Journal of Cheminformatics

REVIEW Open Access

A beginner's approach to deep learning applied to VS and MD techniques

Stijn D'Hondt[1], José Oramas[2] and Hans De Winter[1*]

Abstract

It has become impossible to imagine the fields of biochemistry and medicinal chemistry without computational chemistry and molecular modelling techniques. In many steps of the drug development process in silico methods have become indispensable. Virtual screening (VS) can tremendously expedite the early discovery phase, whilst the use of molecular dynamics (MD) simulations forms a powerful additional tool to in vitro methods throughout the entire drug discovery process. In the field of biochemistry, MD has also become a compelling method for studying biophysical systems (e.g., protein folding) complementary to experimental techniques. However, both VS and MD come with their own limitations and methodological difficulties, from hardware limitations to restrictions in algorithmic capabilities. One solution to overcoming these difficulties lies in the field of machine learning (ML), and more specifically deep learning (DL). There are many ways in which DL can be applied to these molecular modelling techniques to achieve more accurate results in a more efficient manner or expedite the data analysis of the acquired results. Despite steadily increasing interest in DL amidst computational chemists, knowledge is still limited and scattered over different resources. This review is aimed at computational chemists with knowledge of molecular modelling, who wish to possibly integrate DL approaches in their research and already have a basic understanding of the fundamentals of DL. This review focusses on a survey of recent applications of DL in molecular modelling techniques. The different sections are logically subdivided, based on where DL is integrated in the research: (1) for the improvement of VS workflows, (2) for the improvement of certain workflows in MD simulations, (3) for aiding in the calculations of interatomic forces, or (4) for data analysis of MD trajectories. It will become clear that DL has the capacity to completely transform the way molecular modelling is carried out.

Introduction

More than ever, it is absolutely clear how vital in silico techniques have become in the fields of biochemistry and medicinal chemistry. To study fundamental biological processes such as biomolecular recognition, protein folding, or the binding of a potential drug to its target, computational methods have become indispensable. Experimental techniques (e.g., nuclear magnetic resonance, X-ray crystallography, small-angle X-ray scattering) are an important first step in e.g., characterizing the 3D structure of a protein or determining the way in which a small biomolecule binds its target. However, they come with important limitations. A single technique only provides information on certain aspects of a process of interest. The power of experimental techniques lies in combining the data obtained from all experiments to compose a complete and unified look at the way in which a process takes place. This requires time and asks considerable resources. Furthermore, experimental techniques only capture static pictures of structures and complexes, whilst non-accessible intermediate states often deliver

*Correspondence:
Hans De Winter
hans.dewinter@uantwerpen.be
[1] Laboratory of Medicinal Chemistry, Department of Pharmaceutical Sciences, IDLab, University of Antwerp, Universiteitsplein 1, 2610 Wilrijk, Belgium
[2] Department of Computer Science, Sint-Pietersvliet 7, 2000 Antwerp, Belgium

valuable information that could end up being relevant for research in structural biology or drug discovery [1–3].

This is where computational methods come in useful. Widely used are virtual screening (VS) and molecular dynamics (MD) simulations. These techniques allow the evaluation of an entire process in question, rather than stills of the most stable or crystallized conformations. VS aims at predicting the binding affinities between protein–ligand, protein-peptide, or protein–protein complexes. This is a technique that allows for rather fast evaluation of a library of compounds against a query drug target and could be used as an interesting starting point for structure-based drug design. On the other hand, MD can simulate the complex dynamics and conformational landscape of proteins, as well as the binding modes of protein–ligand complexes, using a model describing the physics that oversee all interatomic interactions. These simulations form an enormously powerful tool in providing general information about a biomolecular process and can help guide a study towards logical next steps and experimental techniques [1–3].

Of course, as is the case for experimental setups, computational methods also come with their own limitations. The accuracy of their predictions is limited by the strength of the algorithms in use. For example, high system flexibility of proteins can complicate the accuracy of predictions made by molecular docking algorithms. Improvement of these algorithms is limited by current computing hardware. Algorithms that more closely parallel in vitro conditions ask more resources and, depending on the hardware available to a research group, calculations could ask significant computing time. This also imposes a limit to the timeframes MD simulations can simulate. There are nonetheless many different approaches to counteract current limitations. The availability of computing time on supercomputers, the advances in graphics processing unit (GPU) technology and improvements in methodology have already upscaled the time a simulation can realistically calculate up to several microseconds. Enhanced sampling methods (e.g., umbrella sampling, metadynamics, replica exchange MD, Gaussian accelerated MD, coarse-grained MD) have been extensively developed throughout recent years and make a significant impact on reducing the computational demands of calculations. Finally, a more recent but vital pathway to improve molecular docking and MD simulations is artificial intelligence (AI), more specifically the fields of machine learning/deep learning (ML/DL). With advancements in GPU hardware, training a neural network (NN) has become more feasible timewise, and extensively developed software frameworks like TensorFlow and PyTorch make the development of NNs much more accessible to non-experts [4–6]. Therefore, a deeper look into the basic usages of DL within computational chemistry (complemented with examples) will be the main focus of this review [1–3]. This article is aimed at computational chemists with knowledge of molecular modelling, who wish to possibly integrate DL approaches in their research and already have a basic understanding of the fundamentals of DL and neural networks. Interesting references for further learning are provided [4, 7–9], as well as a glossary (see Table 1) that provides more insight into basic concepts in the field of DL and the DL architectures mentioned throughout this review. A useful dissertation of the different overarching types of DL architectures was made by Sarker [10].

This review entails a non-exhaustive survey of recent and useful applications of DL within the field of molecular modelling, seeing how DL could be employed to improve docking accuracy, simulation efficiency and trajectory analysis [1]. The following sections were drafted to give a general description of how DL techniques could be employed at different stages of a structure-based drug design workflow, followed by examples found in literature of how such applications can be achieved in reality. First, applying DL tools for the improvement of VS methods is discussed in the "Deep learning and virtual screening" section. Then, focus shifts to MD simulations, disserting how DL could aid in guiding along simulations to sample specific objective states ("DL-guided enhanced conformational sampling of protein structures" section), as well as how NNs could learn to calculate interatomic forces and take over simple force fields (FFs) for the necessary MD calculations ("Neural network potentials" section). Lastly follows a discussion on how DL could improve and guide the analysis of MD trajectory data ("DL-guided analysis of MD trajectories" section).

To make this large review more manageable and understandable, the main body of the review is complemented with the "Review highlights" section. This entails a summary of the review paper, presenting a broad overview of the discussed methods throughout each topic, as to make their connections and trade-offs clearer. The "Relevance of DL implementations and key toolset" section summarizes the relevance of these DL implementations and presents a table with a non-exhaustive look at key tools and datasets for DL model development, VS and MD methods, meant to inspire your research. The "DL and VS" and "DL and MD"sections present an outline of the different methods discussed in respectively the "Deep learning and virtual screening"and "Deep learning and molecular dynamics simulations" sections, with tables summarizing all the presented research examples. To conclude this review, the "Conclusions and future perspectives" section offers some final remarks and relevant open research questions.

Table 1 List of key deep learning concepts mentioned throughout the main text and their explanation

DL concept	Explanation
General concepts	
Artificial intelligence (AI)	The ability of computers to mimic human behavior
Machine learning (ML)	ML is a subset of AI, capable of taking certain information as input and then using this to make an informed decision in the future. This process can either take place through optimization of the output predictions by comparing them with the actual expected output (a process called "supervised learning"), or through a process in which the ML algorithm is not explicitly told what to target with the provided input (a process called "unsupervised learning") [204, 205].
Deep learning (DL)	DL is a subset of ML, specifically focused on using NNs of more than one hidden layer to extract useful features or patterns from the input data [4, 9, 10]. DL further differs from ML in the way in which the data can be presented to the algorithm: DL can process virtually any type of information, whilst ML needs to be provided with a suitable input data type, often requiring a preprocessing or feature extraction step [10]. DL is capable of learning underlying features from data to hierarchically build up a concept/representation of the data in question, possibly in an unsupervised manner
Neural network (NN)	A NN consists of a connected series of neurons: building blocks capable of taking information as input, mathematically manipulating that data in some way, and then outputting the resulting value further down the network. Except for the input layer and the output layer, each layer of neurons can be called hidden layers (due to no transparency). The described passing of information through the NN can be called forward propagation, resulting in a prediction result in the output layer. Backpropagation is the reverse process, in which the NN adjusts certain values (weights) between neurons based on the initial outputs of the network versus the desired outputs, as to optimize its predictive power [4, 7, 194, 206].
Dense fully connected neural network (DFCNN)	The most frequent architecture of a NN, in which all input values are densely connected to all neurons of the previous layer of the NN, and their outputs are densely connected to all neurons of the next layer [4, 206].
Rectified linear unit (ReLU)	Part of the mathematical equation inside a neuron is passing an intermediate result through a non-linear activation function, as to introduce non-linearities into the network and enable complex decision-making. The ReLU activation function causes all negative integers to be outputted as 0 but knows no upper boundary for positive integers. In general, models that employ ReLUs in hidden layers tend to train faster and result in more accurate predictions [4, 7, 11, 194, 206, 207].
Model parameters	Model parameters are variables internal to the NN itself. These are values that arise from the processing of data and cannot be adjusted manually by a user whilst developing a network (e.g., weights or biases) [7, 194].
Model hyper-parameters	Model hyperparameters are values that a user can specify manually and that can be tuned to improve a NN (e.g., learning rate, stopping criteria, or regularization technique) [7, 194, 208].
Deep transfer learning (DTL)	Given that DL is a very robust and generalizable technique, the same DL model can often be repurposed for different applications through limited additional learning on a smaller training dataset compared to the datasets needed for the training of a new DL model [8].
Explainable AI (XAI)	A field of computer science focused on the understanding and interpretation of AI systems. There are many methods within the field of XAI to reach proper explainability and interpretability of AI systems, as disserted carefully by Linardatos and coworkers [209].
Concepts related to discriminative learning architectures (Learn to discriminate data between different class labels.)	
Multi-layer perceptron (MLP)	Synonym fora standard DFCNN with forward propagation and backpropagation (given that another name for neuron is perceptron) [10, 210].

Table 1 (continued)

DL concept	Explanation
Convolutional neural network (CNN)	Pattern detection architecture for data of any number of dimensions (most commonly used to analyze 2D data). The model contains special layers called convolutional layers, which analyze specific regions of the input data and create feature (activation) maps based on what features they "see" in those regions. Pooling layers take those feature maps and pool them, simplifying them further, before feeding this dimensionality reduced version of the data to a fully-connected network that can extract and learn the most interesting, essential features of the input [8, 10, 211–213].
Residual neural network (ResNet)	Classical CNN models face the vanishing gradient problem (limited learning in the early layers of the network) as more convolutional layers are added, often resulting in limited performance. A ResNet architecture provides a solution to such problems through a mechanism known as "skip connections". During initial training, this CNN-based model can skip certain layers, which speeds up the first training steps by compression of the initial network and causes appropriate learning in the deepest layers. It is then through retraining that all layers of the network are employed, and more of the feature space of the input can be explored, avoiding vanishing gradients [214].
Recurrent neural network (RNN)	Architecture for the analysis of sequential data: data points that depend on previous data points to be properly understood (e.g., the words in a sentence, the amino acid letters in a protein sequence). In RNNs, it becomes important to integrate a type of memory, in which the output of previous steps forms an additional piece of input for following steps [4, 8, 10, 211, 213].
Long short-term memory (LSTM)	Due to the vanishing gradient problem, the most basic form of RNN is unable to keep information from initial data points in mind the further down the sequence the model is processing data. Therefore, variants on the RNN architecture have been created to offer recurrent connections to earlier memories throughout the network. These variants contain memory cells in their network layers, allowing for the storage of temporal states of the current network. These states can be fed to other parts of the network to introduce a more advanced version of memory. In an LSTM RNN architecture, memory cells contain three gates: an input gate and output gate to control the flow of information in and out of the cell, as well as a forget gate, which determines what information will or will not be stored in a temporal state. In a bidirectional LSTM (BiLSTM) RNN, hidden layers are connected in both directions, so that data can be sent to a cell from both the past and future [10, 211, 215].
Gated recurrent unit (GRU)	A GRU is a simplified version of an LSTM with fewer parameters. A GRU has only two gates, called a reset and an update gate. The flow of information through such a unit is more streamlined compared to an LSTM, with comparable performance (highest on smaller datasets) and faster computing times [10, 216].
Graph neural network (GNN)	GNNs have evolved from CNNs and RNNs and are characterized by the way they represent their input data: as graphs with nodes and edges. For example, molecules can be represented as graphs, in which nodes represent atoms and edges represent bonds and/or noncovalent interactions, both characterized by user-defined features. Each node and edge are a unit of the full NN, capable of processing the input information contained within itself and its neighbors, forming an embedding. This dimensionality reduced representation of the data is then passed along to the first-order neighbors of the starting node, where new embeddings are produced. This message passing process continues until every node of the graph contains information of all other nodes. All the generated embeddings obtained from each node are gathered and summed, as to obtain one single representation of all data contained within the graph. This embedding can then form the input for a follow-up model for classification or regression predictions [217, 218].
Concepts related to generative learning architectures (Hierarchically learn to build up data points from the general features to be found in input data)	

Table 1 (continued)

DL concept	Explanation
Generative adversarial network (GAN)	A GAN can create new data points by first learning the distribution of features and patterns of an input dataset. It consists of two NNs working in tandem: a generator and a discriminator. The generator analyses an input dataset, and from random noise, it learns to create new plausible data points comparable to those in the original (real) set. The discriminator trains itself to predict the probability of a sample being from the original input data rather than from the generated data. During this process, the generator goes into competition with the discriminator in trying to produce samples that are indistinguishable from the real data points, whilst the discriminator tries to better learn feature information for distinguishing real and generated data points. This reciprocity leads to an enhancement of the generated data [4, 7, 10, 46, 219]. Interesting applications of GANs lie in the field of video or voice generation, or as generative chemistry tools (e.g., compound generation) [4, 10, 46, 47, 213, 220].
Auto-encoder (AE)	An AE learns dimensionality reduced representations of data, from which reconstructions of the original input can then be generated. An AE consists of an encoder, code (latent space) and a decoder. The encoder analyses an input and learns the underlying features and patterns of the dataset. Through this dimensionality reduction, it's able to learn how the distribution of features within a dataset is represented. It compresses the original data into a code, which is passed on to the decoder. The decoder then learns to reconstruct the original input from the code. This reconstruction is compared to the original input and the difference is minimized during the loss optimization process. Through this process, a lower-dimensional representation of the data is generated within the latent space of the code. This process is incredibly useful for many general learning tasks, including dimensionality reduction, feature extraction, generative modelling, denoising, and outlier detection [4, 10, 213, 220, 221].
Variational auto-encoder (VAE)	A VAE is a unique AE in the sense that traditional AEs map the code from the input onto a lower-dimensional latent vector, whereas VAEs map their data onto a probability distribution. A normal Gaussian distribution is used most, as it encourages the encoder to distribute the code evenly around the central point of its latent space. The data is assumed to follow a probability distribution, and its mean and variance are attempted to be estimated by the network. The decoder network then attempts to reconstruct the original input by taking samples from the probability distribution. This distribution can afterwards be used to generate new synthetic data points close to those from the original dataset[4, 10, 213, 222, 223].
Wasserstein auto-encoder (WAE)	A type of VAE that develops its latent space in such a way it can intrinsically capture the sequence relationship of peptides [65].
Convolutional variational auto-encoder (CVAE)	A CVAE functions in the same way as a VAE, mapping its data onto a probability distribution as latent space, but the hidden layers extracting or reconstructing the input data make use of convolutions (as in CNNs) [139].
Variational dynamics encoder (VDE)	The predictions made by a VDE are reconstructions of future dynamics, based upon the encoding made by current datapoints. The model is modified in such a way that it is most suitable to be trained on time-series data. For this, it considers a new hyperparameter, a lag time, which represents the time scales of the dynamics that are of interest in the research in question. After training on time-series data, the model can predict the state of the system after a timestep as big as the lag time. When done in an iterative fashion, it creates trajectories of features with dynamics consistent with those of the training system [143].
Restricted Boltzmann machine (RBM)	RBMs are stochastic neural networks with visible and hidden layers that are limited to how they are connected and able to transmit information. A basic RBM architecture consists of two layers, one visible and one hidden. Input data gets transmitted and processed from the visible layer to the hidden layer and back. Afterwards, the output generated at the visible layer is compared to the initial input data. This process repeats for optimization, which leads to the network learning an accurate representation of the input. An RBM is comparable to a VAE, in the sense that it can learn a probability distribution across its input data which can be used for feature selection, dimensionality reduction or classification tasks, and form the input for other learning processes [10, 219, 224].

This list is logically subdivided and sorted into (1) general concepts, (2) concepts related to discriminative learning architectures, and (3) concepts related to generative learning architectures

To make it more comprehensible for the reader, a list of key deep learning concepts mentioned throughout the main text and their explanation is given in Table 1. This list is logically subdivided and sorted into (1) general concepts, (2) concepts related to discriminative learning architectures, and (3) concepts related to generative learning architectures.

Even though the field of medicinal chemistry is mainly used to navigate the review paper through the different topics at hand, the described applications can be extrapolated for many different objectives within different study fields employing computational chemistry. The studies discussed in the different sections all showcase the myriads of contributions DL could make in different study domains, as well as where improvements are still desirable.

Deep learning models applied to molecular modelling

Deep learning and virtual screening

Introduction. Virtual screening comprises of the calculation of the interactions of a compound with a certain drug target, as to predict how favorable the binding of that compound would be to the target [11]. This method generally yields fast and accurate results for the in silico filtering of large compound libraries (e.g., ChEMBL, PubChem, and ZINC databases) during drug discovery, but as with everything, more optimization is definitely possible [12–17]. A possible approach to optimizing VS methods is by infusing its workflow with DL models.

To make the following discussion more comprehensible, it is interesting to divide the different VS methods into two subcategories: structure-based virtual screening (SBVS) and ligand-based virtual screening (LBVS). In general, the output of VS calculations is a scoring function, a measure of the probability of a ligand and its target to bind noncovalently [11]. SBVS predicts target-binding affinity based on the 3D structure of the compound and the drug target. This is mostly done through molecular docking simulations. For SBVS, one of the most widely used docking programs is AutoDock Vina. Other programs are SMINA, GNINA, QuickVina-W, and GLIDE, among others [18–22]. LBVS on the other hand uses as input the molecular and chemical properties of a compound. It bases its prediction of the binding affinity on the similarity between this compound and a known ligand of the drug target. LBVS is mostly done by similarity searches and requires the input molecules to be configured as molecular fingerprints. A molecular fingerprint of a compound is an abstraction that involves turning the molecular and chemical properties of a molecule into a sequence of bits, which can then be compared between molecules. There are many different software packages capable of performing similarity searches (*e.g.*, RDKit, Open Babel, OEChem KT) [23–25]. Each of them supports different fingerprints, although the most common fingerprints are supported by each software package. LBVS steps can also be carried out using pharmacophores, a technique dubbed pharmacophore searching. Other techniques to perform LBVS are also available (*e.g.*, Feature Trees, Topomers, Cresset's FieldScreen, and OpenEye's ROCS software) [26]. For an in-depth background on SBVS and LBVS, the reader is referred to the excellent papers of Vázques and Cleves & Jain, amongst many others [27–29].

Lay-out of "Deep learning and virtual screening" section. The "DL-based molecular fingerprint generation", "Drug-target interaction prediction with DEEPScreen" and "DeepScreening, a DL-based webserver for VS" sections focus on the use of DL for LBVS. The "DL-based molecular fingerprint generation" section starts off with discussing a state-of-the-art ML method, inspired by current Natural Language Processing techniques, capable of generating a new type of molecular fingerprint for virtual screening. The "Drug-target interaction prediction with DEEPScreen" section describes a simple but effective LBVS-derived DL application employing structural information rather than molecular fingerprint information, inspiring readers that DL models do not always have to be convoluted to be valid. The "DeepScreening, a DL-based webserver for VS" section goes further than this and highlights an accessible, user-friendly webserver for LBVS-derived DL model development.

The "Complementary LBVS-derived and SBVS-derived DL models" section touches on the integration of multiple DL models. In this section, an application is exemplified in which a LBVS-derived and a SBVS-derived are integrated, highlighting how multiple models can be chained together to form effective workflows. Next, further bridging the gap between LBVS-based and SBVS-based methods, the "DL as a compound generation tool for further VS steps" section reflects on generative DL models used as compound generation tools, of which the generated compound sets are then used for subsequent traditional VS steps.

Finally, the "DL-based replacements for docking methods: binding affinity predictors" and "DL-based replacements for docking methods: pose predictors" sections delve into SBVS-derived DL models, replacing docking methods as binding affinity or pose predictors. For the "DL-based replacements for docking methods: binding affinity predictors" section, the discussed methods (DeepBindRG and Pafnucy) lay the foundation for binding affinity predictors. By employing general input descriptors and relatively simple architectures, they highlight key drawbacks and limitations of this approach.

Many other architectures are built for these purposes, for which more references are provided. The section ends with an example of a recently published state-of-the-art model (AEV-PLIG). In the "DL-based replacements for docking methods: pose predictors" section, binding pose predictors are described. The selected methods build on top of each other, improving previous limitations to reach two current state-of-the-art models (DiffDock and AlphaFold 3) spearheading how DL can be applied in the field of VS. These SBVS-derived methods are followed by a brief discussion on generalizability and bias in the "DL and VS: is it worth the candle? A discussion on generalizability and bias" section. Lastly, the "DL and VS: or is it the third one who takes it? Generative DL models for the replacement of traditional VS methods" section returns to generative DL models, in order to discuss those capable of completely replacing traditional VS procedures. These architectures are generally quite complex, exceeding the scope of this review, but they are touched upon for completeness.

DL-based molecular fingerprint generation

Mol2vec. Molecular fingerprints are numeric or binary representations of compounds, and form a fundamental basis for many computational techniques, ranging from ML/DL models to similarity searching or clustering. A popular molecular fingerprint is called the Morgan fingerprint [30]. Whilst generating Morgan fingerprints, a Morgan algorithm defines compound substructures within a molecular structure. These compounds substructures are what form the basis for the unsupervised ML method Mol2vec [31]. Mol2vec is inspired by a technique used in Natural Language Processing called Word2vec, which is capable of transforming words into vectors, leading to high-dimensional embeddings of sentences, where vectors of similar words are near in vector space [32]. Mol2vec applies this same technique, where compound substructures are considered the "words" of a sentence, and the "sentence" being the molecule as a whole. The Mol2vec algorithm then calculates a high-dimensional embedding of a molecular structure. It does this by defining feature vectors for each molecular substructure and summing them up to one compound vector of the molecule, in which chemically related substructures are close in vector space. Before being able to apply the Mol2vec algorithm to numerous techniques (*e.g.*, ML methods such as a gradient boosting machine, or NNs), it first needs to be trained on unlabeled data to learn feature vectors of molecular substructures and to sum them up to compound vectors. Mol2vec is thus capable of providing a new representation for compounds, with this vector approach overcoming the drawbacks of many other feature representations (*e.g.*, sparseness and bit collisions) and yielding state-of-the-art performance when applied to different ML/DL tasks. New compounds could even be generated with Mol2vec, by summing the feature vectors of molecular substructures retrieved from a pretrained Mol2vec model.

ProtVec. ProtVec applies a similar strategy for the vectorization of proteins [33]. It generates feature vectors for all the "words" in a protein sequence, with the words being all definable three-amino-acid sequences (considering all possible frameshifts). This results in a protein vector which is bigger in size than those created by Mol2vec. The compound and protein vectors of Mol2vec and ProtVec can be combined, leading to an alignment-independent representations that can be easily applied to datasets of unrelated targets with low sequence similarities [31].

IVS2vec. One clear and powerful example of an implementation of Mol2vec in a novel screening technique, capable of outperforming classical VS methods, is IVS2vec [34]. Inverse Virtual Screening (IVS) is a method to identify protein targets for certain ligands, hence it being the inverse of VS, where ligands are identified for a given protein target. IVS2vec uses the compound vectors generated by Mol2vec as input for a dense fully-connected neural network (DFCNN) to create a DL prediction model. After supervised training, the model is capable of a binary classification into either a class of potential targets with a high possibility of binding with the query ligand, or into a class of targets with low binding possibility. The model is also capable of outputting a binding score, allowing for a more thorough analysis of the prediction results. The NN architecture of IVS2vec consists of a DFCNN using rectified linear unit (ReLU) activation functions, leading to one output prediction value through a sigmoid activation function. Training was done using the PDBbind database of 2017 [35]. Almost 15,000 relevant protein–ligand complexes were selected, after which the ligands were vectorized by Mol2vec as well as the residues making up the binding site pockets of the proteins. Both vectors were then merged into one representation of a protein–ligand complex. These vectors were used as the positive set for the supervised training, whilst a scramble between the compound and protein vectors was made to create a negative set containing nearly 20,000 false complexes. After training and validation, the IVS2vec model was further tested and evaluated on three independent datasets, indicating high performance. Drawbacks to applying an IVS technique such as IVS2vec are the limitations of only being able to use targets currently known and the computational limitations on the number of testable targets. Otherwise, IVS2vec is a strong aid to

drug repurposing or acquiring first indications of possible adverse drug reactions.

Drug-target interaction prediction with DEEPScreen

A disadvantage of using molecular fingerprints as descriptors for the training of DL models for LBVS steps is the fact that pieces of molecular structure information potentially get lost in the fingerprint generation process. Different types of fingerprints deem different structural features important for target binding, which get stored in feature vectors whilst information seen as unimportant isn't captured. This leads to a DL model being presented with only restricted molecular information for its classification or regression predictions. Therefore, it can be interesting to look at other ways to provide a DL architecture with structural input data of the entire molecule, as to allow the network itself to identify all the features relevant for target protein interactions.

DEEPScreen. An interesting example of a DL-based LBVS step employing alternative input data was developed by Rifaioglu et al. and was dubbed DEEPScreen [36]. When presented with drugs or drug candidate compounds, the goal of DEEPScreen is to predict novel interactions with drug targets. The application is a collection of individual convolutional neural networks (CNNs), each being an individual predictor for a target protein. Each predictive model was individually trained and optimized to predict the interactions of small molecule ligands with the target protein. Only ligand information was fed into each model, with loss optimization occurring through a label dictating to which target proteins this ligand is known to bind. The ChEMBL 23 dataset was used to set up manually curated training, validation, and test datasets for each predictive model. Care was taken to ensure each model was trained on balanced amounts of active (*i.e.*, interacting) and inactive (*i.e.*, non-interacting) datapoints for their target. A model takes the small molecule ligands in the form of SMILES representations as input and transforms the SMILES into 200-by-200 pixel 2D structural images. It then runs predictive CNN models on these 2D input images and generates a binary prediction for a ligand as being either active or inactive for the corresponding target protein (Fig. 1). The use of these 2D molecular images of compounds is assumed to lead the CNNs to inherently learn all the molecular structure features of the compounds in question and how they relate to target binding. According to the authors, this should result in higher accuracy of drug target interaction predictions over models using fingerprint featurization approaches. In total, models were developed for 704 target proteins.

