High-Throughput Virtual Screening of Small Molecule Inhibitors for SARS-CoV-2 Protein Targets with Deep Fusion Models

Garrett A. Stevenson¹, Derek Jones¹, Hyojin Kim¹, W. F. Drew Bennett¹, Brian J. Bennion¹, Monica Borucki¹, Feliza Bougret¹, Aidan Epstein¹, Magdalena Franco¹, Brooke Harmon², Stewart He³, Max P. Katz³, Daniel Kirshner¹, Victoria Lao¹, Edmond Y. Lau¹, Jacky Lo¹, Kevin McLoughlin¹, Richard Mosesso², Deepa K. Murugesh¹, Oscar A. Negrete², Edwin A. Saada¹, Brent Segelke¹, Maxwell Stefan², Marisa W. Torres¹, Dina Weilhammer¹, Sergio Wong¹, Yue Yang¹, Adam Zemla¹, Xiaohua Zhang¹, Fangqiang Zhu¹, Felice C. Lightstone¹, Jonathan E. Allen¹,

¹Lawrence Livermore National Laboratory, Livermore, California, USA
²Sandia National Laboratories, Livermore, California, USA
³NVIDIA Corporation, Santa Clara, California, USA

ABSTRACT
Structure-based Deep Fusion models were recently shown to outperform several physics- and machine learning-based protein-ligand binding affinity prediction methods. As part of a multi-institutional COVID-19 pandemic response, over 500 million small molecules were computationally screened against four protein structures from the novel coronavirus (SARS-CoV-2), which causes COVID-19. Three enhancements to Deep Fusion were made in order to evaluate more than 5 billion docked poses on SARS-CoV-2 protein targets. First, the Deep Fusion concept was refined by formulating the architecture as one, coherently backpropagated model (Coherent Fusion) to improve binding-affinity prediction accuracy. Secondly, the model was trained using a distributed, genetic hyper-parameter optimization. Finally, a scalable, high-throughput screening capability was developed to maximize the number of ligands evaluated and expedite the path to experimental evaluation. In this work, we present both the methods developed for machine learning-based high-throughput screening and results from using our computational pipeline to find SARS-CoV-2 inhibitors.

KEYWORDS
deep learning, hyper-parameter optimization, SARS-CoV-2, COVID-19, HPC, GPU, AI

1 INTRODUCTION
The COVID-19 disease caused by the severe acute respiratory syndrome coronavirus (SARS-CoV-2) is responsible for the most recent, severe pandemic in modern human history [33]. At the onset of the COVID-19 pandemic, a worldwide effort began to identify and provide target proteins for vaccine and drug development to neutralize the virus. Two distinct proteins were rapidly solved; the trimeric spike protein (spike), which binds to human ACE2 to enter human cells [3] and the main protease (M⁺pr), which plays a pivotal role in viral gene expression and replication [59]. In response to the pandemic, we participated in a large-scale multi-institutional effort to virtually screen, experimentally test, and optimize therapeutic leads targeting the spike and M⁺pr SARS-CoV-2 protein targets. Two different binding sites from the spike protein (denoted spike1, spike2) and two different conformations of the M⁺pr active site (denoted protease1, protease2) were used in the high-throughput screening calculations.

Experimental tests of drug candidates are expensive and serve as a fundamental bottleneck in drug discovery. As a result, computational chemistry is used extensively to accelerate the discovery process by screening drugs and nominating the strongest candidates for experimental validation [64]. High Performance Computing (HPC) plays a critical role in virtual screening [6, 28, 30, 69, 70] by accelerating computationally expensive calculations and providing the scalability necessary to screen large numbers of candidate molecules. This is crucial, as the chemical space of potential
molecules has been estimated to be on the order of 10^60 [8, 49]. Accurately estimating protein-ligand binding affinities is an important step in drug discovery. However, even computationally expensive, biophysics-based scoring methods find predicting binding free energy a difficult task [27, 64, 72]. Deep learning methods represent an alternative, rapid approach to binding affinity prediction which alleviates the dependence on hand-curated features, which may not capture the mechanism of binding [1, 2].

The two leading deep learning approaches to structure-based binding affinity prediction fall into two categories: 3-dimensional Convolutional Neural Networks (3D-CNNs) [26, 48, 67] and Spatial Graph Convolutional Neural Networks (SG-CNNs) [13, 37, 71]. Fundamentally, 3D-CNN models exploit a voxelized representation of atoms in a 3D grid, which portrays protein-ligand compounds in a Euclidean space for inference. On the other hand, SG-CNN approaches leverage a graph representation of the protein-ligand complex, allowing for multiple â€œedge-typesâ€ to be encoded in the representation (e.g., distinct distance thresholds corresponding to covalent or non-covalent interactions) to sub-select groups of atoms and evaluate their pairwise interactions.

These methods are significantly different in the way they represent compounds and their mechanisms of inference. However, they seek to achieve the same goal: accurate prediction of binding free energy on novel compounds. This observation led to the hypothesis that the 3D-CNN and SG-CNN likely have complementary strengths, which could be exploited by fusing the latent spaces of each modelâ€™s learned features. This approach to â€œFusionâ€ modeling was explored and shown to achieve superior generalization performance in predicting protein-ligand binding on X-ray crystallographic structures and virtually-docked poses of protein-ligand compounds in hold-out test sets [27]. In this work, we detail improvements to Fusion, describe its utility in high-throughput screening, and evaluate its application to the SARS-CoV-2 drug discovery problem. A key innovation reported here is the potential to train a fusion model â€œcoherentlyâ€. While Coherent Fusion comes with the increased computational burden of training a more complex model, the increase in complexity is mitigated by using HPC to perform parallel, distributed training.

2 DEEP FUSION

2.1 Fusion Modeling in Computational Chemistry

Machine learning, specifically deep learning, approaches to protein-ligand binding affinity prediction, represent a promising new development in drug discovery [1, 2, 13, 16, 26, 56, 67, 71, 72]. At a high level, the deep-learned models being proposed for binding affinity prediction are single-pass, feed-forward systems. This fundamental model formulation results in a computational advantage, in that, the models quickly predict binding affinity in one pass over their input. The simplicity and speed of deep learning prediction relative to biophysics-based computations make them especially attractive in the context of massive virtual drug screens.

