-AI-MD application using Cerebras software, which is packaged as a Singularity container and consists of the Cerebras Graph Compiler (CGC), a custom Cerebras distribution of TensorFlow 1.14, and Python 3.7.4.

The Cerebras WSE itself does not run any operating system - compute is triggered by incoming data when training jobs are launched by a Slurm master running on one of the host CPU nodes.

6.3.2 *ThetaGPU.*

ThetaGPU [3] is comprised of twenty-one NVIDIA DGX3 A100 nodes at the Argonne Leadership Computing Facility. Each DGX3 A100 node comprises eight NVIDIA A100 GPUs with a total of 320 gigabytes of GPU memory per node. Each node has two AMD 64-core Rome CPUs with 1 terabyte system memory. An A100 GPU supports 9.7 TFlops in double precision and 19.5 TFlops in single precision. It is architected with a tensor core engine providing for 156 TFlops (TF32), 312 TFlops(FP16 and BF16), among others. The GPUs on a node are interconnected via NVlinkv3 to an NVSwitch. Each DGX3 node has eight Mellanox ConnectX-6 single-port HDR200 PCIe 4.0 x16 adapters providing 200 GB/s. These eight HDR200 interfaces are connected to a dedicated HDR200 fat-tree compute fabric made up of twenty Mellanox QM8790 40-port switches. This fabric also connects the system to a shared 10-petabyte Lustre filesystem. A 15-terabyte node-local NVMe-based partition, comprised of four 4TB solid-state drives in a Raid 0 configuration, offers up to 5 GB/s of I/O performance for both reads and writes on a node.

Software Environment: The software environment on ThetaGPU consists of CUDA 11.0 and cuDNN 8.0 backend. We compiled the OpenMM MD simulation package v 7.5.0 and used a conda environment with Python 3.8.5 comprising of relevant packages, including

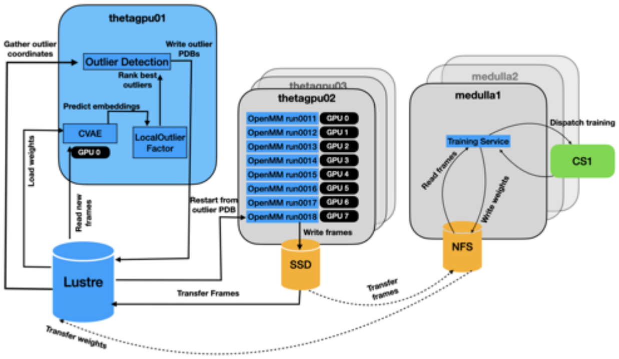

Figure 4: Orchestration of Stream-AI-MD by coupling ThetaGPU and Medulla (Cerebras CS-1) clusters.

dataloaders and analysis packages such as parmed and mdanalysis. For the inference on ThetaGPU, as part of the outlier detection, we use a Tensorflow 1.15 container image from Nvidia.

6.3.3 *Coupling of ThetaGPU and Medulla CS-1 system.*

The ThetaGPU nodes are connected to a border switch via a 40G network interface each. The Medulla CS-1 system is connected to this switch over a 10G interface. Thus, we have a 10 Gbps connectivity between the two systems. As shown in Fig. 4, the MD simulations are deployed on Theta-GPU, the DL/AI training approaches are deployed on the Medulla CS-1 system, and the outlier detection is deployed on the thetagpu node to steer the sampling of conformational space.

The MD simulation and DL/AI runs are loosely coordinated by periodic data transfers between the two systems, such that simulation progress is never blocked on the arrival of data from the AI system. Simulations are guided by the outlier detection algorithm which dispatches MD runs from a priority queue of configurations ranked by a statistical (intrinsic) or biologically-motivated (extrinsic) score. The internal state of the outlier detection algorithm is updated upon the arrival of new CVAE weights from the CS1 training system. All components and their data transfer pathways are provisioned by the orchestrator. Contact maps from the simulation are staged onto the node-local SSD and asynchronously transfer directly to the Medulla CS-1 system for training. The arrival of new contact maps eventually triggers a new round of training, after which the latest model weights are transferred back to ThetaGPU; this transfer in the reverse direction is performed using Globus transfers to the Lustre shared filesystem. The restriction on inbound connections to ThetaGPU compute nodes necessitated this path, and with relatively infrequent transfers and small model weights (on the order of tens of MBs) the overhead did not pose a bottleneck. The outlier detection runs on a full ThetaGPU node and uses both GPUs for CVAE inference and CPUs for local outlier factor calculations. This takes as input the outputs from the simulations and the latest trained model weights to identify the simulations that need to be halted or continued, as well as the new simulations that need to be launched in order to better sample the conformational space.

Table 2: Experiment Configurations

Configuration	Simulation (Sim)	Training (TN)	Outlier Detection (OD)
Baseline on ThetaGPU with Sim	21	-	-
StreamAI-MD - Sim and OD (ThetaGPU) and TN (CS-1)	18	CS1	1

7 PERFORMANCE RESULTS

7.1 Scientific results

Simulation setup and production runs of BBA and nsp16/nsp10 systems. The BBA simulation ensemble is initiated from its unfolded state without secondary structure and a RMSD of 8.22 Å. Its topology is built with OpenMM modeller with Amber99sb-ildn force field and implicit amber99_obc solvent model. 128 - 140 ns total simulation time The simulation are later run with OpenMM CUDA platform at 300K temperature and 1 nm Lennard-Jones cutoff distance. A length constraint is placed upon chemical bonds involving hydrogen atoms. Each step is integrated with Langevin Integrator with a 2 fs timestep. The simulation writes out its configurations every 50 ps. The Stream-AI-MD framework produces 160 individual runs of 128 - 140 ns trajectories ran for 4 hours on 20 Theta-GPU nodes.