After training all models, the performance of DEEPScreen was evaluated on multiple external benchmark datasets and compared to other state-of-the-art ligand-based prediction approaches employing fingerprints (*e.g.*, random forest classifiers, support vector machine classifiers, and logistic regression classifiers). In general, DEEPScreen was seen to outperform the other classifier approaches, reflecting the benefit of employing 2D molecular images as input. Some novel predictions of drug compounds binding to certain drug targets were selected for further validation through a literature-based validation, molecular docking analysis, and in vitro experiments. One case indicated the potential of DEEPScreen to predict novel inhibitors of the enzyme renin with potencies around the levels of investigational drug ligands, encouraging further investigation. In conclusion, the DEEPScreen project succeeded at generating highly optimized, high performance predictive CNN models for 704 different drug target proteins, and all of this has been published as an open access tool. The architecture allows for an independent model to be downloaded separately

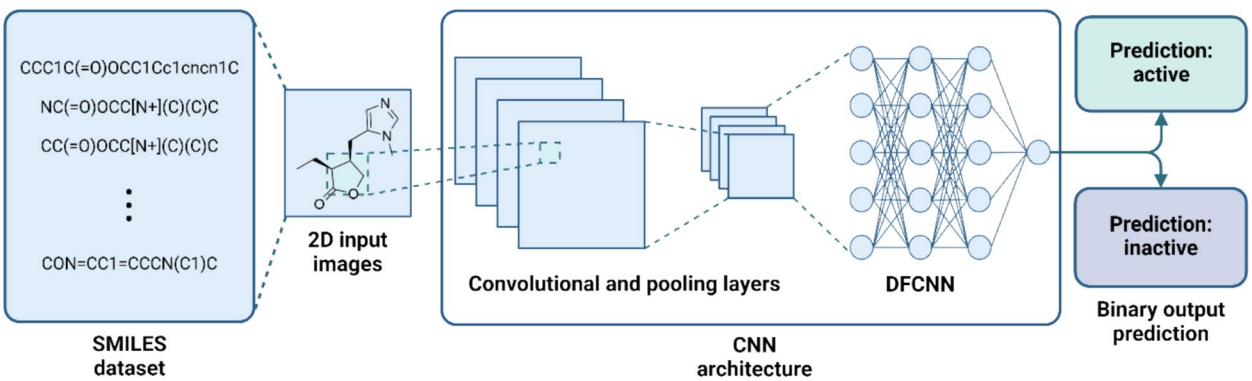

Fig. 1 Overview of the DEEPScreen architecture. Each prediction model included in DEEPScreen takes as input small molecule ligands in the form of SMILES representations, transforms them into 200-by-200 pixel 2D structural images, and then runs a predictive CNN model on them in order to predict whether these ligands are either active (i.e., interacting) or inactive (i.e., non-interacting) against a specific target protein [36]

for the testing of one specific target of interest. Additionally, they focused on creating a reliable open access dataset for ligand-based prediction approaches. Performance could be even more improved by transforming the input into 3D molecular representations (vide infra), although this would cause the models to become more computationally expensive, potentially limiting large scale use. Target proteins with only limited amounts of known ligand interactions, or none at all, were not able to be included in DEEPScreen, forming another limitation.

DeepScreening, a DL-based webserver for VS

DeepScreening. To apply DL models, molecular modelers require not only knowledge of DL and high-quality data, but also user-friendly tools and software. An example of an accessible web server for DL-based VS methods is called DeepScreening [37]. With the use of public or user-provided datasets, this application is capable of training DL models to perform classification or regression tasks. It could perform either VS on a provided chemical library or generate a de novo library of compounds for VS against a drug target. The user can specify a drug target of interest and provide a dataset for training if available, else the ChEMBL 24 database is used for extraction of the drug target and its ligands. The user then selects the features for determining molecular fingerprints of the compounds (for this, the PaDEL open-source software was built in) [38]. The parameters for the training of a classification or a regression model [respectively a DFCNN or a recurrent neural network (RNN)] also need to be specified. After that, a regression model can generate a de novo library of compounds to perform a VS on. Otherwise, the model performs a VS using a provided chemical library [37].

Application examples. Joshi et al. conducted two hybrid VS procedures to find compounds active against a SARS-CoV-2 target enzyme, the 3-chymotrypsin-like protease ($3CL^{pro}$) [39, 40]. In the first part of their research, they conducted a VS of natural compounds against $3CL^{pro}$. First, a predictive regression model was developed using DeepScreening, employing the ChEMBL3927 dataset for training. This first screening was a LBVS step, as the training data was converted into PubChem molecular fingerprint features using the PaDEL tool. The developed DL model consisted of an RNN architecture capturing the correlation between known IC_{50} values of the training compounds and their molecular fingerprints. After training, it was used for the VS step, employing the Selleck database containing 1,611 natural compounds, and leading to 500 selected hits with favorable molecular fingerprint features [39, 41]. These hit compounds were fed into AutoDock Vina for molecular docking simulations, a SBVS step that led to a selection of 39 compounds. The compounds were then subjected to additional screenings (e.g., predictions of their pharmacokinetics, drug-likeness and toxicity, among others), resulting in a selection of three suitable compounds. In the final step, these three compounds were subjected to MD simulations of 100 ns. After further analysis, two final compounds were selected to form stable complexes with $3CL^{pro}$. These could be further analyzed for therapeutic development against SARS-CoV-2 (Fig. 2) [39].

The same hybrid screening workflow was employed for the second part of their research, this time developing an

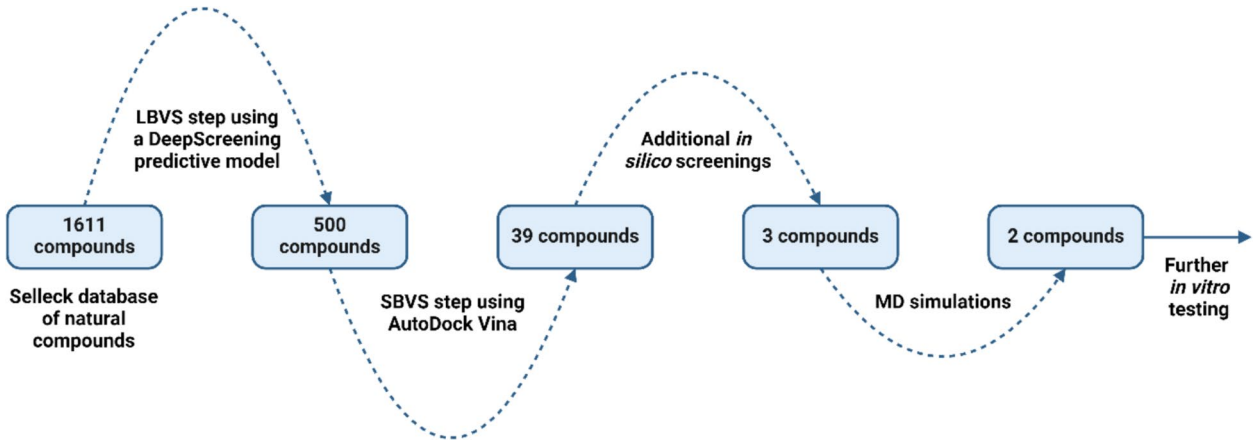

Fig. 2 Overview of the DeepScreening workflow employed by Joshi et al. for the screening of natural compounds against $3CL^{pro}$. Through a LBVS step employing a DL predictive model, a SBVS step employing a traditional molecular docking method, additional in silico screenings for characteristics such as pharmacokinetics and toxicity, and MD simulations, a database of 1,611 compounds was narrowed down to two specific hit compounds for further testing [39]

RNN for drug repurposing of drugs against $3CL^{pro}$. Starting with 9,101 drugs from the DrugBank database, this compound library was narrowed down with the RNN-LBVS step, an AutoDock Vina SBVS step, additional screenings, and MD simulations, to eventually identify two potential drugs to be further tested in vitro and in vivo against SARS-CoV-2 [40]. This research group also adapted the hybrid screening workflow for drug repurposing of FDA-approved drugs against *Candida albicans* dihydrofolate reductase, eventually identifying rifampin, lumacaftor, and paritaprevir as having great potential to inhibit the query enzyme, imploring further exploration of these drugs [42].

Complementary LBVS-derived and SBVS-derived DL models

Drug repurposing DFCNN. Zhang et al. performed a hybrid VS for drug repurposing targeting the viral RNA-dependent RNA polymerase (RdRp) of SARS-CoV-2 [43]. They proposed a hybrid workflow consisting of two consecutive complementary DL models, a classical VS step using AutoDock Vina, MD simulations of the binding pocket, and metadynamics simulations of the entire system. This allowed them to narrow down the 1,906 drugs present in the Approved Drug Library by TargetMol down to four market available drug candidates [44]. The first DL model employed in the study is a DFCNN that takes as input the concatenated compound and protein pocket vector of protein–ligand complexes, as generated by Mol2vec (see the "DL-based molecular fingerprint generation" section). From this, it can predict a protein-drug binding probability through a scoring function. The value of this model as the first step of the hybrid workflow is the fact that the model only considers molecular and chemical information of the compounds as packed in the compound vectors (similar to LBVS). By not taking the information provided by the complex conformations into account, faster prediction speeds could be reached.

DeepBindBC. The second DL model was dubbed DeepBindBC, developed by the same research group [43]. Similar to SBVS, this model also considers spatial information of the protein–ligand interfaces. It requires the same input as generated by AutoDock Vina about a protein–ligand complex structure. With the use of this spatial information, DeepBindBC is better capable of distinguishing non-binders than the DFCNN yet requires more computational time. At the end of the hybrid workflow, the four compounds were tested in vitro. Pralatrexate was the final compound identified as a potential new therapeutic agent against SARS-CoV-2 through the targeting of RdRp.

Zhang et al. also adapted a similar workflow for drug repurposing against tumor necrosis factor-α-induced protein 8-like 2, or TIPE2, impacting cancer and inflammatory diseases [45]. The biggest difference in this study compared to their previous research on RdRp is the scale of the VS steps, scaling up to over 8,000 starting data points. At the end of the thoroughly carried out workflow, four final candidates were selected for in vitro experimental validation. These four compounds, including a low-micromolar affinity binder, offer new information about inhibitors of TIPE2 and can help facilitate further drug development.

DL as a compound generation tool for further VS steps

Next to classification or regression tasks, DL models can also be powerful generative tools, an approach that can be very useful in drug discovery [46]. The next examples make use of that feature by employing generative NN models to investigate chemical space and generate small molecules or their descriptors. The works discussed in this section use generative models to create ligands for subsequent screening through more traditional VS steps (for an illustration of generative models to replace complete VS workflows, see the "DL and VS: or is it the third one who takes it? Generative DL models for the replacement of traditional VS methods" section).

GAN by Andrianov et al. Andrianov and coworkers employed a generative adversarial network (GAN) to identify new HIV-1 entry inhibitors, capable of blocking the CD4-binding site of the viral envelope protein gp120 [47]. The goal of the GAN was to generate new molecular fingerprints comparable to those in the training dataset. Based on these fingerprints, similar or identical compounds were identified from a big chemical library for further testing. The encoder part of the GAN is in and of itself an autoencoder (AE). Getting as input molecular fingerprints, the output of the encoder is a molecular fingerprint that approximating of the original input with probabilities connected to each bit. This output is used for loss optimization of the autoencoder, leading to a latent layer with a normal distribution. For further optimization of this latent layer and thus to correctly learn the features of molecules that bind to gp120, the latent layer is passed on to the discriminator of the GAN, itself a DFCNN. After initial training of the AE as a separate entity, the output of the AE was used to train the decoder to distinguish a randomly generated normal distribution from the encoded normal distribution on the latent layer (Fig. 3).

For this training, a dataset of over 120,000 compounds was generated with the AutoClickChem software package [47, 48]. Through molecular docking using the QuickVina 2 software package, the generated compounds with relatively high binding affinity to gp120 were selected for model training. MACCS fingerprints were calculated for each compound using RDKit. After

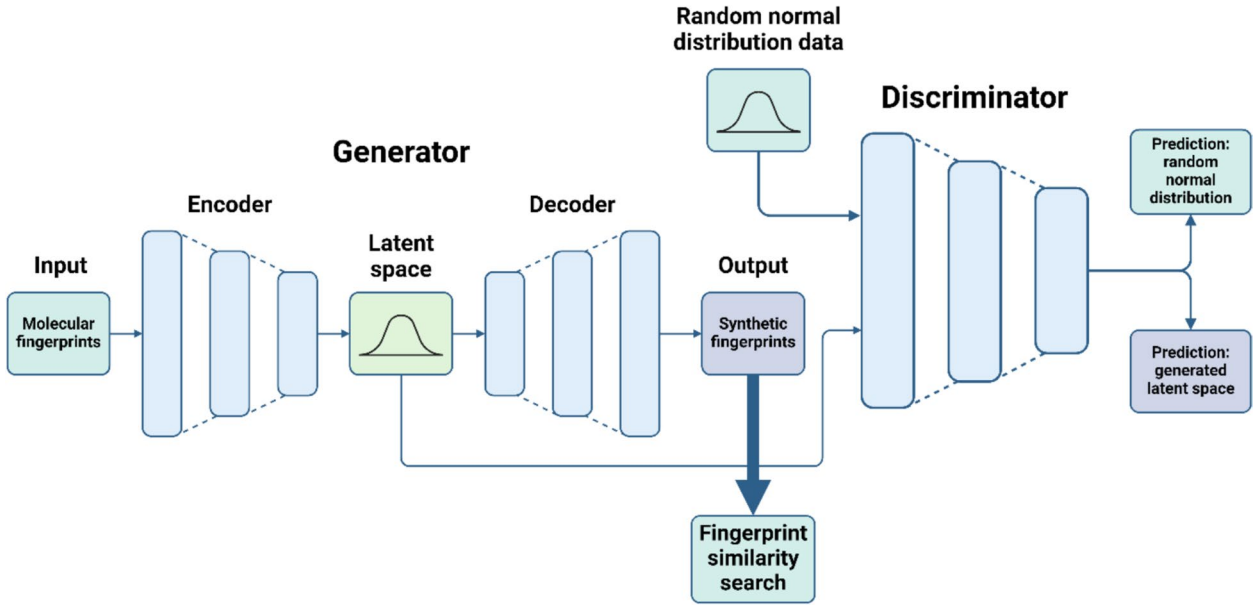

Fig. 3 Overview of the GAN architecture developed by Andrianov et al. for the in silico generation of HIV-1 entry inhibitors. The generator consists of an AE model capable of analyzing molecular fingerprint input and generating new synthetic fingerprints through sampling of its learned latent space. This learned latent space is further optimized through adversarial learning with the discriminator, which gets taught to distinguish the latent space from random normal distribution data. After training and optimization, the generator was used to obtain synthetic fingerprints of strong binders against the HIV-1 viral envelope protein gp120, which were sought out in a chemical library through a fingerprint similarity search for further testing [47]

multiple careful training steps, a trained GAN was produced, capable of generating MACCS fingerprints of molecules with relatively high binding affinity to gp120, as well as with drug-like qualities. The finished GAN model was then used to generate MACCS fingerprints of high binding affinity compounds. These MACCS keys were sought out via fingerprint similarity search in the Drug-Like dataset from the ZINC15 database. After QuickVina docking simulation tests, three compounds were selected for further evaluation of their binding affinities through semi-empirical quantum chemistry and molecular dynamics. When combining all acquired data, the three compounds exhibit high binding affinity to gp120 as well as drug-like physicochemical properties, which make them suitable for the development of novel HIV-1 inhibitors. The entire goal of applying the GAN for this research objective was to narrow down a big library of drug-like compounds, making this workflow less computationally expensive and time-consuming than considering all compounds of the library for molecular docking or MD simulations. A downside of the technique was the limited number of molecular fingerprints generated by the network, leading to possible under sampling of the relevant chemical landscape. Other examples of GANs developed for generating molecules have been described [49–52].

LSTM RNN. Going back to $3CL^{pro}$ in SARS-CoV-2, Arshia et al. conducted another study to identify new drug candidates capable of inhibiting this critical enzyme [53]. For this objective, a generative long short-term memory recurrent neural network (LSTM RNN) was developed through deep transfer learning (DTL). The original network, called LSTM_Chem, was trained purely on ChEMBL datasets and was designed to capture the features of SMILES molecular representations. It can use this information for the generation of new molecules, similar to the training data but with a high degree of structure variation. The model was adapted for the purpose of this research and retrained using SMILES of ChEMBL and ZINC databank compounds. Employing the retrained DL model, 10,000 first generation SMILES were generated. These compounds were tested in RDKit for validity, uniqueness, and originality. After further validation steps, the selected set was used for AutoDock Vina docking simulations with $3CL^{pro}$. The genetic docking algorithm selected 50 compounds from the dataset for further fine-tuning of the LSTM RNN, after which a second generation of 10,000 SMILES was generated. Such a workflow loop was adopted for 10 generations. From these generations, all compounds under a certain binding affinity cut-off score were selected. Hierarchical clustering on this dataset led to four clusters, with the compound with the highest

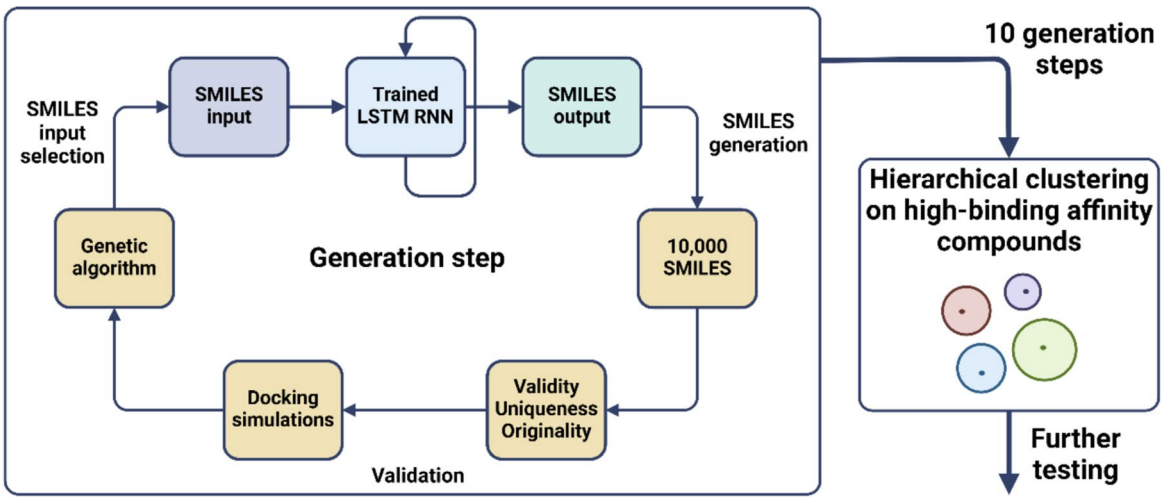

Fig. 4 Overview of the workflow employed by Arshia et al. for the in silico compound generation of $3CL^{pro}$ inhibitors. An LSTM RNN architecture was trained through DTL for the generation of $3CL^{pro}$ binding molecules. Each generation step, the generated molecules were further validated and tested using traditional molecular docking methods. A genetic algorithm then selected a limited number of compounds for further finetuning of the RNN model. After ten generation steps, all molecules with high binding affinity for $3CL^{pro}$ were clustered through a hierarchical clustering method, and the compounds with the highest binding affinity in each cluster were selected for further testing [53]

binding affinity in each cluster being selected for further analysis (Fig. 4). After carrying out extensive MD simulations with these molecules, it was concluded that all four compounds could be potential $3CL^{pro}$ inhibitors. However, these are mere in silico results without further in vitro or in vivo validation. Generation of even more generations of compounds could have led to more suitable compounds than those selected now. Other examples in literature of RNNs developed for generating molecules, which could subsequently be connected to VS workflows, have been described [52, 54–62]. A review discussing even more studies that all applied DL for VS and molecular docking against SARS-CoV-2 targets was written by Sun et al. [63].

WAE. Das et al. used DL generation tools to come up with solutions for another global threat on the rise, namely the growing antimicrobial drug resistance. They developed a variational autoencoder (VAE) for the de novo generation of antimicrobial peptides (AMPs) with desired properties [64]. A Wasserstein autoencoder (WAE) [65] was chosen as the VAE in question. The way in which a WAE is capable of generating a latent space causes the latent space to intrinsically capture the sequence relationship of the peptides, whereas the latent space of a normal VAE fails to do so. The WAE contained a bidirectional gated recurrent unit (GRU) encoder, whereas the decoder was a GRU. The WAE was trained using all known short peptide sequences (i.e., 25 amino acids at most) available on UniProt, represented as text strings. Using all available short peptide sequences for the training—instead of training only on known AMPs—leads to better exploration of plausible peptides beyond known antimicrobial templates. After training, the created latent space of the WAE was passed on to a binary classifier DL model, dubbed Conditional Latent (attribute) Space Sampling (CLaSS). Four bidirectional LSTM classifiers were developed and trained on over multiple thousand labelled peptides sequences to each capture and classify one specific property of a peptide sequence. The most important classifier was whether the sequence is an actual AMP and thus with antimicrobial function. This antimicrobial function classifier was used to skewer the sampling from the latent space of the trained WAE model. Making use of all four CLaSS classifiers, a rejection sampling scheme was created capable of generating molecules with desired attributes. This led to all generated peptide sequences being unique, diverse, optimized and valid AMPs with broad-spectrum potency and low toxicity. With this DL generation and classifier tool, 163 candidate AMPs were selected for further testing using coarse-grained MD simulations, as well as in vitro and in vivo testing. Two compounds were identified to demonstrate low toxicity in mice and high potency against multiple Gram + and Gram- pathogens, as well as to have a low disposition to induce drug resistance in *E. coli*. From this technique, it becomes clear that DL can accelerate the discovery of relevant antimicrobials. Other examples of VAEs developed for generating molecules, which could subsequently be connected to VS workflows, can be found in literature [66–72].

The discussed models throughout this section all generate molecules whose relevance and validity requires additional checks, such as through traditional docking methods. However, most recent efforts have been to develop models capable of generating valid molecules directly by fitting a certain binding pocket. This approach would allow for the complete replacement of traditional VS workflows. Other architectures than those discussed up until this point become relevant in this context, such as graph neural networks (GNNs) [73, 74]. These models will be touched upon in the "DL and VS: or is it the third one who takes it? Generative DL models for the replacement of traditional VS methods" section. Additional reviews discussing de novo drug design with DL architectures can be found [75, 76].

DL-based replacements for docking methods: binding affinity predictors

Current docking software for SBVS steps use scoring functions to estimate protein–ligand binding affinity. The values predicted from these software packages are based on expert knowledge, considering different interaction terms in different proportions. Often, multiple binding conformations as well as ligand and protein flexibility are also considered. Different approaches can lead to varying docking scores, which in turn lead to different conclusions. Multiple ML models have been developed for the scoring of docking results (*e.g.*, RF-score and NNscore), but these models still rely on feature engineering and expert knowledge [77–80]. DL techniques could entail a completely unique view for the prediction of protein–ligand binding affinity. A NN could be capable of learning the binding mode of a protein–ligand complex in an implicit manner, by learning protein–ligand interface contact information from training on a large protein–ligand dataset. Best case, this should allow for more flexibility in the learning of features of complexes, compared to human-based feature selection [81]. This could lead to models capable of predicting the binding affinity of protein–ligand complexes (vide infra) or even predicting the pose of a ligand within a protein ("DL-based replacements for docking methods: pose predictors" section). Knowledge of such approaches is currently still limited, and comes with important limitations, as will be discussed below.

DeepBindRG. Zhang et al. propose a deep neural network (DNN) called DeepBindRG, which is capable of predicting the binding affinity of protein–ligand complexes (Fig. 5) [81]. For this, a residual neural network (ResNet) architecture was built. As input, the research group used a 2D binding interface-related matrix, as to simplify the data to an image-format, whilst keeping as much interface information (atom types, atom pairs, and spatial information) as possible. For this, over 15,000 crystallized protein–ligand complexes were retrieved from the PDBbind database 2018. Several

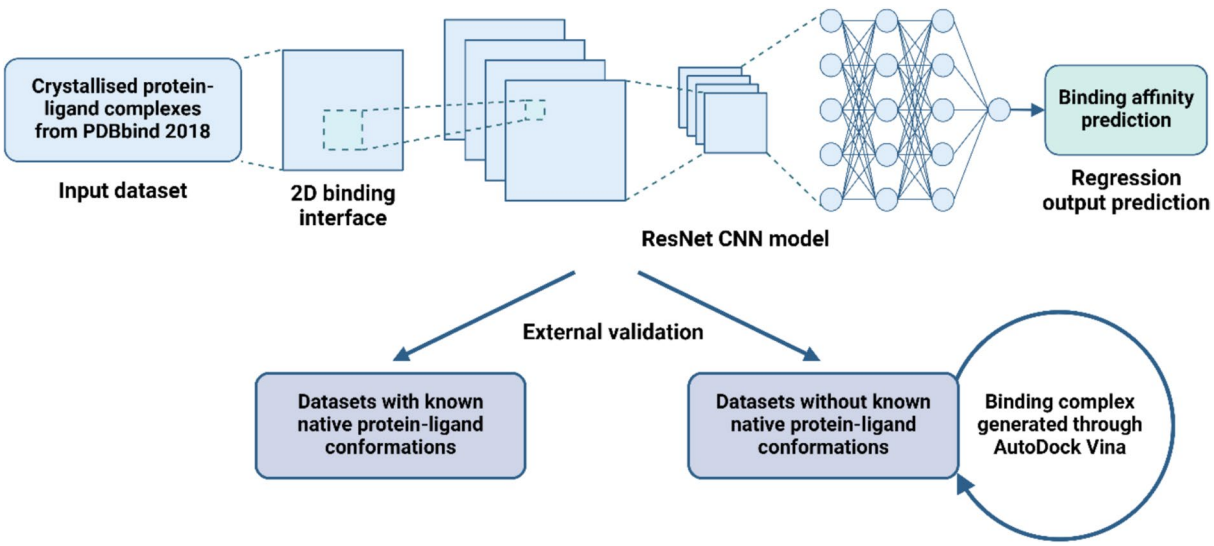

Fig. 5 Overview of the DeepBindRG architecture and the external validation carried out on this DL model by Zhang et al. [81] Crystallized protein–ligand complexes from the PDBbind 2018 database were used as training, validation, and internal test sets for DeepBindRG: a CNN model based on the ResNet architecture. Datapoints were fed to the network as 2D binding interface-related matrices and eventually led to an output prediction of the binding affinity of the ligand to the protein. After training and internal validation, DeepBindRG was further validated using external datasets with either known or unknown native protein–ligand conformations. When unknown, the traditional molecular docking method AutoDock Vina was used to generate the binding complex

extra independent testing sets were selected for further validation steps, containing either known or unknown native protein–ligand conformations. After training and initial validation using the internal validation set, the different external test sets were employed for further validation of the developed ResNet architecture. For the test sets containing complexes without known native conformations, AutoDock Vina was used to generate the protein–ligand binding complex. Different strategies were employed to evaluate the best way to choose which conformation generated by the docking software is near native and is best used for performing the final prediction. First, the prediction results from the test sets with known native conformations were compared to the performance of AutoDock Vina. After analysis, DeepBindRG was found to outperform the classical docking software. However, when analyzing the predictions made on the test sets without known native conformations, it became clear that there was still lots of room for improvement of the current DL method. Inconsistent prediction results indicate that there is an important difference between the positive results seen with predictions on known complex conformations and the results obtained in a setting that would more closely mimic real-world applications, namely when the native conformation isn't known.

