ELSEVIER

Teaser This paper focuses on deep learning approaches in the context of drug discovery for designing new effective molecules, predicting for the desired molecular property profiles and planning synthesis.

Deep learning in drug discovery: opportunities, challenges and future prospects

Antonio Lavecchia

Department of Pharmacy, ``Drug Discovery'' Laboratory, University of Napoli ``Federico II'', via D. Montesano 49, I-80131 Napoli, Italy

Antonio Lavecchia is Professor of Medicinal Chemistry and Head of Drug Discovery Laboratory at the Department of Pharmacy of the University of Napoli Federico II, where he is also Scientific Responsible of the Excellence Laboratory of Molecular Modeling. His research interests are in the field of computational drug discovery. In particular, he has developed and applied computational approaches to accelerate the discovery of drug-like compounds in several therapeutic areas, including cancer, diabetes and dyslipidemia, pain and inflammation, and age-related neurodegenerative disorders. Prof. Lavecchia is the author of more than 130 publications in high-ranked journals, is inventor of three PCT patents, and has delivered numerous invited lectures and seminars. In 2006 he was awarded the Farmindustria Prize for Pharmaceutical Research.

Artificial Intelligence (AI) is an area of computer science that simulates the structures and operating principles of the human brain. Machine learning (ML) belongs to the area of AI and endeavors to develop models from exposure to training data. Deep Learning (DL) is another subset of AI, where models represent geometric transformations over many different layers. This technology has shown tremendous potential in areas such as computer vision, speech recognition and natural language processing. More recently, DL has also been successfully applied in drug discovery. Here, I analyze several relevant DL applications and case studies, providing a detailed view of the current state-of-the-art in drug discovery and highlighting not only the problematic issues, but also the successes and opportunities for further advances.

Introduction

The discovery and development of a new drug is an extremely long, costly, challenging, and inefficient process that takes an average of 10–15 years. Any failure is an enormous financial damage, and in fact, failures are not rare. Despite advances in technology and a very good understanding of biological systems, in the last two decades the pharmaceutical industry has seen a increasingly weakening in research and development productivity due to growing costs, while the absolute number of newly approved drugs has continually diminished because of ever-increasing regulatory obstacles and mounting difficulty in discovering the next blockbuster drug (either in a new disease area or a far superior drug to what is presently in the market) [1]. Over the past decade, progresses in information technology and growing automation have led to the production and collection of huge, hard-to-handle quantities of compound activity and bio-medical data [2,3]. The volume and complexity of these data open to new possibilities (such as to diminish costs and time by improving patients' survival and quality of life or to explore alternative roads of research pathways resulting in personalized cancer drugs), but also set new challenges. Enhancements in computer speed and power make analysis of big data much more practicable. However, it is impossible to handle these huge amounts of data and find suitable patterns by classical statistical approaches and human evaluation alone. Therefore, data

Corresponding author. Lavecchia, A. (antonio.lavecchia@unina.it)

https://doi.org/10.1016/j.drudis.2019.07.006

GLOSSARY

Artificial intelligence Set of theories and techniques used to create machines capable of simulating intelligence.
Autoencoder network A type of neural network that is trained to reconstruct its input at its output. Because there are fewer intermediary hidden units than inputs, the network is forced to learn a short compressed representation at the hidden units, which can be interpreted as a process of abstraction.
Backpropagation A learning algorithm for ANNs used for supervised learning, where connection weights are iteratively updated to decrease the approximation error at the output units.
Big data A large corpus of data, which is typically both structured and unstructured.
Boltzmann Machine Stochastic neural network of symmetrically connected, neuron-like units whose dynamics is governed by an energy function. The input to the network is given through a layer of visible units, while another layer of hidden units is used to model the latent causes of the data. A variant known as Restricted Boltzmann Machine (RBM) is obtained by removing within-layer lateral connections to form a bipartite graph, allowing performing efficient inference and learning.
Convolutional neural networks (CNNs) A class of deep, feed-forward neural networks that is especially amenable to analyzing natural images. The convolution is typically a spatial filter which synthetizes local (neighboring) spatial information.
Deep Belief Network (DBN) Hierarchical generative model composed of a stack of RBMs, which can be greedily trained layer-wise in an unsupervised fashion. The whole network can be eventually fine-tuned with supervised learning to perform discriminative tasks.
Deep learning (DL) ML framework that exploits multiple layers of hidden units to build hierarchical internal representations of the input data.
Dropout Dropout is a regularization method for neural networks that prevents overfitting by randomly setting a fraction of neurons to 0 at each training iteration.
Feed-forward network Layered neural network in which connectivity between layers is one way, starting at the input layer and ending at the output layer.
Generative model A model defined in such a way so as to represent the way we believe the data has been generated. We think of hidden causes that generates the data and also of higher-level hidden causes. Slippery roads may cause accidents, and rain may have caused roads to be slippery.
Long short-term memory (LSTM) LSTM networks aim to prevent the vanishing gradient problem in RNNs by using a memory gating mechanism.
Machine learning A set of statistical tools and algorithms, which are capable of extracting the dominant patterns in data. The data mining can be supervised or unsupervised, with the goal of clustering, classification and prediction.
Max pooling A data down-sampling strategy whereby an input representation (image, hidden-layer output matrix, etc.) is reduced in dimensionality, thus allowing for assumptions to be made about features contained in the down-sampled sub-regions.
Multiple Layer Perceptron (MLP) A MLP is a class of feed-forward ANN, which can distinguish data that are not linearly separable. An MLP usually consists of at least two nonlinear layers.
Overfitting State reached by a learning algorithm when the number of adjustable parameters in a network model is much greater than the number of training data and the algorithm uses the excess capacity to memorize the examples. Although overfitting greatly reduces a network's ability to generalize to new examples, it can be reduced by regularization.
Perceptron A simple neural network model consisting of one unit and inputs with variable weights that can be trained to classify inputs into categories.
Probability distribution Function that specifies the probability of occurrence of all possible states of a system or outcomes in an experiment.
Recurrent network Neural network whose feedback connections allow signals to circulate within it.
Regularization Method to avoid overfitting a network model with many parameters when the training data are limited, such as weight decay, in which all the weights in the network decrease on every epoch of training, and only the weights with large positive gradients survive.
Reinforcement Learning Training a neural network by giving it positive or negative rewards for its actions.

mining has become a progressively important research field, leading to the development of artificial intelligence (AI) and machine learning (ML) (see Glossary). AI is the art of making machines that execute tasks that require intelligence when executed by people. There are two themes of thoughts in AI, namely *weak AI* and *strong AI*. [4] The first is that which can do something very specific, very well. The second is that which thinks like humans, draws on general knowledge, mimics common sense, threatens to become self-aware, and takes over the world. Although no one can predict if or when *strong AI* will appear on the international scene, *weak AI* already today is able of realizing many human jobs better than biological brain. Pandora is very good at choosing what music you might like based on the type of music you liked before [5]. Amazon is very good at guessing that if you bought this, you might like to buy that. Apple's Siri, a virtual assistant for iPhones, provides a bulk of different services, such as sport news, weather reports, answers to user's questions, and so on. The Google's self-driving car (now known as Waymo) drives itself far more safely than any human can drive a car; by 2030 surgical robots will operate better than any human surgeon. In March 2017, the victory recorded by AlphaGo [6], an AI program developed by Google's DeepMind group, over Lee Sedol from South Korea, represented a crucial moment in the history of AI.

AI-based methods are also gaining increasing popularity in the pharmaceutical industry since they offer the capability to accelerate and streamline the drug development pipeline in preclinical studies as well as in subsequent clinical trials [7–13]. ML is a subfield within AI, which aims to develop and utilize algorithms that learn from raw data (Fig. 1). ML tasks, where a set of input-output paired data is provided for training, are referred to as supervised learning tasks. When the task is to predict from a set of potential outcomes, it is a classification task, and when the task is to predict a numeric value, it is a regression task. Output examples from supervised ML are virtual screening predictions

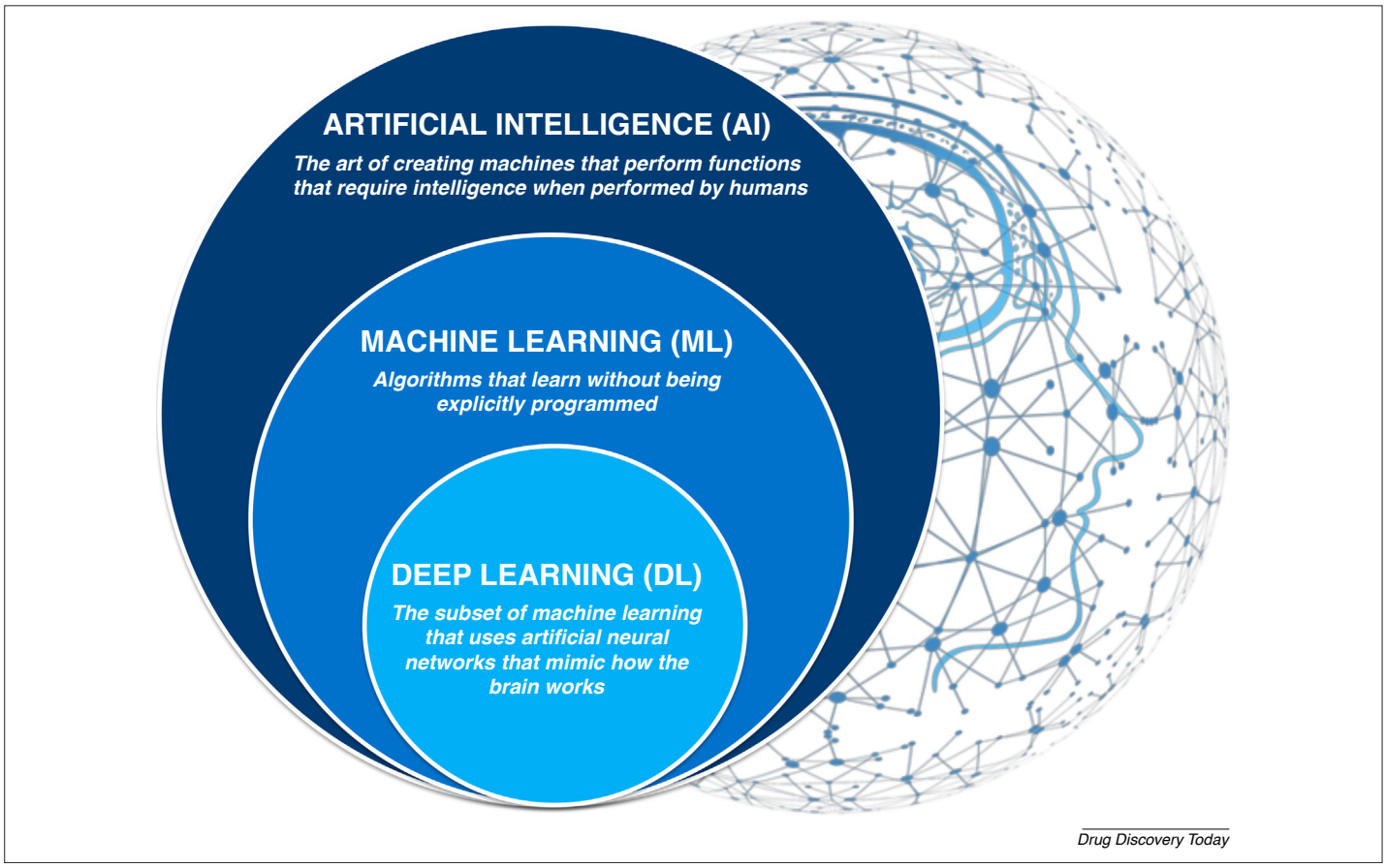

FIGURE 1

The relationship between AI, ML and DL.

