ELSEVIER

Drug Discovery Today: Technologies

Vol. 37, 2020

Editors-in-Chief
Henk Timmerman – VU Amsterdam, Amsterdam, Netherlands
György M. Keserű – Hungarian Academy of Sciences, Budapest, Hungary

DRUG DISCOVERY TODAY TECHNOLOGIES

Artificial Intelligence

A compact review of molecular property prediction with graph neural networks

Oliver Wieder[a], Stefan Kohlbacher[a], Mélaine Kuenemann[b], Arthur Garon[a], Pierre Ducrot[a], Thomas Seidel[a], Thierry Langer[a,*]

[a]University of Vienna, Department of Pharmaceutical Chemistry, Althanstrae 14, A-1090 Vienna, Austria
[b]Servier Research Institute – CentEx Biotechnology, 125 Chemin de Ronde, 78290 Croissy-sur-Seine, France

As graph neural networks are becoming more and more powerful and useful in the field of drug discovery, many pharmaceutical companies are getting interested in utilizing these methods for their own in-house frameworks. This is especially compelling for tasks such as the prediction of molecular properties which is often one of the most crucial tasks in computer-aided drug discovery workflows. The immense hype surrounding these kinds of algorithms has led to the development of many different types of promising architectures and in this review we try to structure this highly dynamic field of AI-research by collecting and classifying 80 GNNs that have been used to predict more than 20 molecular properties using 48 different datasets.

Introduction

The prediction of molecular properties is a fundamental task in the field of drug discovery. Computational methods for their accurate prediction can significantly accelerate the overall process of finding better drug candidates in a faster and cheaper way. This is especially compelling when considering that the average development cost for a new drug is currently estimated to be at around $2.8 billion [1]. The traditional in silico approach for predicting molecular properties has mainly relied on extracting fingerprints or hand-engineered features, which are then used in combination with machine learning algorithms. In an effort to capture features needed for the task at hand, these kinds of molecular representations are inherently biased by domain experts [2]. In order to move beyond this kind of bias to a more general approach, different types of machine learning algorithms have been introduced into the field of molecular property prediction. Especially deep-learning algorithms have seen a resurgence due to not only accelerating computational power, and increasing availability of large data sets but also due to its immense success in related fields such as natural language processing [3] and pattern recognition [4]. These kinds of networks are capable of learning representations in an automated way for a specific task and can therefore eliminate the complicated feature engineering process [5]. In order to use deep-learning algorithms and circumvent the domain specific feature engineering, an appropriate representation for molecules needs to be found. As molecules can be represented as graphs, one approach would be to simply use the molecular graph representation – which lead to the development of graph based neural networks (GNNs) which gained more and more attention and became increasingly popular in the last few years [6–8]. They seem to become one of the most promising deep-learning methods for graph specific tasks, especially due to their success in outperforming traditional machine learning methods in the prediction of quantum mechanics properties [9–12,5], physicochemical properties

*Corresponding author.: T. Langer (thierry.langer@univie.ac.at)

 https://doi.org/10.1016/j.ddtec.2020.11.009

like hydrophobicity [13–15] or the prediction of toxicity [16–19].

Due to the recent acceleration in publications related to molecular property prediction with GNN it often can be difficult to keep up with the current state of this field. For this reason, we provided this review that aims to give a comprehensive overview over the current status-quo of this rapidly developing area of research.

Our contribution is to give an overview over GNNs that have been utilized to predict one or more molecular properties (Fig. 1). We first introduce a neural network classification scheme similar to [6,7] and give a high-level method introduction of all 80 GNN architectures found in more than 63 publications. We then use a similar categorization as in [9] in order to define 5 general categorizations, namely Quantum Chemistry, Pysicochemical Properties, Biophysics, Biological Effect and Synthetic Accessibility, which consist of 20 different molecular properties and 48 different datasets. We then highlight which GNNs have been used in combination with one of them respectively. The overall structure of this survey is as follows:

- A high-level introduction to the different GNN categories used in this review (Section 'Graph neural networks').
- An overview over molecular property predictions with their corresponding GNNs (Section 'Molecular property prediction').

Disclaimer. The reader should be aware that this publication does not aim at providing an exact categorization for the molecular properties or giving a qualitative evaluation of what GNN performs better. Its main focus lies on giving an overview over the status-quo of what properties have been used in combination with GNNs. Otherwise this would be out of the scope of this short review.