The simulation ensembles are initiated with 101 conformations including nsp16/nsp10 complex and detached configurations. To get the separated conformers, nsp16 is displaced from its original binding position up to 10 Å with a uniform 0.1 Å increment. The atomic interactions are described with amber99sb-ildn force field. All systems are solvated by explicit tip3p water model with 0.15 mol/L NaCl in 11^3 nm^3 cubic box. To mimic the body temperature, the simulation is performed at 310 K. The frequency to write a conformer is increased to every 20 ps to provide enough training data. The rest of parameters is similar to BBA simulation. For the equilibrium run, 168 simulations are run for 2.5 hours with most runs producing 11.5 ns trajectories. The Stream-AI-MD production run utilizes 18 ThetaGPU nodes for simulation runs for 2 to 2.5 hours, producing 8-10 ns trajectory from each run.

Folding simulations of BBA. To analyze the results from the folding trajectories, we examined the distributions of the RMSD between the selected outliers and the native state (Fig. 5). Note that as more data is made available to the CS-1, the streaming analysis provides information on outliers that represent some trained change related to the native state. This is done in an unsupervised manner since the inputs to the CVAE are the raw contact maps. Thus, by tracking the collective dynamics of the changes observed in the contact maps, the CVAE learns a latent representation that forces the subsequent iterations to allow the protein to sample conformations that are similar to the native/folded state. This means, at every iteration, the spread of the distribution must point towards lower RMSD values, which can be seen from Fig. 5. We selected the outliers from iterations $\{1, 8, 20, 29, 39\}$ (selected to represent the beginning, middle,

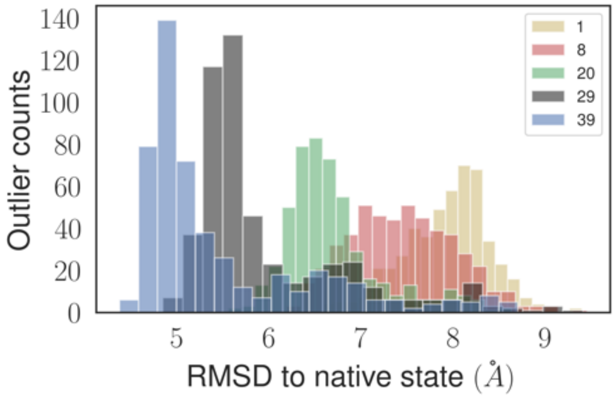

Figure 5: RMSD to native state distributions of 500 outlying conformers at different passes of outlier detection. The starting RMSD distribution is centered around 8 Å and gradually shifts towards 5 Å providing evidence that tail sampling steers the MD simulations toward the folded state.

and simulation end states) and observed that the distribution of the RMSD distinctly moves towards lower RMSDs to the native state.

Corresponding to these results, selecting conformations from the low RMSD range (from the final iteration) also demonstrates that these conformations possess distinct secondary structural features that show features similar to the final folded state of the protein. These observations qualitatively demonstrate that the states that are selected by the CVAE and the LOF detector *preferentially* select conformations that lead towards productive folding pathways. A rigorous analysis of the kinetics of secondary structure formation is necessary to validate our simulations, which we plan to do as part of our future work. It must also be noted that we purposely let our iterations run for very short intervals (only 10 ns between production MD and training/fine-tuning the CVAE) – hence this represents an extreme scenario whereas real production runs may need longer production simulation runs before training/fine-tuning the model.

Simulating the nsp16/nsp10 complex formation process. Given that Stream-AI-MD can potentially sample close to native state of small proteins, we decided to see if we could model the formation of the complex between the nsp16 and nsp10 proteins. As an important drug target, this process is mediated by the interactions between the flexible binding loops of the two proteins and the interface represents a fairly large area of interaction between them. Using Stream-AI-MD, we examined the distribution of the states as observed from tracking the displacement of the center of mass (COM) between the two proteins. Here, we move nsp16 closer to nsp10 – and observe the conformational changes as a consequence of these movements. We compare the Stream-AI-MD simulations with our equilibrium-MD runs (consisting of the same number of conformations) where no AI/ML approaches were used to guide the sampling process.