[14–16], disease diagnosis (under the subcategory classification) [17] or drug efficacy [18] and absorption, distribution, metabolism, excretion and toxicity (ADMET) prediction (under the subcategory regression) [19]. There are two other types of ML: unsupervised learning and reinforcement learning. When input-only data are provided and the algorithm attempts to find something useful in the data, we talk about unsupervised learning. Typical outputs of unsupervised learning include disease subtype discovery from clustering and disease target discovery from feature-finding approaches [20]. In reinforcement learning, the agent (e.g., a real or simulated robot) interacts with the environment for solving a sequential decision task and modifies its action policies to maximize its performance [21]. The outputs from this kind of ML comprise *de novo* drug design under decision-making and experimental designs under execution, both obtainable through modeling and quantum chemistry [22]. A recent application of reinforcement learning is Deep Q-network (DQN)-Docking, an algorithm that applies a DQN to the context of protein-ligand docking prediction [23].

Deep Learning (DL) is a subfield within ML (Fig. 1) based on the use of different artificial neural network (ANN) algorithms that, through a sequence of layers with non-linear processing units, learn high-level abstractions for data [24]. In DL architectures, each layer trains on a distinct set of features based on the previous layer's output. The further you advance into the neural net, the more complex the features your nodes can recognize, since they aggregate and recombine features from the previous layer. For example, a DL algorithm predicting whether an image contains a face or not extracts features such as the first layer perceiving edges, the second layer perceiving shapes such as noses and eyes, and the final layer perceiving face shapes or more complex structures. This is known as feature hierarchy, and it is a hierarchy of increasing complexity and abstraction. It makes DL networks capable of handling very large, high-dimensional data sets with billions of parameters that pass through nonlinear functions. DL has gained huge success in a wide range of applications such as computer games [25], speech recognition [26], visual perception (including hand-writing as well as photo- or object-classification) [27], language translation [28], and self-driving cars. In the drug discovery field, DL applications are rapidly advancing with new articles published almost every week, and there are many recent reviews on various aspects of DL that can be referred to for more detailed information [22,29–37].

Here, I give an overview of DL in the context of drug discovery, define the basic concepts and the rationale of prominent DL architectures and provide recent case studies to demonstrate the practical application of this approach in the field. Finally, I discuss challenges as well as its future potential.

Molecular representation

An important decision before undertaking either generative or predictive modeling of molecules is the choice of representation. Transforming a molecule into vectors of numbers that can be accepted by learning systems is called molecular featurization.

Four kinds of forms that describe molecules can be inputted into DL models: molecular descriptors (e.g. logP, polar surface area, etc.) and fingerprints (vectors that encode molecule structure), molecular graph-based models, simplified molecular input line entry system (SMILES) strings, and grids for convolutional neural networks (CNNs). An in depth description of these molecular representations is beyond the scope of this review. For this the reader is referred to a number of excellent reviews and textbooks on the subject [38–40].

Basic concepts of DL

DL is a set of algorithms in ML that uses ANNs with multiple layers of nonlinear processing units for modeling high-level abstractions contained in data. ANNs have been built with a logic similar to the biological neuronal networks in the human brain. Each neuron can be looked at as a processing unit and an interlinking of such neurons leads to massive computing power that can solve complex operations. The first artificial neuron model was proposed in 1943 by McCulloch and Pitts in terms of a computational model of nervous activity [41]. This model was followed by another, the so-called perceptron [42], a two-layer neural network used for simple tasks, and further expanded in the late 1960s with the introduction of the back-propagation algorithm, used for efficient multiplayer networks training [43]. A perceptron neuron model is shown in Fig. 2a, where x_1, x_2, . . . , x_n are input signals entering the neuron, along with a *bias* term b that can be viewed as the weight to a fixed signal $x_0 = 1$. These inputs are the stimulation levels of a natural neuron. Each input x_i is multiplied by its corresponding weight w_i, then the product $x_i w_i$ is supplied to the body of the neuron. The weights represent the biological synaptic strengths in a natural neuron. The neuron sums all the products for $i = 1, n$. The weighted sum of the products, plus the *bias* b, forms the linear function u_i. Finally, the neuron computes its output y as nonlinear transfer function $f(u_i)$. This function is called the activation (or sometimes transfer) function. It is usually chosen as a monotonically non-decreasing and differentiable function. Sigmoid function is perhaps one of the most popular activation functions because it resembles the behavior of biological neurons. The same output value y can be sent out through multiple ends emerging from the neuron.

Part of the artificial neurons can be connected together to form a network, known as ANN. With different ways of connections, various types of ANNs can be achieved. In the network, there are input layer, several intermediate layers known as hidden layers, and the output layer. Fig. 2b represents schematically this notion. Input, hidden and output layers can be linked in many fashions, which will have an impact on the performance of the network for different tasks. The final result of any input propagated through the network will depend on the pattern of connections in every layer, the functions controlling neuronal activation and the weights associated to each link between neurons. The architecture (pattern of connections) and the activation function play a critical role in the final result obtainable by the network; essentially they define the global model of the network, that has to be in line with the nature of the task to be performed.

When the number of hidden layers is more than two or three, for example hundreds, then that is known as deep neural network (DNN). Such DNNs have the capability to learn and solve problems and the learning is called DL [44] (Fig. 2c). The greater number of hidden layers is, the deeper the network is. This is because each hidden layer extracts more features from the previous layer and generates its own abstract representation. Therefore, to resolve

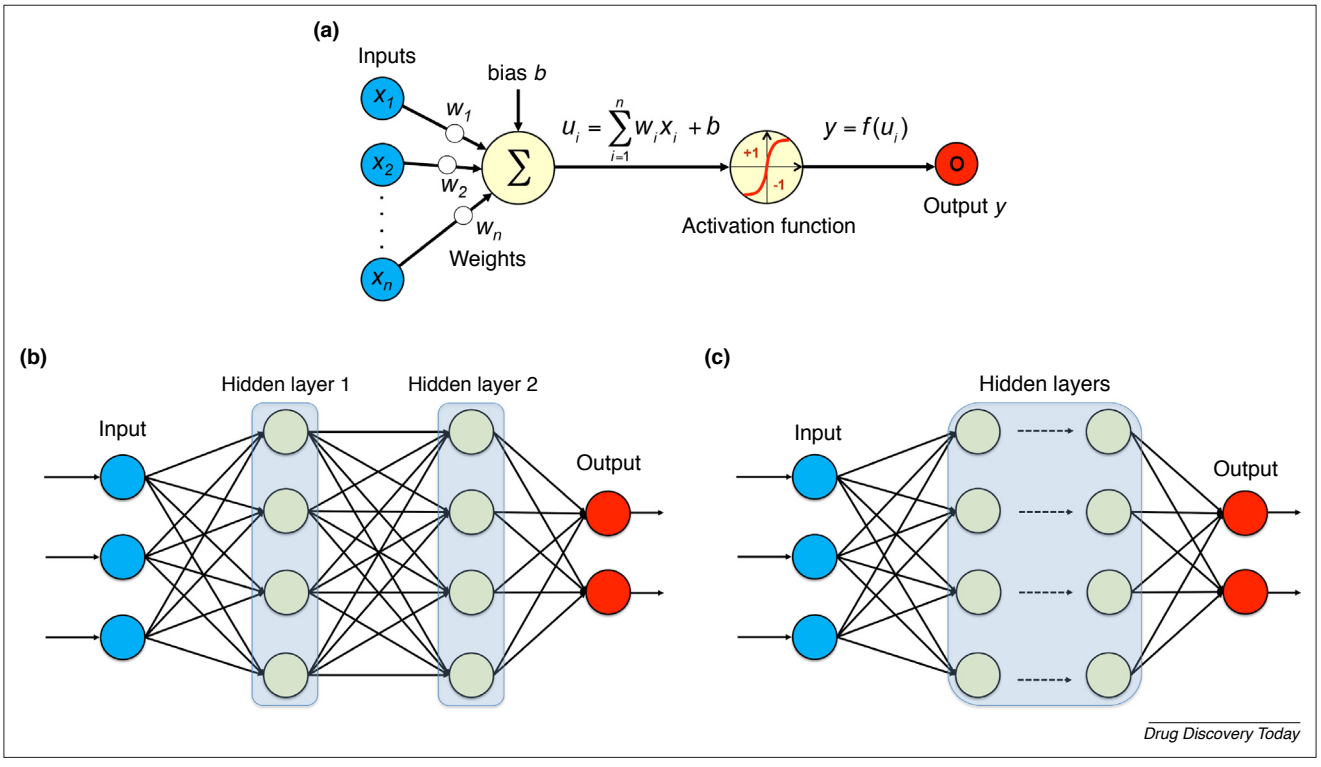

FIGURE 2

more complex features, we have to insert more hidden layers, which make DL able of learning concealed information. The layers are interlinked, with the output of the previous layer being the input of the current layer. DNNs need higher processing power, computing speed, large database and the suitable software with parallel processing. Overfitting still remains a challenge to overcome when DL comes to training very large ANNs or working in domains that offer very small amounts of data. Two new approaches have been recently proposed to prevent this problem: DropOut [45] and DropConnect [46], which is a generalization of the previous. When training with DropOut, a randomly selected subset of activations is dropped within each layer. With DropConnect, we randomly drop the weights. Both of these two techniques, which are possible only for fully connected layers, have been proved to be powerful for regularizing ANNs.

Several open source DL frameworks (Table 1), as well as popular higher level APIs (Application Programming Interfaces) such as Keras and Gluon, were developed to provide an efficient and friendly development environment for researchers to design and implement DNNs. Many of these libraries are well supported, with dozens of active contributors and large user bases, and are very fast in training DNNs with billions of parameters.

DL neural networks can also be built with a variety of architectures, the most commonly used of which are illustrated in Figs. 3 and 4 and described in more detail below. Table 2 summarizes the pros and cons of the most common DL architectures.

Convolutional neural networks (CNNs)

CNNs are feed-forward neural networks, which have made remarkable achievements in the field of image recognition. CNNs are inspired by the mammal's visual cortex [47], which contains very small neuronal cells that are sensitive to specific areas of the visual field, called receptive field. These cells act as local filters over the input space. Hubel and Wiesel further studied this idea in 1962 [48]. In their experiments, they showed that some specific neuronal cells in the brain responded (or excited) only in the presence of edges or lines of a certain orientation. For example, some neurons responded when exposed to vertical edges and some when had horizontal or diagonal edges. All of these neurons, which appeared to be spatially arranged in columnar structures, are able to produce visual perception. This idea of specialized structures within a system having specific assignments is one that machines use as well, and is the basis of CNNs [49]. A simple CNN is a succession of layers. A CNN takes an image and passes it through a series of convolution, activation, pooling/subsampling, and fully-connection layers to get an output (Fig. 3a). This output can be a single class or a probability of classes that best describe the image. Part of the networks may remove the pooling layer or fully-connection layer because of the special task.

The convolution layer is the most important part of a CNN. The parameters in this layer are composed of a set of filters, also called kernels. The main purpose of the convolution task is to extract different features of the input data. The first convolution layer extracts the low-level features such as lines, edges, and corners. Higher-level layers extract the higher-level features. We assume the input has size $M \times M \times D$ and is convolved with K kernels, each of size $n \times n \times D$ apart. Convolution of an input with one kernel yields one output feature. So, the single convolution with K kernels yields K features. Starting from top-left corner of the input, each kernel is shifted from left to right and top to bottom until the kernel gets to the bottom-right corner. For each step, element-by-element multiplication is computed between $n \times n \times D$ elements of the kernel on each position of the kernel. Therefore, $n \times n \times D$ multiplication and addition operations are indispensable to generate one element of one output feature.

To increase the nonlinear ability of the decision functions, an activation layer is used, named Rectified Linear Unit (ReLU) layer. It applies a transformation to the output of the convolution layer by means of an element-wise activation function without changing the dimension of the output. It also helps to train faster.

The pooling layer is responsible for progressively reducing the spatial size of the features. It decreases the number of parameters and calculation. It also makes the features robust against noise and distortion. There are two strategies for subsampling: max pooling and average pooling. Both the pooling functions split the input into non-overlapping two-dimensional space.

The final layer in a CNN is generally a fully connected layer. It is a Multi-Layer Perceptorn (MLP) that utilizes an activation function to compute the weighted sum of all the features of the previous layer to classify the data into target classes.