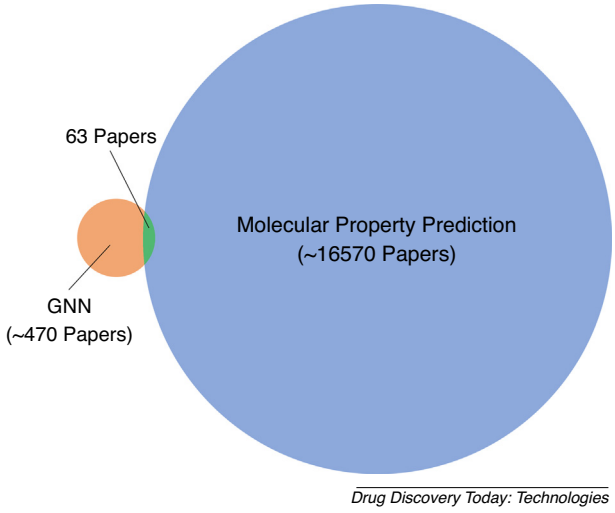


Fig. 1. Shows the proportion of GNN publications to molecular property prediction publications and the intersection of publications included in this survey.

Graph neural networks

This section provides a short introduction to graph neural-networks (GNN) and also outlines their categorization that we will refer to throughout this review. A detailed description of each distinct GNN is out of scope so we can only give a high-level introduction to the different approaches and refer to the corresponding publications listed in Tables 2 and 3 for further implementation details. Common notions and acronyms are given in Table 1 whereas the overall classification scheme is given in Table 2. Overall, we reviewed 80 distinct GNN architectures and split them into three different categories. The first two categories are based on their overall propagation type – namely recurrent graph neural networks (Rec-GNN 'Recurrent graph neural networks') and convolution graph neural networks (Conv-GNN 'Convolutional graph neural networks (Conv-GNN)'). All different variations are being described with respect to their most basic distinction. There exist several different types of networks within one GNN variation – this mostly comes from either using different initial node or edge featurizations, differences in what kind of features are being used during the aggregation (node and or the inclusion of edge features etc.) or additions to the described basic characteristics (GNNs that use a convolution aggregation in addition with some gated output function or attention mechanisms, etc.)

In addition to that we introduce a third category, namely distinct graph neural network architectures (Dist-GNN 'Distinct architectures'). The distinction made is not based on the propagation type, but this category rather consists of a collection of distinct graph based neural-network architectures (Section 'Distinct GNNs') that we wanted to highlight separately as well as possible architectural additions (Section 'Additions') to any kind of graph neural-network architecture like skip-connections, different pooling methods or attention mechanisms.

It is also important to note that not all references in Table 2 do point to the original GNN publication but to a publication which applied them to molecular property predictions. For example GraphSAGE [20] – it has been published in 2017 but Hamilton et al. [20] did not apply it on molecular property predictions. Errica et al. [21] and Hu et al. [19] however did so in 2019 and 2020 respectively. In this case we included both references in Table 3, each time GraphSAGE was used. However, in Table 2 we only included one reference as both refer to the same model, which can be looked up in either of these two publications.

Graphs

A graph in this review is defined as $G = (V, E)$, where V is a set of nodes and E denotes a set of edges. Let $v \in V$ be a

Table 1. Common notations used throughout this publication.

Notations	Definitions
$\odot$	Element-wise product
ρ	A non-linear function (e.g. sogmoid or relu)
$[,]$	Vector concatenation
t	Iterator of t steps
M^T	Matrix transpose
G	A mathematical graph
V	Set of nodes
E	Set of edges
v	Node $v \in V$
e_{uv}	Edge $e_{uv} \in E$ between node u and v
$N(v)$	Neighbors of node v
n	The number of nodes
m	The number of edges
d	The dimension of a node feature vector
b	The dimension of an edge feature vector
$X \in \mathbb{R}^{n \times d}$	Feature matrix of a graph
$x_v \in \mathbb{R}^d$	Feature vector of node v
$x^e_{uv} \in \mathbb{R}^b$	Feature vector of edge e_{uv}
$h_v \in \mathbb{R}^c$	Hidden feature vector of node v
$h^e_{uv} \in \mathbb{R}^a$	Hidden feature vector of edge e_{uv}
$H^{(t)} \in \mathbb{R}^{n \times h}$	Hidden feature matrix of all nodes at iteration t
$W^{(t)}$	Weight matrix of a neural network at iteration t
$A \in \{1,0\}^{\lvert n \rvert x \lvert n \rvert}$	(Unweighted) adjacency matrix
$D \in \mathbb{Z}^{\lvert n \rvert x \lvert n \rvert}$	Degree matrix. $D_{uu} = \sum_{v=1}^{n} A_{uv}$
A_{sym}	Symmetrically normalized A. $A_{sym} = D^{-1/2}AD^{-1/2}$