The remaining challenges of this model are how to generate conformations as close as possible to the native conformation without resorting to AutoDock Vina, and how to identify and select a conformation that is native-like. Another difficulty is the discrepancy between training data and real application data. In the training set, strong binders are dominant. With real-world data however, non-binders would be dominant, and weak binders would be present in higher numbers than strong binders. The currently trained model could thus underperform due to this discrepancy. Still, the DeepBindRG model performs well in general for several independent datasets from different sources. This research helped in uncovering more generalized issues still present for deploying DL models for the prediction of protein–ligand complex binding affinities. This becomes even more apparent when comparing the results of DeepBindRG to Pafnucy (vide infra). The accuracy of both models turned out to be comparable and both face the same difficulties.

Pafnucy. Stepniewska-Dziubinska et al. developed a similar DL model, called Pafnucy, also attempting to predict the binding affinity of protein–ligand complexes [79]. For this, a model made up of a combination of convolutional and dense layers was built, using 4D information as input instead of 2D information. The 4D tensor consists firstly of three dimensions with points defined by Cartesian coordinates, in order to encode the positions of the heavy atoms of the system on a 3D grid. It also contains a fourth dimension consisting of a vector of 19 features per atom in the system (*e.g.*, atom type, hybridization, number of bonds with other heavy atoms and heteroatoms). For training, validation, and testing, almost 15,000 protein–ligand complexes were taken from the PDBbind database 2016. Additional test sets were taken from the Astex Diverse Set [82]. Open Babel was used for the generation of the atom features [24]. In order to allow for better generalization and to avoid sensitivity to the orientation of the protein–ligand complex, every complex was presented to the model in 24 unique orientations, leading to 24 training examples per complex. After building, training, and internal validation of the 3D CNN architecture, the external test sets were run through the model. The prediction results were compared to commonly used scoring functions (*e.g.*, X-Score, ChemScore) obtained through classical VS. Analysis showed that Pafnucy outperforms the classical scoring functions. In the previously discussed paper, Zhang et al. tested Pafnucy on the same four test sets without known native conformations as DeepBindRG and received similar middling results between the two models [81]. This shows that Pafnucy could suffer from the same pitfalls as DeepBindRG when applied to real-world research questions. Many other binding affinity prediction models have been developed throughout recent years, employing CNN or GNN architectures and a range of different input descriptors [83–103].

AEV-PLIG. Valsson et al. recently investigated strategies to enhance the applicability of these DL-based scoring function predictors [104]. They first developed the attention-based GNN model AEV-PLIG (atomic environment vector-protein ligand interaction graph), which is better capable of capturing the complex interplay of interactions determining binding affinity. They evaluated the performance of their model alongside Pafnucy and non-ML-based scoring functions on a variety of benchmarks, such as CASF-2016 (a widely used benchmark for scoring functions) [105]. This evaluation showed no better performance of these ML models compared to standard scoring functions. In order to enhance this performance, the researchers applied a data augmentation strategy on their training data, which used 3D protein–ligand structures modelled using template-based alignment or docking. Such an augmentation strategy significantly improved AEV-PLIG's prediction ability, but it also makes the limits of this current technique clear: these binding affinity predictors still require accurate protein–ligand information for their training. To combat this reliance on traditional molecular docking data, other types of DL models learn to predict the binding poses of protein–ligand complexes themselves.

DL-based replacements for docking methods: pose predictors

Traditional molecular docking methods are quite computationally expensive and still rather inaccurate, either due to limitations in the pose prediction steps or in the current scoring functions. With respect to the latter, DL models such as DeepBindRG, Pafnucy or AEV-PLIG succeed at outperforming current scoring functions through binding affinity predictions. However, they still rely on protein–ligand conformations generated previously, for example through traditional molecular docking methods. Recently, attempts to tackle these problems have proven successful through new DL architectures, attempting to not only predict protein–ligand binding affinities, but also predict the binding poses themselves. Given a protein structure and ligand pair, the first of these models attempted to predict the location of the ligand binding site in the protein, as well as the most optimal binding pose and ligand orientation, all in one shot.

EquiBind. Stärk et al. developed EquiBind, a geometric and graph DL model capable of such one-shot predictions whilst reaching significant speed-ups in computational time compared to traditional docking [106]. This setup succeeds at blind docking, i.e., correctly docking a ligand in a protein without prior knowledge of its binding site. To generate a suitable input for the model, the research group used a clustering technique via RDKit to create a molecular graph of both the ligand and the protein of a complex. This input is fed into an intricate DL architecture consisting of a combination of a graph matching network and a GNN. This type of model allows for the learning that happens in the model to be complacent with certain restrictions. Geometric constraints were added to prevent steric clashes in the direct-shot docking procedure, to allow only biologically plausible flexibility of the ligand within a rigid protein structure, and to ensure that the initial 3D conformations of both compounds don't influence the output predictions. For training and testing, protein–ligand structures from PDBbind were taken. QuickVina-W, GNINA, SMINA, and GLIDE were run on the same test set as a baseline for comparison. Two evaluation metrics were used for this analysis. First, the root-mean-square-deviation (RMSD) was used to show the difference in distance between the atoms at the predicted position of the ligand versus the actual docking pose, whilst the centroid distance acted as a measure of the ability of the model to find the correct binding pocket. When analyzing the obtained test results, it could be concluded that EquiBind is much faster than the traditional molecular docking baseline methods, whilst also generally delivering predictions that are less far off from the true conformer. That said, even with the geometric constraints, cases were still present where the right configuration of ligand atoms within the binding pocket was hard to find. It also lacks the capability of predicting a binding affinity value. The architecture does allow for extra finetuning steps, as to obtain better final predictions at a higher computational cost.

TANKBind. Taking a similar GNN approach, Lu et al. developed TANKBind [107]. This architecture builds in a new form of bias in its predictions through trigonometry constraints, succeeding better at preventing steric clashes and unrealistic conformation predictions. An additional module now also allows for binding affinity predictions. When taking a protein and ligand as input, the model segments the protein into functional blocks, in each of which it then analyses all possible binding sites with the given ligand and eventually outputs one final binding pose with a ligand binding affinity value. The model was trained and evaluated with the same data as EquiBind, and the same four traditional docking methods were used for comparison. This method proves to outperform EquiBind in identification of the binding region and docking pose, whilst reaching the same level of speedups. This, in combination with the state-of-the-art binding affinity prediction module, delivered promising results for DL-based molecular docking methods. Other protein–ligand binding pose prediction models employing GNN architectures can also be found in literature [108, 109].

DiffDock. Both EquiBind and TANKBind saw a significant increase in speed compared to traditional molecular docking methods, but only a relatively small increase in accuracy. Corso et al. approached this docking problem from a different angle when developing DiffDock (Fig. 6) [110]. They didn't want to predict the most fitting binding pose in one shot, rather they wanted to develop a DL architecture that could search the space of possible binding poses in an iterative process guided towards the most optimal one. Through a diffusion generative model, an intricate architecture involving convolutional layers [111], they are capable of sampling poses via a reverse diffusion process. Starting with random conformations of the ligand docked onto the protein, their model learns and transforms a noisy prior distribution into a learned distribution. Throughout this process, it samples possible realistic binding poses and step-by-step refines the system towards a most optimal final binding pose. This diffusion process happens over the degrees of freedom of the system that are relevant for the docking process: translation, rotation, and torsion angles of the ligand relative to the protein. A second module, a confidence model, was added to provide confidence predictions of each sampled binding pose and provide the top ranked pose. The method was built, trained, and tested using PDBbind complexes as data, similar to EquiBind and TANKBind. Its results were once more compared to

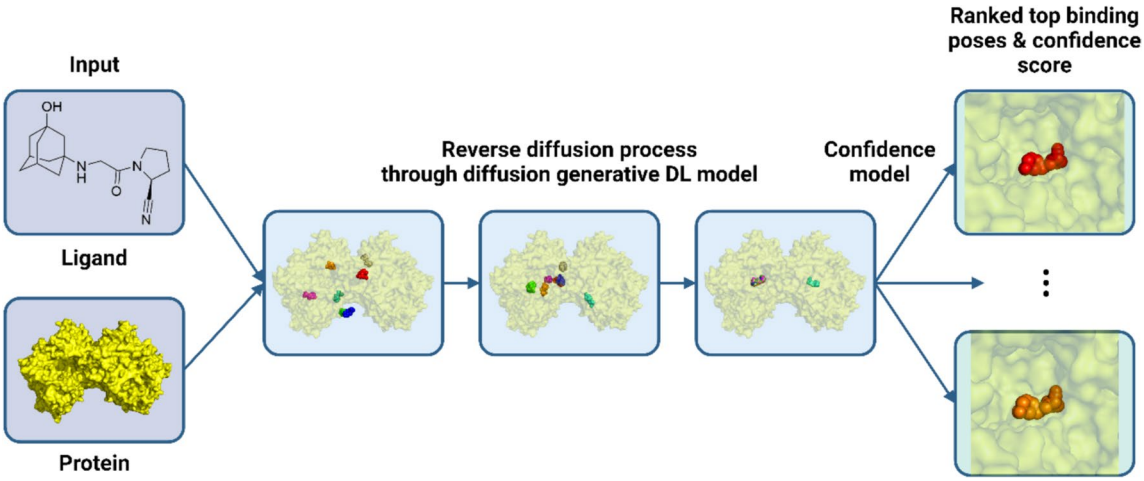

Fig. 6 Overview of the DiffDock architecture by Corso et al. [110] When given separate ligand and protein structures as input, the DL model employs a reverse diffusion process to step-by-step sample more realistic binding poses and refine the system towards binding poses as optimal as possible. A confidence model is then employed on each final binding pose to predict their confidence and provide the top ranked poses

both of these DL docking methods, as well as traditional methods QuickVina-W, GNINA, SMINA, and GLIDE. DiffDock outperforms all the mentioned methods, nearly succeeding at doubling the success rate of a prediction in finding the most optimal binding pose. It reaches high speedups compared to traditional models, albeit slower than one-shot prediction models. Its confidence scores are an accurate indicator of the top sampled pose throughout the diffusion model, providing valuable information for downstream steps in a drug discovery workflow. Where the other DL methods lost their accuracy when employing them on protein structures folded through computational methods instead of the apo-structure determined through crystallization, DiffDock retains much of its accuracy. This information makes it clear that DiffDock is a very interesting application that could be applied to real-world research questions and could provide valuable, accurate docking results for further drug design applications.

AlphaFold 3. This currently most recent version of AlphaFold has managed to progress the field of docking predictions even further [112]. AlphaFold 2 is a computational method, based on DNNs, that is capable of predicting the 3D structure of proteins with unknown tertiary and quaternary structures, purely based on their amino acid sequence and knowledge gained from all proteins with known 3D structures [113]. A detailed description of the architecture developed by Google DeepMind exceeds the scope of this review paper, but depends on attention neural network algorithms. These are seen in transformer networks, namely capable of the storing of information learned at certain points in the network, impacting the weights given to other features; a process mimicking cognitive attention. The current AlphaFold 3 version expands upon the AlphaFold 2 architecture and is capable of handling arbitrary interactions of proteins with other proteins, small molecule ligands, nucleic acids, and modified or non-canonical residues [112]. High performance has already been observed throughout several different tasks. In ligand docking, AutoDock Vina is outperformed on a specific benchmark complex set called PoseBusters (428 liganded protein structures from the PDB), using only protein sequences and ligand identities as inputs, whilst the classical VS software uses bound protein structures as input [114]. The work demonstrates that the highest quality bound structure predictions are made when both the protein and ligand positions are predicted in a joint fashion. Thus, the latest AlphaFold model shows that computational predictions through ML/DL models can outperform traditional docking strategies and could be applied soon as state-of-the-art applications for VS during drug design.

DL and VS: is it worth the candle? A discussion on generalizability and bias

An interesting discussion is led by Volkov et al. which debates what these binding affinity prediction models actually learn [115]. They argue that the drug discovery industry hasn't benefitted yet from these DL models because of poor generalization, certainly towards larger compound libraries. To formulate a hypothesis as to why proper generalization is so difficult to achieve, they performed a comparison between a bunch of DL models recently developed for binding affinity predictions, and the data with which they were trained. Counterintuitively, they concluded that the accuracy of these models

appears independent of training set size, and moreover, that higher complexity of the descriptors of proteins, ligands, and their interactions doesn't translate to higher accuracy of the DL models. With these observations, they hypothesize that DL models simply memorize hidden patterns in the data fed to the model during training, without actually learning underlying biophysical principles of noncovalent interactions. This learning was also observed by Yang et al. as they showed multiple CNN-based binding affinity predictors showing equal performance when trained only on ligand descriptors or protein descriptors compared to protein–ligand interaction descriptors [100]. They also identified the different types of biases that tend to arise in often-employed datasets, artificially enhancing DL model performances. These bias types are further discussed by Chen et al., showing bias can also be induced by analogues and decoys present in datasets [116].

The discussion then becomes whether what current models are learning is something to be regarded as a disadvantage and problem to tackle, or rather a strength of the technique. Regardless of the answer to this question, the way forward to improving binding affinity prediction models should be through increasing the generalization capabilities of a model whilst preserving accuracy. The study by Volkov et al. attempted to remove hidden biases in their training data to improve the generalization capabilities of developed models but were unable to succeed in that respect [115]. They do suggest to only train DL models on protein–ligand interaction descriptors, omitting ligand and protein descriptors, to reduce the risk of overfitting. Sieg et al. propose the need for specific datasets for binding affinity predictor models and discuss guidelines to avoid forms of bias in the training data and validate the model after training [117]. As an example, bias scoring functions could be developed for the selection of datapoints during dataset development. In general, it becomes clear that there are still challenges ahead to better understand and improve upon what binding affinity prediction models learn from, to develop models ready for general applications.

DL and VS: or is it the third one who takes it? Generative DL models for the replacement of traditional VS methods

A shift in interest in recent years can be observed in the resurgence of generative molecular design: a technique that attempts to go beyond virtual screening or docking scores, by using DL models to generate or optimize molecules to fit within a binding pocket of interest [118]. Multiple approaches are being explored for this goal. There are algorithms that build a ligand in 3D utilizing a representation of the target protein structure, e.g., TargetDiff or PILOT (diffusion models), Pocket2Mol or FRAME (GNNs), TacoGFN, and others [118–124]. As another example, there are models that generate/optimize 2D molecule structures through attempting to optimize a structure-explicit scoring function, e.g., AHC (a GRU-based model) or AutoGrow 4 [118, 125–127]. Due to the large variety in possible approaches and a lack of standardized evaluation methods, it is quite difficult to evaluate and compare the models. That said, there are a slew of papers trying to benchmark the 3D methods. They show the models still have a long way to go to becoming truly satisfactory, both in terms of generating valid geometries as well as qualitative molecules for further drug design [128–133]. However, given that this is currently still quite a data-limited field, enormous progress is being made for the models aiding the VS workflow, as well as those trying to overtake the VS-step through generative molecular design. If these methods are implemented in a carefully thought-out manner, they will allow for relevant time- and resource-saving in a drug development workflow [118].

Deep learning and molecular dynamics simulations

Introduction. MD allows for gaining relevant dynamical and kinetic information of protein–ligand systems, whilst being a relatively inexpensive method compared to in vitro counterparts. Next to aiding in drug discovery, MD simulations can be applied for solving biophysical problems, such as protein folding. Modelling these complex processes requires simulations to access biologically relevant timescales. This remains a huge challenge, as the more complex a system to be investigated is, the more computational time calculating one timestep will cost. Ever-improving hardware (current GPU-heavy supercomputers) and the continuous development of enhanced sampling MD approaches has allowed general systems to reach simulation times in the order of milliseconds [134–136]. Systems of around 100,000 atoms can nowadays be simulated on the GPU-partitions of a supercomputer for 100–1,000 ns per day. Whilst interesting processes can be observed within such timescales, relevant sampling of binding/unbinding processes or the conformational space of complex systems (e.g., intrinsically disordered proteins) requires timescales yet unreachable. Applying ML/DL to MD simulations forms a new approach to improving the current state-of-the-art regarding MD and has been successfully applied with promising results.

Lay-out of "Deep learning and molecular dynamics simulations" section. The next sections will dive into different areas in which DL can be of aid to MD, either by enhancing the conformational sampling ("DL-guided enhanced conformational sampling of protein structures" section), through the design of alternative force field potentials ("Neural network potentials" section), or

by boosting the analysis of MD trajectories ("DL-guided analysis of MD trajectories" section). The models discussed in the "DL-guided enhanced conformational sampling of protein structures" section were selected to build on top of each other, from a relatively simple architecture helping in selecting conformations for additional simulations, to more complex workflows that make the architectures more transparent and transferable. The "Neural network potentials" section delves into these alternative force field potentials called neural network potentials (NNPs), showcasing the different generations of NNP architectures and ending with the state-of-the-art models and workflows for the simulation of organic molecules. For the "DL-guided analysis of MD trajectories" section, an often-employed technique for DL-aided MD analysis is showcased through two research examples that use different input descriptors and explanation methods for the DL model. A third research example from the field of genetics demonstrates the broad applicability of such DL-aided MD analysis methods throughout all computational chemistry related fields. This section concludes with a discussion on the need for appropriate input descriptors and validation of analysis results. All-in-all, the "Deep learning and molecular dynamics simulations" section offers a non-exhaustive but inspirational look into how DL models can improve current MD-based workflows, while also touching upon its current limitations.

DL-guided enhanced conformational sampling of protein structures

Proteins can be defined as flexible molecules, with their dynamics being intimately connected to their function [137]. MD simulations can be leveraged to characterize the conformational space of proteins, gaining access to information that would otherwise be difficult to derive with in vitro experiments (e.g., X-ray crystallography, nuclear magnetic resonance). With the current enhanced sampling MD approaches (e.g., metadynamics, replica exchange MD, Gaussian accelerated MD), more of the conformational landscape can be investigated, mitigating the risk of under sampling. Another way in which enhanced sampling can be achieved is by interlacing the MD simulations with active ML components [138] or predictions made by a DL model trained on simulation data. Training a generative network to predict new conformations within the conformational space of a protein could form an interesting technique to obtain starting conformations for additional MD simulations [137]. Guiding such models to form conformations with certain restrictions could guide the simulations towards a desired goal within conformational space. For example, designing a generative model to identify intermediate states in protein folding pathways could help in obtaining a folded protein conformation starting from an unfolded state [139, 140]. It could also be possible to train models to predict the effects of perturbations to MD simulations, such as mutations in a protein sequence, changes in the ionic concentration or solvent type of a system, changes in FF, etc. The impact of such perturbations on the protein dynamics could then be characterized without requiring additional simulations [141].

The computational motifs that are presented here exceed the scope of protein folding and can be extrapolated to be applied to other biophysical problems or drug discovery processes. For example, it could be interesting to use MD simulations to study the movement of a ligand throughout protein channels to and from binding sites. Without enhanced sampling approaches, this migration will very probably not be seen within reasonable timescales. With enhanced sampling, the movement of the ligand can be guided along to and from such protein channels. This could be done with a hybrid MD/DL workflow, guiding the MD simulations along by selecting interesting starting conformations in the protein channel for additional simulations.

Generative AE. Degiacomi describes the development of a generative AE that is trained on MD simulation data and can generate new possible conformations for a protein (Fig. 7) [137]. To create training and test sets, MD simulations were run for a protein of interest, after which frames throughout the whole run were extracted as different conformations for the datasets. This input was fed as flattened Cartesian coordinates into an AE architecture, with the input layer being N-dimensional (N being the degrees of freedom of the system of interest). The AE model leads to an encoding that is a low-dimensional representation of the conformational space of the protein of interest. The decoder is then capable of taking this latent vector and expanding it once more to create an output as close as possible to the initial input structures. Within the conformational space of the latent vector, it is also possible to select any coordinate and generate a new protein conformation from it. This requires the atomic arrangements at those places to be actual plausible molecular structures. Extensive tests were run to test the validity of generated conformations. In the end, it could be concluded that valid conformations were indeed generated, stretching or compression of atoms barely being observed in any structures. The possibility of the model to interpolate structure states in between the input states fed to the model was clearly showcased, indicating that new states cannot be extrapolated from the input data. It was also shown to use the model to predict the structure of a protein undergoing substantial motion when bound to other molecules. The model was seen to perform better and produce a wider range of structures when trained on a flexible protein. Generated

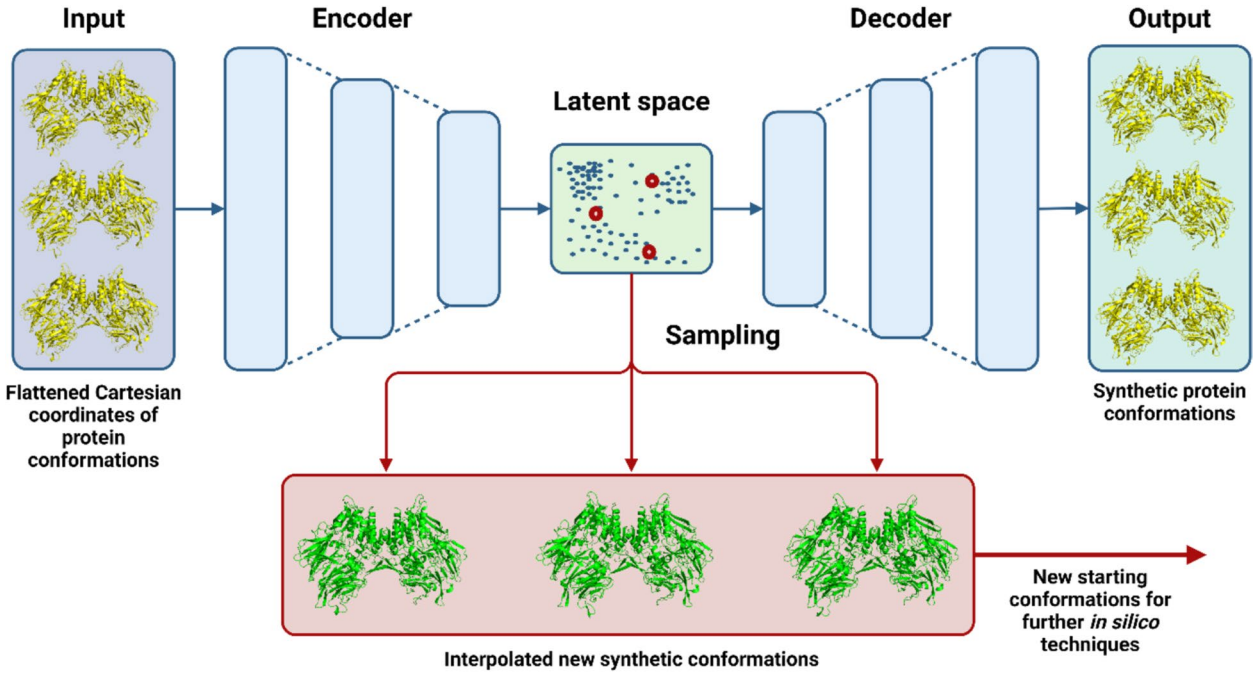

Fig. 7 Overview of the generative AE architecture developed by Degiacomi [137]. Of a protein of interest, the flattened Cartesian coordinates of a dataset of conformations are fed to the AE model as input. After learning, the latent space is a low-dimensional representation of the conformational space of the protein. Through interpolation, it now becomes possible to generate new protein conformations, which can be used as starting conformations for other in silico techniques, such as molecular docking or MD simulations

protein conformations can be coupled to a docking screen, to determine conformations that are closer to a bound state than the starting conformations. These states can be used as starting conformations of new MD simulations, now capable of better sampling the bound states. Thus, the most suitable generated complexes could be refined in a second step using more complex and expensive computational techniques. The current model needs to be developed separately for any protein of interest. That said, it proves it should be possible to, given a large enough dataset, develop a general NN that could be trained quickly through DTL for the solving of a specific conformational sampling problem.

DeepDriveMD. Ma et al. developed a functional example of an MD/DL iterative workflow for protein folding problems (Fig. 8) [139]. A convolutional variational autoencoder (CVAE) was developed to process MD simulation data of a protein of interest, and cluster these conformations in the latent space into regions with certain biophysically relevant features. This then enables the identification of relevant protein conformations as starting coordinates for new MD simulations [142]. The workflow guides MD simulations towards reaching a certain end goal in smaller timeframes and in a relatively short amount of workflow iterations. A CVAE model was developed for two protein test cases, in which the input simulation data was fed into the network as flattened Cartesian coordinates in a contact map representation. The goal for both test cases was to start off with a completely unfolded amino acid sequence and guide the simulations to sample the folded states. The latent spaces in the encodings of the trained CVAE models contain specific latent features that could be used to select conformations with specific characteristics. These latent features are emerging properties of the clustering in the latent space, they are not fed into the network as part of training data. The research group used the RMSD of a conformation compared to the native state of the protein as latent feature for their selection of specific conformations. In cases where the native state isn't known, the RMSD compared to the starting conformation still provides interesting information about which conformations are more folded than the original state. These latent features form the trick to propagating MD simulations towards the folded state of a protein: if an ensemble of MD simulations is carried out at the same time (allowing for parallelization) and their data is fed into the CVAE, it can analyze which MD runs sample novel parts of conformational space and which don't. In the test cases, this means selecting the MD runs that sample conformations with RMSDs closer to the native state.