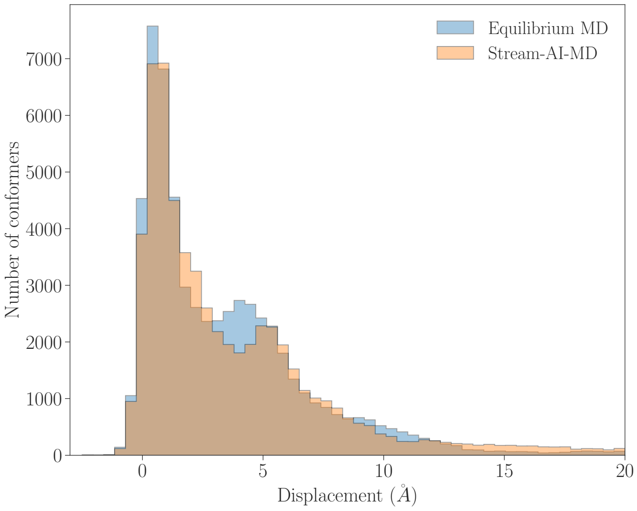

Figure 6: Distance distributions of conformations from the equilibrium-MD (blue) versus Stream-AI-MD (orange) simulations. Despite extremely short time-scale of sampling the landscape, we observe a slight improvement in the sampling of nsp16 and nsp10 that are closer to each other.

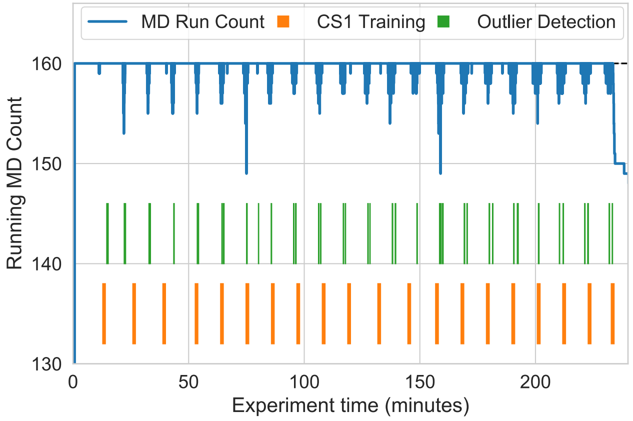

Figure 7: BBA Experiment timeline showing count of running MD simulations with overlaid CS1 training (orange) and outlier detection (green) intervals. Outlier detection time encompasses CVAE forward pass, top outlier ranking, and I/O.

7.2 Computational performance

The computational performance is measured across ThetaGPU and Medulla CS-1 by capturing relevant performance metrics, including timers capturing the entire application.

The blue trace in Figure 7 shows the number of active MD simulations over the span of the 4-hour BBA experiment running on 21 ThetaGPU nodes. An essential feature of Stream-AI-MD illustrated here is that *simulation, learning, and inference are fully overlapping*: the number of active MD runs remains nearly constant at 160

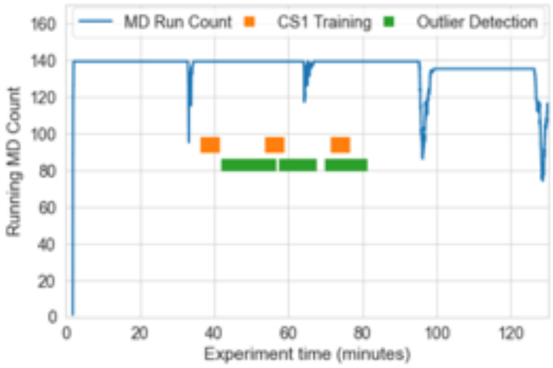

Figure 8: Nsp16/nsp10 complex experiment timeline showing count of running MD simulations with overlaid CS1 training (orange) and outlier detection (green) intervals. Outlier detection time encompasses CVAE forward pass, top outlier ranking, and I/O. The experiment was performed on 19 ThetaGPU nodes (144 GPUs for simulation).

(one simulation per A100 GPU) with brief turnover periods. The overlaid orange and green boxes mark the intervals of CVAE training and outlier detection on this timeline. Because these processes are loosely coupled, we observe an unsynchronized behavior in which multiple rounds of outlier detection may run with the same CVAE model weights. Given the smaller system size in the BBA experiment, the average transfrom from a ThetaGPU gpu node to the Medulla CS-1 system is 1̃58K bytes and this takes an average of 0̃.2sec. We also observe that the overhead of the training and outlier detection is very negligible (<0.1%), and yet, we observe an improvement in both the solution quality and the time it takes to achieve this. This illustrates the impact of using a dedicated AI engine and the end-to-end pipeline of Stream-AI-MD to sample a much larger conformational space and deliver faster insights.

Figure 8 depicts the end-to-end performance of the Stream-AI-MD for the nsp16-10 system executing the simulations on 19 ThetaGPU nodes (144 GPUs), with the training on the CS-1 and the outlier detection on a ThetaGPU node. We observe again that both the training and outlier detection is able to execute concurrently with the simulations. We see the number of simulation configurations remain nearly constant at 1̃44. In this case, we transfer in aggregate 2̃44 MB and this takes in the order of a few minutes. Thus, this stream was unable to saturate the available network interconnect. A overhead here was traced to an issue with the popen system call. In the overall end-to-end pipeline, this time wasn't a bottleneck and the CS-1 was able to complete the training and keep pace with the incoming data streams.

ThetaGPU performance metrics. The performance is obtained by extracting utilization of compute and memory resources. For the nsp16-10 MD simulations, firstly, a flat-profile is obtained with Nvidia NSight Systems [2] that lists out kernels based on their compute time and top two heavy kernels, namely *computeNonbonded* and *composite_2way_fft* were chosen for further analyses. From the metrics obtained from the nsight compute profiler as mentioned above, the critical ones are listed in Table 3. It can be observed that on kernel are memory-bound while the other is compute-bound. The gpu resource utilization could be improved with further optimizations. Since the runs are identical across all gpus, a similar performance is expected to be observed across all nodes.