The concept of fusion is a recent development in deep learning, initially applied to computer vision problems [35, 50, 60, 66]. In fusion, models are combined by integrating different modes of data or approaches for more predictive power. This concept was recently applied to computational chemistry in the form of Fusion models for Atomic and molecular STructures (FAST) [27], where fusion of the two leading deep learning models (3D-CNN, SG-CNN) was shown to improve binding affinity prediction. Specifically, the Late Fusion and Mid-level Fusion models are shown as approximately equivalent or superior to individual SG-CNNs, individual 3D-CNNs, other state-of-the-art deep learning models [26, 56], and physics-based approaches including both Autodock Vina [58] and Molecular Mechanics - Generalized Born / Surface Area (MM/GBSA) which has been shown to improve ligand pose ranking for certain target proteins [19, 64]. The Late Fusion and Mid-level Fusion FAST models were specifically described and made publicly available. In the context of this work, we retrain and optimize the models on data from PDBbind-2019 [62].

\[ pK = -\log(K); \text{ where } K = K_i, K_d, \text{ or } IC_{50} \] (1)

The Late Fusion approach is simple, but its superior performance compared to individual SG-CNNs and 3D-CNNs shows the potential of fusion modeling. In Late Fusion, SG-CNN and 3D-CNN models were separately trained to predict absolute binding affinity (Equation 1), where absolute binding affinity is defined as the negative logarithm of a binding constant \( K \). Binding affinity data in this study is measured as an inhibitory constant \( K_i \), dissociation constant \( K_d \), or inhibitory activity (\( IC_{50} \)), where these measurements are treated as equivalent labels in our calculations. Late Fusion takes the unweighted arithmetic mean across the individually-predicted binding affinity values from the SG-CNN and 3D-CNN models.

The Mid-level Fusion model is also described in [27]. The model is defined by extracting the latent space feature vector from \( Layer^N \) of an N-layer SG-CNN and \( Layer^{M-1} \) of an M-layer 3D-CNN. Each vector of gradients is then processed through model-specific dense layers, concatenated with the originally extracted vectors, and passed through two fusion dense layers for a final prediction. In contrast to Late Fusion, Mid-level Fusion is a non-linear combination of the respective models and its performance has been shown to outperform Late Fusion in some cases.

2.2 Advancing Fusion

Faced with the imperative task of virtually screening molecules for inhibition of SARS-CoV-2 activity, we sought to apply and improve the fusion deep learning approach. First, we updated the previously trained models with data from the latest version of PDBbind (2019) [62], which provides approximately 4000 more compounds than the 2016 version used to train the original models and, thereby, offers the potential for improved generalization.

Our second step looked to improve the 3D-CNN and SG-CNN models through recent developments in the optimization of hyper-parameters. Deep learning models are highly sensitive to a humanâ€™s definition of their hyper-parameters â€œS model architecture, loss function, and optimization algorithm, among others [24]. Automated hyper-parameter optimization was first addressed with parallel searches (grid or random), followed by sequential optimization methods such as Bayesian optimization [4, 21, 53, 55]. Hyper-parameter optimization algorithms have improved over time in their ability to scale [54]. Evolutionary Algorithms (EAs) have been shown to further improve the optimization process [5, 20, 24, 34].
Recently, a leading population-based EA, Population-Based Bandits (PB2) [24], was improved by formulating hyper-parameter optimization as a Gaussian Process (GP) bandit optimization of a time-varying function [45].

Fusion modeling in particular is marked by a significant exposure to hyper-parameters. Both the 3D-CNN and SG-CNN models have their own hyper-parameters which enable them to learn the binding affinity prediction problem optimally in isolation. Additionally, the fusion layers require another set of hyper-parameters necessary to find an optimal non-linear combination of the two models. With this in mind, we saw an opportunity for improving fusion significantly by using a PB2 automated optimization algorithm [45] on Lassen, one of the most powerful high-performance computers in the world [39]. The libraries used to define the model and optimization architectures are PyTorch [46], Pytorch Geometric [14], and Ray/Ray[ Tune] [43].

Finally, a new formulation of fusion, the Coherent Fusion model, was developed as a potential improvement to the previous Late and Mid-level Fusion models. In both the existing fusion approaches, a 3D-CNN and SG-CNN are individually optimized to minimize the mean squared error (MSE) between their predictions of binding free energy and ground-truth experimental values. The existing Mid-level approach then combines the independently optimized models to form a stronger predictor, by learning the latent space strengths of each model. However, in both Late and Mid-level Fusion, the 3D-CNN and SG-CNN weights are unaltered and remain in the state that was optimal for isolated prediction. Given the Late and Mid-level Fusion models\textsuperscript{A2} superior performance compared to their isolated components, we hypothesized that fusion might be further improved by coherently backpropagating gradient through both the fusion layers and the separate models. In doing so, the Coherent Fusion model fine-tunes both the 3D-CNN and SG-CNN heads to cooperatively exploit their strengths in a joint optimization. The drawbacks of Coherent Fusion are both an increased hyper-parameter search space and number of trainable parameters. To address this, we developed a parallel, distributed hyper-parameter optimization training architecture. Compared with the models in [27], the combination of these modifications to the concept of Fusion led to significant differences in the hyper-parameters for the 3D-CNN, SG-CNN, and Fusion layers of the model.

3 OPTIMIZATION AND EVALUATION

3.1 Data

PDBbind-2019 is a curated subset of the larger Protein Data Bank (PDB) [61], which is widely used to tune biophysics- and machine learning-based methods [13, 26, 27, 37, 64]. The PDBbind data set is comprised of crystal structures arranged into two groups (general and refined) based on size (where protein-ligand compounds containing a ligand with molecular weight >1000 Daltons (Da) are excluded from the refined set), data quality (where compounds with a measured IC\textsubscript{50} but no K\textsubscript{i} or K\textsubscript{d} measurements are excluded from the refined set), and resolution of the crystal structure (<2.5 Angstroms). From the refined set, a third, core set is extracted using a clustering protocol based on protein-sequence similarity. The core set is compiled to represent a valid test for scoring methods by creating a high-quality subset of compounds sufficiently different from the general and refined sets. As such, we use the core set as a primary means for evaluating the Fusion methods considered and comparing against published literature.

In this study, we employ the quintile sub-sampling method from [27] to formulate training and validation sets from the PDBbind-2019 general and refined groupings. The sub-sampling is done independently on the general and refined sets and 10% of the examples from each are withdrawn to form the validation set. Quintile
3.2 Training Architecture

Our approach to train the various individual and fusion models was executed iteratively. Lassen uses the IBM Spectrum LSF Job Scheduler [22], which necessitates pausing, rescheduling, and resuming training jobs after a maximum run-time. As the hyper-parameter optimization began to converge, the range of hyper-parameter values was adjusted, when possible, to ensure the lower and upper-bounds of the search space were not limiting factors in model performance. The full scope of hyper-parameters and ranges evaluated for each model are provided in Table 1, where the ranges are binary (T/F), a list of options, uniformly sampled, continuous variables, or not applicable (N/A).

We used the Ray/Ray[Tune] [43] Python library extensively as the foundation for running trials of individual hyper-parameter combinations within the context of a PB2 optimization. Together, Ray and PyTorch provide the ability to accelerate the training process by distributing individual trials across multiple nodes/GPUs. Each of Lassen's 792 GPU nodes is made up of 44 3.45 GHz Power9 CPU cores, 4 NVIDIA Volta V100 GPUs each with 16GB of memory, and 256 GB of main memory. Depending on the complexity of each model, we distributed individual hyper-parameter configurations between 1 and 12 ranks (1 rank = 1 GPU, 10 CPU cores, 64 GB memory) to expedite the training process. Each rank also utilized 24 data workers running in parallel to pre-load future batches. The combination of distributed model training and parallel data loading was central to the feasibility of an experiment with this size/scope.