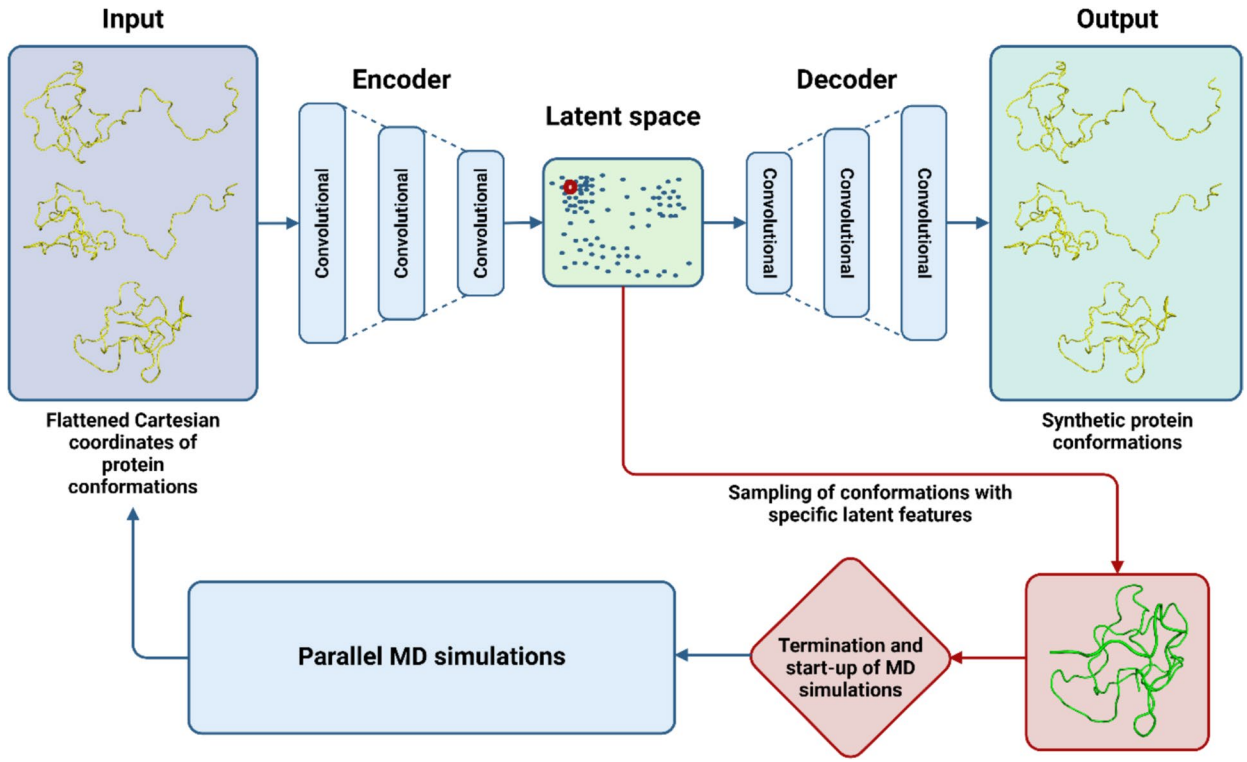

Fig. 8 Overview of the MD/DL iterative workflow developed by Ma et al. for protein folding problems, employing a CVAE architecture [139]. MD simulations are run in parallel for a protein of interest. The conformations generated throughout these simulations are fed to the CVAE as flattened Cartesian coordinates. After learning, the latent space is a low-dimensional representation of the conformational space of the protein, with regions defined by specific latent features/characteristics. This can be used to sample conformations with certain latent features (e.g., the RMSD of a conformation compared to the folded native state/unfolded starting state). Based on these samples, specific simulation runs can be terminated, and new runs can be started from the sampled conformations, in order to speed up the protein folding process

With this model, it becomes possible to create a workflow in which certain simulations that don't deliver new information get cut off, whilst new simulations are booted up that can discover novel parts of the conformational landscape. Thus, a subset of the conformations generated by original MD runs are selected to start new MD simulations. At the same time, other runs get cut off in an iterative manner. This process takes place until the test case proteins are folded, which means reaching a certain RMSD cut-off value close to the native state. One of the two test cases succeeded at sampling the native state of the protein, whilst throughout the other test case partially folded states were extensively sampled. It is hypothesized that through extension of these simulations, complete folding of the protein would also be sampled. These test cases are strong arguments for the use of intermediate data analysis with DL to drive subsequent computations.

The concurrent, coupled MD runs and DL applications sprout forth the requirement of well-thought-out workload and performance balancing. Employing the CVAE in guiding the MD simulations along was proven a worthy undertaking, as the time to train the model took about a nanosecond worth of MD calculations. The workflow's overall performance thus increases by employing the CVAE rather than simply extending the simulation timeframes. Still, the added complexity of adaptive workflows poses significant challenges to not waste time on workload balancing. What it means to have good or bad performance with DL-driven MD simulation workflows was expanded by the research group [140]. In this paper, the integrated approach is also generalized in DeepDriveMD, a framework for carrying out DL-driven MD simulations using self-chosen DL models and MD simulation techniques.

VDE workflow. In the work by Sultan et al., a DL model is trained to perform enhanced sampling together with MD simulations, focusing their efforts on making the developed model transferable to other related systems [141]. According to the authors, more traditional AE networks suffer from low explainability and unclear transferability. Also, due to the input data containing no information about dynamics, these networks could artificially form barriers between states that are actually

kinetically similar. To address these bottlenecks, the research group developed an interesting workflow. First, they designed a new variant of an AE, dubbed a variational dynamics encoder (VDE) [143]. In the VDE loss function's goal of reproducing time-lagged dynamics, it is capable of naturally mapping states kinetically similar to each other in such a way that no artificial barrier is created between them. This model was coupled to a technique used in explainable artificial intelligence (XAI) called saliency mapping, to aid in interpreting the features that contribute to the observed predictions. The magnitude of the derivative of the network's output with respect to the input features can help in understanding what features are important in the decoder's reconstruction from the latent embedding. This VDE architecture was successfully applied in a protein folding test case, where the learned latent coordinate was used as a collective variable (CV) for well-tempered metadynamics simulations. All major conformational states of the protein were clearly distinguishable from each other. This means that the latent variable learned a highly nonlinear transformation capable of separating the most important forms of movement of the molecule. When employing this variable as a CV, it is possible to guide MD simulations to sample the most important dynamical behavior of a protein.

Such a model turns out not easily be scalable to larger systems, as the amount of input features for the VDE would greatly increase, leading to more input nodes and hidden layers, upping calculation time. Therefore, a pre-processing step was added as dimensionality reduction. The dimensionality reduction was achieved through time-structure based independent component analysis (tICA) [144]. The data now selected to be passed on to the VDE and then to MD simulations still allows for maximal exploration of conformational space. It allows for smaller VDE architectures and adds another layer of explainability, since it is possible to analyze what features the tICA modes represent, and thus what the network is accelerating. This leads to the following workflow: dimensionality reduction through tICA, training of a VDE, and lastly using the latent coordinate of the VDE as CV for enhanced sampling MD simulations. According to the research group, this workflow allows for transferability of the learned latent space to closely related proteins. The latent coordinate learned through the training of the VDE is likely to be conserved between highly similar proteins (e.g., mutants) or between similar albeit slightly differing conditions. As a proof of concept, such transferability was tested through the application of this workflow to capture the effects of a mutation on the conformational landscape of a protein domain. One VDE model was trained on data from the wild type domain, after tICA pre-processing. This was used as CV for well-tempered metadynamics simulations on both the wild type and the mutated domain. For both protein domains, walkers were seen to sample the correct folded states, as well as a frequently observed misfolded state.

In short, this work delivers a workflow that is flexible, scalable and transferable across related protein mutants and related simulation conditions. However, it is not easy to determine when transfer of the learned latent coordinate will fail to be predictive and not allow for efficient sampling of a related system. This should be determined case by case, and arbitrarily transferring networks is not something advised. Other generative NNs that have been designed to guide folding sampling towards lesser explored regions of conformational space of proteins and obtain accurate free energy landscapes can be found in literature [145–151].

Neural network potentials

In the previous section, all the described workflows utilized DL models layered on top of MD simulations to guide them towards a desired objective. A completely different approach is the integration of DL models into the MD simulations itself, aiding in speeding up the necessary calculations. The biggest drawback of conventional MD is the approximate method that needs to be used when calculating the potential energies of a system's atoms. As to allow for maximal realism of a simulation, classical MD FFs are designed to calculate the atomistic potentials as close to real-world values as possible, whilst at the same time trying to keep calculation times feasible. This requires the FFs to simplify the descriptions of the interatomic interactions, by which accuracy is lost. Classical FFs are generally only reliable for the sampling of near equilibrium states and cannot be used for the investigation of chemical reactions or transition states. Their accuracy can also vary wildly between systems, leading to the development of many different FFs optimized for different system types. Finding new ways of expediting these calculations without losing any more accuracy proves quite a challenge. In this respect, research is going on to investigate whether ML methods could be used as an alternative approach in the form of ML potentials. ML methods form an interesting alternative to classical FFs in the calculations of atomistic potentials. They offer molecular energy predictions at quantum mechanical (QM)-level accuracy at speeds faster than classical calculations or the relatively fast electronic structure methods such as density functional theory (DFT). At the same time, they allow for transferability. They also don't require knowledge of the functional form of a system, allowing all types of atomic interactions to be described without bias and at a similar

level of accuracy. When these atomistic potentials are predicted using NNs, they are defined as neural network potentials or NNPs, which by now form an integral tool for MD simulations themselves [152, 153].

An extensive review of the current state-of-the-art regarding NNPs was composed by Behler in 2021, and the classification scheme developed in that review will be used as the basis for a short discussion regarding NNPs in the following paragraphs [153]. First, it is important to define a ML potential, of which an NNP is a subgenre. A ML potential can be defined as an analytic expression of the potential energy surface (PES), providing the potential energy and its analytic derivatives as a function of the atomic positions using a ML algorithm. It is constructed using a consistent set of reference electronic structure data. It doesn't contain any assumptions about the functional form of the system, apart from the approximations implicitly included in the chosen reference electronic structure method [153]. An NNP is a ML potential in which the chosen ML algorithm is a DNN. The first NNPs were developed as far back as over 25 years ago, but these were only applicable to low-dimensional systems, meaning that these systems were only allowed to contain a small number of atoms. In the following decades, this field boomed, and a way was discovered to apply NNPs to high-dimensional systems, containing tens of thousands of atoms. Currently, a classification can be made, dividing the different NNPs developed over the years into four generations. First-generation NNPs are those that are functional for low-dimensional systems, with calculations depending only on a few degrees of freedom. Second- to fourth-generation NNPs are all high-dimensional NNPs (HDNNPs), with the third-generation HDNNPs building on top of second-generation HDNNPs by including long-range interactions in the potential energy calculations, and fourth-generation HDNNPs building on top of that by also describing long-range charge transfers. The differences between the four generations will be discussed in more detail below.

First-generation NNPs. These ML potentials are based on a DFCNN, a simple, feed-forward NN that is learned to describe the global potential energy of a system. In its simplest form, a DFCNN takes as input the Cartesian coordinates of a system. Throughout training of the weights of the network, which is done by comparing its predictions to calculations made by a reference electronic structure method such as DFT, it learns to predict the potential energy of the entire system purely based on the current positions of all the atoms in the system (Fig. 9). This is advantageous in the sense that no prior knowledge is needed about the underlying physical principles leading to such energies, such that no bias is included in the predictions. It leads to a very simple functional form, of which derivatives can be calculated that are needed for the calculation of atomistic forces. This all leads to a network that allows for accurate energy predictions and force calculations many orders of

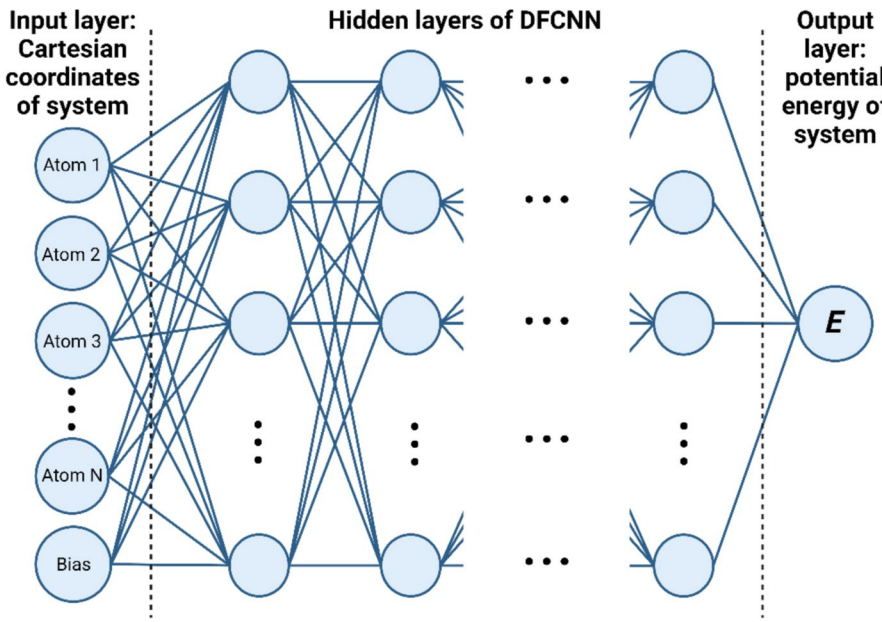

Fig. 9 Overview of 1st generation NNP architectures, which consist of DFCNNs taking as input the Cartesian coordinate vector of the N atoms of a system of interest and output a potential energy prediction for that system [153]

magnitude faster than the classical electronic structure method used for building the reference dataset.

Nevertheless, this first generation came with important drawbacks. The larger the system in question, the bigger the input coordinate vector, which leads to a size increase of the entire network and its hidden layers. Thus, the dimensionality of the network easily increases to sizes at which the calculations become computationally unfeasible. Accurate sampling of high-dimensional spaces remains unattainable with a simple DFCNN. On top of that, in this design each atom's coordinates relate to one neuron in the input layer. As the dimensionality of a DFCNN needs to remain fixed, there is a restriction that only systems with the same dimensionality as those in the training process can be simulated. Cartesian coordinates are also difficult input parameters as they are rotation- and translation-depending: if the system gets rotated or moved around, these coordinate numbers will change, meaning the input of the NN will change, as well as its output. As a sidenote, it is important to address that certain activation functions containing a discontinuity in their derivate, like a ReLU activation function at its origin, cannot be used for the representation of a continuous function such as potential energy [153].

Descriptors of atomic environments. Developing a single NN for the prediction of global potential energies is only reasonable for low-dimensional conformational spaces. To overcome such limitations, a couple of design elements of an NNP had to be revised. Instead of using the Cartesian coordinates of each atom of a system, a new descriptor was developed as input for NNs that describes the atomic interactions of an atom with its surrounding atoms within a certain cut-off radius. This describes the "short-range" energy of that atom, which can be regarded as the full potential energy of the atom under the assumption that most of its interactions take place in a localized chemical environment. For most systems, using a cut-off radius between 6 and 10 Å, errors in potential energy in the order of around 0.1 kcal/mol are observed. Such a cut-off radius is large enough to describe both covalent and close-contact non-covalent interactions of each atom. As an alternative, atom-dependent cut-off radii could be defined [154].

The structural information within such an atomic environment can be taken and converted into a suitable input for NNPs. This led to the development of atomic environment descriptors: different ways in which the features of an environment are described and compiled, as to discriminate different atomic configurations. Important to consider is that input descriptors for high-dimensional systems need to fulfil three requirements: translational, rotational, and permutational invariance. In ML techniques, this is not self-evident, given that such algorithms simply perform calculations on certain input numbers, and so if these numbers change, the output changes also. Many such descriptors of atomic environments conforming to the three invariances have been developed, but the first descriptor that was developed for the construction of HDNNPs is still a widely used descriptor type for NNPs. These are the atom-centered symmetry functions (ACSFs) [154, 155].

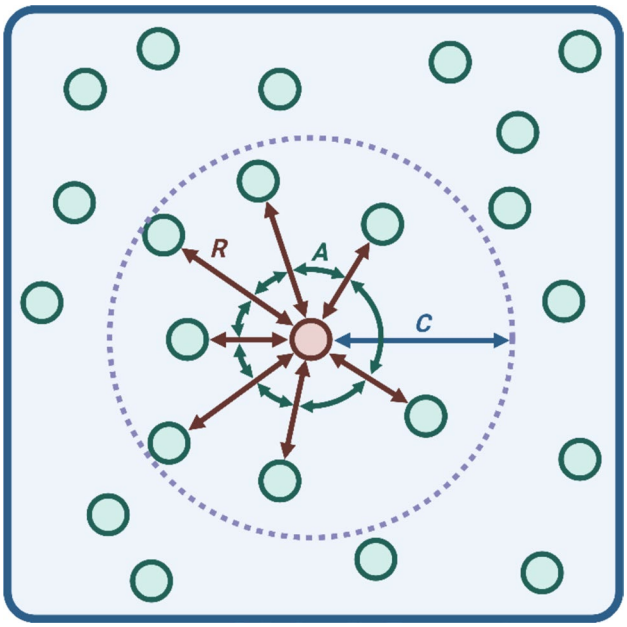

Fig. 10 Schematic representation of radial and angular ACSFs [153]. In this example, the ACSFs are defined for the central atom (red circle) of the system (blue box). A cut-off radius *C* (blue dotted line) defines which neighboring atoms (green circles) are included in the atomic environment of the central atom, and thus in the calculation of the ACSFs. The radial ACSF is the sum of the products of Gaussians and cut-off functions for all atoms within the atomic environment, with the Gaussians describing the distances *R* between the central atom and each neighboring atom (red arrows), whilst the cut-off functions ensure decay to zero in value and slope towards the cut-off radius. The angular ACSF is the sum of the products of the cut-off functions and the angles *A* between the central atom and pairs of two neighboring atoms (green curved arrows)

Atom-centered symmetry functions. In this descriptor, a functional form called a cut-off function is used to define a certain cut-off radius, at which all values decay smoothly to zero (Fig. 10). Within this cut-off sphere, all positions of the neighboring atoms can be described using two symmetry functions called radial ACSFs and angular ACSFs. A radial ACSF is a sum of the products of Gaussians and cut-off functions for all the atoms in the cut-off sphere. A Gaussian describes the distance between the central atom of the atomic environment and its neighbor atom, whilst ensuring decay to zero in value and slope towards the cut-off

radius. Summing all atomic interactions within the cut-off sphere leads to all the information being contained in a single function value, no matter how many atoms are present in the cut-off sphere. Thus, the radial ACSF can describe the distances of neighboring atoms to a central atom in one continuous value. With this, it is still not possible to distinguish different atomic environments in which the same atoms are at the same distances from the central atom, but under different angles from each other. For this, an additional angular ACSF is used, which is a sum of the products of all the terms that describe the angles between the central atom and pairs of two neighboring atoms with a cut-off function. The two ACSFs together can describe the atomic environment of each atom in a system and can be used as input for the development of a ML potential. A more detailed characterization of ACSFs can be found in literature [153–155].

Second-generation NNPs. With these new descriptors acting like a local structural fingerprint of the atomic environment to be considered for the determination of atomic potential energy contributions, HDNNPs can be developed. It is possible to use a separate DFCNN for each atom in the system for the expression of the atomic energy contributions. For each atom, the Cartesian coordinate vector gets converted to a vector of symmetry functions. This forms the input for an atomic NN that outputs an atomic energy contribution. Per atomic NN, two to three hidden layers with 15 to 45 neurons each can predict very accurate atomic potential energies. The atomic NNs are trained in such a way that an atomic NN has the same architecture and weight parameters for all atoms of a certain chemical element. This ensures that all atoms of an element are chemically equivalent and that their potential energy contributions are conforming to permutational invariance by only being a function of their atomic environment. After summation of these contributions, a "short-range energy" of the system is determined, which is regarded as the total potential energy of the system in the case of second-generation HDNNPs (Fig. 11A). This architecture ensures that such an HDNNP is applicable to systems containing a variable number of atoms, as for each element an atomic NN is trained, and for each atom in a given system an atomic NN of that element can be included. This also allows for the application of HDNNPs for systems that are larger than those used for training the weight parameters. Such an architecture is also well suited for parallelization [153].

Third-generation NNPs. Even though second-generation HDNNPs are quite capable of accurately describing potential energies by only considering atomic interactions in a short-range sphere, for many systems long-range electrostatic and dispersion interactions are also crucial. In third-generation HDNNPs, NNs were developed to define environment-dependent atomic charges, from which long-range electrostatic energies can be calculated without truncation. Calculation of these atomic charges is done by a second set of atomic neural networks. They process the atomic environment of each atom of a system (*e.g.*, a vector of ACSFs) with their weights trained in a different fashion compared to the atomic potential energy NNs, in order to output an atomic charge for each atom [153, 156, 157]. First, the atomic charge NNs are trained based on reference atomic charges of a reference dataset, as calculated by a reference electronic structure method (*e.g.*, DFT). From the reference total potential energies, the short-range energies within a given cut-off radius are extracted. This is done by removing the electrostatic energies as computed by the charges given by the atomic charge NNs. The same is done for the short-range forces, by calculating the electrostatic forces and removing them from the reference forces. These short-range energies and forces are then used for the training of the short-range atomic NNs. When applying the new third-generation HDNNP architecture, the set of atom-specific symmetry functions are used as input for both the atomic charge NNs and short-range atomic NNs simultaneously, as to compute both the short-range and electrostatic energies separately. These can then be summed to yield the total potential energy of the system (Fig. 11B).

The need to use reference partial charges for the training of the atomic charge NNs can at first sight appear to be a serious drawback, seeing as different electronic structure calculation methods can yield different partial charges. However, most methods provide very similar partial charges for large interatomic distances, the distances of interest to the atomic charge NNs (beyond a given cut-off distance). The architecture of these HDNNPs is thus quite robust when it comes to choosing different reference electronic structure methods for training. Even though third-generation HDNNPs are a clear advancement in accuracy of the calculated potential energies in comparison to second-generation HDNNPs, they are not frequently used. The reduction of the energy errors often doesn't weigh up to the need to train a second set of NNs, as this increases the computational cost of the technique significantly. NNPs that want to achieve a high transferability towards a wide range of organic molecules could still benefit nicely from considering more long-range interactions [152, 153, 158].

Fourth-generation NNPs. Partial charge determination in third-generation HDNNPs is dependent on local chemical environments. However, in certain systems long-range charge transfers can be significant. For example, ionization states could have an impact on

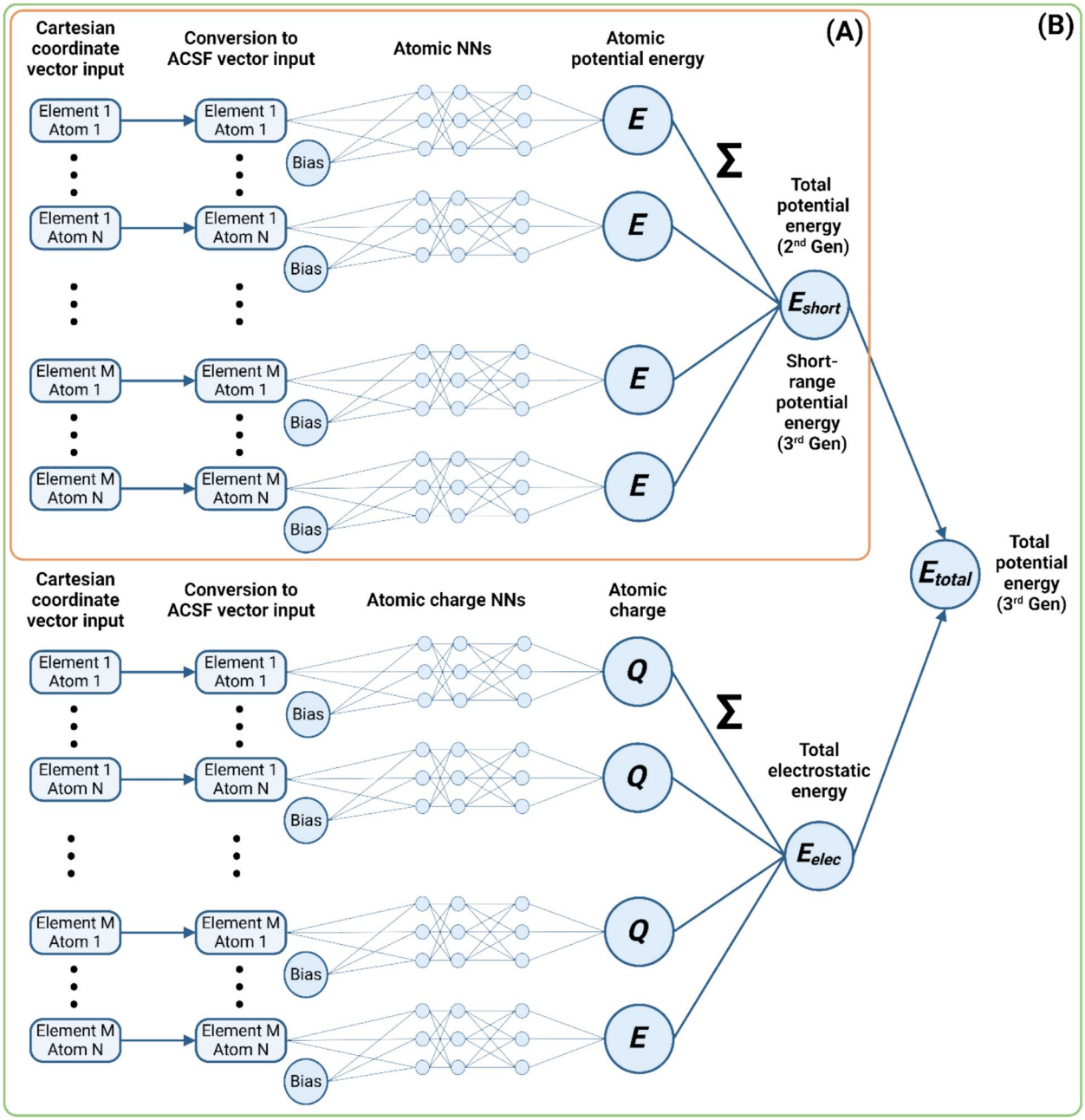

Fig. 11 **A** Overview of 2nd generation HDNNP architectures, in which the Cartesian coordinate vectors of the atoms of a system of interest get converted into a vector of ACSFs, to form the input for individual atomic NNs. These DFCNNs predict the potential energy of each separate atom in the system, delivering the total potential energy of the system when these atomic energies get tallied up. The atomic NNs are trained in the same manner per chemical element. **B** Overview of 3rd generation HDNNP architectures, expanding upon 2nd generation HDNNPs by not only using atomic NNs to predict atomic potential energies per atom of a system, but also atomic charge NNs to predict atomic electrostatic energies. This enables the NNPs to describe more long-range interactions in the system. The short-range potential energies and total electrostatic energies are summed up to provide the total energy of the system. The atomic charge NNs are trained in a different manner per chemical element than atomic NNs [153]

the charge distribution of the entire system, leading to the charges of atoms being dependent on structures of the system outside their local chemical environment. For such systems, second- and third-generation HDNNPs provide incorrect potential energies. Thus, fourth-generation HDNNPs were developed to tackle

systems with non-local charge dependencies. A fourth-generation HDNNP architecture is similar to third-generation HDNNPs in the sense that its total potential energy is once more a sum of short-range energies and long-range electrostatic energies but determined differently [153, 159]. Similar to second-generation HDNNPs, an architecture is built in which the atomic environment of each atom in a system serves as input for independent atomic NNs. However, these NNs are trained to output atomic electronegativities, reproducing a reference dataset obtained from a reference electronic structure method (*e.g.*, DFT). These environment-dependent atomic electronegativities are used in a charge equilibration scheme that depends on the global system in question. This means that certain calculations on these electronegativities lead to atomic charges, from which the long-range electrostatic energy of the system can be calculated [153, 160]. The short-range energies are

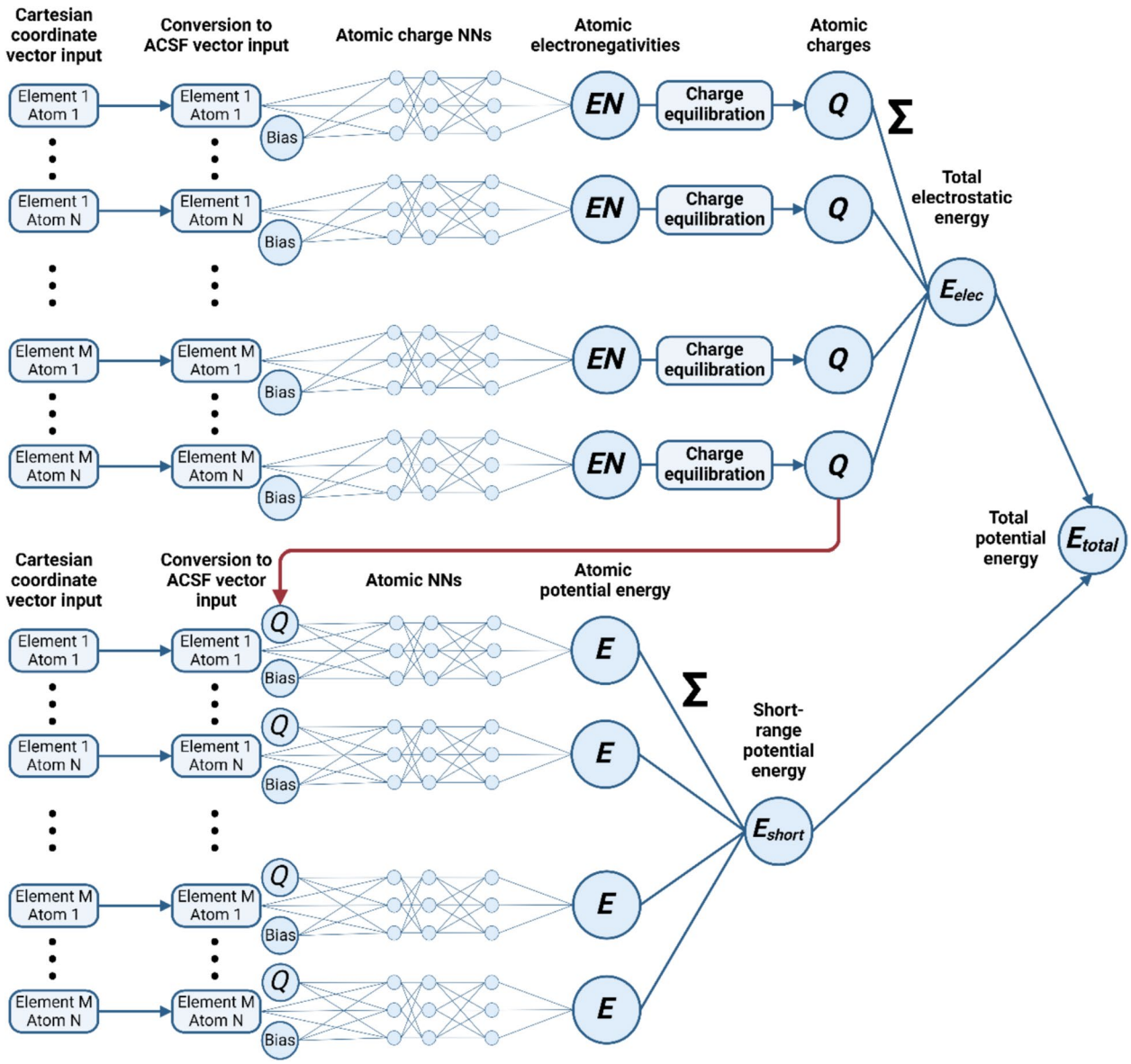

Fig. 12 Overview of 4th generation HDNNP architectures, similar to 3rd generation HDNNPs in the prediction of both atomic charges and atomic potential energies using separate atomic NNs. In this generation, the atomic charge NNs first predict atomic electronegativities that are converted to atomic charges through a charge equilibration scheme depending on the global system of interest. The atomic charges then form an extra level of input for the atomic NNs predicting the short-range atomic potential energies. These two added elements ensure that the architecture considers non-local charge dependencies, resulting in a more accurate total potential energy prediction [153]

calculated as follows. A vector of symmetry functions per atomic environment is used as input for separate atomic NNs, this time trained to output short-range atomic energies. In these atomic NNs an extra input layer neuron is added, feeding as input the atomic charges calculated from the charge equilibration as global descriptors for changes in the local electronic structure. This ensures that both the calculated long-range electrostatic energies and short-range energies adapt to redistributions in the global charge density (Fig. 12). Fourth-generation HDNPPs have a broad applicability and deliver promising results for potential energy calculations of organic molecules [153, 159].