Recurrent neural network (RNN)

A RNN is a kind of ANN where links between units form a directed cycle [50]. This generates an internal state of the network that allows it to display dynamic temporal behavior. Unlike feed-forward networks, RNNs include feedback components that permit signals from one layer to be fed back to a previous layer. Moreover, they are the only neural networks with an internal memory, the use of which is necessary to remove difficulties of learning to store

TABLE 1

A sampling of available open source DL frameworks that implement the methods discussed in this review.

Software	Description	Url
Theano	CPU/GPU symbolic expression compiler in python	http://deeplearning.net/software/theano/
Torch	Provides a Matlab-like environment for state-of-the-art machine learning algorithms in lua	http://torch.ch/
TensorFlow™	Software library for numerical computation using data flow graphs	https://www.tensorflow.org/
MXNet	Software designed for both efficiency and flexibility	https://mxnet.apache.org/
Caffe	Framework made with expression, speed, and modularity in mind	http://caffe.berkeleyvision.org/
Lasagne	Lightweight library to build and train neural networks in Theano	https://lasagne.readthedocs.io/en/latest/
Keras	A theano-based DL library	https://keras.io/
Chainer	A GPU based Neural Network Framework	https://chainer.org/
MatConvNet	A MATLAB toolbox implementing CNNs for computer vision applications	http://www.vlfeat.org/matconvnet/
Deeplearning4J	An Apache 2.0-licensed, distributed neural net library written in Java and Scala	https://deeplearning4j.org/

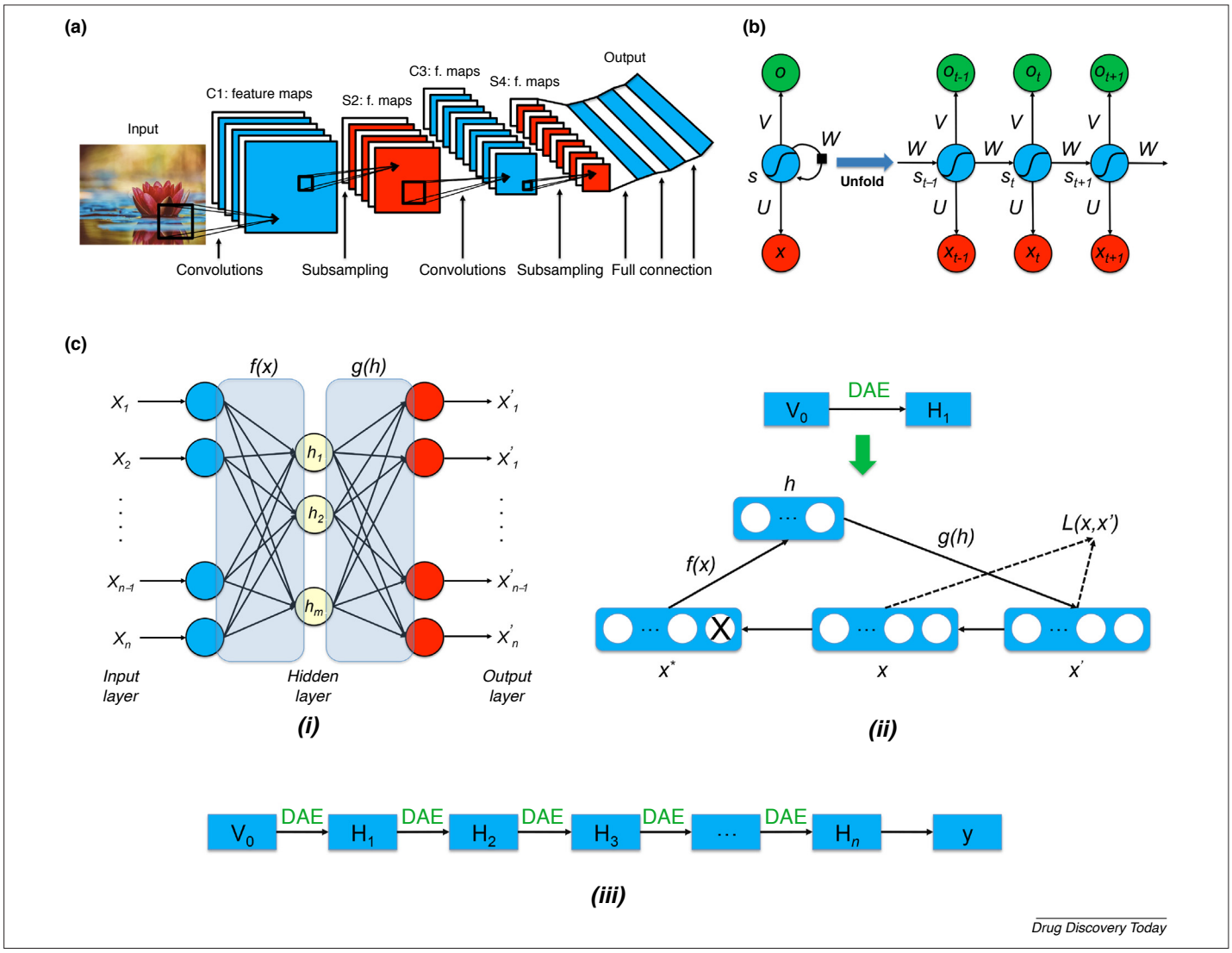

FIGURE 3

Three of the most popular classes of DL architectures in drug discovery data analysis. **(a)** Common CNN architecture including consecutive layers of convolution feature maps and subsampling functions, followed by layers of conventional neural networks. **(b)** Basic RNN architecture consisting of input (*x*), hidden (*s*) and output units (*o*). *U*, *V*, and *W* are the weight matrixes for the transition from *x* to *s*, *s* to *s*, and *s* to *o*, respectively. **(c)** Simple AE (*i*), DAE (*ii*) and SDAE (*iii*) architecture.

long-term information [51]. Thanks to this memory, RNNs are competent for reading a sequence and its context so they are more preferable for sequential data like texts, protein sequences, time series data, audio and video signals. Fig. 3b illustrates the basic RNN architecture. The left part of the figure shows a folded RNN with a self-loop, i.e., the hidden layer *s* is employed to update itself given an input *x*. In the right part of Fig. 3b, the RNN is unfolded as a sequential structure in order to show how it works. The RNN receives a sequence, $x_1, x_2, \ldots, x_t$ as input, where at each step *t*, x_t is a *d*-dimensional feature vector. For example, if the input is a sentence, then each word w_i of the sentence is represented as a vector x_i. At each time-step *t*, the output of the previous step, s_{t-1}, along with the next word vector in the document, x_t, are used to update the hidden state s_t as

$$s_t = f(Ws_{t-1} + Ux_t)$$

where *W* and *U* are the weights of inputs s_{t-1} and x_t, respectively, and *f* is normally nonlinear activation function. The hidden state s_t is the feature representation of the sequence up to time *t* for the input sequence. The initial states s_0 are usually initialized as all 0. Thus, we can utilize s_t to perform various tasks such as sentence completion and document classification. To train an RNN can be used the back-propagation through time (BPTT) [43], which back-propagates the error from time *t* to time 1.

However, the above basic RNN model suffers from gradient vanishing and exploding problem and cannot model the long-term dependences. Therefore, more sophisticate architectures have been proposed for overcoming the drawbacks of regular RNNs including the Long-Short-Term Memories (LSTMs) [52] and the Gated Recurrent Units (GRUs) [53]. Both of these recurrent unit architectures add gates and memory cells (in the case of LSTM) in the hidden layer to control the amount of information entering the unit, the amount that will be stored and the information that will be passed to the next units.

Deep autoencoders (AE)

AEs are a class of neural networks in which the output is set to be equal to the input; i.e., they are unsupervised learning algorithms, which employ a back-propagation algorithm for training [54], and

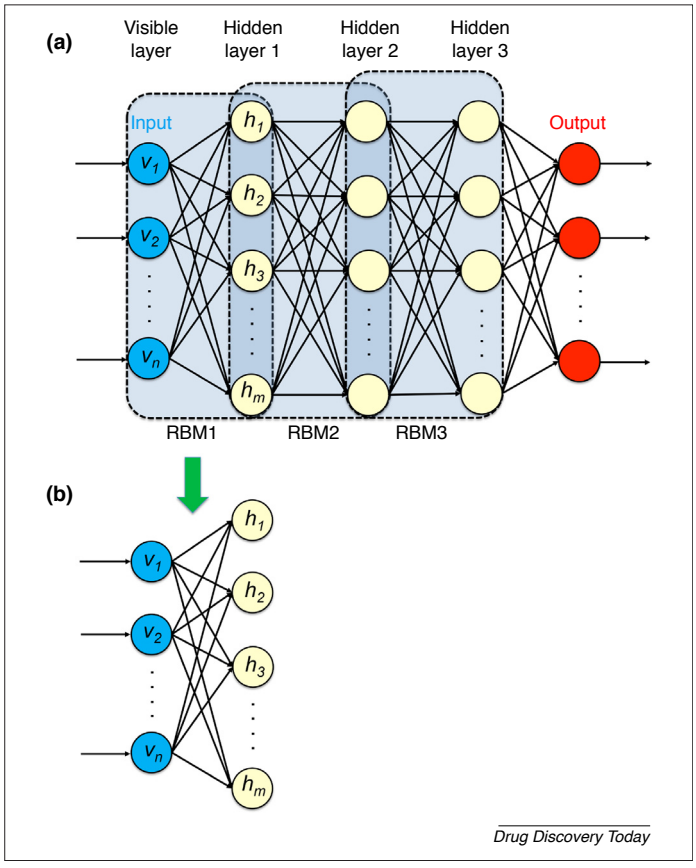

FIGURE 4

(a) DBN with three hidden layers and **(b)** RBM with *n* visible units and *m* hidden units.

have been widely used not only for dimensionality reduction, but also for feature extraction and denoising applications [55].

The network structure of an AE is similar to a multi-layer ANN. The simple structure (Fig. 3c) includes three components: an input layer of raw data to be encoded, a hidden (encoding) layer(s), featuring by a considerable smaller number of neurons, and an output (decoding) layer with the same size (i.e. the same number of neurons) of the input layer. For an AE in Fig. 3c(*i*), supposing that the number of nodes is n in input layer and m in output layer, respectively, the nonlinear activation function is $f(x) = (1 + e^{-x})^{-1}$ in hidden layer and $g(h) = \left(1 + e^{-h}\right)^{-1}$ in output layer, respectively. For the input sample $x = (x_1, x_2, \ldots, x_n)$,the values of the ith hidden node h_i and the jth output node $x_j^{'}$ are as follows:

$$h_i = f(w_i x + b_i) = f\left(\sum_{j=1}^{n} w_{ij} x_j + b_i\right)$$

$$x_j^{'} = g\left(w_j^{'} h + b_j^{'}\right) = g\left(\sum_{k=1}^{m} w_{jk}^{'} h_k + b_j^{'}\right)$$

where w_i is the weight between the input layer and the hidden layer, $w_j^{'}$ is the weight between the hidden layer and the output layer, b_i, $b_j^{'}$ are the bias of hidden layer and output layer, respectively. So, we must find the optimal parameters $\theta = \{w, b, w', b'\}$ which could make the error of x and $x_n' = (x_1^{'}, x_2^{'}, \ldots x_n^{'})$ as small as possible, namely minimize $L(x, x') = \sum_{i=1}^{n} (x_i - x_i^{'})^2$.

When the AE has a hidden layer size greater than or equal to that of the input, it is not guaranteed to learn the weights and can be become simply a unit switch to copy input to output. This problem is addressed by the denoising AE (DAE) [56], in which another layer is added between the input and the hidden layer. This layer adds some noise to the input using either sparsity constraints or dropout techniques (wherein input values are randomly set to 0). This noisy input then goes through learning from the hidden layer to the output layer just like the AE. The loss function of the DAE compares the output with the actual input. Thus, the added noise and the larger hidden layer allow either learning latent structures or adding/removing redundancy to generate the exact signal at the output. The structure of DAE is as shown in Fig. 3c (*ii*). Assuming that the noise proportion is k and the original input information is $x = (x_1, x_2, \ldots, x_n)$, the corrupted new input information is $x^* = (x_1, 0, \ldots, x_n)$ and the number of node equal to 0 is nk. Then, x^*, instead of x, is fed to the input layer in AE. The target value of the output layer still is x'. Then, the training process will be made according to AE. A well-trained DAE can recover its corresponding clean version.