The PB2 hyper-parameter optimization was initialized with a quantile fraction (λ) of 50, a time scale (T) in Epochs, a perturbation interval (t_{ready}) of 100 Epochs, and an objective function (Q) of minimum validation set MSE loss [24, 45]. The procedure begins with a population of initial, randomly sampled hyper-parameter hypotheses. As every trial reaches the perturbation interval (t_{ready}) continue, while the under-performing trials clone PB2 looks at the model's performance and determines if it is above or below the quantile fraction (λ%). The best performing trials (above λ%) continue, while the under-performing trials clone a top-performing configuration (exploiting) and modify it using a

| Hyper-parameter | 3D-CNN | SG-CNN | Fusion |
|-----------------|--------|--------|--------|
| Optimizer | Adam [29] | Adam [29] | Adam [29], AdamW [40], RMSprop [17], Adadelta [9] |
| Activation function | ReLU | ReLU | ReLU, LeReLU [65], SELU [31] |
| Batch size | 8,12,24 | 4,8,12,16 | 1,2,4,5,8,12, 16,24,28,34, 38,48,56 |
| Learning rate | 1e^{-6} - 1e^{-4} | 2e^{-4} - 2e^{-2} | 1e^{-8} - 1e^{-3} |
| Model-Specific Fusion Layers | N/A | N/A | T/F |
| Epochs | 0-150 | 0-350 | 0-500 |
| Pre-trained | T/F | T/F | T/F |
| Batch norm. | T/F | T/F | T/F |
| Dropout 1 (early) | 0.25 | 0 | 0 - 0.50 |
| Dropout 2 (mid) | 0.125 | 0 | 0 - 0.25 |
| Dropout 3 (late) | 0 | 0 | 0 - 0.125 |
| # of Fusion Layers | N/A | N/A | 3,4,5 |
| # of Dense Nodes | 40,64,88, 104,128 | N/A | 8,24,40,64, 88,104,128 |
| Residual Option 1 | T/F | N/A | T/F |
| Residual Option 2 | T/F | N/A | T/F |
| # of Conv. Filters 1 | 32,64,96 | N/A | 32,64,96 |
| # of Conv. Filters 2 | 64,96,128 | N/A | 64,96,128 |
| Non-covalent / Covalent | N/A | 2,3,4, 5,6,7,8 | 2,3,4, 5,6,7,8 |
| Non-covalent / Covalent Neighbor Threshold | N/A | 1.2Å-5.9Å | 1.2Å-5.9Å |
| Non-covalent / Covalent Gather Width | N/A | 8,24,40,64, 88,104,128 | 8,24,40,64, 88,104,128 |

Table 1: Hyper-parameters for each model and their range of values considered by the PB2 optimization.
parallel GP-bandit optimization (exploring). The training process produces both an optimal model and important information about how the hyper-parameters considered effect performance.

### 3.3 Model Architectures

We draw heavily from the original FAST network architectures in [27], which holds detailed descriptions of pre-processing, feature extraction, voxel grid sizing and atom propagation, which were unaltered. The following focuses on updates to the models and the final optimized hyper-parameter configurations for each component. For brevity, we list only the final optimized hyper-parameter values, where the advantage of PB2 is in its ability to learn a schedule of hyper-parameters to converge in an end-state [45].

#### 3.3.1 Individual Models.

The SG-CNN in this work is structurally unaltered from [27], which uses the PotentialNet [13] architecture based on Gated Graph Sequence Neural Networks [36]. The only notable difference is the size of the dense layers were set according to the Non-covalent Gather Width, such that it was sequentially reduced in size by a factor of 1.5 and then 2. A population of 90 SG-CNN trials produced the final model and hyper-parameter configuration given in Table 2.

The 3D-CNN model is slightly modified from the architecture in [27]. The model has dropout above the first two dense layers, 2 additional convolutional layers, the filter sizes begin at 5x5x5 and reduce to 3x3x3, the residual options shown in Figure 1 were fed to the hyper-parameter optimization, and similar to the SG-CNN, the second dense layer size was determined by the optimization and then sequentially reduced by a factor of 2. Again, a population of 90 trials was used, the final hyper-parameter values are given in Table 3, where the optimization converged to using the second residual connection shown in Figure 1 and 32 to 64 filters for the 5x5x5 and 3x3x3 convolutional layers respectively. With this larger 3D-CNN architecture (deeper than in [27]) we found it beneficial to augment the input matrices for the training set by randomly rotating the input data in X, Y, and Z, each with a 10% probability of occurring. This random rotational augmentation was applied only to the voxelized representation of a compound. While the compound is fundamentally the same, altering its presentation to the model helps to prevent overtraining (e.g., learning rotation-dependent features) and to increase the effective size of the training data set.

#### 3.3.2 Late / Mid-level Fusion.

The Late Fusion method was implemented the same as in [27]. On the other hand, the optimization led the Mid-level Fusion model to a modified structure. For Mid-level Fusion, every optional layer (dashed lines) in the yellow Fusion block of Figure 1 was turned on. Table 4 gives the final hyper-parameters for Mid-level Fusion, which are the output of a 180 individual trial population. The other minor differences are a SELU [31] activation was selected over the previous Leaky-ReLU activation [65], a final batch size of 1, and the usage of light dropout instead of none.

#### 3.3.3 Coherent Fusion

The Coherent Fusion model is slightly modified from the architecture in [27]. The model has dropout above the first two dense layers, 2 additional convolutional layers, the filter sizes begin at 5x5x5 and reduce to 3x3x3, the residual options shown in Figure 1 were fed to the hyper-parameter optimization, and similar to the SG-CNN, the second dense layer size was determined by the optimization and then sequentially reduced by a factor of 2. Again, a population of 90 trials was used, the final hyper-parameter values are given in Table 5, where the optimization converged to using the second residual connection shown in Figure 1 and 32 to 64 filters for the 5x5x5 and 3x3x3 convolutional layers respectively. With this larger 3D-CNN architecture (deeper than in [27]) we found it beneficial to augment the input matrices for the training set by randomly rotating the input data in X, Y, and Z, each with a 10% probability of occurring. This random rotational augmentation was applied only to the voxelized representation of a compound. While the compound is fundamentally the same, altering its presentation to the model helps to prevent overtraining (e.g., learning rotation-dependent features) and to increase the effective size of the training data set.