Active learning. A very important aspect to consider is the reference dataset for the training of a HDNNP. The systems and their atomic environments used for training need to be selected carefully, as their functional forms convey how the network learns to define a potential. If a relevant conformational space is mapped through certain reference systems, then the learned HDNNP can be applied to simulations of systems much larger than those used for training, as long as those systems are combinations of the atomic environments learned through the smaller systems. For the most optimal composition of a reference dataset for HDNNPs, a concept called "*active learning*" is applied [153]. First, an initial starting dataset is developed. If the structure of the system to be explored is known, then smaller systems can be determined that include atomic environments relevant for the actual application (including mutations of those systems, such as distortions in the structures). Otherwise, classical MD simulations can be run to sample parts of conformational space. Still, such an initial dataset would be incomplete, lacking certain relevant parts of conformational space. This is where active learning as defined within the field of NNPs comes in: during training of the HDNNP on the initial dataset, it is possible to encounter predictions for structures far from the training data, due to the nonphysical, unbiased functional form of the HDNNP and its highly flexible structure. These differences between the predictions and the structures in the actual training data can be used to identify structures for further description of the conformational space. Through the setup of multiple HDNNPs employing

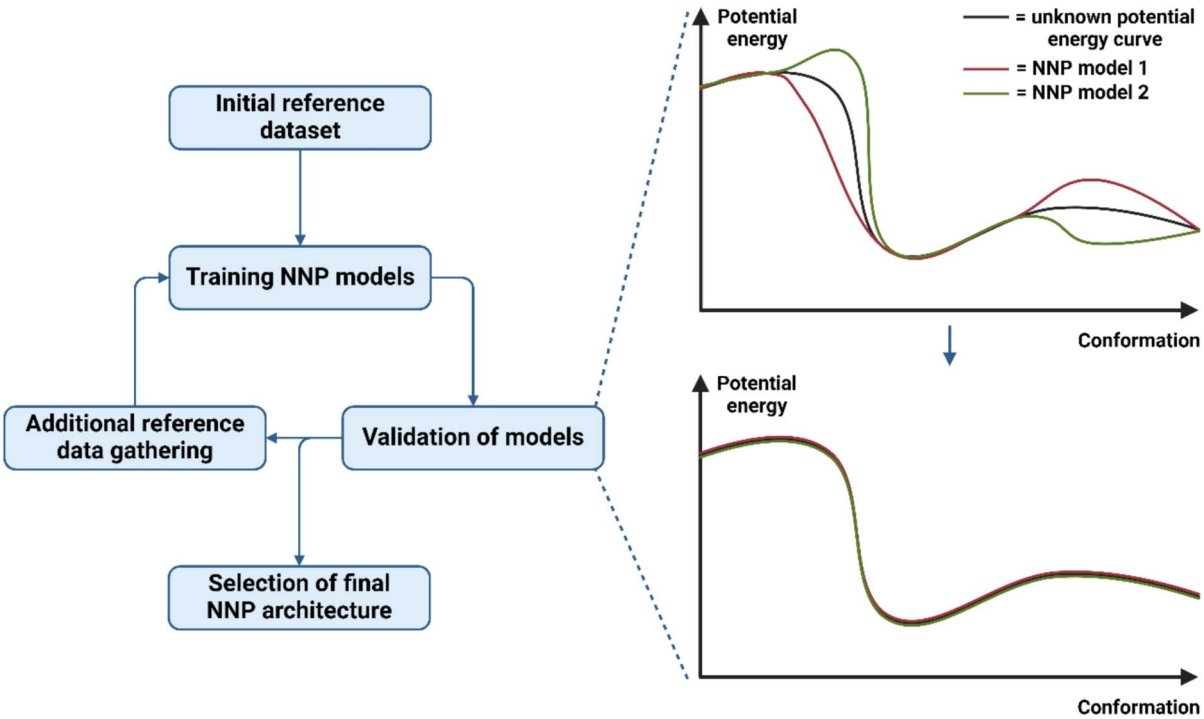

Fig. 13 Overview of the concept of "active learning" [153]. NNP models with differentiating parameters are trained on an initial reference dataset, attempting to capture all the atomic environments relevant for the system of interest to be simulated. Through validation of these models, it is possible to determine what data needs to be added to the reference datasets for further refinement of the NNP models. The models try to learn to describe an unknown potential energy landscape (red and green curves vs. black curve of top right graph). In the conformational regions where the predicted curves differentiate, more information should be provided to the models. These conformations can be obtained through additional MD simulations of the reference structures to provide additional atomic environments for further training. When all trained models converge to describe one potential energy curve (overlapping curves of bottom right graph), the NNPs are optimized, and a final architecture can be selected for the actual application

different parameters, the initial reference dataset can be cheaply expanded to a dataset that properly describes the conformational space to be explored. When the different HDNNPs all describe a potential energy landscape within a certain error threshold, then the HDNNPs are ready for the actual application (Fig. 13).

Next to the structures containing the atomic environments to be used for training, the training dataset should also include reference potential energy landscapes as calculated through certain electronic structure methods. For condensed systems, DFT is most efficient, although still computationally expensive. Active learning offers another advantage here, as selecting structurally distinct conformations to be determined for the improvement of the HDNNP means that only calculations for these structures need to be carried out [153, 161, 162]. During training, all weight parameters need to be optimized, forming a high-dimensional optimization problem. The goal is to find accurate local minima in the conformational landscape with the use of the HDNNP during simulations, which is only possible after careful validation throughout the optimization procedure. HDNNPs exhibit poor performance when it must extrapolate information outside the space of the training set, so active learning should tackle such problems by determining what structures to add to the reference dataset, as to extend the applicability of the HDNNP. Thus, building a suitable reference dataset, training a set of HDNNPs, as well as validating of those architectures are all intertwined processes throughout the entire development process of a final HDNNP [153].

NNP summary. In conclusion, the current state-of-the-art of HDNNPs offers a couple interesting advantages. After training, calculations can be carried out for systems that are much larger than those of the reference dataset, as long as the atomic environments in question are present in the smaller structures of the training set. The current architectures are optimal for parallelization. More conformations per second can be calculated during MD simulations using HDNNPs compared to electronic structure methods, whilst ensuring high accuracy. Even though the same time scales can be obtained using classical FFs, there are additional advantages that could lead to HDNNPs being a better choice of potential over the more conventional methods. HDNNPs don't need knowledge of the physical functional form of the interactions in the system to predict the potential energy with high accuracy. Thus, for systems of which the structure is not fully unraveled yet, the HDNNP can offer interesting observations. Third- and fourth-generation HDNNPs include long-range electrostatic interactions and non-local charge transfers in their observations, leading to even higher accuracies compared to second-generation HDNNPs. That said, these calculations require more computational power, which imposes a need to evaluate case-by-case whether the comparatively small improvement to the accuracy of the potential outweighs the longer computation times. Second-generation HDNNPs are currently the most employed. It is clear that there are still many improvements to be made and challenges to be solved in this ever-growing field. Further improvements in speed and accuracy could be made, as well as a reduction in the demanding step of generating large reference datasets through electronic structures calculations. It could even be interesting to investigate improving the descriptions of atomic environments or gaining a higher level of explainability of the HDNNP predictions [153].

ANAKIN-ME. HDNNPs have already successfully been applied for the simulations of numerous organic molecules [152, 158, 163–173]. One such an example is ANAKIN-ME (Accurate NeurAl networK engINe for Molecular Energies) or ANI for short, a HDNNP trained to be a highly transferable architecture for the potential energy predictions of organic molecules [152, 171]. By now it has been trained on an extensive range of atomic environments of seven elements (H, C, N, O, F, Cl, and S), which together make up about 90% of drug-like molecules. It can make predictions with accuracies comparable to DFT calculations at much higher speedups (~10 [6] factor speedup compared to DFT), and that for several benchmark systems [171]. These calculations still ask more computational time than traditional FF methods.

NNP/MM. Recently, a hybrid method was developed to combat this issue. NNP/MM is a combination of NNPs and molecular mechanics (MM) calculations, where only the most relevant portions of a protein or protein–ligand complex get simulated using NNPs (*e.g.*, the binding pocket), whilst the other parts of the system are simulated with the faster traditional FF calculations. This ultimately boosts the efficiency [172]. Such a hybrid approach has already proven feasible for accurate relative binding free energy calculations, demonstrating higher performance over calculations done on pure MM runs, albeit at slower computation times [173].

DL-guided analysis of MD trajectories

Up until this point, DL techniques have been shown to be applicable in several different ways in computational chemistry. They could aid in virtual screening tests, guide MD simulations to sample specific system states, or even for the calculation of potential energies for simulations. A last facet of the applicability of DL to be explored in this review is its ability to aid in analyzing data created by MD simulations. MD simulations are powerful tools for gaining a better understanding of systems at the molecular level, but efficient sampling of the dynamics of

such systems often requires many trajectories to be calculated. This generates massive amounts of data. On top of that, oftentimes a study wants to observe very specific dynamic elements, e.g., the difference in conformations formed by a protein after ligand binding. Such differences could be extremely subtle, involving only small and specific parts of the system. Finding such low-density signals in big datasets often proves to be a difficult challenge. It is here that DL tools can form a valuable asset. NNs can find complex patterns in high-dimensional datasets, where humans normally would run the risk of overlooking relevant information and introducing human bias in the analysis. DL techniques can thus form interesting tools in steering the characterization of large, high-dimensional datasets. Such approaches have been employed successfully several times in recent research. The following paragraphs will go over a couple interesting examples and approaches of how to apply DL techniques for the data analysis of MD simulations.

CNN by Plante et al. Plante et al. employed a DL approach when analyzing the trajectory data of MD simulations run on G protein-coupled receptors (GPCRs) [174]. They attempted to uncover GPCR conformational differences when different ligands are bound to certain GPCRs, as to better understand functional selectivity and further the rational drug design targeting these receptors. To enhance the analysis of these ligand-bound GPCR MD trajectories, DNNs were trained. They learned to classify picture-like representations of the GPCR states (with ligands removed) into categories divided by which ligands the NN predicts that protein state is bound to. If it is understood how the NN makes its classification decisions by computing what parts of the protein the NN pays most attention to, then structural motifs that indicate the subtle differences in structural dynamics of the GPCRs when binding to different ligands will reveal themselves.

This design approach was applied to two well-characterized GPCRs. MD simulations were run on models of the proteins inserted in a membrane-like structure and docked with ligands representing functionally-selective classes (i.e., a full agonist, an inverse agonist, and a partial agonist). From the MD trajectories, conformations were extracted to create datasets containing tens of thousands of datapoints per protein, further divided into training, validation and test sets. After extracting the ligands from each trajectory datapoint, it was chosen to then convert the data into a 2D picture-like format for CNN training. Such a transformation is obtained by generating a 2D picture, containing as many pixels as the protein in question contains atoms, and representing each atom by coloring each corresponding pixel with RGB values according to the XYZ coordinates of the atom. Each pixel always represents the same atom of the system. Before this conversion is carried out, each atom is subjected to a positional and orientational scrambling procedure to remove translational and rotational bias from the original trajectory. Together with a label of the ligand that was bound to the conformation in question, these 2D representations formed the input for the CNN and the loss optimization process. Per protein, a densely connected CNN was created and tested, based on an established implementation called DenseNet [175]. Each conformation was classified purely on structural differences into categories predicting to what ligand that conformation must be bound to. After optimization with the training and validation sets using their corresponding ligand labels, the accuracy of the developed CNNs was evaluated using the test sets. The networks of both proteins were capable of correctly labelling more than 99% of the test set conformations.

Now that NNs are created that can accurately predict these pharmacological class labels, the key factor that makes this analysis technique so interesting lies in the determination of the molecular features that led to those predictions. For this, a sensitivity analysis was carried out through saliency mapping, computing the gradient of the NN classification score for a specific label with respect to each of the pixels of the input image [176]. The more attention the network paid to a certain pixel for the determination of the class label, the higher this gradient will be. Thus, attention maps can be created, highlighting the pixels with the highest gradients, which allows for determination of the atoms that are important for a certain classification decision. From this, it becomes possible to analyze and evaluate the structural features that were deemed important for the network when specific ligands are bound to the protein in question (Fig. 14). For both GPCRs, the regions most relevant for the binding of full agonists, inverse agonists, and partial agonists became easily traceable, and were analyzed in detail. It was shown that though similar structural motifs are involved in the pharmacological response of GPCRs of different receptor families, the receptors still react differently to the binding of similar ligand classes. Despite the limited scope of the current framework, from the analysis it can be suggested that this method is generalizable and is of interest for further studies regarding functional selectivity of GPCRs. In a similar vein, the same research group developed the interesting Rare Event Detection protocol, employing an unsupervised ML technique called non-negative matrix factorization [177, 178].

GLOW. Do et al. developed the "Gaussian accelerated molecular dynamics, deep Learning and free energy prOfiling Workflow", or GLOW, to predict molecular

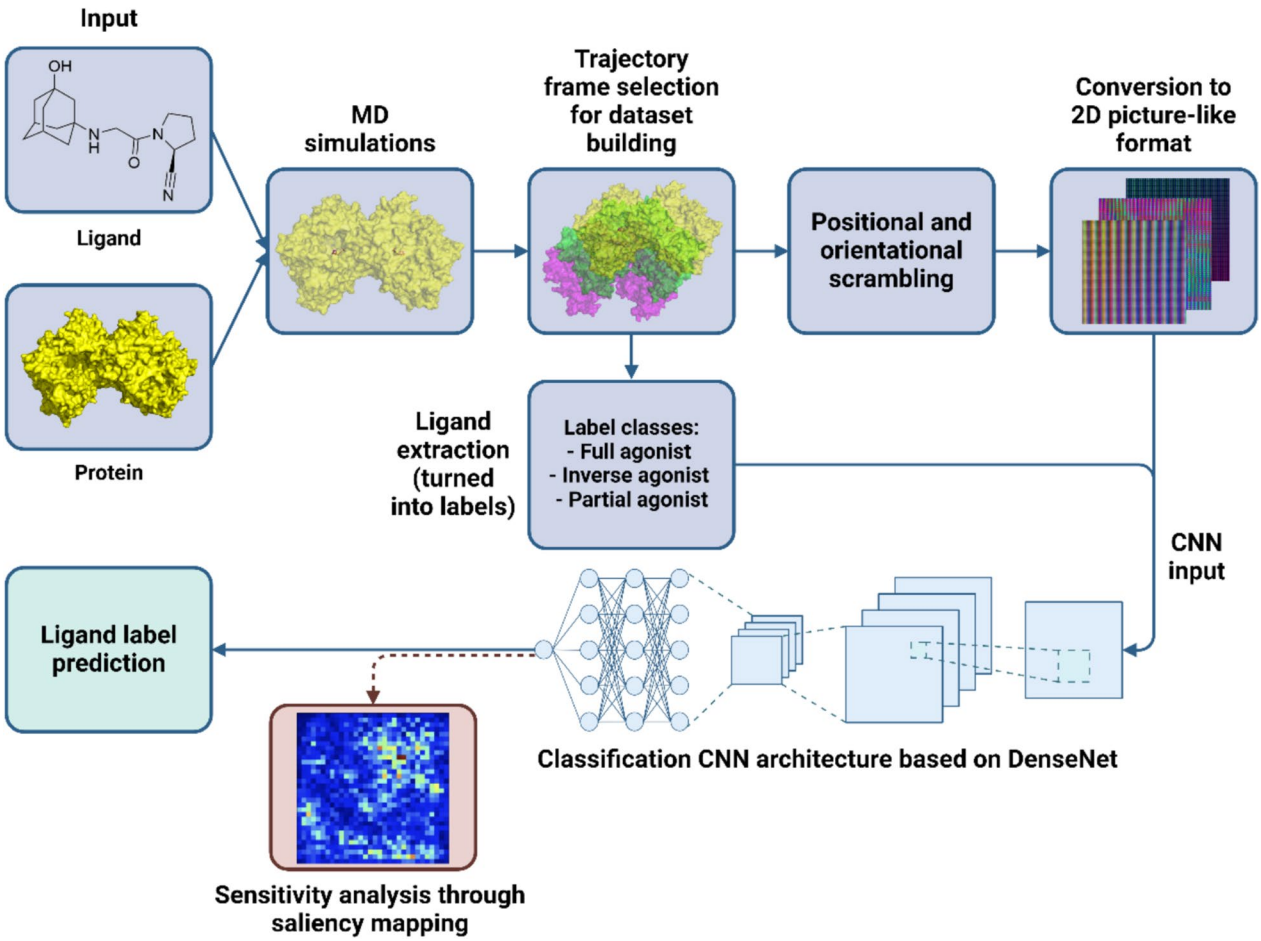

Fig. 14 Overview of the MD trajectory data analysis workflow developed by Plante et al. [174] Ligands representing functionally-selective classes (e.g., full agonists, inverse agonists, partial agonists) are docked onto proteins of interest to create systems for MD simulations. Relevant frames from those simulations are selected for the development of DL training datasets. From these frames the ligand atoms are extracted, but a label is provided with each frame detailing the class of the ligand previously bound in the conformation to allow loss optimization. The protein conformations undergo a positional and orientational structure scrambling procedure to remove bias, after which they are translated into a 2D picture-like format. Each pixel in a picture represents an atom of the protein conformation, with its RGB values corresponding to the XYZ coordinates of the atom. A CNN model based on the DenseNet architecture is then developed and trained on the pictures and class labels to predict the label of the ligand that was bound in a conformation. After optimization, the network's decisions can be analyzed using saliency mapping, as to show the protein regions/structural features relevant for the binding of different ligands

determinants and map free energy landscapes of biomolecules [179]. This study once more focused on GPCRs, attempting to observe the impact of the binding of allosteric ligands on the structural dynamics of GPCRs. For this, GLOW was developed and used for the characterization of the activation and allosteric modulation of the adenosine A_1 receptor (A_1AR). Information was to be obtained about the dynamic structural differences of the GPCR in four different settings: (1) when bound to an agonist; (2) an agonist and an intracellular G protein; (3) an agonist, an intracellular G protein, and a positive allosteric modulator; and (4) an antagonist. Gaussian accelerated MD simulations were carried out on these systems. The obtained trajectories were transformed into a representation suitable as input for a DNN. For 150,000 selected frames of each system, the residue contact maps were calculated using MDTraj and Contact Map Explorer [180]. Such a 2D binary matrix represents all the distances between each possible pair of amino acid residues. The obtained residue contact maps were converted to grayscales images, split into training, validation, and test sets, and then formed the input for CNN architectures. After training, the overall accuracy of the final network when classifying the validation set for each system was over 99%.

The next step of GLOW involves analyzing which residues the developed DNN pays most attention to, and thus which structural elements undergo dynamical changes when bound to specific ligands. For this, first hierarchical agglomerative clustering was used to cluster the conformations obtained for each system. For the most populated structural cluster of each system, the residue contact map was used to calculate attention maps of residue contact gradients, determined by calculating those gradients via gradient-based pixel attribution. Thus, the pixels with the highest gradient will be those deemed most important by the network for its classification decisions. Several criteria were used to then select residue contacts for the computation of the free energy profiles of each system, by reweighting the Gaussian accelerated simulations via the PyReweighting toolkit [181, 182]. By determining the most important residue contacts, it became possible to more efficiently study those structural dynamics relevant in GPCR activation and allosteric modulation. It also allowed for the determination of potential free energy surfaces. Descriptions were made about the loose coupling of the extracellular domains that bind ligands, as well as the intracellular domains that bind G proteins during GPCR activation. It was also described how the extracellular loop 2 plays a critical role in the allosteric modulation of A_1AR. The binding of the positive allosteric modulator was seen to stabilize the GPCR-G protein complex by increasing agonist binding affinity and reducing G protein mobility. In this manner, GLOW provided further insight on top of confirming findings seen in previous studies. In summary, a workflow like that of the study of Plante et al. was described with GLOW but showed the variety possible for choosing how to adapt MD trajectory data into a suitable input for DNNs. It also showed how the decision-making process of the DNN can be analyzed in different ways to gain a better understanding of the structural dynamics of protein systems [174, 179].

DL-RP-MDS. Whereas the previous workflows focused on uncovering the structural dynamics of proteins related to ligand binding, MD simulations and DL analysis tools can also be massively useful in other related areas of computational chemistry. Tam et al. developed the Deep Learning Ramachandran Plot-Molecular Dynamics Simulations workflow, or DL-RP-MDS, for the functional classification of genetic variants (Fig. 15) [183]. Many of the currently identified genetic missense variants are simply classified as variants of uncertain significance, seeing as functional information is lacking to properly determine their impact. To overcome this limitation, the research group focused on developing a new platform for the high-throughput classification of genetic variants, currently targeting the classification of missense variants into benign and deleterious subgroups. Previously, the research group had developed RP-MDS [184]. To measure the impact of genetic variants on protein function, they postulated that this impact is reflected by the stability of the protein

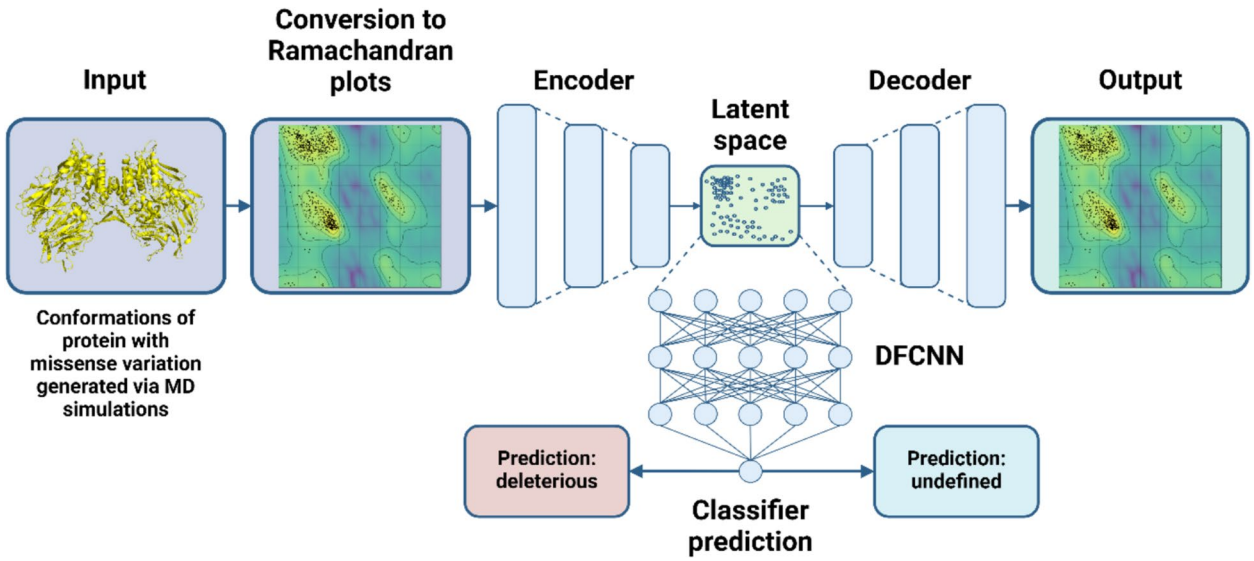

Fig. 15 Overview of the DL-RP-MDS method developed by Tam et al. [183]. To measure the impact of missense variations on protein function, an AE architecture was built and trained. It takes as input the Ramachandran plots of conformations of the query protein with a missense variation of interest, generated using MD simulations. Through its reconstruction of the input via its encoder and decoder layers, it learns a low-dimensional latent representation of the Ramachandran plot input data. This latent space forms the input of a DFCNN classifier that predicts the variants of the protein to either be deleterious or undefined (i.e., benign)

structure. Thus, MD simulations were run on query proteins with missense variations of interest. The torsion angles ϕ and Ψ of the protein secondary structural backbone throughout the obtained trajectories were assimilated to form Ramachandran plots. Alterations that could be seen in the backbone of the proteins reflected the impact of the swapped amino acids. When attempting to classify those mutations as either benign or deleterious, some problems quickly became clear. It was difficult to manually analyze small structural changes and their impact on the function of the protein. Even more difficult was setting a cut-off at which mutations were seen as benign or pathogenic, certainly if for a specific protein not enough mutations were known as "training data". These types of limitations could be overcome with the power of DL. Thus, this follow-up study set out to teach an AE architecture to generate a probabilistic classification, based on Ramachandran plots from MD simulations of proteins and their known missense variant structures. After training, the AE had learned to encode the complex torsional configurations observed in the Ramachandran plots into a low-dimensional latent representation. This reduces the complexity of the original plots but retains crucial information. The latent space then formed the input of a DFCNN classifier, tasked with learning to classify the data by predicting the probability of the variants as being either deleterious or undefined (*i.e.*, benign).