Table 3: GPU Performance Metrics for two MD Simulation kernels

Metric	*computeNonbonded*	*composite_2way_fft*
SM utilization %	65.33 %	19.12%
IPC	2.45 inst/cycle	0.98 inst/cycle
Memory utilization%	41.2%	28.23%
Memory throughput	47.05 GB/s	243.87 GB/s
Memory bandwidth	32.59%	25.88%
Achieved Occupancy	36.06%	19.00%
DRAM utilization	3.30%	18%

Table 4: Model Description

Model	Parameters
input	(28, 28) / (448, 448)

Conv1
Conv2
Conv3
Conv4
	stride = (1, 1) / (2,2)
	kernel = (5, 5)
	filter = 100

Dense
Var Emb
Dense
Dense
	units = 64
	latent_dim = 10
	units = 64
	units = 64

DeConv1
DeConv2
DeConv3
DeConv4
	stride = (1, 1) / (2, 2)
	kernel = (5, 5)
	filter = 100

Loss	BCE + KL divergence

Medulla CS-1 performance metrics. The model run on the CS-1 is the CVAE described in Figure 3 is depicted in Table 4. While choosing the hyper-parameters, our goals were to optimize time-to-accuracy and reduce the time taken to run the end-to-end workflow, while preventing premature over-fitting of the network. Because the application is bottlenecked by simulation speed, our objective for the CS-1 was simply to select a batch size which would ensure that we could run enough training iterations per simulation transfer for model learning to keep pace with the simulation.

To these ends, we selected a batch size of 512 with a corresponding learning rate of 2e-5. This allowed us to balance training performance with that of the simulations, and exhibited the desired model convergence properties. With these settings, CS-1 achieved training throughput of 48,000 samples/sec on BBA system and 5,700 samples/sec on nsp16/nsp10 complex system. In standalone training runs outside the Stream-AI-MD simulation and inference loop, CS-1 achieved training throughput on BBA and nsp16/nsp10 complex systems of up to 52,000 samples/sec and 6200 samples/sec, respectively. In the case of the nsp16/nsp10 complex system, the large local memory capacity of the WSE was required to enable these runs – at a batch size of 512, the given model in this configuration would not fit on a single GPU.

Another variable we explored was the number of samples seen by the neural network per training loop of the workflow to see if shorter and more frequent trainings helped us achieve faster training to accuracy. To do this, we ran experiments using two modes of data transfer steps from ThetaGPU - comparing convergence properties when we used (1) 40960 samples per loop vs. when we used only (2) 20480 samples per loop.

We observed that, indeed, as we reached 40960 samples per training loop (12 training iterations) in mode 1, where 20480 were newly generated samples and 20480 samples were from the previous iteration, we observed better network convergence as compared to when using only 20480 samples per training (10240 new samples, 10240 old samples) in mode 2, with triple the number of trainings (36) in the same time period.

8 IMPLICATIONS

The work here reports on the demonstration and scientific value of a novel heterogeneous HPC system for AI-driven MD simulations to better inform our understanding of the nsp16/nsp10 PPI within the SARS-CoV-2 proteome. Given its biological role in evading the host (human) immune response, the need for understanding how this complex enzyme functions by "partnering" with nsp10 assumes immediate importance. Further these simulations are providing quantitative insights into how the intrinsic flexibility of loops around the binding surface of both the proteins mediates the overall interaction. These simulations will be useful in elucidating conformational intermediates that mediate the complex formation, which will be further useful downstream to probe the mechanism of inhibiting this important drug target. Thus, our results have immediate implications on the use of AI/ML in simulating complex biological phenomena.

Stream-AI-MD also has implications for how AI and high performance computing (HPC) workflows will intersect in the future. In particular, AI/ML methods have been making significant inroads in driving approximate methods for simulating complex biological systems. AI/ML methods are now effective in extracting complex patterns from MD trajectory datasets, summarizing the implicit statistical relationships between atoms/groups of atoms in a biomolecular system of interest, and predicting specific properties from the data. Further, AI/ML methods have been used in the context of guiding adaptive molecular simulation campaigns. Applications such as Stream-AI-MD therefore represent a growing "motif" of workloads that aim to tightly couple AI/ML with MD simulations.

Here, the key difference between AI/ML methods applied to problems in computer vision or imaging and bio-molecular systems is that the underlying physical principles are generally well encoded, thus providing a well defined framework for characterizing simulation datasets. Thus, a number of AI/ML applications are focused on learning force field parameters from simulations, driving multiscale or coarse-grained simulations from all-atom simulations and building generative models for complex biological systems. However, AI/ML approaches need large amounts of training data and need relatively large amount of compute resources even for running modest size training workloads. This gives rise to an intrinsic imbalance between AI/ML and HPC workloads, which need to be carefully managed.