node with feature vector x_v and $e_{uv} \in E$ be an edge pointing from u to v with feature vector x^e_{uv}. The adjacency matrix A shows the connectivity of the nodes and is binary if the graph is unweighted. It is defined as a $n \times n$ matrix with $A_{uv} = 1$ if $e_{uv} \in E$ and $A_{uv} = 0$ if $e_{uv} \notin E$. The symmetrically-normalized adjacency matrix is defined as $A_{sym} = D^{-1/2}AD^{-1/2}$, where D is the degree matrix defined as $D \in \mathbb{Z}^{\lvert V \rvert x \lvert V \rvert}$. In general, molecular graphs are undirected, unweighted and mostly heterogeneous. A graph is undirected if and only if A is symmetric and where $e_{uv} = e_{vu}$ and undirected graphs are considered to be a special case of directed graphs, where $e_{uv} \neq e_{vu}$. Heterogeneous graphs contain different types of nodes and edges with their corresponding featurizations.

Learning approaches

There exist several different strategies for training GNNs. Depending on the task at hand and the available data, this can be done via supervised, unsupervised, semi-supervised or reinforcement learning. Typical tasks can include node, edge or graph classification, link prediction or graph regression.

Supervised learning can be utilized for different graph-level tasks such as node, edge or graph classification as well as graph regression tasks. The main objective in supervised learning is to reduce the loss between the predicted and the true value, which can be either a class label or a numeric value. Most common loss functions for classification tasks are cross-entropy and negative log-likelihood whereas for regression tasks often functions such as root mean squared error or root mean absolute error are being used.

Unsupervised learning is applied when no class labels are available. In such a case, the end-to-end learning can be done by, e.g. reconstructing the whole graph in order to learn a representation that contains the graph structure as well as information about it or by removing certain parts of the graph like nodes or edges and then predict them. Popular tasks include link prediction, node classification or representation learning for downstream tasks.

Semi-supervised learning includes both, labeled and unlabeled data and it is mostly used in the case when not enough labeled data is available. In such instances, the information contained by the graph representation is enriched via the unsupervised learning setting in order to perform better in the downstream supervised task.

Reinforcement learning is another learning approach that differs from supervised or unsupervised learning. The typical framework of RL consists of an environment, a set of possible actions that can be performed by an agent, a current state and a scoring function. Based on the current state, the agent performs an action which has an impact on the environment and leads to a new state. The performed action is being evaluated by the scoring function and depending on the reward, the agent learns whether this action in the current state of the environment leads to a higher reward or not.

Recurrent graph neural networks

Recurrent graph neural networks (Rec-GNNs) were among the first graph based neural networks to be utilized for molecular property prediction (Fig. 3) and their main difference to convolution based graph neural networks (Section 'Convolutional graph neural networks (Conv-GNN)') is how the information is being propagated. Rec-GNNs apply the same weight-matrices in an iterative way till an equilibrium is reached whereas Conv-GNNs apply different weights at each timestep t (Fig. 2). The earliest approaches of Rec-GNNs were based on directed, acyclic graphs [12] in supplementary material. Nevertheless, the first Rec-GNN to be utilized for molecular property predictions was introduced by Scarselli et al. [55] and they relaxed these constraints by also being capable of dealing with undirected and cyclic graph representations. They introduced the term graph neural networks (GNN) and applied a local transition function (also called

Table 2. Shows the categorization of all GNNs used for molecular property prediction in this review.