In order to learn more complex features, it is necessary to stack the DAE to form a DL model called stacked DAE (SDAE) [57]. The training process of SDAE is divided into two parts: layer-by-layer pre-training and fine-tuning. Its structure is as shown in Fig. 3c (*iii*). The layer-by-layer pre-training is an unsupervised learning process. Firstly, the input layer and the first hidden layer would be seen as a DAE. After the first DAE trained, the first hidden layer, as the input layer in the second DAE, would construct the second DAE with the second hidden layer. And so on until nth DAE trained. The fine-tuning is a supervised learning process, in which the multiplayer DAE and the output layer in the top layer will be considered as a multiplayer back-propagation neural network (BPNN). The multiplayer BPNN uses the stochastic gradient descent to optimize the parameters.

Deep belief network (DBN)

As the learning process becomes more complex, DL will need more time on training. In this case, we normally cannot get satisfactory results. One solution for this problem is DBN [58], which not only can be used to identify and classify data, but also can produce data. DBNs are made up of multi-layers of neurons. These neurons are splitted into hidden layers and visible layers. Visible layers receive data, while hidden layers extract features, so they are named feature detectors. As depicted in Fig. 4a, a DBN is composed of several layers of Restricted Boltzmann Machine (RBM), which is a generative stochastic ANN capable to learn a probability distribution over the set of inputs [54].

A RBM consists of a visible layer unit and a hidden layer unit, but without visible-visible and hidde-hidden connections, as shown in Fig. 4b. This network can be trained in a supervised and unsupervised fashion and is able to represent a large number of features of the inputs, so hidden layer units can represent up to $2n$ features. The network can be trained to respond to a single question to a total of $2n$ questions. However, a single hidden layer RBM cannot extract all the features from the input data, as it is unable to determine the relationship between the variables. Hence, multiple layers of RBMs are used one after another to extract nonlinear features.

TABLE 2

Comparison of different DL algorithms.

Network	Feature	Pros	Cons
CNN	It consists of convolutional filters which transform 2D into 3D and it is very good for 2D data	Strong nonlinear mapping ability The powerful ability of extracting features Very good performance Model learns rapidly	It needs a great deal of labeled data for classification. The train time is long. Too much training samples lead to larger network size. Different tasks require separate training
RNN	It has the capability to learn sequences. The weights are shared across all steps and neurons	Learns sequential events Can model time dependencies Performs significantly better and is less expensive when working on complex tasks with large amounts of data	There are many issue due to a vanishing gradient and the need for big datasets Complex to build the right architecture suitable for a specific problem Does not yield better results if the prepared data is relatively small
AE	It is used in unsupervised learning and is designed mainly for extraction and reduction of the dimensionality of features. The number of input is equal to number of output	It does not need labeled data There are different variations: DAE, SDAE, Conventional AE for greater robustness	It needs a pre-training step Its training may disappear
DBN	It has unidirectional connection and is used in both supervised and unsupervised ML. The hidden layers of each sub-network serves as visible layer for the next layer	The greedy strategy (used in each layer) and the inference tractable maximize directly the likelihood	The initialization makes the training process computationally expensive

Training of a DBN consists of a greedy, unsupervised, layer-by-layer pre-training process and a supervised, global fine-tuning process. The former provides good initial values for all parameters, while the latter searches the optimal state based on the given initial parameters of the network. This hybrid training approach in terms of unsupervised pre-training and supervised fine-tuning increases the generative performance and the discriminative power of the network [59]. So, DBN becomes increasingly complex and highly desirable for solving classification problems.

DL applications in molecular property and activity prediction

DL has been employed in numerous cases for property and activity prediction. In many of these studies a comparison to other ML techniques has been made demonstrating that DL achieves comparable or better performance than other ML techniques for different properties including prediction of biological activity, ADMET properties and physico-chemical parameters.

One of the first applications of DL in drug discovery dates to 2012, when a competition on the prediction of drug properties and activities organized by the pharmaceutical company Merck was won by a multitask deep feed-forward algorithm developed in academia, with an improvement of about 15% in relative accuracy even over Merck's proprietary systems [60]. The report disclosed that the performance of DNN changes depending on the activation function used and the network architecture (number of hidden layers as well as number of neurons in each layer).

Mayr et al. developed the DeepTox pipeline, an ensemble DL-based model for predicting toxic effects of chemical compounds, that won the Tox21 toxicology prediction challenge in 2014 on a dataset containing 12,000 drugs and environmental chemicals with up to 12 different toxicity endpoints [61]. Of particular significance is the ability of this model to outperform other ML approaches in 9 out of 12 toxic endpoints. Following these works, many groups proved that massively multitask DL architectures perform better than single-task and Random Forrest (RF) models in property prediction [62–64]. Others also compared several ML approaches with different ChEMBL data sets using random split and temporal cross validation to show the superiority of DL [65].

Subramanian et al. demonstrated that conventional ML methods such as RF and simple DL networks are able of predicting the inhibitory activity against beta-Secretase 1 (BACE-1) from only a few hundred compounds and, mostly, in the absence of a very large training data set [66]. Aliper et al. demonstrated how DL algorithms trained on large transcriptional response data sets, can classify various drugs into therapeutic categories solely based on their transcriptional profiles in different cell lines combined with pathway information [67].

Other strategies from both cheminformatics and ML algorithms directly operate on graphs (i.e. molecular structure) rather than making use of predefined molecular descriptors. Lusci and colleagues, for example, developed a DL approach called UG-RNN (Undirected Graph Recursive Neural Networks) that encodes molecules into undirected graph-based systems to effectively predict the water solubility of compounds [68]. One benefit of this approach is that it relies only marginally on the identification of suitable molecular descriptors because the appropriate representations are learned automatically from the data. Similarly, Xu et al. used the same approach to model drug-induced liver injury (DILI) [69]. The model was trained on 475 drugs, with an external test set of 198 drugs. The authors generated a model with a performance exceptional compared to other models (AUC = 0.955).

More recently, attempts to utilize CNN to model chemical data were also reported. Duvenaud et al. introduced a CNN approach to generate continuous molecular fingerprints directly from molecular graphs [70]. The information about each atom and its neighbors are provided to the neural network, and neural fingerprints

are used to predict new features for the graphs. Other notable works on graph neural networks are included in Ref [71–75].

In Ref [75], Coley and coworkers introduced a CNN embedding of attributed molecular graphs to predict molecular properties (octanol and aqueous solubility, melting point, and biological toxicity), which enhanced the performance of Duvenaud's model. The model used a molecular tensor combining the bond-level and atom-level attributes to describe attributed graph representation of molecules. The full workflow of this approach can be seen in Fig. 5.

Very recently, Gilmer et al. suggested that many of these methodologies can be considered as specific examples of a general message passing formalism, and coined the term message passing neural networks (MPNNs) to refer to them together [76]. Other strategies use SMILES strings to train RNN models. For example, Bjerrum proposed SMILES enumeration, a single-line text uniquely representing one molecule, as the raw input to LSTM cell-based neural network to build predictive models without the requirement to create molecular descriptors [77]. Interestingly, the author showed that SMILES enumeration produces statistically more robust QSAR models both when predicting single SMILES, but even more when taking the average prediction using enumerated SMILES for the same molecule. Recent work has even explored the concept of using 2D molecule images with minimal chemical information to develop models that were on average equivalent to contemporary DNN models trained on molecular fingerprints [78,79].

DL applications in de novo *design*

Several DL-based techniques have been projected for molecular *de novo* design and molecular property extraction. For example, Gómez-Bombarelli et al. introduced a variational AE (VAE) combined to a MLP to create new molecules with desired properties automatically [80]. The network consisted of an encoder that converts discrete SMILES strings into continuous vectors in latent space, a decoder that can make these vectors back to the discrete SMILES strings, and a MLP-based predictor that predicts the property of the molecules (Fig. 6a). To find the continuous vectors with high predictive value of the property, the Bayesian optimization is leveraged. However, the model has the disadvantage to produce many cases of invalid SMILES that do not correspond to plausible chemical structures. To overcome the limitation of generating valid molecules, Kusner et al. [81] and Dai et al. [82] developed the grammar VAE that explicitly adds syntactic and semantic constraints to SMILES strings using context free and attribute grammars (Fig. 6b). In 2017, Kadurin et al. [83] trained an adversarial AE (Fig. 6c), which combines the properties of both a generator and a discriminator, on a dataset of molecules with different tumor growth inhibition activity. The resulting model then was employed to create fingerprints of molecules with desired properties. Further examination of the generated molecules revealed that new molecular fingerprints matched closely to already known highly effective anticancer drugs such as anthracyclines. Subsequently, authors proposed an improved architecture named druGAN that also incorporated further molecular parameters such as solubility and permitted generating more chemically different molecules [84]. This new model clearly exhibited enhancement in terms of feature extraction, generation ability and reconstruction error, suggesting a great potential in drug discovery. Blaschke et al. also applied the adversarial AE to generate ligands specific to the dopamine type 2 receptor [85]. However, it should be noted that generative AEs might suffer from mode collapse [86] such that they generate only molecules with low diversity. Although models that do not suffer from mode collapse were proposed [87], a suitable metric should be able to evaluate the internal diversity of the generated molecules.

Recently, there has been an increasing interest in using the RNN for the *de novo* design of molecules. Segler et al. introduced the

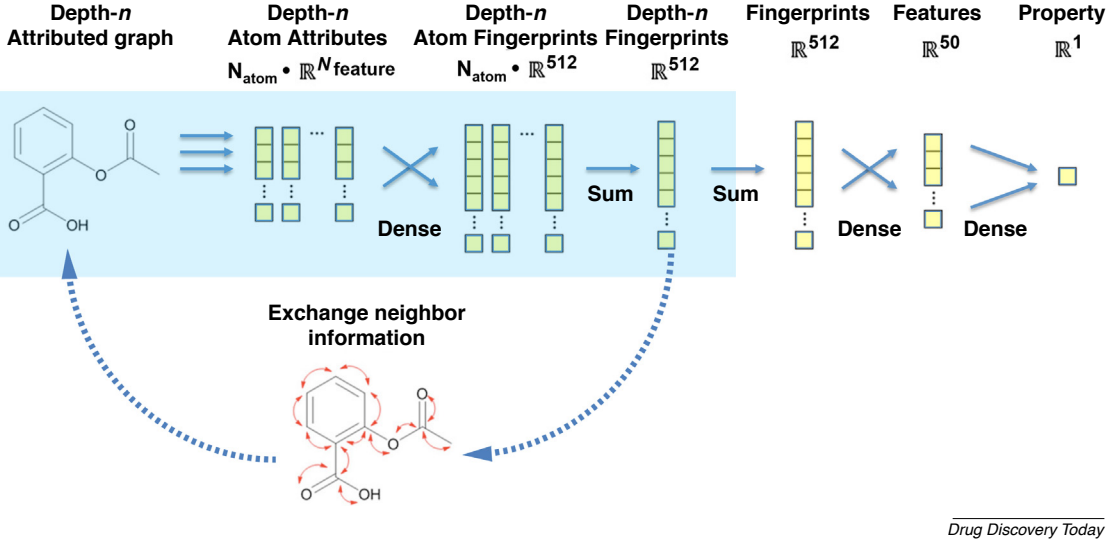

FIGURE 5

Architecture model of graph-based CNN for molecular embedding [75]. In this model, each molecule is represented as an undirected graph containing nodes (atoms) and edges (bonds). Atom feature vectors are iteratively updated based on those calculated at the previous depth. The neighboring atom vectors are combined linearly with associated bond vectors and passed through a nonlinear activation function to obtain new atom vectors. Atom feature vectors are transformed into longer atom fingerprint vectors through another learned mapping so that the resulting fingerprint vectors can be simply summed to get the