Separately, recent advances in the development of domain-specific AI computer systems have significant potential to accelerate or augment traditional high performance computing systems for science. We see this across a range of applications from AI-augmented biological and physics-based simulation, to AI-enabled signal and data processing from experimental instrument facilities. This work represents an initial indication of that potential for a specific application. As a result, in the near future we anticipate increasing prevalence of AI accelerators within traditional supercomputing facilities. This will influence both infrastructure and scientific research planning. A key question we plan to investigate in ongoing work is how best to integrate AI and HPC infrastructure, particularly to meet the unique needs of scientific applications and enable novel research.

Methods like Stream-AI-MD are not restricted to biological systems, but can easily adapted to any domain where molecular simulations are used (e.g., materials systems). Such extensions are straightforward; however, one could extend such motifs to other application domains such as population-based epidemiological simulations. Here, real time information could be continually fed into AI/ML models that infer parameter settings for simulation systems and can guide real-time model simulations to predict accurate disease tracking information. These capabilities are critical in the context of ongoing pandemics such as COVID-19.

Continuing work along these lines should consider the impact such systems on e.g. interconnect fabric, as well as novel byte-addressable data management and processing systems, to supporting bursting streaming data. Novel workflows, resource managers, scheduling systems and policies will also be needed for researchers to effectively use such heterogeneous systems and enabling automated end-to-end scientific explorations driven by AI.

ACKNOWLEDGMENTS

We thank the ALCF HPC Systems and Network Administration team. We also thank Drs. Shantenu Jha, Hyungro Lee, Igor Yakushin, Todd Munson, Kristopher Keipert, and Ian Foster for helpful discussions. Research was supported by the DOE Office of Science through the National Virtual Biotechnology Laboratory, a consortium of DOE national laboratories focused on response to COVID-19, with funding provided by the Coronavirus CARES Act. This research was supported as part of the CANDLE project by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration. This research used resources of the Argonne Leadership Computing Facility (ALCF), which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357. Anda Trifan acknowledges support from the United States Department of Energy through the Computational

Sciences Graduate Fellowship (DOE CSGF) under grant number: DE-SC0019323

REFERENCES

[1] [n.d.]. Nvidia Nsight Compute Profiler. https://docs.nvidia.com/nsight-compute. Accessed: 2020-10-03.

[2] [n.d.]. Nvidia Nsight Systems Profiler. https://developer.nvidia.com/nsight-systems. Accessed: 2020-10-03.

[3] [n.d.]. ThetaGPU Supercomputing System. https://www.alcf.anl.gov/support-center/theta/theta-thetagpu-overview. Accessed: 2020-10-03.

[4] Brandon J Anson, Mackenzie E Chapman, Emma K Lendy, Sergii Pshenychnyi, TD Richard, Karla JF Satchell, and Andrew D Mesecar. 2020. Broad-spectrum inhibition of coronavirus main and papain-like proteases by HCV drugs. (2020).

[5] Michelle R. Arkin, Yinyan Tang, and James A. Wells. 2014. Small-Molecule Inhibitors of Protein-Protein Interactions: Progressing toward the Reality. *Chemistry Biology* 21, 9 (2014), 1102 – 1114. https://doi.org/10.1016/j.chembiol.2014.09.001

[6] Hossam M. Ashour, Walid F. Elkhatib, Md. Masudur Rahman, and Hatem A. Elshabrawy. 2020. Insights into the Recent 2019 Novel Coronavirus (SARS-CoV-2) in Light of Past Human Coronavirus Outbreaks. *Pathogens* 9, 3 (2020). https://doi.org/10.3390/pathogens9030186

[7] Debsindhu Bhowmik, Shang Gao, Michael T. Young, and Arvind Ramanathan. 2018. Deep clustering of protein folding simulations. *BMC Bioinformatics* 19, 18 (2018), 484. https://doi.org/10.1186/s12859-018-2507-5

[8] Luigi Bonati, Yue-Yu Zhang, and Michele Parrinello. 2019. Neural networks-based variationally enhanced sampling. *Proceedings of the National Academy of Sciences* 116, 36 (2019), 17641–17647. https://doi.org/10.1073/pnas.1907975116 arXiv:https://www.pnas.org/content/116/36/17641.full.pdf

[9] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. 2000. LOF: Identifying Density-Based Local Outliers. *SIGMOD Rec.* 29, 2 (May 2000), 93–104. https://doi.org/10.1145/335191.335388

[10] Carlos J. Camacho, Zhiping Weng, Sandor Vajda, and Charles DeLisi. 1999. Free Energy Landscapes of Encounter Complexes in Protein-Protein Association. *Biophysical Journal* 76, 3 (1999), 1166 – 1178. https://doi.org/10.1016/S0006-3495(99)77281-4

[11] Pablo Ceres and Adam Zlotnick. 2002. Weak ProteinProtein Interactions Are Sufficient To Drive Assembly of Hepatitis B Virus Capsids. *Biochemistry* 41, 39 (2002), 11525–11531. https://doi.org/10.1021/bi0261645 arXiv:https://doi.org/10.1021/bi0261645 PMID: 12269796.