Graph neural network category	Variant	Approach	GNN-Name
Convolution graph-neural-network (Conv-GNN)	Spatial	GCN	ChemNet [22] GCN [23] NN4G [24] CNN [25] EAGCN [13] MGE-CNN [16] AGNN [14] GFN [26] GraphSAGE [21] MxPool [27] DGCNN [28] DCNN [29] Siamese-GCN [30] 3DGCN [31] ECC [29] InfoGraph [32] IterRefLSTM [30] CapsGNN [33] GCAPS-CNN [33] MGN [2] Patchy-San [34] Deep-LRP [35] SN-GCN [36] ExGCN [37] Att/Gate-GCN [38] GAT [19] PotentialNet [39] MT-PotentialNet [40] C-SGEN [14] attnLSTM [30] IGN [41]
		MPNN	MPNN [5] SELU-MPNN&E/AMPNN [17] D-MPNN [10] DiffPool [21] MV-GNN^{cross} [42] SAMPN [43] ASGN [44] GraphNet [45] DGGNN [46] R-GCN [47] GSN [48] OT-GNN [15] GCN+VN [49] CCN [41] GPNN [11]
	Spectral	–	LanczosNet [11] SpecConv [50] RGAT [51] AGCN [52] EigenGCN [53] PIN [48] SGC [14] ChebyNet [11]
Recurrent graph-neural-network (Rec-GNN)	Recurrent	RNN	UG-RNN [54] R-GNN [55] MGCN [1] in supplementary material GGRNet [12] IGNN [2] in supplementary material GPNN [11]
		GRU	E/AMPNN [17] DGGNN [46] MSGG [3] in supplementary material PotentialNet [39] MT-PotentialNet [40] GGNN [37]
		LSTM	IterRefLSTM [30] attnLSTM [30] GatedGCN [48]
Distinct GRAPH-NEURAL-NETWORK ARchitectures (Dist-GNN)	Distinct Approaches	Weisfeiler–Lehman	1-2-3-GNN [4] in supplementary material PPGN [41] 3WL-GNN [5] in supplementary material StructPool [18] GIN [6] in supplementary material GIN +VN [49] RP-GIN [7] in supplementary material
		Transformer	PAGTN [8] in supplementary material MAT [9] in supplementary material
	Additions	Attention	GAT [19] CapsGNN [33] Att/Gate-GCN [38] E/AMPNN [17] RGAT [51] IterRefLSTM [30] AGNN [14] SAGPool [10] in supplementary material MV-GNN^{cross} [42] SAMPN [43] ExGCN [37] MSGG [3] in supplementary material EAGCN [13] attnLSTM [30]
		Skip Connection	PAGTN [8] in supplementary material Att/Gate-GCN [38] GGRNet [12] ExGCN [37] C-SGEN [14] GatedGCN [48]
		Super-Node Pooling	SN-GCN [36] GWM [11] in supplementary material GIN+VN [49] GCN+VN [49] SortPool [28] SAGPool [10] in supplementary material StructPool [18] MxPool [27] EigenGCN [53] DiffPool [21] set2set [10] in supplementary material LRP [35] RP [7] in supplementary material gPool [10] in supplementary material

aggregation function) $M_w(\cdot)$ which updates the node's hidden feature vector $h_v^{(t)}$ at time t via

$$h_u^{(t)} = \sum_{v \in N(u)} \rho(M_w([x_u, x_{u,v}^e, x_v, h_u^{(t-1)}])) \quad (1)$$

where x_u and x_{uv}^e are the labels of current node and edge respectively, x_v are the neighboring node labels and $h_u^{(t-1)}$ is the $(t-1)$ hidden feature vector. $h^{(0)_u}$ is initialized randomly. The parametrization of the neural network $M_w(\cdot)$ is the same for all t. After t iterations, the output function $O_w(\cdot)$ takes the hidden features and generates the node's output vector $\hat{y}$.

It is defined as

$$\hat{y} = O_w\left(\sum_{v \in G} h_v^{(t)}\right) \quad (2)$$

These two functions can be seen as a vanilla Rec-GNN (RNN in Fig. 2). The two different states can be summed up as information diffusion function or aggregation function and output function.

More sophisticated Rec-RNN include aggregation functions that are similar to gated GNNs – two approaches include aggregation functions such as gated recurrent units (GRU [13] in supplementary material) or long-short-term memory (LSTM [14] in supplementary material) networks. Several different modifications of GRU and LSTM networks have been introduced and their vanilla approach can be defined for a GRU or LSTM aggregate as

$$\begin{aligned} h_u^{(t)} &= \textstyle\sum_{v \in N(u)} \mathrm{GRU}_w(h_u^{(t-1)}) \\ h_u^{(t)} &= \textstyle\sum_{v \in N(u)} \mathrm{LSTM}_w(h_u^{(t-1)}) \end{aligned} \quad (3)$$

Table 3. Summary of what GNN was used with which dataset and its corresponding category.