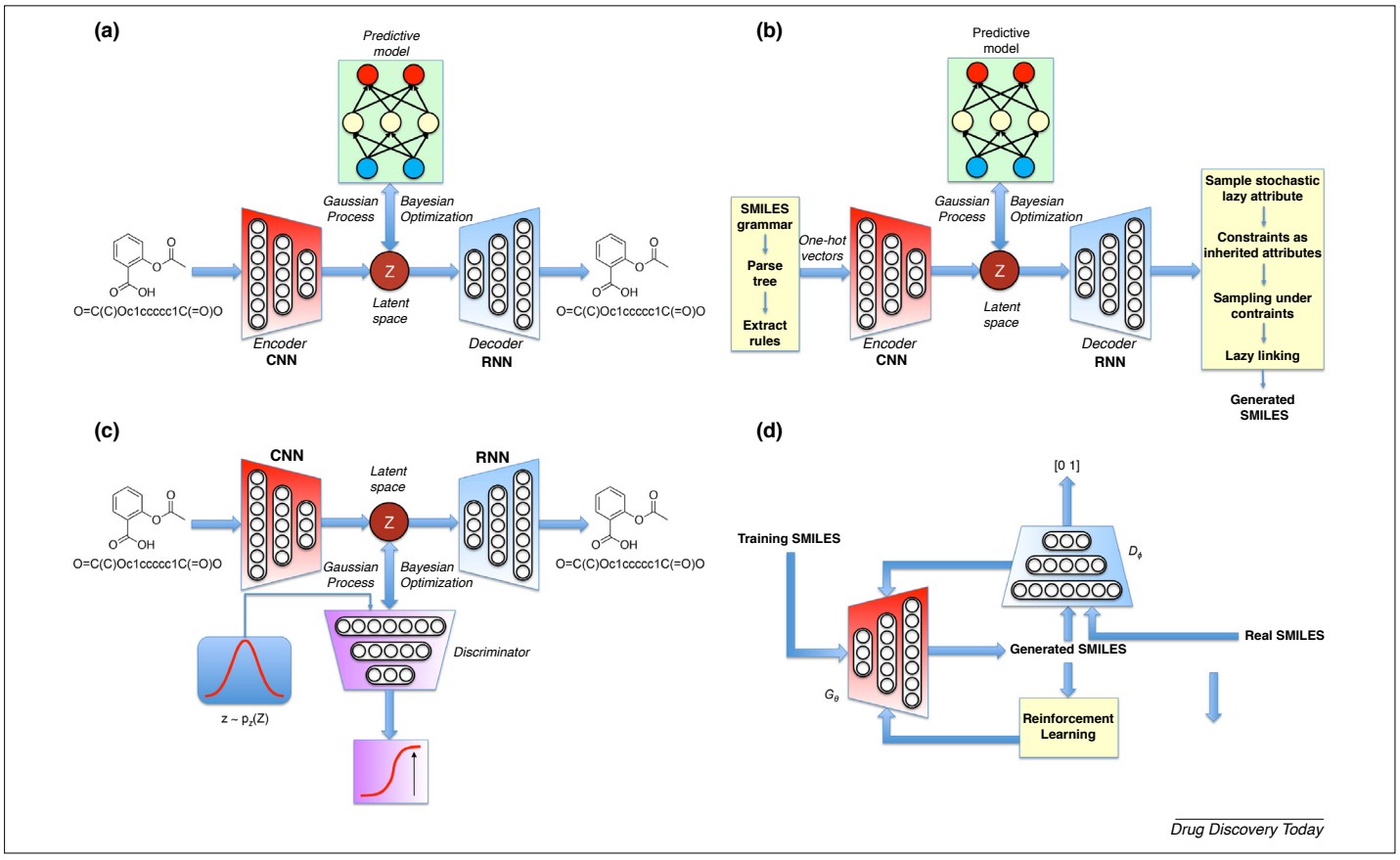

FIGURE 6

Current frameworks of RNN-, VAE-, and reinforcement learning-based SMILES generative models. **(a)** VAE-based *de novo* molecular generative model [80] consisting of an encoder, a decoder and a MLP-based predictor. **(b)** Syntax-directed VAE [82] utilizing a context-free grammar-based decoder to capture the syntax and incorporates at same time ring bonds and valence of atoms to address the semantics. **(c)** Adversarial AE [83], in which an adversarial network is added to an AE standard to induce the latent vector code z to match a certain target distribution $p_z(z)$. **(d)** Objective-Reinforced Generative Adversarial Networks (ORGAN) [92], in which the GAN framework is combined with reward functions with reinforcement learning to produce SMILES strings. G_θ is a LSTM-based generator parameterized by θ, that generates high-quality sequences $X_{1:T} = (x_1, \ldots, x_T)$ of length T. The discriminator D_φ parameterized by φ is a CNN specifically for sequence classification tasks.

notion of transfer and reinforcement learning for generating focused molecule libraries using RNNs on SMILES, a string-based representation derived from molecular graphs [88]. The generative LSTM-based model pretrained on a general corpus of molecules was either fine-tuned on a small number of known actives, or coupled to an external scoring function for creating novel structures with desirable activity against a specific biological target. Another similar LSTM-based RNN model with different applications was also developed by Gupta et al. for *de novo* drug design and fragment-based drug discovery [89]. In addition, Jaques et al. [90], Olivecrona et al. [91], and Guimaraes et al. [92] proposed a reinforcement learning method to modify a pretrained molecular generative model so as to impose several desirable properties (such as logP, synthetic accessibility and quantitative estimate of drug-likeness) in molecules produced from the generative model (Fig. 6d).

CNN applications in predicting drug-target interactions

Drug-target scoring is a key step in a structure-based drug design pipeline. Choosing a suitable binding pose and predicting the binding affinity of a drug-target complex increases the chances of a successful virtual screening application [93]. Specifically, CNN scoring functions have shown considerable skill in pose/affinity prediction and active/inactive detection for drug-target complexes, demonstrating an exceptional performance when compared with several well-performing scoring functions developed with both linear and nonlinear methods. For example, Ragoza et al. demonstrated that multilayer CNN models can successfully learn to differentiate between correct and incorrect binding poses when trained on 3D drug-target structures [94]. To train and test the model as well as compare its performance with that of simple docking softwares, the Database of Useful Decoys-Enhanced (DUD-E), which contains a large number of experimentally verified actives and property-matched decoys, was utilized. The authors found that the scoring function obtained based on the CNN algorithm performed significantly better than Autodock Vina [95] in terms of predicting both binding poses and affinities. Wallach et al. built AtomNet, a deep CNN, for bioactivity prediction of small molecules in drug discovery applications [96]. The authors evaluated the accuracy of the model on DUD-E benchmark platform. AtomNet outperformed previous algorithms and reached an AUC greater than 0.9 on 57.8% of the targets. However, in a recent chemRxiv preprint, Chen et al. clearly showed that the high performance of CNN models trained on DUD-E is not attributable to having learned the features of protein-ligand interactions

but rather to analogue and decoy bias inherent in the DUD-E dataset, suggesting extreme caution in using this dataset when applied to ML-based method development [97]. Jimenéz et al. presented DeepSite, a deep CNN trained with protein structural data that treats proteins as 3D images [98]. The network was used for ligand binding site detection and demonstrated superior performance than the state-of-the-art. In 2018, the same authors [99] employed a 3D graph CNN model, named K_{DEEP}, to predict ligand-protein binding affinities, and obtained a Pearson correlation coefficient of 0.82, with a root mean square error of 1.27 in pK units between the predicted affinity and the experimental values. Gomes et al. [100] introduced an algorithm named Atomic Convolutional Neural Network (ACNN) for predicting the energy gap between a bounded drug-target complex and an unbounded state using spatial CNN directly from atomic coordinates. To process the input, the authors employed radial pooling filters with learnable mean and variance. Such filters provided a summary of the atomic environment and a representation that was invariable with respect to the arrangement of atoms and the orientation of the complex. The workflow of this method can be seen in Fig. 7. Authors found that ACNN either outperforms or performs competitively with cheminformatics and ML-based methods in predicting the binding free energy of a subset of drug-target complexes available in the PDB_{Bind} dataset.

Although some promising results have been achieved with CNNs, it is important to realize that, in many cases, their ability to improve results compared to presently used scoring functions has not been firmly established.

DL applications in planning chemical syntheses

Several ML and DL tools have been developed and applied for forward synthetic prediction to control the outcome of chemical reactions [101–106]. These tools usually use several statistical learning means to make the prediction job, such as reaction type classification [103] and automatic identification of the reaction center [104]. Most of the current DL methods for retrosynthetic analysis utilize similar tools and differ between them essentially in the fashion in which molecules are represented. In particular, Liu et al. formulated a sequence-to-sequence (seq2seq) model for retrosynthetic reaction prediction that converts (e.g., translates) a product SMILES string to reactant(s) SMILES string [107]. The authors made use of 50 000 reactions from US patents to train the LSTM network and achieved similar accuracy to rule-based approaches. However, owing to the nature of the translation

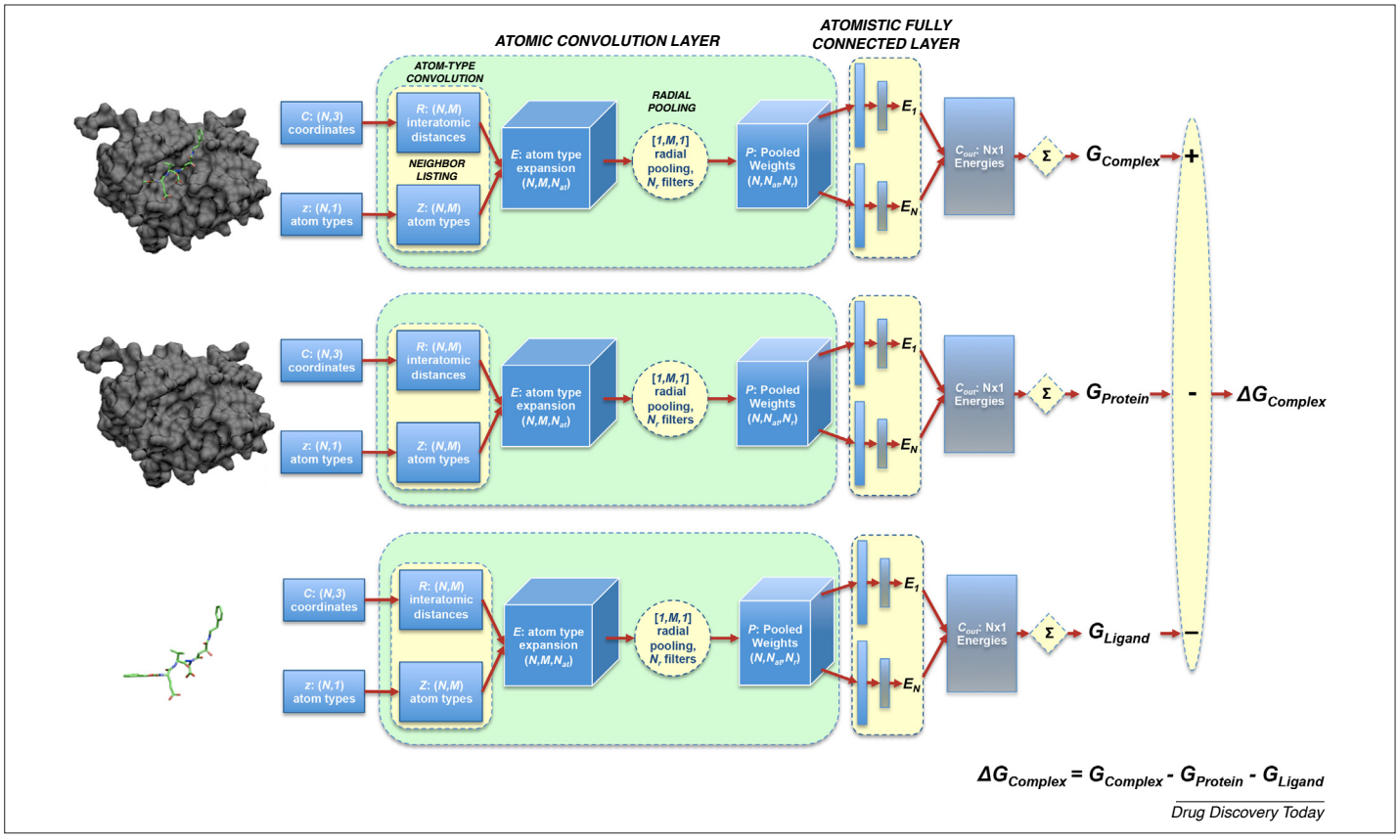

FIGURE 7

Architecture of ACNN for prediction of drug-target binding affinity [100]. The model includes: i) the (*N,M*) distance matrix *R*, which is constructed accepting as input a (*N,3*) coordinate matrix *C*, and the (*N,M*) matrix Z, an atomic number list of neighboring atoms (atom types), which is also constructed from the (*N,1*) atomic number vector *z*; ii) the atom type convolution layer, that creates the output matrix *E* of shape (N,M,N_{at}), where N_{at} is the number of unique atomic numbers (atom types) present in the molecular system; iii) the radial pooling layer, which aims to down-sample the output of the atom type convolution for preventing over-fitting, as well as reducing the number of parameters learned. It generates the output pooled matrix *P* of shape (N, N_{at}, N_r), where N_r is the number of desired radial filters; and iii) the atomistic fully connected network, that flatten the output of the radial pooling layer returning as output the energy E_i

process, the model tends to predict chemically unrealizable precursors for a product. Segler et al. developed a DNN for both retrosynthesis and reaction prediction that could be trained with large reaction sets (3.5 million) extracted from the Reaxys chemical database [108] and obtained a top-ten accuracy of 97% for reaction prediction and 95% in retrosynthesis. In 2018, the same authors projected an updated model for chemical synthesis prediction, composed by three DNNs combined with a Monte Carlo Tree Search (MCTS) framework [109]. This algorithm was trained on 12 million reactions from the Reaxys database, providing suggestions with extraordinary promptness and at a level comparable to human-driven synthesis planning.