[12] Matteo T. Degiacomi. 2019. Coupling Molecular Dynamics and Deep Learning to Mine Protein Conformational Space. *Structure* 27, 6 (2019), 1034 – 1040.e3. https://doi.org/10.1016/j.str.2019.03.018

[13] Kalyani Dhusia, Zhaoqian Su, and Yinghao Wu. 2020. Using Coarse-Grained Simulations to Characterize the Mechanisms of Protein–Protein Association. *Biomolecules* 10, 7 (2020). https://doi.org/10.3390/biom10071056

[14] Carl Doersch. 2016. Tutorial on Variational Autoencoders. arXiv:stat.ML/1606.05908

[15] Sean Ekins, Melina Mottin, Paulo R.P.S. Ramos, Bruna K.P. Sousa, Bruno Junior Neves, Daniel H. Foil, Kimberley M. Zorn, Rodolpho C. Braga, Megan Coffee, Christopher Southan, Ana C. Puhl, and Carolina Horta Andrade. 2020. Déjà vu: Stimulating open drug discovery for SARS-CoV-2. *Drug Discovery Today* 25, 5 (2020), 928 – 941. https://doi.org/10.1016/j.drudis.2020.03.019

[16] Adrian H. Elcock, David Sept, and J. Andrew McCammon. 2001. Computer Simulation of ProteinProtein Interactions. *The Journal of Physical Chemistry B* 105, 8 (2001), 1504–1518. https://doi.org/10.1021/jp003602d arXiv:https://doi.org/10.1021/jp003602d

[17] José Antonio Encinar and Javier A. Menendez. 2020. Potential Drugs Targeting Early Innate Immune Evasion of SARS-Coronavirus 2 via 2'-O-Methylation of Viral RNA. *Viruses* 12, 5 (2020). https://doi.org/10.3390/v12050525

[18] Cunliang Geng, Li C. Xue, Jorge Roel-Touris, and Alexandre M. J. J. Bonvin. 2019. Finding the G spot: Are predictors of binding affinity changes upon mutations in protein–protein interactions ready for it? *WIREs Computational Molecular Science* 9, 5 (2019), e1410. https://doi.org/10.1002/wcms.1410 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/wcms.1410

[19] Don L. Gibbons, Marie-Christine Vaney, Alain Roussel, Armelle Vigouroux, Brigid Reilly, Jean Lepault, Margaret Kielian, and Félix A. Rey. 2004. Conformational change and protein–protein interactions of the fusion protein of Semliki Forest virus. *Nature* 427, 6972 (2004), 320–325. https://doi.org/10.1038/nature02239

[20] Norman Goodacre, Prajwal Devkota, Eunhae Bae, Stefan Wuchty, and Peter Uetz. 2020. Protein-protein interactions of human viruses. *Seminars in Cell Developmental Biology* 99 (2020), 31 – 39. https://doi.org/10.1016/j.semcdb.2018.07.018 SI: Protein-protein interactions in health and disease.

[21] Carlos X. Hernández, Hannah K. Wayment-Steele, Mohammad M. Sultan, Brooke E. Husic, and Vijay S. Pande. 2018. Variational encoding of complex dynamics. *Phys. Rev. E* 97 (Jun 2018), 062412. Issue 6. https://doi.org/10.1103/PhysRevE.97.062412

[22] Lingyan Jin, Weiru Wang, and Guowei Fang. 2014. Targeting Protein-Protein Interaction by Small Molecules. *Annual Review of Pharmacology and Toxicology* 54, 1 (2014), 435–456. https://doi.org/10.1146/annurev-pharmtox-011613-140028 arXiv:https://doi.org/10.1146/annurev-pharmtox-011613-140028 PMID: 24160698.

[23] Lingyan Jin, Weiru Wang, and Guowei Fang. 2014. Targeting Protein-Protein Interaction by Small Molecules. *Annual Review of Pharmacology and Toxicology* 54, 1 (2014), 435–456. https://doi.org/10.1146/annurev-pharmtox-011613-140028 arXiv:https://doi.org/10.1146/annurev-pharmtox-011613-140028 PMID: 24160698.

[24] Travis Johnston, Boyu Zhang, Adam Liwo, Silvia Crivelli, and Michela Taufer. 2017. In situ data analytics and indexing of protein trajectories. *Journal of Computational Chemistry* 38, 16 (2017), 1419–1430. https://doi.org/10.1002/jcc.24729 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcc.24729

[25] Alfredo Jost Lopez, Patrick K. Quoika, Max Linke, Gerhard Hummer, and Jürgen Köfinger. 2020. Quantifying Protein–Protein Interactions in Molecular Simulations. *The Journal of Physical Chemistry B* 124, 23 (2020), 4673–4685. https://doi.org/10.1021/acs.jpcb.9b11802 arXiv:https://doi.org/10.1021/acs.jpcb.9b11802 PMID: 32379446.