Through thorough validation, the DL-RP-MDS method was shown to be able to successfully classify missense variants into benign or deleterious mutations with an accuracy of over 98%. This computational method was extensively compared to RP-MDS and 22 other in silico genetic variant classification methods and was shown to have the highest performance of all. It reached the highest sensitivity and specificity out of all the methods, as long as a balanced amount of training data was provided. The major advantages of this in silico method are that it overcomes for the most part the overprediction problem of deleterious variants (by enhancing the benign training data using a generative sampling technique), it sees an improvement in accuracy over all other in silico methods, and it provides a continuous value for its classification predictions. The method showcases the need of such classification methods to be gene-specific, considering the intrinsic structural differences between the genes. However, limitations are still present. The classification may still be skewed towards deleterious variant predictions. Pathogenic variants may also not influence the structure of the corresponding protein, not allowing the network to recognize it as pathogenic. Lastly, the model needs lots of finetuning per specific gene and protein. In summary, the study showed that structural change is a valuable property for variant classification, and that this analysis workflow is readily applicable for the classification of other unclassified missense variants. Venanzi et al. also focused their work on predicting the impact of point mutations on the activity of proteins [185]. To address this protein engineering challenge, they employed many parallel ML techniques [amongst which a multi-layer perceptron (MLP)]. In this process, they noted the importance of dynamical information gained through MD simulations in addition to traditional sequence and structural information for the training and testing of qualitative ML models.

Discussion on dimensionality reduction tools. Fleetwood et al. made an extensive analysis comparing many ML techniques as dimensionality reduction tools for streamlining the analysis and feature extraction process of MD simulations [186]. Multiple DNNs were discussed in the paper. They divided the techniques discussed in their review into two categories: supervised and unsupervised learning. Within the supervised learning category, they included MLPs, and within the unsupervised learning category, they included restricted Boltzmann machines (RBMs) and AEs. The performances of all the different ML tools were benchmarked on a newly developed toy model mimicking MD simulations. These benchmarks were used to build a checklist to aid other researchers in making a more well-thought-out decision when choosing a ML technique for trajectory data analysis.

First, the unsupervised learning methods, RBMs and AEs, were evaluated. When employing Cartesian coordinates as input type for the training of NNs, it quickly became clear that it was very difficult for the NNs to distinguish the atomic features important to dynamics seen in the toy model systems from those irrelevant. This once more shows that Cartesian coordinates form a bad descriptor of an atomic environment for DL model training. However, a performance similar as with principal component analysis was obtained when using interatomic distances as input features. For the AE, performance dipped the larger the system became for the training of the NN, indicating that AEs are more difficult to train on higher dimensional systems. When looking at the supervised learning methods, MLPs were easily successful at identifying the most important features from systems for both Cartesian and internal coordinate input features. However, it did also identify irrelevant atoms as important, indicating lower specificity for this technique. In general, unsupervised learning methods seemed to underperform compared to supervised methods in cases where the labels of the input data points were known (in this context meaning that the

most important features in a certain state of a system were known), seeing as they can't employ this valuable information.

Many different versions of the NNs were tested through the variation of different hyperparameters. From this, it was concluded that optimization of the hyperparameters to a certain degree led to improvements in performance, but that small changes around the most optimal values only resulted in small changes with regards to which features were deemed important by the network. Conclusions from the toy model analysis showed the importance of only using appropriate input parameters for DL training (*e.g.*, internal atom distances, backbone dihedral angles). It also showed that it isn't always necessary to invest many resources in finding the most optimal set of hyperparameters for a NN, as long as the hyperparameters are properly tuned to some degree. And lastly, that the choice of ML technique or NN depends on the information available for training (*e.g.,* choosing between unsupervised and supervised learning). It appears most useful to combine multiple ML/DL techniques, forming a cohesive picture when combined. To this end, the research group developed a checklist to help other researchers in identifying the most optimal ML/DL techniques to help solve their research questions. The underlying message of the review conveys the immense usefulness of ML/DL tools in aiding along the data analysis of biomolecular simulations, as long as it is merely used as a toolbox to aid scientists in their analysis and not deployed autonomously.

Summary. This past section offered a non-exhaustive look into how different DL techniques could be employed to aid in the analysis of MD trajectory data for a multitude of research purposes. It is clear that many different options can be considered, depending on the input features available for training. Testing multiple different architectures and combining the insight they offer should be considered the ideal workflow, more valuable than spending the same number of resources on perfecting the hyperparameter optimization of a single network. Converting the trajectory data into suitable input features should be regarded as essential to the success of these analysis techniques.

Review highlights

Relevance of DL implementations and key toolset

In the current age of Big Data, ever-growing amounts of data are available in the fields encompassing computational chemistry, transforming DL into a very powerful and advantageous technique to implement in many different workflows. In silico techniques already complemented in vitro and in vivo techniques within the fields of biochemistry and medicinal chemistry, validating results and delivering valuable additional insights. However, many of the widely employed techniques come with their own limitations. Hardware limitations restrict the types of algorithms that can be used, limiting in their own right the level of accuracy that can be achieved within reasonable computation times. One way in which similar or even higher levels of accuracy can be reached at faster speeds is through the implementation of AI, of which DL is by now a well explored and validated avenue. All the different DL architectures are focused on employing NNs to extract useful patterns or information from input data to then make informed decisions or predictions. The large amounts of data needed to train DL models for various processes is becoming more readily available by the day, and research in the field of XAI and Interpretable Machine Learning allows for more transparent models from which more reliable conclusions can be drawn (as further discussed by Jiménez-Luna et al. for the field of drug discovery) [187]. This review discusses how DL could be implemented in important in silico molecular modelling techniques such as VS and MD simulations to improve drug discovery workflows.

The most important tool in the toolkit of a scientist who is looking to implement DL techniques into their workflows, is the dataset to train their model. Given the fact that a DL model is a data-driven intelligent system, it can only be as powerful as the data that is fed to it. Thus, to properly develop a NN, a dataset of high quality is certainly as vital as the code to set up the network. Picking out the right dataset, both in terms of quality as well as quantity, is a key first step in the development of a DL model. It is vital to reflect upon the data required in terms of what information it can provide to the NN and how this information reflects onto the subject the network needs to be trained on. The quantity of data in a dataset can vary wildly from a couple hundred examples to billions of data points. More data means more computational time needed to train a model but could on the other hand be interesting for better generalization of the NN. The quality of the data is very important as well and often requires pre-processing steps to eliminate noisy data points and wrong labels. A dataset can be manually created depending on the type of project to be executed, but there are also many high-quality repositories available online that provide well-maintained, high-quality datasets for training AI models. Examples of such repositories are the UCI Machine Learning Repository, Kaggle and the Google Dataset Search program [188–190]. Within the field of drug discovery, compound libraries can vital, of which ChEMBL, PubChem, and ZINC databases are good examples [12–17]. However, care should be taken to mix data from different sources, as very

recently it has been shown that this can lead to adding more noise to the data [191].

Once a dataset has been thoughtfully selected for the development of a deep learning model, the data needs to be further pre-processed to create highly curated and balanced data. This comes in the form of formatting, eliminating data points with missing values or assigning actual values to those that are missing. Another important factor to pay attention to is the presence of skewed data in datasets. It is crucial to balance out such inequalities in the data by down sampling the majority class and upscaling the minority class with certain factors. This is done to avoid bias of the model towards the majority class (e.g., active or inactive compounds, bound or unbound ligands) [192]. Feature scaling is also an important preprocessing step to keep track of, as most DL algorithms benefit from learning features that are similar in scale. Features from different objects can only be properly compared if the other attributes of the objects are similar in context, ensuring that each attribute will have an equal contribution to DL model predictions. Feature scaling is most often done through normalization and standardization techniques. Normalization binds all values between [0,1] or [−1,1], whilst standardization transforms the data to have a zero mean and variance of one [193]. Finally, care needs to be taken to split up the entire dataset into three subsets: a training, validation, and test set. All three sets, though possibly differing in size, need to be similar to each other and may not contain skewed data. The model can then be iteratively trained and validated on the training and validation subset, whilst optimizing its hyperparameters: a process called cross-validation. The test set is reserved for the final test of the model before use [194, 195].

Of course, data isn't the only important element needed for the development of DL applications within the field of computational chemistry. Throughout the "Deep learning models applied to molecular modelling" section, many different tools were shown to be vital for the setup of VS steps, MD simulations, and the DL models weaved into the workflows of the discussed research examples. Table 2 gives a noncomprehensive summary of all these mentioned tools, including software libraries, analysis tools, databases and benchmark datasets, to introduce researchers to the enormous toolkit at their disposal.

DL and VS

There are many ways in which DL could improve VS steps. DL models could form a primary screening step, mimicking either LBVS or SBVS steps in narrowing down a large library of compounds to a more concise dataset that could then be used for traditional VS methods. DL model predictions ask less computational time than traditional docking screens, so this hybrid method could improve the throughput rate of an entire VS approach. Another way in which DL could be implemented for VS steps is by using generative models to generate a dataset for traditional VS methods. A large compound library could be used to train a generative learning model to generate its own compounds with specific, predefined characteristics (e.g., compounds with binding capacity towards the target protein, synthesizable compounds, drug-like molecules). Caution with this method of sampling is warranted, as this generative step forms the first step of a long drug discovery process, so it is advantageous to sample enough of the chemical landscape as to not overlook promising molecular structures.

A DL model could also be developed to perform a task similar to a docking screen and predict either binding affinities of complexes, or even a best fitting binding pose. A final approach could go even further than this and use generative models to develop molecules that optimally fit within a binding pocket of interest. While these last options are incredibly powerful, seeing as they entirely eliminate the need for traditional docking computations and are thus capable of the highest speedups, they do require careful consideration throughout their development. The training dataset for the models needs to be heavily curated to avoid overfitting or skewering, as to allow for proper generalization of the model to obtain accurate results when applying it to external data. Most models still have a long way to go, but models like DiffDock and AlphaFold 3 are the current state-of-the-art performing docking pose predictions and protein–ligand binding affinity predictions, while outperforming traditional docking methods in accuracy, as well as speed. Both form interesting applications for new drug development projects. A summary of the methods discussed throughout the "Deep learning and virtual screening" section is given in Table 3.

DL and MD

Imbuing MD simulations and their analyses with DL approaches can be compelling for a plethora of reasons. Observing biological/biochemical processes by employing classical MD simulations is not self-evident, requiring long computational time periods. Even though hardware is improving, and enhanced sampling MD methods provide new ways for better sampling of longer timescales, DL has now also become a feasible and interesting approach. Instead of trying to reach convergence on the sampling of a certain process of interest simply by for example running enough parallel simulations from different starting points, it is now possible to guide along

Table 2 Summary of the tools mentioned throughout this review used to setup VS steps, MD simulations, or DL models

Name DL/VS/MD-tool	Description	Application example(s)
ML/DL software libraries		
PyTorch [6]	Open-source ML/DL framework for NN development and training	[37, 141]
TensorFlow [5]	Open-source ML/DL framework for NN development and training	[137, 140, 174, 179]
VS tools		
AutoDock Vina [196, 197]	Open-source program for molecular docking	[39, 40, 43, 53, 81]
SMINA [19]	Fork of AutoDock Vina focused on improving scoring functions and energy minimization	[106, 110]
GNINA [20, 198]	Fork of SMINA, employing CNNs for improved support of scoring functions and ligand optimization	[106, 110]
QuickVina 2 [199]	Fork of AutoDock Vina using heuristics to reach significant speedups at similar accuracy	[47]
QuickVina-W [21]	Update of QuickVina 2, providing the ability to dock blindly if the docking site is unknown	[106, 110]
GLIDE [22]	Schrödinger software package for ligand-receptor docking	[106, 110]
RDKit [23]	Open-source cheminformatics toolbox	[34, 47, 53, 106]
Open Babel [24]	Open-source cheminformatics toolbox	[79]
OEChem KT [25, 200, 201]	OpenEye Scientific programming library for chemistry and cheminformatics	[103]
PaDEL [38]	Open-source software for calculating molecular descriptors and fingerprints	[37, 39]
AutoClickChem [48]	Currently out-of-date open-source software for automated in silico chemical synthesis	[47]
MD tools		
GROMACS [134]	Open-source software for high-performance MD simulation and output analysis	[183]
Amber [135]	Suite of biomolecular simulation programs	[179]
NAMD [136]	Open-source software for high-performance simulation of large biomolecular systems	[137, 174]
MDTraj [180]	Open-source Python library for MD trajectory manipulation and analysis	[179]
PyReweighting [181]	Open-source Python toolkit to facilitate the reweighting of accelerated MD simulations	[179]
General databases for ML/DL development		
UCI ML repository [188]	Collection of accessible databases for ML/DL training and analysis	
Kaggle [189]	Data science community platform	
Google Dataset Search [190]	Search engine for freely available online data	
Compound library databases		
ChEMBL [12–14]	Manually curated database of bioactive molecules with drug-like properties	[36, 37, 39, 53]
PubChem [15]	World's largest free chemical information database	[36]
ZINC [16, 17]	Curated collection of commercially available chemical compounds for VS	[47, 53, 174]
PDBbind [35]	Collection of experimentally measured binding affinity data for biomolecular complexes	[34, 79, 81, 106, 110]
Selleck [202]	Bioactive compound libraries that consist of small molecules with validated biological and pharmacological activities	[39]
TargetMol [44]	Research supplier for compound libraries of small molecule compounds	[43]
UniProt [203]	Freely accessible resource for protein sequence and functional information	[64]
Benchmarking sets		
Astex Diverse Set [82]	Diverse, high-quality test set for the validation of protein–ligand docking performance	[79]
PoseBusters [114]	Python package to perform standard quality checks on DL-based protein–ligand docking methods	[112]
CASF-2016105	Open-access benchmark to assess and compare scoring functions in several metrics	[104]

This is a noncomprehensive list, meant to inspire researchers of the range of tools at their disposal. It is divided into (1) software libraries used to develop ML/DL models, (2) the various mentioned VS tools and (3) MD tools, (4) databases useful for training DL models, in general as well as (5) compound library databases, and lastly (6) mentioned tools useful for benchmarking certain DL models

Table 3 Summary of DL models mentioned throughout the "Deep learning and virtual screening" section of this review used to aid in performing VS workflows

DL-VS method/tool	Description
LBVS-type screening step	
IVS2vec [34]	DFCNN that uses as input ligand compound vectors generated by Mol2vec (a ML method producing high-dimensional embeddings of molecular structures) and is capable of a binary protein classification: proteins with either a high or a low possibility of binding with a query ligand
DEEPScreen [36]	Collection of 704 CNNs, each an individual predictor of favorable interactions between a query protein and small molecule ligands
DeepScreening [37, 39, 40, 42]	Freely accessible web server capable of training DL models for either classification tasks or regression tasks. For classification tasks, DFCNNs perform binding probability predictions on a provided chemical library. For regression tasks, RNNs generate a de novo compound library and then perform binding probability predictions against a query protein
Drug repurposing DFCNN by Zhang et al. [43, 45]	DFCNN that uses as input ligand compound vectors generated by Mol2vec and that is capable of predicting protein–ligand binding probabilities. It uses only molecular and chemical information of the compound vectors, not considering spatial information
DeepBindBC [43, 45]	DFCNN that uses as input protein–ligand complex structures as generated by AutoDock Vina (thus considering spatial information) and that can estimate protein–ligand binding probabilities
Generative model for VS steps	
GAN by Andrianov et al. [47]	GAN that consists of an AE encoder and DFCNN discriminator, that is able to generate molecular fingerprints of compounds similar to those from its training set, as to then identify comparable compounds from existing compound libraries for further use in a drug discovery workflow
LSTM RNN by Arshia et al. [53]	LSTM RNN retrained through DTL from a network called LSTM_Chem capable of capturing the features of SMILES molecular representations. It was retrained through 10 generations of refinements to learn to generate SMILES of unique, original and valid compounds, each generation with better binding affinity to a query protein
WAE by Das et al. [64]	A type of VAE with a GRU encoder and decoder, able to capture the features of short peptide sequences (max. 25 amino acids). Using this model's latent space and four bidirectional LSTM classifier models, the architecture can generate diverse, valid AMPs with broad-spectrum potency and low toxicity, used for further in silico, in vitro and in vivo testing
Binding affinity predictor	
DeepBindRG [81]	ResNet that uses as input 2D binding interface-related matrices of protein–ligand complexes and predicts their binding affinity
Pafnucy [79]	Model built of convolutional and dense layers, capable of using 4D input information (3D coordinates and an additional feature vector) to predict the binding affinity of protein–ligand complexes
AEV-PLIG [104]	Attention-based GNN model that uses as input protein–ligand interaction graphs to capture the interplay of interactions determining binding affinity and predict binding affinities for the query complexes
Pose predictor	
EquiBind [106]	Combination of a graph matching network and GNN that uses as input protein–ligand complex graphs to perform one-shot predictions of the most optimal binding poses of query ligands in proteins (without binding affinity values)
TANKBind [107]	Similar GNN approach to EquiBind, using an additional bias parameter set to better prevent steric clashes and unrealistic conformations during the one-shot binding pose predictions. It also includes an additional module that allows for binding affinity predictions
DiffDock [110]	Diffusion generative model that starts with random conformations of a query ligand docked onto a protein and uses a reverse diffusion process to sample realistic protein–ligand binding poses and iteratively refine the system towards a most optimal final binding pose prediction
AlphaFold 3 [112]	Attention-based architecture capable of predicting the 3D structure of proteins with unknown tertiary and quaternary structures based on their amino acid sequence, as well as predict interactions with other proteins, small molecule ligands, nucleic acids, and modified or non-canonical residues
Generative model to replace VS steps	
TargetDiff [119]	3D equivariant diffusion model that can generate 3D molecular structures befitting a query protein binding site, together with a binding affinity estimation
PILOT [120]	3D equivariant diffusion model that can generate 3D molecular structures befitting a query protein binding site (while maintaining high synthetic accessibility), together with a binding affinity estimation
Pocket2Mol [121]	E(3)-equivariant generative network that consists of a GNN generating 3D molecular structures befitting a query protein binding site (while maintaining drug properties such as drug likeness and synthetic accessibility) and a sampling algorithm that helps sample structures conditioned on the query pocket representation

Table 3 (continued)

DL-VS method/tool	Description
FRAME [124]	Series of SE(3)-equivariant generative networks capable of generating 3D molecular structures befitting a query protein binding site. From a starting molecule, the architecture selects locations on which to add certain molecular fragments to better fit a protein pocket in question, until a certain user-specified goal is reached (e.g., molecular weight)
TacoGFN [122]	GFlowNet-based approach that can generate 3D molecular structures befitting a query protein binding site combined with a binding affinity estimation
AHC [127]	Reinforced Genetic Algorithm employing neural models to build, evolve and optimize 2D molecular structures with binding affinity to a query protein through attempting to optimize a structure-explicit scoring function
AutoGrow 4 [126]	Genetic Algorithm that evolves and optimizes 2D molecular structures from random seeds to compounds with binding affinity to a query protein through attempting to optimize a structure-explicit scoring function

This is a noncomprehensive list of available models, meant to inspire researchers of the range of techniques at their disposal. It is divided into (1) models used to perform LBVS-type screening steps, (2) generative models used to generate datasets for further VS steps, (3) models used to predict binding affinity values of complexes or (4) predict poses of ligands within query proteins, and (5) generative models used to entirely replace other VS techniques, generating molecules that fit within a binding pocket of interest

Table 4 Summary of DL models mentioned throughout the "Deep learning and molecular dynamics simulations" section of this review used to aid in performing MD workflows

DL-MD method/tool	Description
DL-guided enhanced conformational sampling	
AE by Degiacomi [137]	AE model with a latent space encoded from the flattened Cartesian coordinate systems of MD simulation frames of a query protein, from which new protein conformations could be interpolated as starting structures for follow-up MD simulations
DeepDriveMD [140]	Workflow for protein folding problems employing a CVAE model with a latent space encoded from contact map representations of the flattened Cartesian coordinates of MD simulation frames of a query protein. These conformations get clustered in the latent space in regions with biophysically relevant features, from which protein conformations could be identified as starting coordinates for follow-up MD simulations, in order to speed up the sampling of a protein folding process
VDE workflow by Sultan et al. [141]	Workflow to sample the most important dynamical behavior of a protein, employing a VDE architecture with a latent space encoded from through-tICA-dimensionality-reduced conformational states of the query protein. The latent coordinate of the VDE was used as CV for well-tempered metadynamics simulations for the sampling of the most important dynamics of the system
Neural network potentials	
ANAKIN-ME [152, 171]	Accurate NeurAl network engine for Molecular Energies or ANI, a HDNNP trained to be a highly transferable architecture for the potential energy predictions of organic molecules
NNP/MM [172]	A hybrid method combining NNPs and MM calculations, where specific regions of a system are simulated using NNPs and the other parts through faster traditional MM calculations, in order combine the accuracy and efficiency strengths of the two separate simulation methods
DL-guided analysis of MD trajectories	
CNN by Plante et al. [174]	CNN that helps uncovering conformational differences when different ligands are bound to a query protein. The model is trained on 2D scrambled pixel maps of protein conformations of protein–ligand complex MD simulations to predict what ligand that protein state is bound to. This architecture is coupled to an explanation technique highlighting the protein regions in each frame that the network paid attention to for its classification decision. This allows for the analysis of structural features possibly undergoing dynamical differences for each simulation system type
GLOW [179]	CNN that helps uncovering conformational differences when different ligands are bound to a query protein. The model is trained on 2D residue contact maps of protein conformations of protein–ligand complex Gaussian accelerated MD simulations to predict what ligand that protein state is bound to. This architecture is coupled to an explanation technique highlighting the protein regions in each frame that the network paid attention to for its classification decision. This allows for the analysis of structural features possibly undergoing dynamical differences for each simulation system type, as well as map the free energy landscapes of these conformations
DL-RP-MDS [183]	Workflow for the functional classification of genetic missense variants into benign or deleterious subgroups. MD simulations are run for a query protein and its missense variations of interest. Then, an AE is trained on the Ramachandran plots of the obtained simulation frames. The latent space of the AE is connected to a classification DFCNN to predict the missense variants to be either benign or deleterious

This is a noncomprehensive list of available models, meant to inspire researchers of the range of techniques at their disposal. It is divided into (1) models used for enhanced conformational sampling, (2) models used as NNPs and (3) models used to guide the analysis of MD trajectories

the sampling in the simulations using DL techniques. Generative models can be trained to identify intermediate states in such processes based on data provided by an initial set of MD simulations, after which follow-up simulations can be initiated from these identified states. Such models can also be trained to predict the impact of system perturbations. If, for such approaches, general DL models can be developed that are both scalable and explainable, and that can be retrained through DTL for specific research objectives, then significant speedups in these sampling methods can be achieved. Concurrent MD/DL workflows do require efficient workload and performance balancing.

A second approach is the use of NNPs during MD simulations: NNs capable of predicting the molecular energies of a system at each timestep of a simulation. Such predictions can offer QM-levels of accuracy at high speedups compared to traditional QM or possibly even MM methods. NNP models can describe atomic interactions without bias, and if trained in a broad manner through a suitable reference dataset built using active learning, they can be transferred between systems. ANI is a current state-of-the-art, highly transferable architecture that was built for simulations of organic molecules [152, 171]. Hybrid NNP/MM methods are also possible, combining the strengths of both methods in a synergistic manner, much like QM/MM methods [172].

Lastly, DL models can be immensely useful for the analysis of MD trajectory data. MD simulations often cause large amounts of high-dimensional data to be generated, and DL tools can aid in finding complex patterns that other analysis methods could overlook. DL models can be trained to make certain classification or regression predictions on the trajectory data, after which XAI techniques can be employed to determine the features present in the data that led the model to make those predictions. Such techniques can be feature-engineered to lead researchers to investigate specific elements of the data, based on what the model deems important. The most valuable approach appears to be to train multiple architectures on the same data and combine the observations they provide to reach satisfactory conclusions. If the features of the trajectory data can be properly converted into suitable input descriptors for DL training, then these approaches can form powerful techniques for guiding along difficult analyses. A summary of the methods discussed throughout the "Deep learning and molecular dynamics simulations" section is given in Table 4.

Conclusions and future perspectives

This review focused on DL implementations throughout molecular modelling techniques and provided extensive examples of recent approaches in the field found in literature. It becomes clear that not only the type of architecture employed and the setup of the model is critical to the success of an approach, but also the quality and quantity of the provided data. The way in which that data is converted into suitable input for a NN is equally important. These different elements require carefulness at each step, trial-and-error and troubleshooting as needed, and a critical mind that vigorously analyses any obtained results. There are still many improvements to be made in all the approaches discussed above, but at a rapid pace, DL is transforming the field of computational chemistry and accelerating the discoveries made within.

Among these improvements is the need for more high-qualitative data, certainly in the field of medicinal chemistry/biochemistry, such as protein–ligand binding affinity data. More generalized benchmarking datasets and tools for DL applications are also desired, since it is currently often a difficult challenge to compare different models. During model development, more emphasis should be placed on transferability of models, allowing for the training of general architectures for a certain task (*e.g.*, generative molecular design, generalizable MD analysis techniques, or faster, more accurate HDNNPs). These types of models could then be incorporated into generalized workflows, allowing research groups to compare their findings. Another critical element to incorporate in new research is a focus on explainability and transparency. Using the available techniques in the field of XAI to ensure it is understood how a model comes to its predictions is essential to prevent models that are either too general, overfit, or introduce bias in their results. Lastly, this field would benefit from more easily accessible DL tools, making the training and development of models more comprehensive for the general scientific community.