### Table 2: Final hyper-parameters for the SG-CNN

| Hyper-parameter               | Value  |
|------------------------------|--------|
| Epochs                       | 213    |
| Batch size                   | 16     |
| Learning rate                | 2.66e^-3 |
| Non-covalent K               | 3      |
| Covalent K                   | 6      |
| Non-covalent Neighbor Threshold | 5.22Å |
| Covalent Neighbor Threshold  | 2.24Å  |
| Non-covalent Gather Width    | 128    |
| Covalent Gather Width        | 24     |

### Table 3: Final hyper-parameters for the 3D-CNN

| Hyper-parameter               | Value  |
|------------------------------|--------|
| Epochs                       | 75     |
| Batch size                   | 12     |
| Learning rate                | 4.90e^-4 |
| Batch normalization          | F      |
| # of Dense Nodes             | 128    |
| # of Conv. Filters 1         | 32     |
| # of Conv. Filters 2         | 64     |
| Residual Option 1            | F      |
| Residual Option 2            | T      |

### Table 4: Final hyper-parameters for Mid-level Fusion

| Hyper-parameter               | Value  |
|------------------------------|--------|
| Epochs                       | 64     |
| Batch size                   | 1      |
| Learning rate                | 4.03e^-4 |
| Batch normalization          | F      |
| Optimizer                    | Adam [29] |
| Activation function          | SELU [31] |
| Residual Fusion Layers       | T      |
| Dropout Rate 1 (early)       | 0.251  |
| Dropout Rate 2 (mid)         | 0.125  |
| Dropout Rate 3 (late)        | ~0     |
| # of Fusion Layers           | 5      |

### Table 5: Final hyper-parameters for Coherent Fusion

| Hyper-parameter               | Value  |
|------------------------------|--------|
| Pre-trained                  | T      |
| Epochs                       | 18     |
| Batch size                   | 48     |
| Learning rate                | 1.08e^-4 |
| Batch normalization          | F      |
| Optimizer                    | Adam [29] |
| Activation function          | SELU [31] |
| Residual Fusion Layers       | F      |
| Dropout Rate 1 (early)       | 0.386  |
| Dropout Rate 2 (mid)         | 0.247  |
| Dropout Rate 3 (late)        | 0.055  |
| # of Fusion Layers           | 4      |
Table 6: Performance of Fusion models on the PDBbind core set crystal structures

| Model         | RMSE | MAE  | $R^2$ | Pearson R | Spearman R |
|---------------|------|------|-------|-----------|------------|
| Pafnucy [56]  | 1.42 | 1.13 | -     | 0.78      | -          |
| Mid-level Fusion | 1.38 | 1.10 | 0.596 | 0.778     | 0.757      |
| Late Fusion  | 1.33 | 1.07 | 0.623 | 0.813     | 0.805      |
| Coherent Fusion | 1.30 | 1.05 | 0.640 | 0.807     | 0.802      |
| KDeep [26]   | 1.27 | -    | -     | 0.82      | 0.82       |

3.3.3 Coherent Fusion. In developing the Coherent Fusion model, it was unclear whether the same 3D-CNN and SG-CNN hyperparameter configurations found to be optimal in isolation would also be ideal for their collaborative prediction. As such, we gave the optimization the option to load the models individually trained for prediction or re-define their structure and train each head from scratch. Using the pre-trained models led to a significant improvement in validation loss. Therefore, Table V gives the final hyperparameters for the best performing Coherent Fusion model, which loads the weights from the SG-CNN in Table 2 and 3D-CNN model described in Table 3.

The Coherent Fusion model experiment optimized a population of 270 individual trials to produce a best performer. Interestingly, the Coherent Fusion model converged to exclude the model-specific dense layer options the Mid-level Fusion model uses (Figure 1) and used a simpler (4 fusion layers) architecture overall. Additionally, the Coherent Fusion used a larger batch size of 48 and significantly stronger dropout. Across the board, the Coherent Fusion model preferred a simpler Fusion architecture with significantly stronger regularization. Our intuition for this phenomenon is that the Coherent Fusion model adjusting a larger set of learned parameters allows for a simpler architecture, faster convergence, and heavier regularization compared to the Mid-level Fusion model, which serves as preliminary evidence of a stronger predictor.

3.4 Evaluation Results

Over 60,000 Lassen GPU hours were used to optimize the various models. All model iterations and intervals were not run across the same number of nodes, but our training architecture was run at its peak across 66 Lassen nodes capable of over 7,300 TFLOPS using 2904 CPU cores and 264 GPUs to train in parallel.

In Table 6, the Coherent Fusion model is shown to outperform the Late and Mid-level Fusion methods on the PDBbind core set of 290 compounds. While the difference between the Late and Coherent Fusion methods is only 0.03 RMSE, the genetic optimization of Coherent Fusion produced several nearly identical models, which consistently performed better than Late Fusion in all evaluated metrics. Importantly, the Coherent Fusion model converged to a model structure using an automated process that exceeds the performance of the hand-crafted fusion architecture used in the original Mid-level Fusion [27]. Additionally, we provide a comparison to two other deep learning approaches (KDeep [26] and Pafnucy [56]) to view Coherent Fusion’s performance in a wider scope. While the PDBbind core set is a standard benchmark for machine learning methods [26, 27, 56], high-throughput virtual screening overwhelmingly relies to docked poses of compounds for drug discovery. To follow suit, we leveraged ConveyorLC [70] to compare Coherent Fusion against physics-based scoring functions in the docking space. The noisier docking data also provides insight into whether the machine learning model was over-trained and how robust it is to noise when scoring more realistic data.

197 compounds from the PDBbind core set were successfully evaluated by ConveyorLC with the physics-based Autodock Vina algorithm [58] and MM/GBSA methods for comparison with Coherent Fusion. Each compound was then filtered by $RMSE$ where each of the 197 compounds were checked for a pose with $RMSE < 1Å$ such that a correct pose was found and sufficiently similar to the crystal structure from PDBbind. Using the binding affinity values from PDBbind as ground-truth for the docking poses, Vina achieved a Pearson correlation coefficient of 0.579, MM/GBSA scored 0.591, and Coherent Fusion reached 0.745. To further examine the performance difference between the three methods, binding affinity prediction can be cast as a binary classification problem [27]. Figure 2 shows the results on a subset of the 197 core set compounds. Positive and negative classes were created from 57 stronger binders and 71 weaker binders, respectively. Because the set of strong vs. weak docked poses is small (128 total), we elected to compare the different methods using a Precision-Recall Curves and $F_1$-scores which give a much more direct picture of how each model is performing than a ROC curve provides.
The nominally small scoring improvements such as, Coherent vs. Late Fusion (0.02 MAE) or Coherent Fusion vs. MM/GBSA (0.06 $F_1$-score), have an amplified value in large-scale screening. For example, consider a hypothetical virtual screen evaluating 1 million compounds to subsequently purchase 100 compounds (0.01%) for experimentation. If the results from classifying the PDBBind core set docking complexes translated to the top 100 of 1 million candidates with a factor of 10 decrease in precision, a virtual screen using MM/GBSA would produce 7 true positives and Coherent Fusion-based screen would produce 9 true positives in $< 1/100^{th}$ of the computational time (Table 7). While significant caveats apply, any additional true positive binders are valuable, as in practice, inhibitory compounds are hard to come by and must meet additional pharmacokinetic and safety requirements. With this analysis, we considered the Coherent Fusion model valuable and validated for use in screening for SARS-CoV-2 inhibitors.