General category	Molecular properties	Dataset name	Prediction task	GNN name
Quantum chemistry	Coordinates (PES)	COD/CSD	Regression	DGGNN [46]
	Partial charges	Chembl Sub-DS	Regression	GraphNet [45]
	Energies	QM7	Regression	D-MPNN [10,42] GGRNet [12] MPNN [9,10,12] [8] in supplementary material [42] GCN [9,10,12] [8] in supplementary material [42] PAGTN [8] in supplementary material MV-GNN^{cross} [42] MGCN [42]
		QM8	Regression	GGRNet [12] MPNN [9–12,39,17] [8] in supplementary material [42] GCN [9–12,17] [8] in supplementary material [11,42] PAGTN [8] in supplementary material MV-GNN^{cross} [42] LanczosNet [11] GGNN [11] DCNN [11] PotentialNet [39] ChebyNet [11] GraphSAGE [11] E/AMPNN [17] D-MPNN [10,42] MGCN [42] GPNN [11]
		QM9	Regression	MPNN [5,9,35,10] [2,1] in supplementary material [12] [4,8] in supplementary material D-MPNN [10] GGRNet [12] CCN [41] GAT [2] in supplementary material GCN [9,10] [8] in supplementary material [12] [2,11] in supplementary material DGGNN [46] 1-2-3-GNN [4] in supplementary material [35] Deep-LRP [35] ChebyNet [2] in supplementary material ASGN [44] MGCN [1] in supplementary material IGNN [2] in supplementary material PAGTN [8] in supplementary material InfoGraph [44] R-GCN [2] in supplementary material GWM [11] in supplementary material GIN [2,11] in supplementary material LanczosNet [2] in supplementary material GGNN [2,1,11] in supplementary material RGAT [11] in supplementary material PPGN [35] DiffPool [41]
		OPV	Regression	ASGN [44] InfoGraph [44]
Physicochemical properties	Polar surface area	ZINC Sub-DS	Regression	Att/Gate-GCN [38] GCN [38]
		HPLC EPSA DS	Regression	MT-PotentialNet [40]
	Bioavailability	AMGEN-PXR-DS	Classification	ChemNet [22]
	Octanol solubility	Abrahams-DS	Regression	CNN [25]
	Aqueous solubility	Huuskonen	Regression	UG-RNN [54]
		Intrinsic solubility DS	Regression	UG-RNN [54]
		Solubility challenge DS	Regression	UG-RNN [54]
		ESOL	Regression	UG-RNN [54,25] [3] in supplementary material GCN [23,25,14,9,37,39,17,10] [3,8,9] in supplementary material [42] E/AMPNN [17]
	Aqueous solubility	ESOL	Regression	D-MPNN [10,15] PAGTN [8] in supplementary material OT-GNN [15] AGNN [14] PotentialNet [39] ExGCN [37] ChebyNet [52] SpecConv [52] GGNN [37] MPNN [17,42,14,9,10] [3,8] in supplementary material CNN [25] SGC [14] 3DGCN [31] AGCN [52] MV-GNN^{cross} [42] MSGG [3] in supplementary material C-SGEN [14] MAT [9] in supplementary material EAGCN [9] in supplementary material
		ZINC Sub-DS	Regression	GSN [48] GAT [48] [5] in supplementary material GIN [48] [5] in supplementary material GatedGCN [48] MPNN [5] in supplementary material [48] 3WL-GNN [5] in supplementary material GCN [5] in supplementary material GraphSAGE [5] in supplementary material
		OChem	Regression	SAMPN [43] MPNN [43]
	Boiling/melting point	Alkane DS	Regression	NN4G [24]
		Bradley-good DS	Regression	D-MPNN [10] MPNN [10] GCN [10] CNN [25]

Table 3 (*Continued*)