[26] Harry Jubb, Alicia P. Higueruelo, Anja Winter, and Tom L. Blundell. 2012. Structural biology and drug discovery for protein–protein interactions. *Trends in Pharmacological Sciences* 33, 5 (2012), 241 – 248. https://doi.org/10.1016/j.tips.2012.03.006

[27] Can Li, Daniel Belkin, Yunning Li, Peng Yan, Miao Hu, Ning Ge, Hao Jiang, Eric Montgomery, Peng Lin, Zhongrui Wang, Wenhao Song, John Paul Strachan, Mark Barnell, Qing Wu, R. Stanley Williams, J. Joshua Yang, and Qiangfei Xia. 2018. Efficient and self-adaptive in-situ learning in multilayer memristor neural networks. *Nature Communications* 9, 1 (2018), 2385. https://doi.org/10.1038/s41467-018-04484-2

[28] Kresten Lindorff-Larsen, Stefano Piana, Ron O. Dror, and David E. Shaw. 2011. How Fast-Folding Proteins Fold. *Science* 334, 6055 (2011), 517–520. https://doi.org/10.1126/science.1208351 arXiv:https://science.sciencemag.org/content/334/6055/517.full.pdf

[29] Sen Liu, Qiang Zheng, and Zhiying Wang. 2020. Potential covalent drugs targeting the main protease of the SARS-CoV-2 coronavirus. *Bioinformatics* 36, 11 (04 2020), 3295–3298. https://doi.org/10.1093/bioinformatics/btaa224 arXiv:https://academic.oup.com/bioinformatics/article-pdf/36/11/3295/33329387/btaa224.pdf

[30] Andreas Mardt, Luca Pasquali, Hao Wu, and Frank Noé. 2018. VAMPnets for deep learning of molecular kinetics. *Nature Communications* 9, 1 (2018), 5. https://doi.org/10.1038/s41467-017-02388-1

[31] Mikita M. Misiura and Anatoly B. Kolomeisky. 2020. Role of Intrinsically Disordered Regions in Acceleration of Protein–Protein Association. *The Journal of Physical Chemistry B* 124, 1 (2020), 20–27. https://doi.org/10.1021/acs.jpcb.9b08793 arXiv:https://doi.org/10.1021/acs.jpcb.9b08793 PMID: 31804089.

[32] Frank Noé. 2020. *Machine Learning for Molecular Dynamics on Long Timescales.* Springer International Publishing, Cham, 331–372. https://doi.org/10.1007/978-3-030-40245-7_16

[33] Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. 2019. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science* 365, 6457 (2019). https://doi.org/10.1126/science.aaw1147 arXiv:https://science.sciencemag.org/content/365/6457/eaaw1147.full.pdf

[34] Frank Noé, Gianni De Fabritiis, and Cecilia Clementi. 2020. Machine learning for protein folding and dynamics. *Current Opinion in Structural Biology* 60 (2020), 77 – 84. https://doi.org/10.1016/j.sbi.2019.12.005 Folding and Binding Proteins.

[35] Albert C. Pan, Daniel Jacobson, Konstantin Yatsenko, Duluxan Sritharan, Thomas M. Weinreich, and David E. Shaw. 2019. Atomic-level characterization of protein–protein association. *Proceedings of the National Academy of Sciences* 116, 10 (2019), 4244–4249. https://doi.org/10.1073/pnas.1815431116 arXiv:https://www.pnas.org/content/116/10/4244.full.pdf

[36] Jerry M. Parks and Jeremy C. Smith. 2020. How to Discover Antiviral Drugs Quickly. *New England Journal of Medicine* 382, 23 (2020), 2261–2264. https://doi.org/10.1056/NEJMcibr2007042 arXiv:https://doi.org/10.1056/NEJMcibr2007042 PMID: 32433861.

[37] Arvind Ramanathan, Akash Parvatikar, Srinivas C. Chennubhotla, Yang Mei, and Sangita C. Sinha. 2020. Transient Unfolding and Long-Range Interactions in Viral BCL2 M11 Enable Binding to the BECN1 BH3 Domain. *Biomolecules* 10, 9 (2020). https://doi.org/10.3390/biom10091308

[38] Arvind Ramanathan, Andrej Savol, Virginia Burger, Chakra S. Chennubhotla, and Pratul K. Agarwal. 2014. Protein Conformational Populations and Functionally Relevant Substates. *Accounts of Chemical Research* 47, 1 (2014), 149–156. https://doi.org/10.1021/ar400084s arXiv:https://doi.org/10.1021/ar400084s PMID: 23988159.

[39] Arvind Ramanathan, Andrej J. Savol, Pratul K. Agarwal, and Chakra S. Chennubhotla. 2012. Event detection and sub-state discovery from biomolecular simulations using higher-order statistics: Application to enzyme adenylate kinase. *Proteins: Structure, Function, and Bioinformatics* 80, 11 (2012), 2536–2551. https://doi.org/10.1002/prot.24135 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.24135

[40] Arvind Ramanathan, Ji Oh Yoo, and Christopher J. Langmead. 2011. On-the-Fly Identification of Conformational Substates from Molecular Dynamics Simulations. *Journal of Chemical Theory and Computation* 7, 3 (2011), 778–789. https://doi.org/10.1021/ct100531j arXiv:https://doi.org/10.1021/ct100531j PMID: 26596308.