Abbreviations

$3CL^{pro}$	3 Chymotrypsin-like protease
A_1AR	Adenosine A_1 receptor
ACSF	Atom-centered symmetry function
AE	Auto-encoder
AI	Artificial intelligence
AMP	Antimicrobial peptide
CNN	Convolutional neural network
CV	Collective variable
CVAE	Convolutional variational autoencoder
DFCNN	Dense fully connected neural network
DFT	Density functional theory
DL	Deep learning
DNN	Deep neural network
DTL	Deep transfer learning
FF	Force field
GAN	Generative adversarial network
GNN	Graph neural network
GPCR	G protein-coupled receptor
GPU	Graphics processing unit
GRU	Gated recurrent unit
HDNNP	High-dimensional neural network potential
IVS	Inverse virtual screening

LBVS	Ligand-based virtual screening
LSTM	Long short-term memory
MD	Molecular dynamics
ML	Machine learning
MLP	Multi-layer perceptron
MM	Molecular mechanics
NN	Neural network
NNP	Neural network potential
PES	Potential energy surface
QM	Quantum mechanics
RBM	Restricted Boltzmann machine
RdRp	RNA-dependent RNA polymerase
ReLU	Rectified linear unit
ResNet	Residual neural network
RMSD	Root-mean-square-deviation
RNN	Recurrent neural network
SBVS	Structure-based virtual screening
tICA	Time-structure-based independent component analysis
VAE	Variational autoencoder
VDE	Variational dynamics encoder
VS	Virtual screening
WAE	Wasserstein autoencoder
XAI	Explainable artificial intelligence

Acknowledgements
Figures 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 created with BioRender.com. Complexes shown in Fig. 6 created with the open access version of DiffDock on Huggingface.co.

Author contributions
S.D. wrote the main manuscript text, H.D.W and J.A.M. reviewed the manuscript.

Availability of data and materials
No datasets were generated or analysed during the current study.

Declarations

Competing interests
The authors declare no competing interests.

Received: 20 December 2024 Accepted: 12 March 2025
Published online: 08 April 2025

References

1. Wang J, Bhattarai A, Do HN, Miao Y (2022) Challenges and frontiers of computational modelling of biomolecular recognition. QRB Discov 3:1–12. https://doi.org/10.1017/QRD.2022.11
2. Hollingsworth SA, Dror RO (2018) Molecular dynamics simulation for All. Neuron 99:1129–1143. https://doi.org/10.1016/J.NEURON.2018.08.011
3. De Vivo M, Masetti M, Bottegoni G, Cavalli A (2016) Role of molecular dynamics and related methods in drug discovery. J Med Chem 59:4035–4061. https://doi.org/10.1021/ACS.JMEDCHEM.5B01684
4. Amini A, Amini A, Lolla S. (2024) MIT 6.S191 | Introduction to deep learning. http://introtodeeplearning.com/.
5. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C et al (2016) TensorFlow: large-scale machine learning on heterogeneous distributed systems. ArXiv. https://doi.org/10.48550/arXiv.1603.04467
6. Paszke A, Gross S, Massa F, Lerer A, Bradbury Google J, Chanan G et al. (2019) PyTorch: an imperative style, high-performance deep learning library. In: NIPS'19: proceedings of the 33rd international conference on neural information processing systems. arXiv, 8026–37. https://doi.org/10.48550/arXiv.1912.01703. https://pytorch.org/.
7. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT press, Cambridge, MA
8. Alzubaidi L, Zhang J, Humaidi AJ, Al-Dujaili A, Duan Y, Al-Shamma O et al (2021) Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. J Big Data 8:1–74. https://doi.org/10.1186/S40537-021-00444-8
9. Shinde PP, Shah S (2018) A review of machine learning and deep learning applications. In: 2018 Fourth international conference on computing, communication control and automation (ICCUBEA 2018). 1–6. https://doi.org/10.1109/ICCUBEA.2018.8697857
10. Sarker IH (2021) Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. SN Comput Sci 2(6):1–20. https://doi.org/10.1007/S42979-021-00815-1
11. Carpenter KA, Cohen DS, Jarrell JT, Huang X (2018) Deep learning and virtual drug screening. Future Med Chem 10(21):2557–2567. https://doi.org/10.4155/FMC-2018-0314
12. Mendez D, Gaulton A, Bento AP, Chambers J, De Veij M, Félix E et al (2019) ChEMBL: towards direct deposition of bioassay data. Nucleic Acids Res 47(D1):D930–D940. https://doi.org/10.1093/NAR/GKY1075
13. Davies M, Nowotka M, Papadatos G, Dedman N, Gaulton A, Atkinson F et al (2015) ChEMBL web services: streamlining access to drug discovery data and utilities. Nucleic Acids Res 43(W1):W612–W620. https://doi.org/10.1093/NAR/GKV352
14. Zdrazil B, Felix E, Hunter F, Manners EJ, Blackshaw J, Corbett S et al (2024) The ChEMBL database in 2023: a drug discovery platform spanning multiple bioactivity data types and time periods. Nucleic Acids Res 52(D1):D1180–D1192. https://doi.org/10.1093/NAR/GKAD1004
15. Kim S, Chen J, Cheng T, Gindulyte A, He J, He S et al (2023) PubChem 2023 update. Nucleic Acids Res 51(D1):D1373–D1380. https://doi.org/10.1093/NAR/GKAC956
16. Irwin JJ, Tang KG, Young J, Dandarchuluun C, Wong BR, Khurelbaatar M et al (2020) ZINC20 - a free ultralarge-scale chemical database for ligand discovery. J Chem Inf Model 60(12):6065–6073. https://doi.org/10.1021/ACS.JCIM.0C00675
17. Tingle BI, Tang KG, Castanon M, Gutierrez JJ, Khurelbaatar M, Dandarchuluun C et al (2023) ZINC-22—a free multi-billion-scale database of tangible compounds for ligand discovery. J Chem Inf Model 63(4):1166–1176. https://doi.org/10.1021/ACS.JCIM.2C01253
18. Cheng T, Li Q, Zhou Z, Wang Y, Bryant SH (2012) Structure-based virtual screening for drug discovery: a problem-centric review. AAPS J 14(1):133–141. https://doi.org/10.1208/S12248-012-9322-0
19. Koes DR, Baumgartner MP, Camacho CJ (2013) Lessons learned in empirical scoring with smina from the CSAR 2011 benchmarking exercise. J Chem Inf Model 53(8):1893–1904. https://doi.org/10.1021/CI300604Z
20. McNutt AT, Francoeur P, Aggarwal R, Masuda T, Meli R,Ragoza M et al (2021) GNINA 1.0: molecular docking with deep learning. J Cheminform 13(1):1–20. https://doi.org/10.1186/S13321-021-00522-2
21. Hassan NM, Alhossary AA, Mu Y, Kwoh CK (2017) Protein-ligand blind docking using QuickVina-W with inter-process spatio-temporal integration. Sci Rep 7(1):1–13. https://doi.org/10.1038/s41598-017-15571-7
22. Halgren TA, Murphy RB, Friesner RA, Beard HS, Frye LL, Pollard WT et al (2004) Glide: a new approach for rapid, accurate docking and scoring. 2. Enrichment factors in database screening. J Med Chem 47(7):1750–1759. https://doi.org/10.1021/JM030644S
23. RDKit. https://www.rdkit.org/.
24. O'Boyle NM, Banck M, James CA, Morley C, Vandermeersch T, Hutchison GR (2011) Open babel: an open chemical toolbox. J Cheminform 3(10):1–14. https://doi.org/10.1186/1758-2946-3-33
25. Molecular Modeling Software | OpenEye Scientific. https://www.eyesopen.com/.
26. Tresadern G, Bemporad D, Howe T (2009) A comparison of ligand based virtual screening methods and application to corticotropin releasing factor 1 receptor. J Mol Graph Model 27(8):860–870. https://doi.org/10.1016/J.JMGM.2009.01.003
27. Vázquez J, López M, Gibert E, Herrero E, Javier LF (2020) Merging ligand-based and structure-based methods in drug discovery: an overview of combined virtual screening approaches. Molecules 25(20):4723. https://doi.org/10.3390/MOLECULES25204723
28. Cleves AE, Jain AN (2020) Structure- and ligand-based virtual screening on DUD-E+: performance dependence on approximations to the

binding pocket. J Chem Inf Model 60(9):4296–4310. https://doi.org/10.1021/ACS.JCIM.0C00115
29. Cereto-Massagué A, Ojeda MJ, Valls C, Mulero M, Garcia-Vallvé S, Pujadas G (2015) Molecular fingerprint similarity search in virtual screening. Methods 71:58–63. https://doi.org/10.1016/J.YMETH.2014.08.005
30. Rogers D, Hahn M (2010) Extended-connectivity fingerprints. J Chem Inf Model 50(5):742–754. https://doi.org/10.1021/CI100050T
31. Jaeger S, Fulle S, Turk S (2018) Mol2vec: unsupervised machine learning approach with chemical intuition. J Chem Inf Model 58(1):27–35. https://doi.org/10.1021/ACS.JCIM.7B00616
32. Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. In: 1st international conference on learning representations, ICLR 2013. arXiv. https://doi.org/10.48550/arXiv.1301.3781
33. Asgari E, Mofrad MRK (2015) Continuous distributed representation of biological sequences for deep proteomics and genomics. PLoS ONE 10(11):e0141287. https://doi.org/10.1371/JOURNAL.PONE.0141287
34. Zhang H, Liao L, Cai Y, Hu Y, Wang H (2019) IVS2vec: a tool of Inverse Virtual Screening based on word2vec and deep learning techniques. Methods 166:57–65. https://doi.org/10.1016/j.ymeth.2019.03.012
35. Wang R, Fang X, Lu Y, Wang S (2004) The PDBbind database: collection of binding affinities for protein-ligand complexes with known three-dimensional structures. J Med Chem 47(12):2977–2980. https://doi.org/10.1021/JM030580L
36. Rifaioglu AS, Nalbat E, Atalay V, Martin MJ, Cetin-Atalay R, Doğan T (2020) DEEPScreen: high performance drug–target interaction prediction with convolutional neural networks using 2-D structural compound representations. Chem Sci 11(9):2531–2557. https://doi.org/10.1039/C9SC03414E
37. Liu Z, Du J, Fang J, Yin Y, Xu G, Xie L (2019) DeepScreening: a deep learning-based screening web server for accelerating drug discovery. Database. https://doi.org/10.1093/DATABASE/BAZ104
38. Yap CW (2011) PaDEL-descriptor: an open source software to calculate molecular descriptors and fingerprints. J Comput Chem 32(7):1466–1474. https://doi.org/10.1002/JCC.21707
39. Joshi T, Joshi T, Pundir H, Sharma P, Mathpal S, Chandra S (2020) Predictive modeling by deep learning, virtual screening and molecular dynamics study of natural compounds against SARS-CoV-2 main protease. J Biomol Struct Dyn 39(17):6728–6746. https://doi.org/10.1080/07391102.2020.1802341
40. Joshi T, Sharma P, Mathpal S, Joshi T, Maiti P, Nand M et al (2022) Computational investigation of drug bank compounds against 3C-like protease (3CLpro) of SARS-CoV-2 using deep learning and molecular dynamics simulation. Mol Divers 26(4):2243–2256. https://doi.org/10.1007/S11030-021-10330-3
41. Compound Libraries for High Throughput/Content Screening | 96-Well. https://www.selleckchem.com/screening/natural-product-library.html.
42. Joshi T, Pundir H, Chandra S (2022) Deep-learning based repurposing of FDA-approved drugs against Candida albicans dihydrofolate reductase and molecular dynamics study. J Biomol Struct Dyn 40(18):8420–8436. https://doi.org/10.1080/07391102.2021.1911851
43. Zhang H, Yang Y, Li J, Wang M, Saravanan KM, Wei J et al (2020) A novel virtual screening procedure identifies Pralatrexate as inhibitor of SARS-CoV-2 RdRp and it reduces viral replication in vitro. PLoS Comput Biol 16(12):e1008489. https://doi.org/10.1371/JOURNAL.PCBI.1008489
44. Compound Libraries | Inhibitors | Virtual Screening - TargetMol. https://www.targetmol.com/index.
45. Zhang H, Li J, Saravanan KM, Wu H, Wang Z, Wu D et al (2021) An integrated deep learning and molecular dynamics simulation-based screening pipeline identifies inhibitors of a new cancer drug target TIPE2. Front Pharmacol 12:772296. https://doi.org/10.3389/FPHAR.2021.772296
46. Bian Y, Xie XQ (2021) Generative chemistry: drug discovery with deep learning generative models. J Mol Model 27(3):1–18. https://doi.org/10.1007/S00894-021-04674-8
47. Andrianov AM, Nikolaev GI, Shuldov NA, Bosko IP, Anischenko AI, Tuzikov AV (2022) Application of deep learning and molecular modeling to identify small drug-like compounds as potential HIV-1 entry inhibitors. J Biomol Struct Dyn 40(16):7555–7573. https://doi.org/10.1080/07391102.2021.1905559
48. Durrant JD, McCammon JA (2012) AutoClickChem: click chemistry in silico. PLoS Comput Biol 8(3):e1002397. https://doi.org/10.1371/JOURNAL.PCBI.1002397
49. Maziarka Ł, Pocha A, Kaczmarczyk J, Rataj K, Danel T, Warchoł M (2020) Mol-CycleGAN: a generative model for molecular optimization. J Cheminform 12(1):1–18. https://doi.org/10.1186/S13321-019-0404-1
50. Méndez-Lucio O, Baillif B, Clevert DA, Rouquié D, Wichard J (2020) De novo generation of hit-like molecules from gene expression signatures using artificial intelligence. Nat Commun 11:10. https://doi.org/10.1038/s41467-019-13807-w
51. Prykhodko O, Johansson SV, Kotsias PC, Arús-Pous J, Bjerrum EJ, Engkvist O et al (2019) A de novo molecular generation method using latent vector based generative adversarial network. J Cheminform 11:74. https://doi.org/10.1186/S13321-019-0397-9
52. Sanchez-Lengeling B, Outeiral C, Guimaraes GL, Aspuru-Guzik A (2017) Optimizing distributions over molecular space. An objective-reinforced generative adversarial network for inverse-design chemistry (ORGANIC). ChemRxiv. https://doi.org/10.26434/CHEMRXIV.5309668.V3
53. Arshia AH, Shadravan S, Solhjoo A, Sakhteman A, Sami A (2021) De novo design of novel protease inhibitor candidates in the treatment of SARS-CoV-2 using deep learning, docking, and molecular dynamic simulations. Comput Biol Med 139:104967. https://doi.org/10.1016/J.COMPBIOMED.2021.104967
54. Moret M, Friedrich L, Grisoni F, Merk D, Schneider G (2020) Generative molecular design in low data regimes. Nat Mach Intell 2:171–180. https://doi.org/10.1038/S42256-020-0160-Y
55. Gupta A, Müller AT, Huisman BJH, Fuchs JA, Schneider P, Schneider G (2018) Generative recurrent networks for de novo drug design. Mol Inform 37(1–2):1700111. https://doi.org/10.1002/MINF.201700111
56. Segler MHS, Kogej T, Tyrchan C, Waller MP (2018) Generating focused molecule libraries for drug discovery with recurrent neural networks. ACS Cent Sci 4(1):120–131. https://doi.org/10.1021/ACSCENTSCI.7B00512
57. Merk D, Friedrich L, Grisoni F, Schneider G (2018) De novo design of bioactive small molecules by artificial intelligence. Mol Inform 37(1–2):1700153. https://doi.org/10.1002/MINF.201700153
58. Zheng S, Yan X, Gu Q, Yang Y, Du Y, Lu Y et al (2019) QBMG: Quasi-biogenic molecule generator with deep recurrent neural network. J Cheminform 11:5. https://doi.org/10.1186/S13321-019-0328-9
59. Olivecrona M, Blaschke T, Engkvist O, Chen H (2017) Molecular de-novo design through deep reinforcement learning. J Cheminform 9:48. https://doi.org/10.1186/S13321-017-0235-X
60. Arús-Pous J, Blaschke T, Ulander S, Reymond JL, Chen H, Engkvist O (2019) Exploring the GDB-13 chemical space using deep generative models. J Cheminform 11:20. https://doi.org/10.1186/S13321-019-0341-Z
61. Arús-Pous J, Johansson SV, Prykhodko O, Bjerrum EJ, Tyrchan C, Reymond JL et al (2019) Randomized SMILES strings improve the quality of molecular generative models. J Cheminform 11:71. https://doi.org/10.1186/S13321-019-0393-0
62. Ertl P, Lewis R, Martin E, Polyakov V (2018) In silico generation of novel, drug-like chemical matter using the LSTM neural network. ArXiv. https://doi.org/10.48550/arXiv.1712.07449
63. Sun Y, Jiao Y, Shi C, Zhang Y (2022) Deep learning-based molecular dynamics simulation for structure-based drug design against SARS-CoV-2. Comput Struct Biotechnol J 20:5014–5027. https://doi.org/10.1016/J.CSBJ.2022.09.002
64. Das P, Sercu T, Wadhawan K, Padhi I, Gehrmann S, Cipcigan F et al (2021) Accelerated antimicrobial discovery via deep generative models and molecular dynamics simulations. Nat Biomed Eng 5:613–623. https://doi.org/10.1038/s41551-021-00689-x
65. Tolstikhin I, Bousquet O, Gelly S, Schölkopf B (2019) Wasserstein auto-encoders. In: 6th International Conference on Learning Representations, ICLR 2018. arXiv. https://doi.org/10.48550/arXiv.1711.01558
66. Blaschke T, Olivecrona M, Engkvist O, Bajorath J, Chen H (2018) Application of generative autoencoder in de novo molecular design. Mol Inform 37(1–2):1700123. https://doi.org/10.1002/MINF.201700123

67. Gómez-Bombarelli R, Wei JN, Duvenaud D, Hernández-Lobato JM, Sánchez-Lengeling B, Sheberla D et al (2018) Automatic chemical design using a data-driven continuousrepresentation of molecules. ACS Cent Sci 4(2):268–276. https://doi.org/10.1021/ACSCENTSCI.7B00572
68. Sattarov B, Baskin II, Horvath D, Marcou G, Bjerrum EJ, Varnek A (2019) De novo molecular design by combining deep autoencoder recurrent neural networks with generative topographic mapping. J Chem Inf Model 59(3):1182–1196. https://doi.org/10.1021/ACS.JCIM.8B00751
69. Mohammadi S, O'Dowd B, Paulitz-Erdmann C, Goerlitz L (2019) Penalized variational autoencoder for molecular design. ChemRxiv. https://doi.org/10.26434/CHEMRXIV.7977131.V2
70. Samanta B, De A, Jana G, Chattaraj KP, Ganguly N, Gomez-Rodriguez M (2019) NEVAE: a deep generative model for molecular graphs. Proc AAAI Conf Artif Intell 33(1):1110–1117. https://doi.org/10.1609/aaai.v33i01.33011110
71. Simonovsky M, Komodakis N. (2018) GraphVAE: towards generation of small graphs using variational autoencoders. In: 27th international conference on artificial neural networks (ICANN 2018) - Lecture notes in computer science, 11139, pp 412–422. https://doi.org/10.1007/978-3-030-01418-6_41
72. Imrie F, Bradley AR, Van Der Schaar M, Deane CM (2020) Deep Generative Models for 3D Linker Design. J Chem Inf Model 60(4):1983–1995. https://doi.org/10.1021/ACS.JCIM.9B01120
73. Mercado R, Rastemo T, Lindelöf E, Klambauer G, Engkvist O, Chen H et al (2020) Practical notes on building molecular graph generative models. Appl AI Lett 1(2). https://doi.org/10.1002/ail2.18
74. Xiong J, Xiong Z, Chen K, Jiang H, Zheng M (2021) Graph neural networks for automated de novo drug design. Drug Discov Today 26(6):1382–1393. https://doi.org/10.1016/J.DRUDIS.2021.02.011
75. Chen H, Engkvist O, Wang Y, Olivecrona M, Blaschke T (2018) The rise of deep learning in drug discovery. Drug Discov Today 23(6):1241–1250. https://doi.org/10.1016/J.DRUDIS.2018.01.039
76. Lavecchia A (2019) Deep learning in drug discovery: opportunities, challenges and future prospects. Drug Discov Today 24(10):2017–2032. https://doi.org/10.1016/J.DRUDIS.2019.07.006
77. Ballester PJ, Mitchell JBO (2010) A machine learning approach to predicting protein-ligand binding affinity with applications to molecular docking. Bioinformatics 26(9):1169–1175. https://doi.org/10.1093/BIOINFORMATICS/BTQ112
78. Durrant JD, McCammon JA (2010) NNScore: a neural-network-based scoring function for the characterization of protein-ligand complexes. J Chem Inf Model 50(10):1865–1871. https://doi.org/10.1021/CI100244V
79. Stepniewska-Dziubinska MM, Zielenkiewicz P, Siedlecki P (2018) Development and evaluation of a deep learning model for protein–ligand binding affinity prediction. Bioinformatics 34(21):3666–3674. https://doi.org/10.1093/BIOINFORMATICS/BTY374
80. Li H, Sze KH, Lu G, Ballester PJ (2021) Machine-learning scoring functions for structure-based virtual screening. Wiley Interdiscip Rev Comput Mol Sci 11(1):e1478. https://doi.org/10.1002/WCMS.1478
81. Zhang H, Liao L, Saravanan KM, Yin P, Wei Y (2019) DeepBindRG: A deep learning based method for estimating effective protein-ligand affinity. PeerJ 7:e7362. https://doi.org/10.7717/PEERJ.7362
82. Hartshorn MJ, Verdonk ML, Chessari G, Brewerton SC, Mooij WTM, Mortenson PN et al (2007) Diverse, high-quality test set for the validation of protein-ligand docking performance. J Med Chem 50(4):726–741. https://doi.org/10.1021/JM061277Y
83. Cang Z, Wei G (2017) TopologyNet: topology based deep convolutional and multi-task neural networks for biomolecular property predictions. PLoS Comput Biol 13(7):e1005690. https://doi.org/10.1371/JOURNAL.PCBI.1005690ISBN:1111111111
84. Gomes J, Ramsundar B, Feinberg EN, Pande VS (2017) Atomic convolutional networks for predicting protein-ligand binding affinity. ArXiv. https://doi.org/10.48550/arXiv.1703.10603
85. Jiménez J, Škalič M, Martínez-Rosell G, De Fabritiis G (2018) KDEEP: protein-ligand absolute binding affinity prediction via 3D-convolutional neural networks. J Chem Inf Model 58(2):287–296. https://doi.org/10.1021/ACS.JCIM.7B00650
86. Li Y, Rezaei MA, Li C, Li X. (2019) DeepAtom: a framework for protein-ligand binding affinity prediction. In: Proceedings - 2019 IEEE international conference on bioinformatics and biomedicine, BIBM 2019, pp 303–310. https://doi.org/10.1109/BIBM47256.2019.8982964
87. Zheng L, Fan J, Mu Y (2019) OnionNet: a multiple-layer intermolecular-contact-based convolutional neural network for protein-ligand binding affinity prediction. ACS Omega 4(14):15956–15965. https://doi.org/10.1021/ACSOMEGA.9B01997
88. Hassan-Harrirou H, Zhang C, Lemmin T (2020) RosENet: improving binding affinity prediction by leveraging molecular mechanics energies with an ensemble of 3D convolutional neural networks. J Chem Inf Model 60(6):2791–2802. https://doi.org/10.1021/ACS.JCIM.0C00075
89. Kwon Y, Shin WH, Ko J, Lee J (2020) AK-Score: accurate protein-ligand binding affinity prediction using an ensemble of 3D-convolutional neural networks. Int J Mol Sci 21(22):8424. https://doi.org/10.3390/IJMS21228424
90. Wang S, Liu D, Ding M, Du Z, Zhong Y, Song T et al (2021) SE-OnionNet: a convolution neural network for protein-ligand binding affinity prediction. Front Genet 11:607824. https://doi.org/10.3389/FGENE.2020.607824
91. Xie L, Xu L, Chang S, Xu X, Meng L (2020) Multitask deep networks with grid featurization achieve improved scoring performance for protein–ligand binding. Chem Biol Drug Des 96(3):973–983. https://doi.org/10.1111/CBDD.13648
92. Zhu F, Zhang X, Allen JE, Jones D, Lightstone FC (2020) Binding affinity prediction by pairwise function based on neural network. J Chem Inf Model 60(6):2766–2772. https://doi.org/10.1021/ACS.JCIM.0C00026
93. Ahmed A, Mam B, Sowdhamini R (2021) DEELIG: a deep learning approach to predict protein-ligand binding affinity. Bioinform Biol Insights 15. https://doi.org/10.1177/11779322211030364
94. Jiang D, Hsieh CY, Wu Z, Kang Y, Wang J, Wang E et al (2021) InteractionGraphNet: a novel and efficient deep graph representation learning framework for accurate protein-ligand interaction predictions. J Med Chem 64(24):18209–18232. https://doi.org/10.1021/ACS.JMEDCHEM.1C01830
95. Kumar S, Hyun KM (2021) SMPLIP-score: predicting ligand binding affinity from simple and interpretable on-the-fly interaction fingerprint pattern descriptors. J Cheminform 13(1):1–17.https://doi.org/10.1186/S13321-021-00507-1
96. Liu Q, Wang PS, Zhu C, Gaines BB, Zhu T, Bi J et al (2021) OctSurf: efficient hierarchical voxel-based molecular surface representation for protein-ligand affinity prediction. J Mol Graph Model 105:107865. https://doi.org/10.1016/J.JMGM.2021.107865
97. Seo S, Choi J, Park S, Ahn J (2021) Binding affinity prediction for protein–ligand complex using deep attention mechanism based on intermolecular interactions. BMC Bioinform 22(1):1–15. https://doi.org/10.1186/S12859-021-04466-0
98. Son J, Kim D (2021) Development of a graph convolutional neural network model for efficient prediction of protein-ligand binding affinities. PLoS ONE 16(4):e0249404. https://doi.org/10.1371/JOURNAL.PONE.0249404
99. Shen H, Zhang Y, Zheng C, Wang B, Chen P (2021) A cascade graph convolutional network for predicting protein–ligand binding affinity. Int J Mol Sci 22(8):4023. https://doi.org/10.3390/IJMS22084023/S1
100. Yang J, Shen C, Huang N (2020) Predicting or pretending: artificial intelligence for protein-ligand interactions lack of sufficiently large and unbiased datasets. Front Pharmacol 11:508760. https://doi.org/10.3389/FPHAR.2020.00069
101. Jones D, Kim H, Zhang X, Zemla A, Stevenson G, Bennett WFD et al (2021) Improved protein-ligand binding affinity prediction with structure-based deep fusion inference. J Chem Inf Model 61(4):1583–1592. https://doi.org/10.1021/ACS.JCIM.0C01306
102. Karlov DS, Sosnin S, Fedorov MV, Popov P (2020) GraphDelta: MPNN scoring function for the affinity prediction of protein-ligand complexes. ACS Omega 5(10):5150–5159. https://doi.org/10.1021/ACSOMEGA.9B04162
103. Feinberg EN, Sur D, Wu Z, Husic BE, Mai H, Li Y et al (2018) PotentialNet for molecular property prediction. ACS Cent Sci 4(11):1520–1530. https://doi.org/10.1021/ACSCENTSCI.8B00507
104. Valsson Í, Warren MT, Deane CM, Magarkar A, Morris GM, Biggin PC (2025) Narrowing the gap between machine learning scoring functions and free energy perturbation using augmented