General category	Molecular properties	Dataset name	Prediction task	GNN name
		Hydrocarbon DS	Regression	ChemNet [22]
	Passive-membrane-perm.	P_app DS	Regression	MT-PotentialNet [40]
	Blood-brain-perm.	BBBP	Classification	D-MPNN [10,15,42] PAGTN [8] in supplementary material MAT [9] in supplementary material MPNN [17,9,42,10] [3,8] in supplementary material MGCN [42] MV-GNN^{cross} [42] E/AMPNN [17] MSGG [3] in supplementary material OT-GNN [15] EAGCN [9] in supplementary material GAT [19] GIN [19] GCN [19,9,17,42,10] [3,8,9] in supplementary material UG-RNN [3] in supplementary material GraphSAGE [19]
	Hydrophobicity	ZINC Sub-DS	Regression	Att/Gate-GCN [38] GCN [38]
		Az-logD DS	Regression	AGCN [52] ChebyNet [52] SpecConv [52]
		logD DS	Regression	MT-PotentialNet [40]
		LIPO DS	Regression	EAGCN [13] E/AMPNN [17] D-MPNN [10,15] GWM [11] in supplementary material AGNN [14] PT-GNN [15] PAGTN [8] in supplementary material MV-GNN^{cross} [42] MSGG [3] in supplementary material MPNN [17,42,14,9,10,43] [8] in supplementary material GIN [11] in supplementary material SAMPN [43] C-SGEN [14] ExGCN [37] GGNN [37] [11] in supplementary material GCN [14,9,37,42,17,10] [8,11] in supplementary material SGC [14] RGAT [11] in supplementary material
	Solvation free energy	FreeSolv	Regression	MPNN [14,9,10,42] [3] in supplementary material D-MPNN [10,42] EAGCN [13] [9] in supplementary material SN-GCN [36] AGNN [14] MV-GNN^{cross} [42] UG-RNN [3] in supplementary material GCN [36,14,9,10] [3] in supplementary material [42] [9] in supplementary material SGC [14] MAT [9] in supplementary material
	Solvation Free Energy	FreeSolv	Regression	MSGG [3] in supplementary material C-SGEN [14] MGCN [42] 3DGCN [31]
		HFE-DS	Regression	AGCN [52] ChebyNet [52] SpecConv [52]
	Metabolic stability	MetStab DS	Classification	MAT [9] in supplementary material GCN [9] in supplementary material EAGCN [9] in supplementary material
Biophysics	Affinity	PDBbind	Regression	D-MPNN [10] MPNN [10,14,9] [3] in supplementary material GCN [10,14,9] [3] in supplementary material MSGG [3] in supplementary material AGNN [14] C-SGEN [14] PotentialNet [39] UG-RNN [3] in supplementary material SGC [14]
	Efficacy	Gamo DS	Regression	GCN [23]
	Activity	NCI AIDS DS	Classification	MGN [2]
		ChEMBL Sub-DS	Classification	MPNN [10] GCN [10,8] D-MPNN [10]
		Estragon $\alpha\beta$ DS	Classification	MAT [9] in supplementary material GCN [9] in supplementary material EAGCN [9] in supplementary material
		DPP4 DS	Classification	SpecConv [50]
		PCBA DS	Classification	D-MPNN [10] SN-GCN [36] MPNN [9,10] GCN [36,9,10] GIN [49] GCN [49] GIN+VN [49] GCN+VN [49]
		MUV	Classification	SELU-MPNN&E/AMPNN [17] attnLSTM [30] Siamese-GCN [30] MPNN [17,9,10] SN-GCN [36] IterRefLSTM [30] D-MPNN [10] GraphSAGE [19] GAT [19] RP-GIN [7] in supplementary material GIN [19] GCN [19,9] [7] in supplementary material [36,17,10]
		HIV	Classification	EAGCN [13] E/AMPNN [17] D-MPNN [10] Deep-LRP [35] GAT [19,48,35] GWM [11] in supplementary material SN-GCN [36] GSN [48] MPNN [17,48,9,10] RGAT [11] in supplementary material

Table 3 (*Continued*)