4 HIGH-THROUGHPUT SCREENING

Our SARS-CoV-2 effort screened over 500 million compounds against each of the 4 Mpro and spike targets, drawing compounds from four public virtual compound libraries [33]. The ZINC database [57] was used to create a set of "world-approved 2018" drugs from a list of FDA-approved and "world-not-FDA" approved drugs. An additional 1.5 million compounds were selected from ChEMBL [15]; 18 million compounds were drawn from eMolecules [11], and the remaining compounds came from Enamine’s list of drug-like compounds estimated to be synthetically feasible [12].

SMILES strings [63] from the eMolecules and Enamine databases and the 2D SDF structures from the ZINC and ChEMBL libraries were downloaded, respectively. Both forms of input were imported into MOE and written in parallel to HDF5 files. In the context of Lassen's Job Scheduler LSF [22], we formulated the Fusion evaluation jobs as many, individual 4 node processes, each of which executes a set docking complexes translated to the top 100 of 1 million candidates.

4.1 Physics-based Screening Pipeline

As with our comparison between Coherent Fusion and physics-based binding affinity scoring functions, we leveraged the existing ConveyorLC [70] tool chain to search for candidate inhibitors of the two binding sites from spike and active sites for two SARS-CoV-2 protease crystal structures. ConveyorLC is made up of four parallelized programs each designed to handle a specific task in the molecular docking and re-scoring processes. ConveyorLC uses CDT1Receptor to perform protein preparation, CDT2Ligand for ligand preparation, CDT3Docking performs the molecular docking, and finally CDT4mmgbsa handles MM/GBSA re-scoring. Further details on the exact execution, pre-processing, and parameters used in the docking simulations can be found in previous studies [33, 70].

The Vina scoring used in CDT3Docking, operates at approximately one minute per compound per CPU core. On a single Lassen node with 40 CPU cores (each core has 4 hardware threads) using 8 Monte Carlo simulations per compound, Vina is able to dock ≈10 docking poses per second. In contrast, a single-point MM/GBSA score takes 10 minutes per docking pose per CPU core. Because of its computational cost, MM/GBSA is often used as a re-scoring function to refine an already filtered set of compounds. [64]. Even on a Lassen node, MM/GBSA is only capable of re-scoring ≈0.067 poses per second.

4.2 Distributed Fusion Predictions

In order to screen millions of compounds against SARS-CoV-2, we developed a scalable architecture around the Coherent Fusion model for rapid evaluation (Figure 3). The Coherent Fusion model occupies 1.5 GB GPU memory, which fits on each 16GB NVIDIA Volta V100 GPU. The remainder of the GPU memory is used to simultaneously load 56 individual docked poses into a batch alongside each model. The 4 model instances on each node were given 12 parallel data loaders to accelerate inference. Each model in every job is assigned approximately one minute per compound per CPU core. On a single Lassen node with 40 CPU cores (each core has 4 hardware threads) using 8 Monte Carlo simulations per compound, Vina is able to dock ≈10 docking poses per second. In contrast, a single-point MM/GBSA score takes 10 minutes per docking pose per CPU core. Because of its computational cost, MM/GBSA is often used as a re-scoring function to refine an already filtered set of compounds. [64]. Even on a Lassen node, MM/GBSA is only capable of re-scoring ≈0.067 poses per second.

Figure 3: Structure of a single Fusion scoring job (top). A job begins with 2 million poses to score, divides them per node, then each node assigns poses to its ranks and scores. Individual ranks (bottom) take their assigned poses, begin loading batches into memory and feeding them to the GPU for inference. Finally, identifiers and predictions are collected and written in parallel.
each assigned to evaluate an independent set of 2 million poses, which is approximately 200,000 compounds. This format was also a response to our encountering a wide range of errors (bad metadata, node failure, broken pipe errors, etc...), which led to our pipeline being tailored for fault tolerance. With this architecture, when a job fails it has minimal impact on overall throughput (another job takes its place), the reason for failure is easier to pinpoint (log files are smaller and easier to parse), and only a small set of compounds are affected or need to be rescheduled.

Table 7: Throughput for Fusion prediction single job (2 million poses) and peak performance (125 parallel jobs)

| Metric                  | Single Job | Peak |
|-------------------------|------------|------|
| Avg. Startup            | 20 min.    | ?    |
| Avg. Evaluation         | 280 min.   | ?    |
| Avg. File Output        | 6.5 min.   | ?    |
| Poses per sec.          | 108        | 13,594 |
| Poses per hour          | 338,800    | 48,600,000 |
| Compounds per hour      | 33,880     | 4,860,000 |

To create each 4 node job, we relied heavily on Horovod [52], which is based on MPI concepts and uses MPI for cross-node communication. With 4 GPUs on each node, each job is a 16-rank distributed process. Each rank runs a Python script and is given a specific GPU, CPU cores, and memory allocation to execute the evaluation. At the beginning of a job, we simply divide the set of compounds assigned to the job by the number of ranks and assign each rank the subset with its index. When evaluation completes, the ranks use allgather to compile the results and subsequently write out HDF5 files. File output was identified as a bottleneck to the evaluation process early on, which was mitigated by assigning each rank compounds to be written in the same files and directories. The output file format was designed to mirror the output format of ConveyorLC’s CDT3Docking process [70] for interpretation with existing tools and further evaluation of pharmacokinetic and safety properties [33].

Table 7 includes a breakdown of how time is spent in an individual job, where the initial 20 minutes are consumed by loading HPC modules, an Anaconda [23] environment, initializing the Horovod ranks, loading an instance of the Fusion model onto each GPU, and pre-loading the initial batches of data for evaluation. The bulk of a job is, as expected, spent in an evaluation period, where batches are loaded, evaluated, and predictions are stored in parallel across all ranks. Finally, once the ranks gather together. The file writing process begins and completes about 6.5 minutes later yielding an average total run-time of approximately 5.1 hours.

With jobs designed for scalability, we regularly ran more than 10 at a time during the SARS-CoV-2 screening effort. However, at set times the majority of Lassen nodes were made available to accelerate evaluation. The peak of which was an allotment of 500 nodes for Fusion screening. The impact of a large number of nodes is clearly seen in Table 7, where throughput was increased more than 100 times. Ultimately, during several hours of evaluation at scale, the Coherent Fusion model used more than 14,010 TFLOPS of Lassen’s compute power to screen nearly 5 million compounds per hour. The throughput advantage of Fusion is clear and compared with Vina and MM/GBSA, the Fusion model scoring code provides a 2.7x and 403x speed increase, respectively.

4.3 Single Job Scalability & Bottlenecks

While fault tolerance encouraged the use of many individual jobs each with a small number of nodes, the optimal scale of an individual job was not immediately obvious. Several factors contributed to selecting 4 nodes as optimal including: the 12 hour job run-time limit on Lassen, the startup overhead of a job, the benefit of additional nodes, and the stability of the Python libraries used.