Coley et al. produced for a given set of reactants all probable product candidates from a set of reaction templates obtained from US patents (candidate enumeration) and predicted the outcome of the reaction by ranking the candidates with a neural network (candidate ranking) [102]. The model correctly assigned the major product rank 1 in 71.8% of cases, rank $\leq$3 in 86.7% of cases, and rank $\leq$5 in 90.8% of cases. To obviate the problems of the hybrid template/neural network strategy, the same authors in a later research omitted templates and used a trainable model for candidate enumeration in addition to ranking [104]. The Weisfeller–Lehman Network (WLN), a graph CNN that incorporates information from neighboring atoms into each atom's representation, scored candidate reactions. The model demonstrated significantly higher accuracy than the template-based method and was several orders of magnitude faster, extending its application to a larger data set of about 400 000 reactions.

AI implementations also excel at optimizing existing chemical reactions. Recently, Zhou and co-workers [110] reported an autonomous platform for the optimization of chemical reaction conditions through the Deep Reaction Optimizer (DRO), an algorithm based on deep reinforcement learning. This model achieved better performance after running on real reactions, showing its learning ability.

CNNs applications in biomedical imaging

Recent advances in automated image-based high-content microscopy provide a new opportunity for drug discovery and preclinical development, as well as scalable and systematic phenotypic profiling of complex compound libraries, offering itself as a complementary method to target-based in vitro screening [111]. Image-based profiling analysis has demonstrated the ability to enhance the preclinical drug development at almost any stage of the pipeline from target identification over mechanism of action prediction to toxicity profiling.

In the past few years, CNN-based methods have been successfully used to address challenging problems in biomedical imaging because they learn multi-level internal representations of the data [44,112,113]. In drug discovery, these methodologies have been applied for image-based high-content screening experiments, which aim to identify targets and drugs that modulate phenotypes (such as subcellular localization of specific proteins) [114], and image-based profiling experiments, which aim to profile cells upon exposure to genetic, pathogenic or chemical perturbations [115]. Kraus et al. combined deep CNNs with multiple instance learning to classify small molecules by mechanism of action from full-size microscopy images of cancer cells [116]. Dürr et al. successfully used CNNs for phenotypic classification of cells treated with different biochemical compounds [117]. Godinez et al. developed CNNs approaches in order to accurately predict small molecule mechanism of action [118].

In biology, CNNs have been used for pixelwise image segmentation of nucleus, cytoplasm, cell, and nuclear membranes [119], neuronal membranes detection [120] and mitosis identification in histology images of the breast cancer [121]. Besides pixel-level tasks, DL has also been applied to cell- and tissue-level examination. Pärnamaa et al. used a 11-layer CNN for classifying fluorescent protein subcellular localization using microscopy yeast cell images [122]. Ferrari et al. employed CNNs to count bacterial colonies in agar plates [123]. Eulenberg et al. constructed a deep CNN framework combined with classification and visualization based on non-linear dimension reduction to analyze single cell flow cytometry images [124]. Furthermore, a CNN-based network was developed to segment and recognize neural stem cells in images taken by bright field microscope [125]. In the context of DL diagnostic tools, CNNs have also shown promise in improving the accuracy and efficiency of histopathological diagnosis [126] and in classifying cancer patients using immunohistochemistry of tumor tissues [127].

Finally, in the medical field, CNNs algorithms have been extensively used in analyzing medical images obtained from magnetic resonance imaging (MRI) [128–132], computed tomography (CT) scan [133–137], positron emission tomography (PET) [138], mammography [139], ultrasound [140], radiographic imaging [141], mass-spectrometry imaging (MSI) [142] and X-ray, to segment, classify, denoise, discover anomalies and diseases from these images.

Concluding remarks and future prospects

Recent progresses in AI, including the growth and development of more complex ML techniques, have made a huge impact on the drug development process. AI technology can address some of the main challenges notably reducing cost, time, and work demands during the early stages of drug discovery, by exploiting in silico approaches for *de novo* drug design, synthesis prediction, and bioactivity prediction. An extension technique from ML is DL. In this article, I have emphasized the theoretical foundations and exemplary recent applications of DL in chemoinformatics and drug discovery.

The DL technologies are a step ahead of ML technologies. Traditional ML techniques rely on deeply handcrafted features in extraction or engineering. For instance, to get decent results in image classification, several preprocessing procedures have to be applied, such as filters, edge detection, and so on. The biggest advantage of DL is that most, if not all, features can be learned automatically from the data, provided that enough (sometimes million) training data examples are available. DL models have feature detector units at each layer that gradually extract more complex and invariant features from the original raw input signals. Lower layers detect simple features that are then fed into higher layers, which in turn detect more complex features. In contrast, traditional ML models present very few layers that map the original input features into a problem-specific feature space. Conventional ML approaches solve the issue by breaking down the issue, solving different parts first, and then joining the results finally to give output, while DL approaches solve the issue using an

end-to-end approach, that is, directly feeding the inputs and corresponding outputs, without specifying any rules or patterns to the networks and without decomposing encoding training and decoding training into two separate steps. Unlike classical ML architectures, DL architectures take long time to get trained because of the huge number of parameters and relatively huge datasets, which limits its application generally. For example, if one has only a small amount of activity data available, it is difficult to predict the biological activity of the new molecules because a few data cannot cover a sufficient chemical space. The one-shot learning method [143,144], which requires only limited training samples, can be used in solving such a problem. Finally, conventional ML methods are quite restrictive, whereas DL methods can be applied to a wide range of applications and various domains. A large part of it goes to the transfer learning that allows making use of pre-trained DL networks for different tasks within the same domains. Table 3 compares DL and traditional ML techniques, giving excellent insight into both. Although we are discussing here all the advantages about DL over the other conventional ML approaches, we cannot assert that DL is superior to the traditional ML methods. There is no single algorithm that can solve all ML problems more efficiently than others. It all depends on specific application cases. For example, when the problem involves combined sets of compound or target protein input descriptors, the performance of DL does not significantly differ from other conventional ML algorithms. Conversely, DL has shown much better performance in biomedical image clustering and analysis as well as in predicting biological activity and toxicity, chemical reactions and *de novo* molecular design. Consequently, it can be deduced that DL together with ML can make enormous progresses collaboratively in the development of drug discovery perspectives.

Although significant progress has been made in the field of DL application in drug discovery, it remains a field with many significant challenges, such as the dependency on using experimental data for training and validation, and the scoring of compounds binding to proteins. Substantial advancement is being made in the field of transfer learning [145], one-shot learning [144], and conformal prediction [146], with recent enhancements in free-energy perturbation [147], which contributes to solving the scoring challenge.

Since the training of deep models with a large number of free parameters represents a complex optimization problem, many research works have been devoted to the development of efficient learning methods for deep networks. The strategies proposed to tackle problems of training deep architectures include the development of more efficient optimizers, the use of activation functions based on local competition, the use of well-designed initialization methods and the use of skip connections between layers with the aim of improving the flow of information. However, deep network training still has several problems caused by the stacking of several non-linear transformations that need to be addressed.

Further research into the generation of new descriptors (including fingerprint-based systems) as well as the development of supplementary criteria for a correct representation of a chemical system will be essential for the near future. The availability and plasticity of DL models can accelerate future developments via learned features and theory-informed models. Furthermore, since our human mind isn't infallible and we do not have the data to test the success of medicinal chemistry programs, we should try not to set the benchmark for DL methods too high. Finally, to take full advantage of AI techniques in the future, significant resources will be needed for data care, integration and management. The experimental confirmation of the AI effectiveness in drug discovery programs is a crucial factor in understanding how AI can contribute to medicinal chemistry and how it can be innovated and improved. Considerable investments in terms of budget and

TABLE 3

DL versus traditional ML algorithms.

Comparison Area	DL	ML
Data dependencies	Performance increases as the scale of data increases	Performance remains unchanged as the scale of data increases
Hardware dependencies	Algorithms heavily depend on high-end machines because they require GPUs, which is a major part of their working	Algorithms can work on low-end machines
Execution time training	Algorithms take a long time to train because there are so many parameters to train	Algorithms take much less time to train, ranging from a few seconds to a few hours
Problem-solving approach	The problem is solved end-to-end , that is it is learned directly from the data	The problem is broken down into different parts, that can be solved individually and then combined to get the result
Functioning	Subset of ML that takes data as an input and makes wise decisions using an ANN	Super-set of DL that takes data as an input, parses that data and tries to make decisions based on what it has learned while being trained
Feature engineering	Algorithms try to learn high-level features from the raw data, does not depend on hand-crafted features like local binary patterns and, most importantly, perform a hierarchical feature extraction	Algorithms are not able to extract meaningful features from the data, and depend on hand-crafted features as an input to perform well
Interpretability	Algorithms are not easy to interpret	Algorithms are interpretable regarding what parameters they chose and why they chose those parameters
Output	It is usually a score, an element, text, speech, etc	It is usually a numerical value like a score or a classification

reorganization of laboratory operations are indispensable during this phase to prove the future value of AI in rapidly synthesizing and testing the identified molecules. It will not be easy to convince the chemists to synthesize molecules that are not necessarily part of a drug discovery program, but this will likely expand the process moving forward. As such, in addition to playing an increasingly important role in drug discovery, AI and DL technologies must necessarily receive support from those scientists who will use AI systems closely and need to design, synthesize and test the compounds as the technologies associated with drug discovery become automated, while guaranteeing that the skilled human mind is deeply involved in the process.

For the future, we think that the collection of advanced ML systems can make the encouraging progression of technical enhancement. Furthermore, data fusion methods and efficiency evaluation of chemical structure data together with measured end points of relevance will also contribute to improve the performance of AI technologies in the field of drug discovery. In addition, because in this area of research the data are still limited in terms of volume and, even when accessible, have a high variability, it is expected that more advanced DL methods will be able to handle the effect of noisy data. Also, the associated algorithms should be able to tolerate these disarray datasets. While large amounts of data are available in many applications, however in some areas, large amount of data are rarely available. So, more flexible models will need to be developed to achieve an enhanced learning ability when only a limited amount of data is available. The creation of new automated platforms and software can considerably increase ML growth to a certain degree. In the context of DL, the most important future direction is to combine the conventional deep architectures with the aim to achieve higher performance and higher levels of integration between chemistry data, biomedical data, omics data and theoretical computation results (e.g., molecular dynamics simulation, molecular docking and quantum chemistry calculation). Finally, it prospects a future trend in the use of quantum ML for biochemical efforts involved in early phases of drug discovery [148]. Quantum ML, which sits at the intersection of quantum physics and ML, is emerging as a powerful methodology allowing quantum speed-ups and improving classical ML algorithms [149–151]. Recently, Wiebe et al. [152] have shown that quantum computing is capable of reducing the time required to train a RBM, while also providing a richer framework for DL than its classical analog. As a result, the field is open to real explorations of how chemical wave function data can be used by quantum algorithms for concrete drug discovery applications.