General category	Molecular properties	Dataset name	Prediction task	GNN name
				GIN+VN [49] 3DGCN [31] GatedGCN [48] GCN+VN [49] RP-GIN [7] in supplementary material
		NCI1	Classification	GIN [19,48,35,49] [11] in supplementary material GCN [19,48,9] [7] in supplementary material [35,36,17,10,49] [11] in supplementary material GraphSAGE [19,35] DGCNN [21,48,26] [6] in supplementary material [27,33,53] [4,10] in supplementary material Patchy-San [34,26] [4,6] in supplementary material [29,33] SAGPool [10] in supplementary material [27] GraphSAGE [21,27] GSN [48] IGN [41,48] GIN [6] in supplementary material [21,26,27] PIN [48] 1-2-3-GNN [4] in supplementary material AGCN [52] ECC [29,21,33] CapsGNN [33,26] set2set [10] in supplementary material [53]
	Activity	NCI1	Classification	gPool [10] in supplementary material [27] EigenGCN [53] DiffPool [21,27,53] [10] in supplementary material SpecConv [52] MxPool [27] CCN [41] GCAPS-CNN [33] PPGN [41,48] DCNN [29,48] [6,4] in supplementary material GFN [26] ChebyNet [52]
		BACE	Classification	MPNN [10,9] [3,8] in supplementary material [42] GCN [10,9] [3] in supplementary material [19] [8,9] in supplementary material [42] D-MPNN [10,42,15] MV-GNN^{cross} [42] MSGG [3] in supplementary material GIN [19] 3DGCN [31] EAGCN [9] in supplementary material PAGTN [8] in supplementary material GraphSAGE [19,53] GAT [19] OT-GNN [15] MAT [9] in supplementary material MGCN [42]
Biological effect	Side effects	SIDER	Classification	SELU-MPNN&E/AMPNN [17] D-MPNN [10,42] IterRefLSTM [30] SpecConv [52] AGCN [52] MV-GNN^{cross} [42] GIN [19] MPNN [17,9,10,42] ChebyNet [52] MGCN [42] GCN [19,9,10,17,42] GraphSAGE [19] GAT [19] attnLSTM [30] Siamese-GCN [30]
	Toxicity	MUTAG DS	Classification	R-GNN [55] GIN [6] in supplementary material [26,18] RGAT [51] IterRefLSTM [51] DGCNN [28,26] [6] in supplementary material [33,53] Patchy-San [34,26] [6] in supplementary material [29,33] GSN [48] DCNN [29] [6] in supplementary material DiffPool [41,53] IGN [41] PPGN [41] PIN [48] ECC [29,33] InfoGraph [32] R-GCN [47] CNN [41] GCN [53] CapsGNN [33,26] EigenGCN [53] StructPool [18] GFN [26] GCAPS-CNN [33] set2set [53] GraphSAGE [53]
		Tox21	Classification	CNN [25] E/AMPNN [17] PotentialNet [39] attnLSTM [30] Siamese-GCN [30] GWM [11] in supplementary material RGAT [51] SN-GCN [36] IterRefLSTM [30,51] MPNN [17,9,10,42] Deep-LRP [35] AGCN [52] MV-GNN^{cross} [42] SpecConv [52] ChebyNet [52] RP-GIN [7] in supplementary material GIN [19] GCN [19,9] [7] in supplementary material [36,39,17,10,42] GAT [19] ExGCN [37] GGNN [37] GraphSAGE [19] D-MPNN [10,42] MGCN [42]
		ToxCast	Classification	MPNN [10,9] D-MPNN [10] SN-GCN [36] AGCN [52] SpecConv [52] ChebyNet [52] GIN [19] GCN [19,9,36,10] GraphSAGE [19] GAT [19]
		ClinTox	Classification	MPNN [10,9,42] D-MPNN [10,42] AGCN [52] MV-GNN^{cross} [42] SpecConv [52]

Table 3 (*Continued*)

General category	Molecular properties	Dataset name	Prediction task	GNN name
				GIN [19] GCN [19,9,10,42] GraphSAGE [19] GAT [19] MGCN [42] ChebyNet [52]
		PTC DS	Classification	GIN [6] in supplementary material [18] DGCNN [28] [6,4] in supplementary material Patchy-San [34][6,4] in supplementary material PPGN [41] DCNN [41] [6,4] in supplementary material
		PTC DS	Classification	InfoGraph [32] PIN [48] StructPool [18] IGN [41] 1-2-3-GNN [4] in supplementary material [41] ECC [41]
	Toxicity	AOT DS	Classification & Regression	MGE-CNN [16]
	ADMET	General ADMET DS	Regression	MT-PotentialNet [40]
Synthetic accessibility	Synthetic accessibility	ZINC-DS	Regression	Att/Gate-GCN [38] GCN [38]

where GRU_w or $LSTM_w$ is the aggregation function with the same parametrization for the whole neural network.