data. Commun Chem 8(1):1–12. https://doi.org/10.1038/s42004-025-01428-y
105. Su M, Yang Q, Du Y, Feng G, Liu Z, Li Y et al (2019) Comparative assessment of scoring functions: the CASF-2016 update. J Chem Inf Model 59(2):895–913. https://doi.org/10.1021/ACS.JCIM.8B00545
106. Stärk H, Ganea OE, Pattanaik L, Barzilay R, Jaakkola T (2022) EquiBind: geometric deep learning for drug binding structure prediction. In: Proceedings of the 39th international conference on machine learning (PMLR 2022), 162, pp 20503–20521. arXiv. https://doi.org/10.48550/arXiv.2202.05146
107. Lu W, Technologies G, Wu Q, Zhang J, Rao J, Li C et al (2022) TANKBind: trigonometry-aware neural networks for drug-protein binding structure prediction. In: 36th conference on neural information processing systems (NeurIPS 2022), 35, pp 7236–7249. https://doi.org/10.1101/2022.06.06.495043
108. Lim J, Ryu S, Park K, Choe YJ, Ham J, Kim WY (2019) Predicting drug-target interaction using a novel graph neural network with 3D structure-embedded graph representation. J Chem Inf Model 59(9):3981–3988. https://doi.org/10.1021/ACS.JCIM.9B00387
109. Torng W, Altman RB (2019) Graph convolutional neural networks for predicting drug-target interactions. J Chem Inf Model 59(10):4131–4149. https://doi.org/10.1021/ACS.JCIM.9B00628
110. Corso G, Stärk H, Jing B, Barzilay R, Jaakkola T (2023) DiffDock: diffusion steps, twists, and turns for molecular docking. In: 11th international conference on learning representations (ICLR 2023). arXiv. https://doi.org/10.48550/arXiv.2210.01776
111. Cao H, Tan C, Gao Z, Xu Y, Chen G, Heng PA et al (2024) A survey on generative diffusion models. IEEE Trans Knowl Data Eng 36(7):2814–2830. https://doi.org/10.1109/TKDE.2024.3361474
112. Abramson J, Adler J, Dunger J, Evans R, Green T, Pritzel A et al (2024) Accurate structure prediction of biomolecular interactions with AlphaFold 3. Nature. https://doi.org/10.1038/s41586-024-07487-w
113. Jumper J, Evans R, Pritzel A, Green T, Figurnov M, Ronneberger O et al (2021) Highly accurate protein structure prediction with AlphaFold. Nature 596:583–589. https://doi.org/10.1038/s41586-021-03819-2
114. Buttenschoen M, Morris GM, Deane CM (2024) PoseBusters: AI-based docking methods fail to generate physically valid poses or generalise to novel sequences. Chem Sci 15(9):3130–3139.https://doi.org/10.1039/D3SC04185A
115. Volkov M, Turk JA, Drizard N, Martin N, Hoffmann B, Gaston-Mathé Y et al (2022) On the frustration to predict binding affinities from protein-ligand structures with deep neural networks. J Med Chem 65(11):7946–7958. https://doi.org/10.1021/ACS.JMEDCHEM.2C00487
116. Chen LI, Cruz AI, Ramsey S, Dickson CJ, Duca JS, Hornak V et al (2019) Hidden bias in the DUD-E dataset leads to misleading performance of deep learning in structure-based virtual screening. PLoS ONE 14(8):e0220113. https://doi.org/10.1371/journal.pone.0220113
117. Sieg J, Flachsenberg F, Rarey M (2019) In need of bias control: evaluating chemical data for machine learning in structure-based virtual screening. J Chem Inf Model 59:947–961. https://doi.org/10.1021/acs.jcim.8b00712
118. Thomas M, Bender A, de Graaf C (2023) Integrating structure-based approaches in generative molecular design. Curr Opin Struct Biol 79:102559. https://doi.org/10.1016/J.SBI.2023.102559
119. Guan J, Qian WW, Peng X, Su Y, Peng J, Ma J (2023) 3D equivariant diffusion for target-aware molecule generation and affinity prediction. Iin: 11th international conference on learning representations, ICLR 2023. https://doi.org/10.48550/arXiv.2303.03543
120. Cremer J, Le T, Noé F, Clevert DA, Schütt KT (2024) PILOT: equivariant diffusion for pocket-conditioned de novo ligand generation with multi-objective guidance via importance sampling. Chem Sci 15(36):14954–14967. https://doi.org/10.1039/D4SC03523B
121. Peng X, Luo S, Guan J, Xie Q, Peng J, Ma J (2022) Pocket2Mol: efficient molecular sampling based on 3D protein pockets. In: Proceedings of the 39th international conference on machine learning, PMLR (162), pp 17644–17655. https://doi.org/10.48550/arXiv.2205.07249
122. Shen T, Seo S, Lee G, Pandey M, Smith JR, Cherkasov A et al (2023) TacoGFN: target-conditioned GFlowNet for structure-based drug design. Gener AI Biol. https://doi.org/10.48550/arXiv.2310.03223
123. Feng W, Wang L, Lin Z, Zhu Y, Wang H, Dong J et al (2024) Generation of 3D molecules in pockets via a language model. Nat Mach Intell 6(1):62–73. https://doi.org/10.1038/s42256-023-00775-6
124. Powers AS, Yu HH, Suriana P, Rohan, Koodli V, Lu T et al (2023) Geometric deep learning for structure-based ligand design. ACS Cent Sci 9(12):2257–2267. https://doi.org/10.1021/ACSCENTSCI.3C00572
125. Thomas M, O'Boyle NM, Bender A, de Graaf C (2022) Augmented Hill-Climb increases reinforcement learning efficiency for language-based de novo molecule generation. J Cheminform 14(1):1–22. https://doi.org/10.1186/S13321-022-00646-Z
126. Spiegel JO, Durrant JD (2020) AutoGrow4: an open-source genetic algorithm for de novo drug design and lead optimization. J Cheminform 12(1):1–16. https://doi.org/10.1186/S13321-020-00429-4
127. Fu T, Gao W, Coley CW, Sun J (2022) Reinforced genetic algorithm for structure-based drug design. Adv Neural Inf Process Syst 35:12325–12338
128. Baillif B, Cole J, McCabe P, Bender A (2024) Benchmarking structure-based three-dimensional molecular generative models using GenBench3D: ligand conformation quality matters. ArXiv. https://doi.org/10.48550/arXiv.2407.04424
129. Jocys Z, Grundy J, Farrahi K (2024) DrugPose: benchmarking 3D generative methods for early stage drug discovery. Digital Discovery 3(7):1308–1318. https://doi.org/10.1039/D4DD00076E
130. Lin H, Zhao G, Zhang O, Huang Y, Wu L, Liu Z et al (2024) CBGBench: fill in the blank of protein-molecule complex binding graph. ArXiv. https://doi.org/10.48550/arXiv.2406.10840
131. Liu H, Qin Y, Niu Z, Xu M, Wu J, Xiao X et al (2024) How good are current pocket-based 3D generative models?: The benchmark set and evaluation of protein pocket-based 3D molecular generative models. J Chem Inf Model 10:37. https://doi.org/10.1021/ACS.JCIM.4C01598
132. Thomas M, O'Boyle NM, Bender A, De Graaf C (2024) MolScore: a scoring, evaluation and benchmarking framework for generative models in de novo drug design. J Cheminform 16(64):1–20. https://doi.org/10.1186/s13321-024-00861-w
133. Zheng K, Lu Y, Zhang Z, Wan Z, Ma Y, Zitnik M et al (2024) Structure-based drug designbenchmark: do 3D methods really dominate? ArXiv, pp 1–31. https://doi.org/10.48550/arXiv.2406.03403
134. Páll S, Zhmurov A, Bauer P, Abraham M, Lundborg M, Gray A et al (2020) Heterogeneous parallelization and acceleration of molecular dynamics simulations in GROMACS. J Chem Phys 153(13):134110. https://doi.org/10.1063/5.0018516/199476
135. Case DA, Aktulga HM, Belfon K, Cerutti DS, Cisneros GA, Cruzeiro VWD et al (2023) AmberTools. J Chem Inf Model 63(20):6183–6191. https://doi.org/10.1021/acs.jcim.3c01153
136. Phillips JC, Hardy DJ, Maia JDC, Stone JE, Ribeiro JV, Bernardi RC et al (2020) Scalable molecular dynamics on CPU and GPU architectures with NAMD. J Chem Phys 153(4):044130. https://doi.org/10.1063/5.0014475
137. Degiacomi MT (2019) Coupling molecular dynamics and deep learning to mine protein conformational space. Structure 27(6):1034-1040.e3. https://doi.org/10.1016/J.STR.2019.03.018
138. Noé F (2018) Machine learning for molecular dynamics on long timescales. ArXiv. https://doi.org/10.48550/arXiv.1812.07669
139. Ma H, Bhowmik D, Lee H, Turilli M, Young MT, Jha S et al (2020) Deep generative model driven protein folding simulations. In: Foster I, Joubert GR, Kucera L, Nagel WE, Peters F (eds) Parallel computing: technology trends (Advances in parallel computing; vol 36). IOS Press BV, Amsterdam, pp 45–55. https://doi.org/10.48550/arXiv.1908.00496
140. Lee H, Ma H, Turilli M, Bhowmik D, Jha S, Ramanathan A et al (2019) DeepDriveMD: deep-learning driven adaptive molecular simulations for protein folding. ArXiv. https://doi.org/10.48550/arXiv.1909.07817
141. Sultan MM, Wayment-Steele HK, Pande VS (2018) Transferable neural networks for enhanced sampling of protein dynamics. J Chem Theory Comput 14(4):1887–1894. https://doi.org/10.1021/acs.jctc.8b00025
142. Bhowmik D, Gao S, Young MT, Ramanathan A(2018) Deep clustering of protein folding simulations. BMC Bioinform 19:47–58. https://doi.org/10.1186/S12859-018-2507-5
143. Hernández CX, Wayment-Steele HK, Sultan MM, Husic BE, Pande VS (2018) Variational encoding of complex dynamics. Phys Rev E 97:062412. https://doi.org/10.1103/PhysRevE.97.062412

144. Schultze S, Grubmüller H (2021) Time-lagged independent component analysis of random walks and protein dynamics. J Chem Theory Comput 17(9):5766–5776. https://doi.org/10.1021/ACS.JCTC.1C00273
145. Chen W, Tan AR, Ferguson AL (2018) Collective variable discovery and enhanced sampling using autoencoders: Innovations in network architecture and error function design. J Chem Phys 149(7):072312. https://doi.org/10.1063/1.5023804
146. Chiavazzo E, Covino R, Coifman RR, Gear CW, Georgiou AS, Hummer G et al (2017) Intrinsic map dynamics exploration for uncharted effective free-energy landscapes. Proc Natl Acad Sci U S A 114(28):E5494–E5503. https://doi.org/10.1073/pnas.1621481114
147. Mardt A, Pasquali L, Wu H, Noé F (2018) VAMPnets for deep learning of molecular kinetics. Nat Commun 9:5. https://doi.org/10.1038/s41467-017-02388-1
148. Ribeiro JML, Bravo P, Wang Y, Tiwary P (2018) Reweighted autoencoded variational Bayes for enhanced sampling (RAVE). J Chem Phys 149(7):072301. https://doi.org/10.1063/1.5025487
149. Chen W, Ferguson AL (2018) Molecular enhanced sampling with autoencoders: on-the-fly collective variable discovery and accelerated free energy landscape exploration. J Comput Chem 39(25):2079–2102. https://doi.org/10.1002/JCC.25520
150. Wu H, Mardt A, Pasquali L, Noe F (2019) Deep generative Markov state models. In: 32nd conference on neural information processing systems (NeurIPS 2018). arXiv. https://doi.org/10.48550/arXiv.1805.07601
151. Sun L, Vandermause J, Batzner S, Xie Y, Clark D, Chen W et al (2022) Multitask machine learning of collective variables for enhanced sampling of rare events. J Chem Theory Comput 18(4):2341–2353. https://doi.org/10.1021/acs.jctc.1c00143
152. Smith JS, Isayev O, Roitberg AE (2017) ANI-1: an extensible neural network potential with DFT accuracy at force field computational cost. Chem Sci 8(4):3192–3203. https://doi.org/10.1039/C6SC05720A
153. Behler J (2021) Four generations of high-dimensional neural network potentials. Chem Rev 121(16):10037–10072. https://doi.org/10.1021/ACS.CHEMREV.0C00868
154. Behler J, Parrinello M (2007) Generalized neural-network representation of high-dimensional potential-energy surfaces. Phys Rev Lett 98(14):146401. https://doi.org/10.1103/PHYSREVLETT.98.146401
155. Behler J (2011) Atom-centered symmetry functions for constructing high-dimensional neural network potentials. J Chem Phys 134(7):074106. https://doi.org/10.1063/1.3553717
156. Artrith N, Morawietz T, Behler J (2011) High-dimensional neural-network potentials for multicomponent systems: applications to zinc oxide. Phys Rev B 83(15):153101. https://doi.org/10.1103/PHYSREVB.83.153101
157. Morawietz T, Sharma V, Behler J (2012) A neural network potential-energy surface for the water dimer based on environment-dependent atomic energies and charges. J Chem Phys 136(6):064103. https://doi.org/10.1063/1.3682557
158. Yao K, Herr JE, Toth DW, McKintyre R, Parkhill J (2018) The TensorMol-0.1 model chemistry: a neural network augmented with long-range physics. Chem Sci 9(8):2261–2269. https://doi.org/10.1039/C7SC04934J
159. Ko TW, Finkler JA, Goedecker S, Behler J (2021) A fourth-generation high-dimensional neural network potential with accurate electrostatics including non-local charge transfer. Nat Commun 12:398. https://doi.org/10.1038/s41467-020-20427-2
160. Faraji S, Ghasemi SA, Rostami S, Rasoulkhani R, Schaefer B, Goedecker S et al (2017) High accuracy and transferability of a neural network potential through charge equilibration for calcium fluoride. Phys Rev B 95(10):104105. https://doi.org/10.1103/PHYSREVB.95.104105
161. Behler J (2015) Constructing high-dimensional neural network potentials: a tutorial review. Int J Quantum Chem 115(16):1032–1050. https://doi.org/10.1002/QUA.24890
162. Schran C, Brezina K, Marsalek O (2020) Committee neural network potentials control generalization errors and enable active learning. J Chem Phys 153(10):104105. https://doi.org/10.1063/5.0016004
163. Jose KVJ, Artrith N, Behler J (2012) Construction of high-dimensional neural network potentials using environment-dependent atom pairs. J Chem Phys 136(19):194111. https://doi.org/10.1063/1.4712397
164. Gastegger M, Behler J, Marquetand P (2017) Machine learning molecular dynamics for the simulation of infrared spectra. Chem Sci 8(10):6924–6935. https://doi.org/10.1039/C7SC02267K
165. Gastegger M, Kauffmann C, Behler J, Marquetand P (2016) Comparing the accuracy of high-dimensional neural network potentials and the systematic molecular fragmentation method: a benchmark study for all-trans alkanes. J Chem Phys 144(19):194110. https://doi.org/10.1063/1.4950815
166. Gastegger M, Marquetand P (2015) High-dimensional neural network potentials for organic reactions and an improved training algorithm. J Chem Theory Comput 11(5):2187–2198. https://doi.org/10.1021/ACS.JCTC.5B00211
167. Gastegger M, Schwiedrzik L, Bittermann M, Berzsenyi F, Marquetand P (2018) WACSF - weighted atom-centered symmetry functions as descriptors in machine learning potentials. J Chem Phys 148(24):241709. https://doi.org/10.1063/1.5019667
168. Schran C, Behler J, Marx D (2020) Automated fitting of neural network potentials at coupled cluster accuracy: protonated water clusters as testing ground. J Chem Theory Comput 16(1):88–99. https://doi.org/10.1021/ACS.JCTC.9B00805
169. Litman Y, Behler J, Rossi M (2020) Temperature dependence of the vibrational spectrum of porphycene: a qualitative failure of classical-nuclei molecular dynamics. Faraday Discuss 221:526–546. https://doi.org/10.1039/C9FD00056A
170. Topolnicki R, Brieuc F, Schran C, Marx D (2020) Deciphering high-order structural correlations within fluxional molecules from classical and quantum configurational entropy. J Chem Theory Comput 16(11):6785–6794. https://doi.org/10.1021/ACS.JCTC.0C00642
171. Devereux C, Smith JS, Davis KK, Barros K, Zubatyuk R, Isayev O et al (2020) Extending the applicability of the ANI deep learning molecular potential to sulfur and halogens. J Chem Theory Comput 16(7):4192–4202. https://doi.org/10.1021/acs.jctc.0c00121
172. Galvelis R, Varela-Rial A, Doerr S, Fino R, Eastman P, Markland TE et al (2023) NNP/MM: accelerating molecular dynamics simulations with machine learning potentials and molecular mechanics. J Chem Inf Model 63(18):5701–5708. https://doi.org/10.1021/ACS.JCIM.3C00773
173. Zariquiey FS, Galvelis R, Gallicchio E, Chodera JD, Markland TE, de Fabritiis G (2024) Enhancing protein-ligand binding affinity predictions using neural network potentials. J Chem Inf Model 64(5):1481–1485. https://doi.org/10.1021/acs.jcim.3c02031
174. Plante A, Shore DM, Morra G, Khelashvili G, Weinstein H (2019) A machine learning approach for the discovery of ligand-specific functional mechanisms of GPCRs. Molecules 24(11):2097. https://doi.org/10.3390/MOLECULES24112097
175. DenseNet. https://keras.io/api/applications/densenet.
176. Simonyan K, Vedaldi A, Zisserman A (2014) Deep inside convolutional networks: visualising image classification models and saliency maps. In: 2nd international conference on learning representations (ICLR 2014). arXiv. https://doi.org/10.48550/arXiv.1312.6034
177. Plante A, Weinstein H (2021) Ligand-dependent conformational transitions in molecular dynamics trajectories of GPCRs revealed by a new machine learning rare event detection protocol. Molecules 26(10):3059. https://doi.org/10.3390/MOLECULES26103059
178. Lee DD, Seung HS (1999) Learning the parts of objects by non-negative matrix factorization. Nature 401(6755):788–791. https://doi.org/10.1038/44565
179. Do HN, Wang J, Bhattarai A, Miao Y (2022) GLOW: a workflow integrating gaussian-accelerated molecular dynamics and deep learning for free energy profiling. J Chem Theory Comput 18(3):1423–1436. https://doi.org/10.1021/acs.jctc.1c01055
180. McGibbon RT, Beauchamp KA, Harrigan MP, Klein C, Swails JM, Hernández CX et al (2015) MDTraj: a modern open library for the analysis of molecular dynamics trajectories. Biophys J 109(8):1528–1532. https://doi.org/10.1016/J.BPJ.2015.08.015
181. Miao Y, Sinko W, Pierce L, Bucher D, Walker RC, McCammon JA (2014) Improved reweighting of accelerated molecular dynamics simulations for free energy calculation. J Chem Theory Comput 10(7):2677–2689. https://doi.org/10.1021/CT500090Q
182. Miao Y, Feher VA, McCammon JA (2015) Gaussian accelerated molecular dynamics: unconstrained enhanced sampling and free energy

calculation. J Chem Theory Comput 11(8):3584–3595. https://doi.org/10.1021/ACS.JCTC.5B00436
183. Tam B, Qin Z, Zhao B, Wang SM, Lei CL (2023) Integration of deep learning with Ramachandran plot molecular dynamics simulation for genetic variant classification. iScience 26(3):106122. https://doi.org/10.1016/J.ISCI.2023.106122
184. Tam B, Sinha S, Wang SM (2020) Combining Ramachandran plot and molecular dynamics simulation for structural-based variant classification: Using TP53 variants as model. Comput Struct Biotechnol J 18:4033–4039. https://doi.org/10.1016/J.CSBJ.2020.11.041
185. Elia Venanzi NA, Basciu A, Vargiu AV, Kiparissides A, Dalby PA, Dikicioglu D (2024) Machine learning integrating protein structure, sequence, and dynamics to predict the enzyme activity of bovine enterokinase variants. J Chem Inf Model 64(7):2681–2694. https://doi.org/10.1021/ACS.JCIM.3C00999
186. Fleetwood O, Kasimova MA, Westerlund AM, Delemotte L (2020) Molecular insights from conformational ensembles via machine learning. Biophys J 118(3):765–780. https://doi.org/10.1016/J.BPJ.2019.12.016
187. Jiménez-Luna J, Grisoni F, Schneider G (2020) Drug discovery with explainable artificial intelligence. Nat Mach Intell 2(10):573–584. https://doi.org/10.1038/s42256-020-00236-4
188. Markelle K, Longjohn R, Nottingham K. The UCI Machine Learning Repository. https://archive.ics.uci.edu/.
189. Kaggle. https://www.kaggle.com/.
190. Google Dataset Search. https://datasetsearch.research.google.com/.
191. Landrum GA, Riniker S (2024) Combining IC50 or Ki values from different sources is a source of significant noise. J Chem Inf Model 64(5):1560–1567. https://doi.org/10.1021/ACS.JCIM.4C00049
192. Johnson JM, Khoshgoftaar TM (2019) Survey on deep learning with class imbalance. J Big Data 6(1):1–54. https://doi.org/10.1186/S40537-019-0192-5
193. Ozsahin DU, Taiwo Mustapha M, Mubarak AS, Said Ameen Z, Uzun B (2022) Impact of feature scaling on machine learning models for the diagnosis of diabetes. In: 2022 international conference on artificial intelligence in everything (AIE 2022). https://doi.org/10.1109/AIE57029.2022.00024
194. Buduma N, Buduma N, Papa J, Locascio N (2022) Fundamentals of deep learning: designing next-generation machine intelligence algorithms, 2nd edn. O'Reilly Media Inc., Sebastopol
195. Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning. Springer, New York
196. Trott O, Olson AJ (2010) AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. J Comput Chem 31(2):455–461. https://doi.org/10.1002/JCC.21334
197. Eberhardt J, Santos-Martins D, Tillack AF, Forli S (2021) AutoDock Vina 1.2.0: new docking methods, expanded force field, and python bindings. J Chem Inf Model 61(8):3891–3898. https://doi.org/10.1021/ACS.JCIM.1C00203
198. Ragoza M, Hochuli J, Idrobo E, Sunseri J, Koes DR (2017) Protein-ligand scoring with convolutional neural networks. J Chem Inf Model 57(4):942–957. https://doi.org/10.1021/ACS.JCIM.6B00740
199. Alhossary A, Handoko SD, Mu Y, Kwoh CK (2015) Fast, accurate, and reliable molecular docking with QuickVina 2. Bioinformatics 31(13):2214–2216. https://doi.org/10.1093/BIOINFORMATICS/BTV082
200. Marcou G, Rognan D (2007) Optimizing fragment and scaffold docking by use of molecular interaction fingerprints. J Chem Inf Model 47(1):195–207. https://doi.org/10.1021/CI600342E
201. Stahl M, Mauser H (2005) Database clustering with a combination of fingerprint and maximum common substructure methods. J Chem Inf Model 45(3):542–548. https://doi.org/10.1021/CI050011H
202. Selleckchem.com - Bioactive Compounds Expert. 2025. https://www.selleckchem.com/.
203. Consortium TU, Bateman A, Martin MJ, Orchard S, Magrane M, Adesina A et al (2025) UniProt: the universal protein knowledgebase in 2025. Nucleic Acids Res 53(D1):D609–D617. https://doi.org/10.1093/NAR/GKAE1010ISBN:1471210510
204. Glielmo A, Husic BE, Rodriguez A, Clementi C, Noé F, Laio A (2021) Unsupervised learning methods for molecular simulation data. Chem Rev 121(16):9722–9758. https://doi.org/10.1021/ACS.CHEMREV.0C01195
205. Sarker IH (2021) Machine learning: algorithms, real-world applications and research directions. SN Comput Sci 2(3):1–21. https://doi.org/10.1007/S42979-021-00592-X
206. Goh GB, Hodas NO, Vishnu A (2017) Deep learning for computational chemistry. J Comput Chem 38(16):1291–1307. https://doi.org/10.1002/JCC.24764
207. Xu B, Wang N, Chen T, Li M (2015) Empirical evaluation of rectified activations in convolutional network. ArXiv. https://doi.org/10.48550/arXiv.1505.00853
208. Yang L, Shami A (2020) On hyperparameter optimization of machine learning algorithms: theory and practice. Neurocomputing 415:295–316. https://doi.org/10.1016/J.NEUCOM.2020.07.061
209. Linardatos P, Papastefanopoulos V, Kotsiantis S (2020) Explainable AI: a review of machine learning interpretability methods. Entropy 23(1):18. https://doi.org/10.3390/E23010018
210. Van Efferen L, Ali-Eldin AMT (2017) A multi-layer perceptron approach for flow-based anomaly detection. In: 2017 international symposium on networks, computers and communications, ISNCC 2017. https://doi.org/10.1109/ISNCC.2017.8072036
211. Sarker IH (2021) Deep cybersecurity: a comprehensive overview from neural network and deep learning perspective. SN Comput Sci 2:154. https://doi.org/10.1007/S42979-021-00535-6
212. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324. https://doi.org/10.1109/5.726791
213. Yang X, Wang Y, Byrne R, Schneider G, Yang S (2019) Concepts of artificial intelligence for computer-assisted drug discovery. Chem Rev 119(18):10520–10594. https://doi.org/10.1021/ACS.CHEMREV.8B00728
214. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), 770–778.https://doi.org/10.1109/CVPR.2016.90
215. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780. https://doi.org/10.1162/NECO.1997.9.8.1735
216. Chung J, Gulcehre C, Cho K, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. In: NIPS'14: Deep learning and representation learning workshop. arXiv. https://doi.org/10.48550/arXiv.1412.3555
217. Corso G, Stark H, Jegelka S, Jaakkola T, Barzilay R (2024) Graph neural networks. Nat Rev Methods Primers 4(17):1–13. https://doi.org/10.1038/s43586-024-00294-7
218. Zhang Z, Chen L, Zhong F, Wang D, Jiang J, Zhang S et al (2022) Graph neural network approaches for drug-target interactions. Curr Opin Struct Biol 73:102327. https://doi.org/10.1016/J.SBI.2021.102327
219. Deng L (2014) A tutorial survey of architectures, algorithms, and applications for deep learning. APSIPA Trans Signal Inf Process 3(1):e2. https://doi.org/10.1017/atsip.2013.9
220. Hoseini P, Zhao L, Shehu A (2021) Generative deep learning for macromolecular structure and dynamics. Curr Opin Struct Biol 67:170–177. https://doi.org/10.1016/J.SBI.2020.11.012
221. Liu W, Wang Z, Liu X, Zeng N, Liu Y, Alsaadi FE (2017) A survey of deep neural network architectures and their applications. Neurocomputing 234:11–26. https://doi.org/10.1016/J.NEUCOM.2016.12.038
222. Zhang G, Liu Y, Jin X (2020) A survey of autoencoder-based recommender systems. Front Comput Sci 14(2):430–450. https://doi.org/10.1007/S11704-018-8052-6
223. Kingma DP, Welling M (2019) An introduction to variational autoencoders. Found Trends Mach Learn 12(4):307–392. https://doi.org/10.1561/2200000056
224. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507. https://doi.org/10.1126/SCIENCE.1127647

Publisher's Note