Conflicts of interest

The authors declare no conflicts of interest.

References

1 Scannell, J.W. *et al.* (2012) Diagnosing the decline in pharmaceutical R&D efficiency. *Nat. Rev. Drug Discov.* 11 (3), 191–200
2 Papadatos, G. *et al.* (2015) Activity, assay and target data curation and quality in the ChEMBL database. *J. Comput. Aided Mol. Des.* 29 (9), 885–896
3 Kim, S. *et al.* (2016) PubChem substance and compound databases. *Nucleic Acids Res.* 44 (D1), D1202–1213
4 Searle, J.R. (1980) Minds, brains, and programs. *Behav. Brain Sci.* 3 (3), 417–424
5 Turnbull, D. *et al.* (2008) Five approaches to collecting tags for music. pp. 225–230
6 Silver, D. *et al.* (2016) Mastering the game of Go with deep neural networks and tree search. *Nature* 529 (7587), 484
7 Lavecchia, A. (2015) Machine-learning approaches in drug discovery: methods and applications. *Drug Discov. Today* 20 (3), 318–331
8 Huang, Z. *et al.* (2017) Data mining for biomedicine and healthcare. *J. Healthcare Eng.* 7107629
9 Zhang, Y. *et al.* (2017) Computer-aided clinical trial recruitment based on domain-specific language translation: a case study of retinopathy of prematurity. *J. Healthcare. Eng.* 7862672
10 Merk, D. *et al.* (2018) De novo design of bioactive small molecules by artificial intelligence. *Mol. Inform.* 37 (1-2),
11 Menden, M.P. *et al.* (2013) Machine learning prediction of cancer cell sensitivity to drugs based on genomic and chemical properties. *PLoS One* 8 (4), e61318
12 Schneider, G. (2018) Automating drug discovery. *Nat. Rev. Drug Discov.* 17 (2), 97–113
13 Leber, A. *et al.* (2017) Modeling new immunoregulatory therapeutics as antimicrobial alternatives for treating Clostridium difficile infection. *Artif. Intell. Med.* 78, 1–13
14 Pérez-Sánchez, H. *et al.* (2014) Improving drug discovery using hybrid softcomputing methods. *Appl. Soft Comput.* 20, 119–126
15 Cano, G. *et al.* (2017) Automatic selection of molecular descriptors using random forest: Application to drug discovery. *Expert Syst. Appl.* 72, 151–159
16 Jiménez, F. *et al.* (2018) A methodology for evaluating multi-objective evolutionary feature selection for classification in the context of virtual screening. *Soft Comput.* 1–26
17 Guncar, G. *et al.* (2018) An application of machine learning to haematological diagnosis. *Sci. Rep.* 8 (1), 411
18 Li, H. *et al.* (2017) A novel multi-target regression framework for time-series prediction of drug efficacy. *Sci. Rep.* 7, 40652
19 Maltarollo, V.G. *et al.* (2015) Applying machine learning techniques for ADME-Tox prediction: a review. *Expert Opin. Drug Metab. Toxicol.* 11 (2), 259–271
20 Young, J.D. *et al.* (2017) Unsupervised deep learning reveals prognostically relevant subtypes of glioblastoma. *BMC Bioinf.* 18 (Suppl 11), 381
21 Arulkumaran, K. *et al.* (2017) *A brief survey of deep reinforcement learning. arXiv preprint arXiv:1708.058662*017
22 Chen, H. *et al.* (2018) The rise of deep learning in drug discovery. *Drug Discov. Today* 23 (6), 1241–1250
23 Serrano, A. *et al.* (2018) Accelerating drugs discovery with deep reinforcement learning: an early approach. In *Proceedings of the 47th International Conference on Parallel Processing Companion*. 6 ACM
24 Schmidhuber, J. (2015) Deep learning in neural networks: an overview. *Neural Netw.* 61, 85–117
25 Silver, D. *et al.* (2016) Mastering the game of Go with deep neural networks and tree search. *Nature* 529 (7587), 484–489
26 Deng, L. *et al.* (2013) New types of deep neural network learning for speech recognition and related applications: An overview. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. pp. 8599–8603, IEEE
27 Russakovsky, O. *et al.* (2015) Imagenet large scale visual recognition challenge. *Int. J. Comput. Vision* 115 (3), 211–252
28 Wu, Y. *et al.* (2016) *Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.081442*016
29 Ching, T. *et al.* (2018) Opportunities and obstacles for deep learning in biology and medicine. *J. R. Soc. Interface* 15 (141),
30 Goh, G.B. *et al.* (2017) Deep learning for computational chemistry. *J. Comput. Chem.* 38 (16), 1291–1307
31 Gawehn, E. *et al.* (2018) Advancing drug discovery via GPU-based deep learning. *Expert Opin. Drug Discov.* 13 (7), 579–582
32 Gawehn, E. *et al.* (2016) Deep learning in drug discovery. *Mol. Inform.* 35 (1), 3–14
33 Jing, Y. *et al.* (2018) Deep learning for drug design: an artificial intelligence paradigm for drug discovery in the big data era. *AAPS J.* 20 (3), 58
34 Rifaioglu, A.S. *et al.* (2018) Recent applications of deep learning and machine intelligence on in silico drug discovery: methods, tools and databases. *Brief. Bioinform.* 10, 1–36
35 Ghasemi, F. *et al.* (2018) Neural network and deep-learning algorithms used in QSAR studies: merits and drawbacks. *Drug Discov. Today* 23, 1784–1790
36 Ghasemi, F. *et al.* (2018) Deep neural network in QSAR studies using deep belief network. *Appl. Soft Comput.* 62, 251–258
37 Ghasemi, F. *et al.* (2017) The role of different sampling methods in improving biological activity prediction using deep belief network. *J. Comput. Chem.* 38 (4), 195–203

38 Xu, Y. *et al.* (2018) An overview of neural networks for drug discovery and the inputs used. *Expert Opin. Drug Discovery* 13 (12), 1091–1102
39 Elton, D.C. *et al.* (2019) *Deep learning for molecular generation and optimization-a review of the state of the art. arXiv preprint arXiv:1903.04388*2019
40 (2019) *Deep Learning for the Life Sciences* (1th ed.), O'Reilly Media, Sebastopol
41 McCulloch, W.S. and Pitts, W. (1943) A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophys.* 5 (4), 115–133
42 Rosenblatt, F. (1958) The perceptron: a probabilistic model for information storage and organization in the brain. *Psychol. Rev.* 65 (6), 386
43 Werbos, P.J. (1990) Backpropagation through time: what it does and how to do it. *Proc. IEEE* 78 (10), 1550–1560
44 LeCun, Y. *et al.* (2015) Deep learning. *Nature* 521 (7553), 436–444
45 Srivastava, N. *et al.* (2014) Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15 (1), 1929–1958
46 Wan, L. *et al.* (2013) Regularization of neural networks using dropconnect. In *International Conference on Machine Learning*. 1058–1066
47 Matsugu, M. *et al.* (2003) Subject independent facial expression recognition with robust face detection using a convolutional neural network. *Neural Netw.* 16 (5-6), 555–559
48 Hubel, D.H. and Wiesel, T.N. (1962) Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *J. Physiol.* 160, 106–154
49 Shin, H.C. *et al.* (2016) Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE Trans. Med. Imaging* 35 (5), 1285–1298
50 Olurotimi, O. (1994) Recurrent neural network training with feedforward complexity. *IEEE Trans. Neural Netw.* 5 (2), 185–197
51 Bengio, Y. *et al.* (1994) Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Netw.* 5 (2), 157–166
52 Hochreiter, S. and Schmidhuber, J. (1997) Long short-term memory. *Neural Comput.* 9 (8), 1735–1780
53 Cho, K. *et al.* (2014) *On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259*2014
54 Hinton, G.E. and Salakhutdinov, R.R. (2006) Reducing the dimensionality of data with neural networks. *Science* 313 (5786), 504–507
55 Deng, L. and Yu, D. (2014) Deep learning: methods and applications. *Found. Trends Signal Process.* 7 (3–4), 197–387
56 Vincent, P. *et al.* (2008) Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*. pp. 1096–1103, ACM
57 Vincent, P. *et al.* (2010) Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.* 11 (Dec), 3371–3408
58 Hinton, G.E. *et al.* (2006) A fast learning algorithm for deep belief nets. *Neural Comput.* 18 (7), 1527–1554
59 Chen, X.-W. and Lin, X. (2014) Big data deep learning: challenges and perspectives. *IEEE Access* 2, 514–525
60 Ma, J. *et al.* (2015) Deep neural nets as a method for quantitative structure–activity relationships. *J. Chem. Inf. Model.* 55 (2), 263–274
61 Mayr, A. *et al.* (2016) DeepTox: toxicity prediction using deep learning. *Front. Environ. Sci.* 3, 80
62 Ramsundar, B. *et al.* (2015) *Massively multitask networks for drug discovery. arXiv preprint arXiv:1502.02072*2015
63 Unterthiner, T. *et al.* (2014) Deep learning as an opportunity in virtual screening. In *Proceedings of the deep learning workshop at NIPS* vol. 27, 1–9
64 Ramsundar, B. *et al.* (2017) Is multitask deep learning practical for pharma? *J. Chem. Inf. Model.* 57 (8), 2068–2076
65 Lenselink, E.B. *et al.* (2017) Beyond the hype: deep neural networks outperform established methods using a ChEMBL bioactivity benchmark set. *J. Cheminf.* 9 (1), 45
66 Subramanian, G. *et al.* (2016) Computational modeling of β-secretase 1 (BACE-1) inhibitors using ligand based approaches. *J. Chem. Inf. Model.* 56 (10), 1936–1949
67 Aliper, A. *et al.* (2016) Deep learning applications for predicting pharmacological properties of drugs and drug repurposing using transcriptomic data. *Mol. Pharm.* 13 (7), 2524–2530
68 Lusci, A. *et al.* (2013) Deep architectures and deep learning in chemoinformatics: the prediction of aqueous solubility for drug-like molecules. *J. Chem. Inf. Model.* 53 (7), 1563–1575
69 Xu, Y. *et al.* (2015) Deep learning for drug-induced liver injury. *J. Chem. Inf. Model.* 55 (10), 2085–2093
70 Duvenaud, D.K. *et al.* (2015) Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*. 2224–2232
71 Kearnes, S. *et al.* (2016) Molecular graph convolutions: moving beyond fingerprints. *J. Comput.-Aided Mol. Des.* 30 (8), 595–608
72 Li, J. *et al.* (2017) *Learning graph-level representation for drug discovery. arXiv preprint arXiv:1709.03741*2017
73 Xu, Y. *et al.* (2017) Deep learning based regression and multiclass models for acute oral toxicity prediction with automatic chemical feature extraction. *J. Chem. Inf. Model.* 57 (11), 2672–2685
74 Schütt, K.T. *et al.* (2017) Quantum-chemical insights from deep tensor neural networks. *Nat. Commun.* 8, 13890
75 Coley, C.W. *et al.* (2017) Convolutional embedding of attributed molecular graphs for physical property prediction. *J. Chem. Inf. Model.* 57 (8), 1757–1772
76 Gilmer, J. *et al.* (2017) *Neural message passing for quantum chemistry. arXiv preprint arXiv:1704.01212*2017
77 Bjerrum, E.J. (2017) *SMILES enumeration as data augmentation for neural network modeling of molecules. arXiv preprint arXiv:1703.07076*2017
78 Goh, G.B. *et al.* (2017) *Chemception: a deep neural network with minimal chemistry knowledge matches the performance of expert-developed QSAR/QSPR models. arXiv preprint arXiv:1706.06689*2017
79 Goh, G.B. *et al.* (2018) How much chemistry does a deep neural network need to know to make accurate predictions? In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. pp. 1340–1349, IEEE
80 Gómez-Bombarelli, R. *et al.* (2018) Automatic chemical design using a data-driven continuous representation of molecules. *ACS Cent. Sci.* 4 (2), 268–276
81 Kusner, M.J. *et al.* (2017) Grammar variational autoencoder. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. 1945–1954.JMLR.org
82 Dai, H. *et al.* (2018) *Syntax-directed variational autoencoder for structured data. arXiv preprint arXiv:1802.08786*2018
83 Kadurin, A. *et al.* (2017) The cornucopia of meaningful leads: Applying deep adversarial autoencoders for new molecule development in oncology. *Oncotarget* 8 (7), 10883
84 Kadurin, A. *et al.* (2017) druGAN: an advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico. *Mol. Pharmaceutics* 14 (9), 3098–3104
85 Blaschke, T. *et al.* (2018) Application of generative autoencoder in de novo molecular design. *Mol. Inf.* 37 (1-2), 1700123
86 Unterthiner, T. *et al.* (2017) *Coulomb GANs: provably optimal nash equilibria via potential fields. arXiv preprint arXiv:1708.08819*2017
87 Popova, M. *et al.* (2018) Deep reinforcement learning for de novo drug design. *Sci. Adv.* 4 (7), eaap7885
88 Segler, M.H.S. *et al.* (2017) Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS Cent. Sci.* 4 (1), 120–131
89 Gupta, A. *et al.* (2018) Generative recurrent networks for de novo drug design. *Mol. Inf.* 37 (1-2), 1700111
90 Jaques, N. *et al.* (2017) Sequence tutor: Conservative fine-tuning of sequence generation models with kl-control. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. 1645–1654.JMLR.org
91 Olivecrona, M. *et al.* (2017) Molecular de-novo design through deep reinforcement learning. *J. Cheminf.* 9 (1), 48
92 Guimaraes, G.L. *et al.* (2017) *Objective-reinforced generative adversarial networks (ORGAN) for sequence generation models. arXiv preprint arXiv:1705.10843*2017
93 Lavecchia, A. and Di Giovanni, C. (2013) Virtual screening strategies in drug discovery: a critical review. *Curr. Med. Chem.* 20 (23), 2839–2860
94 Ragoza, M. *et al.* (2017) Protein–ligand scoring with convolutional neural networks. *J. Chem. Inf. Model.* 57 (4), 942–957
95 Trott, O. and Olson, A.J. (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
96 Wallach, I. *et al.* (2015) *AtomNet: A deep convolutional neural network for bioactivity prediction in structure-based drug discovery. arXiv preprint arXiv:1510.02855*2015
97 Chen, L. *et al.* (2019) *Hidden Bias in the DUD-E Dataset Leads to Misleading Performance of Deep Learning in Structure-Based Virtual Screening. ChemRxiv*2019. http://dx.doi.org/10.26434/chemrxiv.7886165.v1
98 Jiménez, J. *et al.* (2017) DeepSite: protein-binding site predictor using 3D-convolutional neural networks. *Bioinformatics* 33 (19), 3036–3042
99 Jiménez, J. *et al.* (2018) K DEEP: Protein-ligand absolute binding affinity prediction via 3D-convolutional neural networks. *J. Chem. Inf. Model.* 58 (2), 287–296
100 Gomes, J. *et al.* (2017) *Atomic convolutional networks for predicting protein-ligand binding affinity. arXiv preprint arXiv:1703.10603*2017
101 Segler, M.H.S. and Waller, M.P. (2017) Modelling chemical reasoning to predict and invent reactions. *Chem.–A Eur. J.* 23 (25), 6118–6128
102 Coley, C.W. *et al.* (2017) Prediction of organic reaction outcomes using machine learning. *ACS Cent. Sci.* 3 (5), 434–443