The first parameter explored in development of the Fusion screening capability was the number of poses to evaluate per job. We found it possible to complete up to 5 million scored poses on 4 nodes under the 12 hour Lassen time limit. However, the prevalence of unpredictable errors in the docking data, featurization steps, and inter-node/rank communication, led us to instead assign 2 million poses to each job. While 2 million poses is less efficient (i.e., startup overhead is a larger percentage of each job’s run-time), in practice this decision led to less wasted computational time, as the Fusion scoring code does not write results until it finishes scoring all poses. In future work, efficiency will be improved by creating a separate, parallel process per rank to write results as they are computed, but due to the urgent need for SARS-CoV-2 predictions we elected to mitigate unpredictable errors by narrowing the size of each Fusion job to balance fault-tolerance and computational efficiency.

The next two parameters explored were the batch size per rank and number of nodes per job. The batch size per rank effects the number of times data is transferred from CPU->GPU and predictions from GPU->CPU. For the Mpro and spike target sites, we found up to 56 poses (each with voxelized and connectivity representations) could consistently fit on the NVIDIA V100 GPU. Figure 4 displays the effect three different batch sizes had on a single job.
In practice, the performance difference between each batch size was small with a batch size of 56 yielding a ≈10 minute run-time advantage over batch size 12.

The Fusion scoring code under-utilized the Lassen GPUs, which led to consistent and relatively small offsets in run-time by batch size. This is due to the computational cost of pre-processing (e.g., file reading and data featurization), which is the most significant bottleneck in the evaluation process. Despite using 12 parallel data loaders per rank, the GPU is intermittently waiting to evaluate more poses. In future work, further optimization of the parallel data loaders will increase GPU utilization and improve throughput, which increases the value of Fusion for large-scale virtual screening.

To determine the optimal number of nodes per job and performance benefit of additional nodes, we evaluated the run-times of Fusion jobs using 1, 2, 4, and 8 nodes per job. Figure 4 also displays the performance of each number of nodes where the same set of 10 jobs was run for every point evaluated. The variance between sets is also plotted surrounding each line, but was found to be small (< 5 minutes) and as a result is not clearly visible.

A significant factor in choosing the number of nodes for each job was stability of the Python libraries used by the Fusion model. Inter-node and inter-rank communication errors were increasingly prevalent as the number of nodes/ranks in a job increased and the percentage of failed jobs for each number of nodes was ≈2% for 1 and 2 nodes, ≈3% for 4 nodes, and ≈20% of jobs failed when using 8 nodes. The instability was caused by the specific combination of Horovod [52] and PyTorch [46] used on the POWER9 architecture, which has since been updated. However, a 20% job failure rate eliminated 8 nodes per job as a candidate configuration.

The results of our scalability experiments led to a final selection of 2 million poses per job using 4 nodes. This was the result of several different factors and adjustments including the observation that when using 500 Lassen nodes at the same time, the LSF scheduler encountered problems simultaneously running 250 2-node jobs, which was solved by using 125 4-node Fusion jobs instead.

### 5 SARS-COV-2 RESULTS

The high-throughput virtual drug screening pipeline described in Section 4 produced several computational results for each compound screened. The Fusion model AZs binding-affinity prediction was one of the three energy calculations (Vina, MM/GBSA, Fusion), which were used as a component of a hand-tailored cost function designed to filter which compounds to purchase for experimental evaluation and which were less likely to be successful. Full details of the ranking and reasoning may be found in [33] and computational predictions are made available at https://url-excluded [38]. Virtual screening output on the computational side fed directly into an experimental process to physically interrogate candidate molecules.

#### 5.1 Experimental Validation

Experimental testing of the candidate binders which were screened and purchased to target M\(^{\text{pro}}\) used a fluorescence resonance energy transfer (FRET) based activity assay or a SDS-PAGE gel protein cleavage assay. After assay optimization, additional screens were run in order to down select compounds. For example, compounds from the ZINC database [57] were down selected to an additional testing of 19 compounds, which yielded 4 candidates inhibiting the activity of M\(^{\text{pro}}\) at 100\(\mu\text{M}\) and 300 compounds against M\(^{\text{pro}}\) protease1 (blue) and 81 compounds against M\(^{\text{pro}}\) protease2 (orange). The spike assays were evaluated at 10\(\mu\text{M}\) and include 151 compounds against spike1 (green) and 113 compounds against spike2 (red). Compounds which exhibited ≤1% inhibition (no experimental binding activity) are excluded.

Figure 5: Coherent Fusion predicted binding affinity vs. experimental percentage of inhibition at 100\(\mu\text{M}\) for 130 compounds against M\(^{\text{pro}}\) protease1 (blue) and 81 compounds against M\(^{\text{pro}}\) protease2 (orange). The spike assays were evaluated at 10\(\mu\text{M}\) and include 151 compounds against spike1 (green) and 113 compounds against spike2 (red). Compounds which exhibited ≤1% inhibition (no experimental binding activity) are excluded.

On the other hand, compounds predicted to inhibit the SARS-Cov-2 spike protein were screened by both a pseudo-typed virus assay and a biolayer interferometry competitive assay (BLI). Here the candidate compounds are being evaluated for their ability to inhibit ACE2-spike binding and in parallel, the spike binding candidates were screened using a cell-based infection assay at 10\(\mu\text{M}\). Further details of the experimental design, assays, results, and discussion can be found in [33].

#### 5.2 Connecting Predictions and Experimental Results

Given the ground-truth experimental values for physically tested compounds, we enabled to retrospectively evaluate the accuracy of each computational approach which generated a prediction. Some of the obvious questions to ask are: “Which method was most correlated with the experimental results?”, “Are the most accurate scoring functions the same for all four M\(^{\text{pro}}\) and spike targets?”, and “Which of the scoring methods is most accurate for the strongest experimental inhibitors?”. Each experimentally prosecuted compound can be traced back to its virtually docked poses for either
the two M\textsuperscript{pro} or two spike binding targets. This means each scoring method may have predicted binding affinity values for up to 40 poses per compound (10 poses maximum per binding site). While in the computational domain we have predicted binding affinity values, the output from the M\textsuperscript{pro} and spike assays is a percentage of inhibition normalized between 0 and 100%. These values are produced at a given concentration of the candidate drug (100 µM for M\textsuperscript{pro} targets and 10 µM for spike targets), which gives context to the overall strength of a binder.

For each scoring method (Vina, MM/GBSA, Coherent Fusion) the results per compound were aggregated and represented by the strongest prediction across all poses for each binding site (maximum for Coherent Fusion, minimum for Vina and MM/GBSA). Because of the computational cost of MM/GBSA, we instead use the ATOM Modeling PipeLine (AMPL) MM/GBSA predicted MM/GBSA values, which have been shown to be highly correlated with actual MM/GBSA calculations and were trained to predict MM/GBSA scores on each specific target [42].