[41] João Marcelo Lamim Ribeiro, Pablo Bravo, Yihang Wang, and Pratyush Tiwary. 2018. Reweighted autoencoded variational Bayes for enhanced sampling (RAVE). *The Journal of Chemical Physics* 149, 7 (2018), 072301. https://doi.org/10.1063/1.5025487 arXiv:https://doi.org/10.1063/1.5025487

[42] Monica Rosas-Lemus, George Minasov, Ludmilla Shuvalova, Nicole L. Inniss, Olga Kiryukhina, Joseph Brunzelle, and Karla J. F. Satchell. 2020. High-resolution structures of the SARS-CoV-2 2'-O-methyltransferase reveal strategies for structure-based inhibitor design. *Science Signaling* 13, 651 (2020). https://doi.org/10.1126/scisignal.abe1202 arXiv:https://stke.sciencemag.org/content/13/651/eabe1202.full.pdf

[43] Monica Rosas-Lemus, George Minasov, Ludmilla Shuvalova, Nicole L Inniss, Olga Kiryukhina, Grant Wiersum, Youngchang Kim, Robert Jedrzejczak, Natalia I Maltseva, Michael Endres, et al. 2020. The crystal structure of nsp10-nsp16 heterodimer from SARS-CoV-2 in complex with S-adenosylmethionine. *bioRxiv* (2020).

[44] M. Salim, T. Uram, J. T. Childers, V. Vishwanath, and M. Papka. 2019. Balsam: Near Real-Time Experimental Data Analysis on Supercomputers. In *2019 IEEE/ACM 1st Annual Workshop on Large-scale Experiment-in-the-Loop Computing (XLOOP)*. 26–31.

[45] Benjamin Schuler, Alessandro Borgia, Madeleine B Borgia, Pétur O Heidarsson, Erik D Holmstrom, Daniel Nettels, and Andrea Sottini. 2020. Binding without folding – the biomolecular function of disordered polyelectrolyte complexes. *Current Opinion in Structural Biology* 60 (2020), 66 – 76. https://doi.org/10.1016/j.sbi.2019.12.006 Folding and Binding Proteins.

[46] Megan Scudellari. 2020. The sprint to solve coronavirus protein structures—and disarm them with drugs. *Nature* 581, 7808 (2020), 252–255.

[47] Zahra Shamsi, Kevin J. Cheng, and Diwakar Shukla. 2018. Reinforcement Learning Based Adaptive Sampling: REAPing Rewards by Exploring Protein Conformational Landscapes. *The Journal of Physical Chemistry B* 122, 35 (09 2018), 8386–8395. https://doi.org/10.1021/acs.jpcb.8b06521

[48] Felix B Sheinerman, Raquel Norel, and Barry Honig. 2000. Electrostatic aspects of protein–protein interactions. *Current Opinion in Structural Biology* 10, 2 (2000), 153 – 159. https://doi.org/10.1016/S0959-440X(00)00065-8

[49] Graham R. Smith and Michael J.E. Sternberg. 2002. Prediction of protein–protein interactions by docking methods. *Current Opinion in Structural Biology* 12, 1 (2002), 28 – 35. https://doi.org/10.1016/S0959-440X(02)00285-3

[50] Cerebras Systems. 2019. *Wafer-Scale Deep Learning, Presentation at HotChips 2019.* Retrieved October 3, 2020 from https://www.youtube.com/watch?v=QF9oObzMBpU&t=3715.

[51] M. Taufer, S. Thomas, M. Wyatt, T. M. Anh Do, L. Pottier, R. F. da Silva, H. Weinstein, M. A. Cuendet, T. Estrada, and E. Deelman. 2019. Characterizing In Situ and In Transit Analytics of Molecular Dynamics Simulations for Next-Generation Supercomputers. In *2019 15th International Conference on eScience (eScience)*. 188–198.

[52] Kei Terayama, Ai Shinobu, Koji Tsuda, Kazuhiro Takemura, and Akio Kitao. 2019. evERdock BAI: Machine-learning-guided selection of protein-protein complex structure. *The Journal of Chemical Physics* 151, 21 (2019), 215104. https://doi.org/10.1063/1.5129551 arXiv:https://doi.org/10.1063/1.5129551

[53] Gerrit J. J. van den Burg and Christopher K. I. Williams. 2020. An Evaluation of Change Point Detection Algorithms. arXiv:stat.ML/2003.06222

[54] Natalia Vassilieva. 2020. *Neural Network Parallelism at Wafer Scale.* Retrieved October 3, 2020 from https://www.cerebras.net/data-model-pipeline-parallel-training-neural-networks/.

[55] Yihang Wang, João Marcelo Lamim Ribeiro, and Pratyush Tiwary. 2019. Past–future information bottleneck for sampling molecular reaction coordinate simultaneously with thermodynamics and kinetics. *Nature Communications* 10, 1 (2019), 3573. https://doi.org/10.1038/s41467-019-11405-4

[56] Willy Wriggers, Kate A. Stafford, Yibing Shan, Stefano Piana, Paul Maragakis, Kresten Lindorff-Larsen, Patrick J. Miller, Justin Gullingsrud, Charles A. Rendleman, Michael P. Eastwood, Ron O. Dror, and David E. Shaw. 2009. Automated Event Detection and Activity Monitoring in Long Molecular Dynamics Simulations. *Journal of Chemical Theory and Computation* 5, 10 (2009), 2595–2605. https://doi.org/10.1021/ct900229u arXiv:https://doi.org/10.1021/ct900229u PMID: 26631775.

[57] Jun Zhang, Yi Isaac Yang, and Frank Noé. 2019. Targeted Adversarial Learning Optimized Sampling. *The Journal of Physical Chemistry Letters* 10, 19 (10 2019), 5791–5797. https://doi.org/10.1021/acs.jpclett.9b02173