103 Wei, J.N. *et al.* (2016) Neural networks for the prediction of organic chemistry reactions. *ACS Cent. Sci.* 2 (10), 725–732
104 Jin, W. *et al.* (2017) Predicting organic reaction outcomes with weisfeiler-lehman network. In *Advances in Neural Information Processing Systems*. 2607–2616
105 Fooshee, D. *et al.* (2018) Deep learning for chemical reaction prediction. *Mol. Syst. Des. Eng.* 3 (3), 442–452
106 Coley, C.W. *et al.* (2018) Machine learning in computer-aided synthesis planning. *Acc. Chem. Res.* 51 (5), 1281–1289
107 Liu, B. *et al.* (2017) Retrosynthetic reaction prediction using neural sequence-to-sequence models. *ACS Cent. Sci.* 3 (10), 1103–1113
108 Segler, M.H.S. and Waller, M.P. (2017) Neural-symbolic machine learning for retrosynthesis and reaction prediction. *Chem.–A Eur. J.* 23 (25), 5966–5971
109 Segler, M.H.S. *et al.* (2018) Planning chemical syntheses with deep neural networks and symbolic AI. *Nature* 555 (7698), 604
110 Zhou, Z. *et al.* (2017) Optimizing chemical reactions with deep reinforcement learning. *ACS Cent. Sci.* 3 (12), 1337–1344
111 Scheeder, C. *et al.* (2018) Machine learning and image-based profiling in drug discovery. *Curr. Opin. Syst. Biol.* 10, 43–52
112 Shen, D. *et al.* (2017) Deep learning in medical image analysis. *Annu. Rev. Biomed. Eng.* 19, 221–248
113 Falk, T. *et al.* (2019) U-Net: deep learning for cell counting, detection, and morphometry. *Nat. Methods* 16 (1), 67
114 Moffat, J.G. *et al.* (2017) Opportunities and challenges in phenotypic drug discovery: an industry perspective. *Nat. Rev. Drug Discovery* 16 (8), 531
115 Schirle, M. and Jenkins, J.L. (2016) Identifying compound efficacy targets in phenotypic drug discovery. *Drug Discovery Today* 21 (1), 82–89
116 Kraus, O.Z. *et al.* (2016) Classifying and segmenting microscopy images with deep multiple instance learning. *Bioinformatics* 32 (12), i52–i59
117 Dürr, O. and Sick, B. (2016) Single-cell phenotype classification using deep convolutional neural networks. *J. Biomol. Screening* 21 (9), 998–1003
118 Godinez, W.J. *et al.* (2017) A multi-scale convolutional neural network for phenotyping high-content cellular images. *Bioinformatics* 33 (13), 2010–2019
119 Ning, F. *et al.* (2005) Toward automatic phenotyping of developing embryos from videos. *IEEE Trans. Image Process.* 14 (9), 1360–1371
120 Cireşan, D. *et al.* (2012) Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in neural information processing systems*. 2843–2851
121 Cireşan, D.C. *et al.* (2013) Mitosis detection in breast cancer histology images with deep neural networks. In *International Conference on Medical Image Computing and Computer-assisted Intervention*. pp. 411–418, Springer
122 Pärnamaa, T. and Parts, L. (2017) Accurate classification of protein subcellular localization from high-throughput microscopy images using deep learning. *G3: Genes, Genomes, Genet.* 7 (5), 1385–1392
123 Ferrari, A. *et al.* (2017) Bacterial colony counting with convolutional neural networks in digital microbiology imaging. *Pattern Recognit.* 61, 629–640
124 Eulenberg, P. *et al.* (2016) Deep learning for imaging flow cytometry: cell cycle analysis of Jurkat cells. *bioRxiv* 081364
125 Jiang, B. *et al.* (2015) Convolutional neural networks in automatic recognition of trans-differentiated neural progenitor cells under bright-field microscopy. In *In Instrumentation and Measurement, Computer, Communication and Control (IMCCC), 2015 Fifth International Conference on*. pp. 122–126, IEEE
126 Litjens, G. *et al.* (2016) Deep learning as a tool for increased accuracy and efficiency of histopathological diagnosis. *Sci. Rep.* 6, 26286
127 Vandenberghe, M.E. *et al.* (2017) Relevance of deep learning to facilitate the diagnosis of HER2 status in breast cancer. *Sci. Rep.* 7, 45938
128 Kamnitsas, K. *et al.* (2017) Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation. *Med. Image Anal.* 36, 61–78
129 Havaei, M. *et al.* (2017) Brain tumor segmentation with deep neural networks. *Med. Image Anal.* 35, 18–31
130 Havaei, M. *et al.* (2016) Deep learning trends for focal brain pathology segmentation in MRI. In *Machine Learning for Health Informatics*. pp. 125–148, Springer
131 Kleesiek, J. *et al.* (2016) Deep MRI brain extraction: a 3D convolutional neural network for skull stripping. *NeuroImage* 129, 460–469
132 Nie, D. *et al.* (2016) 3D deep learning for multi-modal imaging-guided survival time prediction of brain tumor patients. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. pp. 212–220, Springer
133 Fritscher, K. *et al.* (2016) Deep neural networks for fast segmentation of 3D medical images. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. pp. 158–165, Springer
134 Gao, X.W. *et al.* (2017) Classification of CT brain images based on deep learning networks. *Comput. Methods Programs Biomed.* 138, 49–56
135 Roth, H.R. *et al.* (2016) Improving computer-aided detection using convolutional neural networks and random view aggregation. *IEEE Trans. Med. Imaging* 35 (5), 1170–1181
136 Hoo-Chang, S. *et al.* (2016) Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE Trans. Med. Imaging* 35 (5), 1285
137 Shen, W. *et al.* (2017) Multi-crop convolutional neural networks for lung nodule malignancy suspiciousness classification. *Pattern Recognit.* 61, 663–673
138 Ypsilantis, P.-P. *et al.* (2015) Predicting response to neoadjuvant chemotherapy with PET imaging using convolutional neural networks. *PloS One* 10 (9), e0137036
139 Kooi, T. *et al.* (2017) Large scale deep learning for computer aided detection of mammographic lesions. *Med. Image Anal.* 35, 303–312
140 Chen, H. *et al.* (2015) Standard plane localization in fetal ultrasound via domain transferred deep neural networks. *IEEE J. Biomed. Health. Inf.* 19 (5), 1627–1636
141 Lee, S. *et al.* (2015) FingerNet: Deep learning-based robust finger joint detection from radiographs. In *Biomedical Circuits and Systems Conference (BioCAS), 2015 IEEE*. pp. 1–4, IEEE
142 Behrmann, J. *et al.* (2017) Deep learning for tumor classification in imaging mass spectrometry. *Bioinformatics* 34 (7), 1215–1223
143 Vinyals, O. *et al.* (2016) Matching networks for one shot learning. In *Advances in neural information processing systems*. 3630–3638
144 Altae-Tran, H. *et al.* (2017) Low data drug discovery with one-shot learning. *ACS Cent. Sci.* 3 (4), 283–293
145 Simoes, R.S. *et al.* (2018) Transfer and multi-task learning in QSAR modeling: advances and challenges. *Front Pharmacol.* 9, 74
146 Ahlberg, E. *et al.* (2017) Using conformal prediction to prioritize compound synthesis in drug discovery. In *The 6th Symposium on Conformal and Probabilistic Prediction with Applications,(COPA 2017), 13-16 June, 2017, Stockholm, Sweden*. 174–184
147 Shivakumar, D. *et al.* (2010) Prediction of absolute solvation free energies using molecular dynamics free energy perturbation and the OPLS force field. *J. Chem. Theory Comput* 6 (5), 1509–1519
148 Cao, Y. *et al.* (2018) Potential of quantum computing for drug discovery. *IBM J. Res. Dev.* 62 (6), 6 1-6: 20
149 Biamonte, J. *et al.* (2017) Quantum machine learning. *Nature* 549 (7671), 195
150 Lloyd, S. *et al.* (2013) *Quantum algorithms for supervised and unsupervised machine learning. arXiv preprint arXiv:1307.04112*013
151 Rebentrost, P. *et al.* (2014) Quantum support vector machine for big data classification. *Physical review letters* 113 (13), 130503
152 Wiebe, N. *et al.* (2014) *Quantum deep learning. arXiv preprint arXiv:1412.3489*2014