5.3 Analyzing Computational Predictions

Following this aggregation, the output is a single prediction per compound tied to a single percent inhibition. Each method can then be viewed as a scatter plot comparing predicted vs. actual experimental results as in Figure 5. We computed Pearson and Spearman correlation coefficients for each method across all experimentally tested M\textsuperscript{pro} and spike compounds. However, most experimentally tested compounds are negatives (≤ 1% inhibition), which gives correlation coefficients near 0 for each of the three evaluated methods (table excluded for brevity). In an attempt to focus our analysis on the relative strengths and weaknesses of the different scoring methods and not the difficulty of the overall binding affinity prediction problem, we computed correlation coefficients for each method on the subset of compounds for which any experimental binding (>1%) was observed.

Table 8 shows the correlations for all methods where the absolute value of the Vina and MM/GBSA scores are used. AMPL MM/GBSA gives the best correlation for the protease1 target, Coherent Fusion for the protease2 and spike1 targets, and Vina scores the spike2 binding site best. However, across the board, it is clear that even when limiting the analysis to >1% inhibition, the correlations for each method remain low and the interpretation of near-zero correlation coefficients is unavailing.

While removing all the non-inhibitors gives a glimpse into which methods are most correlated with the SARS-CoV-2 binders, it also removes the context of those predictions. That is, the overall prediction strength for each method is somewhat obscured as the range of each method’s prediction values is limited to its minimum and maximum prediction in the smaller set of SARS-CoV-2 binders. With this in mind, we sought to answer the question of which scoring methods were most accurate for the strongest experimental inhibitors by including non-binding compounds and casting the prediction problem as a binary classification of compounds with >33% inhibition (positive class) and compounds with ≤ 33% inhibition (negative class). A threshold of 33% was chosen to avoid severe class
imbalances caused by higher thresholds. This problem formulation is similar to that in Figure 2 where we set a threshold to separate stronger binders from weaker binders. This approach is applicable in practice, as virtual screens eventually down select to a small set of candidates for final analysis, purchase, and experimental testing.

Figure 6 displays Precision/Recall Curves and \( F_1 \)-scores using a threshold of 33% to separate the positive and negative binding examples for each target. The y-axis of each P/R curve is limited to 0.35 to observe different model behaviors. Although the curves for each target appear different from the P/R plots in Figure 2, they provide information about how each model performs in practice on a noisier experimental screen, where the cutoff separating active from inactive molecules is less clear.

While the P/R Curves in Figure 6 give low \( F_1 \)-scores and precisions, two important observations arise. First, for each of the four binding sites, we computed Cohen's Kappa statistic (\( \kappa \)) to compare each model with a random classifier \([7, 32]\). Equation 2 gives the equation for computing \( \kappa \):

\[
\kappa = \frac{\rho_o - \rho_e}{1 - \rho_e}
\]

where \( \rho_o \) is the observed accuracy and \( \rho_e \) is the expected accuracy. A random classifier achieves a \( \kappa = 0 \) by guessing according to the frequency of the positive and negative classes. Every model on each target was measured to achieve a \( \kappa > 0 \) (with the exception of Vina on spike1), indicating the models are consistently better than random across the targets. In the context of P/R Curves, a random classifier's \( F_1 \)-Scores performance is expressed as a horizontal line (Figure 6) with constant expected precision equal to the proportion of positive examples in the data set.

In practice, the three binding affinity prediction models served as input to a hand-crafted cost function \([33]\), to select compounds for experimentation. The outcome of which produced 9 distinct combinations for final analysis, purchase, and experimental testing. The outcome of which produced 9 distinct combinations for final analysis, purchase, and experimental testing. In that, while 500+ million candidates were evaluated, only a fraction (2.1\% to 3.5\%) of the best candidates were tested experimentally, yet the models successfully yielded a 10.4\% hit rate.

Figure 7 presents four top compounds bound to two of the four sites/configurations used in the virtual screen (\( M^{\text{pro}}/\text{protease1} \) and spike/spike1). The \( M^{\text{pro}}/\text{protease1} \) compounds reached 100% inhibition and had Coherent Fusion predicted binding affinities of 8.5 (Fig. 7a) and 8.1 (Fig. 7b). The spike RBD compounds evaluated at 10\( \mu \)M had predicted affinities and experimental inhibitions at 7.6/100\% (Fig. 7c) and 8.3/98\% (Fig. 7d).

For the \( M^{\text{pro}} \) protease1 binding site, AMPL MM/GBSA is the best predictor of experimental binding. Additionally, between the two \( M^{\text{pro}} \) binding sites, protease1 and protease2, the AMPL MM/GBSA predictions on protease1 jointly achieve a higher \( F_1 \)-score and Pearson correlation coefficient than all other methods. In fact, for each of the four targets, not only do the best >33\% \( F_1 \)-scores match with the best >1\% Pearson/Spearman correlations, but the relative differences across target binding sites are also in agreement.

Coherent Fusion reaches the maximal \( F_1 \) and correlation coefficients for the \( M^{\text{pro}} \) protease2 and spike1 binding sites. Across all model / binding site combinations, Coherent Fusion's \( F_1 \)-Scores performance on the spike1 protein is the top performer, though followed closely by AMPL MM/GBSA on the same target site. This is unexpected, as the protease targets are much larger and nominally thought to provide better opportunity for drug-like compounds to bind.

However, target specific strengths are not unexpected \([64]\). Especially where the \( M^{\text{pro}} \) sites are large protein pockets and the spike targets are much smaller. Interestingly, the maximal >1\% correlations and >33\% \( F_1 \)-scores across all targets favor the spike proteins. The differing concentrations between the \( M^{\text{pro}} \) and spike experiments is important to note, as \( M^{\text{pro}} \) binders were evaluated at 100\( \mu \)M, which is a higher drug concentration and, therefore, allows for weaker binders to exhibit higher observed percentages of inhibition.
6 CONCLUSION AND FUTURE WORK
Deep Fusion was improved by coherent backpropagation and distributed, genetic hyper-parameter optimization. The optimization automatically produced an updated version of the Coherent Fusion model, which was shown to exceed the performance of hand-crafted Fusion model variants and alternative machine learning methods on crystal structures from the PDBbind core set benchmark. In evaluating noisier docked poses of the same test set, Coherent Fusion also achieved an improvement in correlation and $F_1$-score relative to the physics-based Vina and MM/GBSA methods.

We utilized Coherent Fusion to screen over 500 million compounds across four SARS-CoV-2 binding sites. This was achieved by designing a high-throughput distributed architecture for fault-tolerant scoring at scale. Using parallel data loaders and LassenÅÁs 4 GPUs per node, the Coherent Fusion model is currently capable of screening ≈30 poses per second per node. Coherent Fusion was used as input to a weighted cost function across binding affinity and other scoring models in order to sub-select compounds for experimentation.

Among the nearly 1000 compounds tested experimentally, several inhibited activity of $\text{M}^{\text{pro}}$ and spike. In analyzing which binding affinity methods were most correlated with the experimental results, we found the optimal performer to vary by target. However, two of the four targeted binding sites were best predicted by Coherent Fusion both in terms of >1% inhibiting correlation with the experimental results and >33% inhibiting binary classification. Although overall predictive power of all of the scoring functions is limited, any opportunity for computational enrichment of strong binders is needed to alleviate the otherwise prohibitive time and cost of the physical experimental screens.

In future work, we aim to use our baseline Coherent Fusion model from this work to fine tune and predict for specific protein target types and binding sites. We believe introducing target specificity to the models and thereby reducing the scope of the binding affinity prediction problem will increase the value of relative differences in the modelÅÁs binding affinity predictions. Our high-throughput architecture can also be accelerated, as the GPUs on each Lassen node were observed to be under-utilized. Increased numbers of parallel data loaders were observed to decrease the overall stability of each individual evaluation job indicating some refactoring is necessary.

ACKNOWLEDGMENTS
The authors gratefully acknowledge Professor Xinquan Wang (Tsinghua University) for providing early access to his crystal structure of the ACE2-RBD complex. The authors gratefully acknowledge extensive computer time provided by Livermore Computing. Part of this 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. The authors thank Lawrence Livermore National Laboratory for funding Laboratory Directed Research and Development projects 20-ERD-065 and 20-ERD-062. Part of this research was also supported by the American Heart Association under CRADA TC02274-4 and the National Nuclear Security Administration through the Accelerating Therapeutics for Opportunities in Medicine (ATOM) Consortium under CRADA TC02349. This work was funded in part by DTRA under award HDTRA1036045. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of EnergyÅÁs National Nuclear Security Administration under contract DE-NA0003525. All work performed at Lawrence Livermore National Laboratory is performed under the auspices of the U.S. Department of Energy under Contract DE-AC52-07NA27344. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

REFERENCES
[1] Qurrat Ul Ain, Antoniya Aleksandrova, Florian D Roessler, and Pedro J Ballester. 2015. Machine-learning scoring functions to improve structure-based binding affinity prediction and virtual screening. Wiley Interdisciplinary Reviews: Computational Molecular Science 5, 6 (2015), 405–424.
[2] Pedro Ballester and John Mitchell. 2010. A machine-learning approach to predicting protein-ligand binding affinity with applications to molecular docking. Bioinformatics (Oxford, England) 26 (03 2010), 1169–75. https://doi.org/10.1093/bioinformatics/btp112
[3] Sandrine Belouzard, Jean K Millet, Beth N Licitra, and Gary B Whitaker. 2012. Mechanisms of coronavirus cell entry mediated by the viral spike protein. Viruses 4, 6 (2012), 1011–1033.
[4] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for hyper-parameter optimization. In 25th annual conference on neural information processing systems (NIPS 2011), Vol. 24. Neural Information Processing Systems Foundation.
[5] PA Castillo, J Carpio, JJ Merelo, Alberto Prieto, V Rivas, and Gustavo Romero. 2000. Evolving multilayer perceptrons. Neural Processing Letters 12, 2 (2000), 115–128.
[6] Tiejun Cheng, Qingliang Li, Zhigang Zhou, Yani Wang, and Stephen H Bryant. 2012. Structure-based virtual screening for drug discovery: a problem-centric review. The AAPS journal 14, 1 (2012), 133–141.
[7] Jacob Cohen. 1960. A coefficient of agreement for nominal scales. Educational and psychological measurement 20, 1 (1960), 37–46.
[8] Kurt LM Drew, Hikam Bainam, Prashanna Klwaraunjoo, Bo Yu, and Johannes Reynisson. 2012. Size estimation of chemical space: how big is it? Journal of Pharmaceutical and Pharmacology 64, 4 (2012), 490–495.
[9] John Duch, Eld Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. Journal of machine learning research 12, 7 (2011).
[10] Sally R Ellingson, Brian Davis, and Jonathan Allen. 2020. Machine learning and ligand binding predictions: a review of data, methods, and obstacles. Bioinformatics & Biophysica Acta (BBA)-General Subjects 1864, 6 (2020), 129545.
[11] eMolecules. 2021. eMolecules. https://www.emolecules.com/
[12] Enamine. 2021. Enamine. https://enamine.net/
[13] Evan N Feinberg, Debniul Sur, Zhenglin Wu, Brooke E Husie, Huanghao Mai, Yang Li, Saihsai Sun, Jianyi Yang, Bharath Ramsundar, and Vijay S Pande. 2018. PotentialNet for molecular property prediction. ACS central science 4, 11 (2018), 1520–1530.
[14] Matthias Fey and Jan Eric Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. arXiv preprint arXiv:1903.02428 (2019).
[15] Anna Gaulton, Louisa J Bells, A Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, et al. 2012. CHEMBL: a large-scale bioactivity database for drug discovery. Nucleic acids research 40, D1 (2012), D1100–D1107.
[16] Joseph Gomes, Bharath Ramsundar, Evan N Feinberg, and Vijay S Pande. 2017. Atomic convolutional networks for predicting protein-ligand binding affinity. arXiv preprint arXiv:1703.10603 (2017).
[17] Alex Graves. 2013. Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850 (2013).
[18] Chemical Computing Group ULC. 2021. Molecular Operating Environment. https://www.chemcomp.com/MOE-Molecular_Operating_Environment.htm
[19] Tingjuan Hou, Junmei Wang, Youyoug Li, and Wei Wang. 2011. Assessing the performance of the molecular mechanics/Poisson Boltzmann surface area and molecular mechanics/generlized Born surface area methods. II. The accuracy of ranking poses generated from docking. Journal of computational chemistry 32, 5 (2011), 866–877.
[68] Xiaohua Zhang, Horacio Perez-Sanchez, and Felice C Lightstone. 2017. A comprehensive docking and MM/GBSA rescoring study of ligand recognition upon binding antithrombin. Current topics in medicinal chemistry 17, 14 (2017), 1631–1639.

[69] Xiaohua Zhang, Sergio E Wong, and Felice C Lightstone. 2013. Message passing interface and multithreading hybrid for parallel molecular docking of large databases on petascale high performance computing machines. Journal of computational chemistry 34, 11 (2013), 915–927.

[70] Xiaohua Zhang, Sergio E Wong, and Felice C Lightstone. 2014. Toward fully automated high performance computing drug discovery: a massively parallel virtual screening pipeline for docking and molecular mechanics/generalized Born surface area rescoring to improve enrichment.

[71] Jingbo Zhou, Shuang Li, Liang Huang, Haoyi Xiong, Fan Wang, Tong Xu, Hui Xiong, and Dejing Dou. 2020. Distance-aware Molecule Graph Attention Network for Drug-Target Binding Affinity Prediction. arXiv preprint arXiv:2012.09624 (2020).

[72] Fangqiang Zhu, Xiaohua Zhang, Jonathan E. Allen, Derek Jones, and Felice C. Lightstone. 2020. Binding Affinity Prediction by Pairwise Function Based on Neural Network. Journal of Chemical Information and Modeling 60, 6 (2020), 2766–2772. https://doi.org/10.1021/acs.jcim.0c00026 PMID: 32338892.