It is important to note that there exist neural networks in Table 2 that are both, Rec-GNN and Conv-GNN. This applies for, e.g. [17] because they use a MPNN approach in their E/AMPNN networks with a GRU update function (not aggregation) – same applies for [11] (GPNN) and [30] (IterRefLSTM&attnLSTM). Additionally, Feinberg et al. [39,40] both utilize PotentialNet, which uses GRU as an update function but different weightings for each edge type during the aggregation. Therefore, it is also in both categories. We also consider all LSTM and GRU approaches to be RNNs but not vice versa.

Convolutional graph neural networks (Conv-GNN)

The main difference to Rec-GNNs is that the aggregation function uses different weights for each timestep t or for each relational feature (relational graph neural networks [51,47] and [7] in supplementary material). Conv-GNNs can be further divided into two main categories – **spectral** and **spatial** Conv-GNNs.

For both types of Conv-GNNs, several different architectures have been developed (Table 2). The common denominator within spectral methods is that they are based on the eigendecomposition of the laplacian matrix L. For most spectral Conv-GNNs applications L is the symmetrically normalized (L_{sym}), which can then be defined as

$$L_{sym} = D^{1/2} L D^{1/2} \\ L = D - A \tag{4}$$

where D is the degree matrix. The eigendecomposition of the L can then be defined as

$$L = V \Lambda V^{(T)} \tag{5}$$

where Λ and V are eigenvalues and eigenvectors of L respectively. Their aggregation function is generally defined as

$$H^t = V(V^{(T)} H^{(t-1)} \odot V^{(T)} \Theta^{(t-1)}_{spectral}) \tag{6}$$

where $H^{(t)}$ is the hidden feature matrix at timestep t and $\Theta^{(t-1)}_{spectral}$ is a matrix of kernel parameters, which can be shared over the whole graph. The hidden feature matrix can be initialized as $H^0 = X \in \mathbb{R}^{n \times d}$. There exist several smaller and

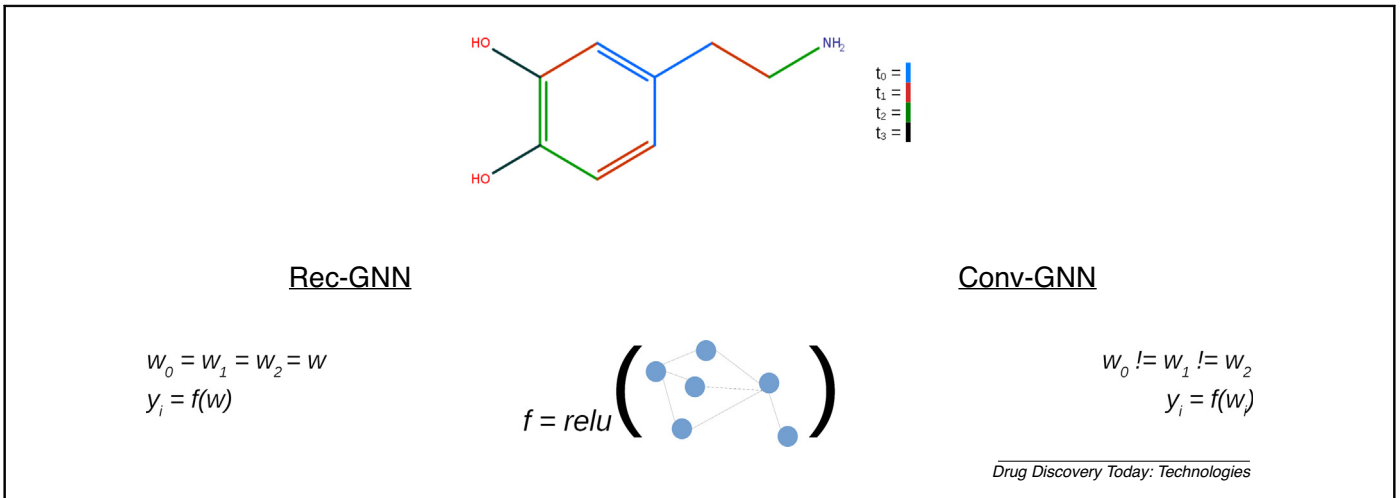
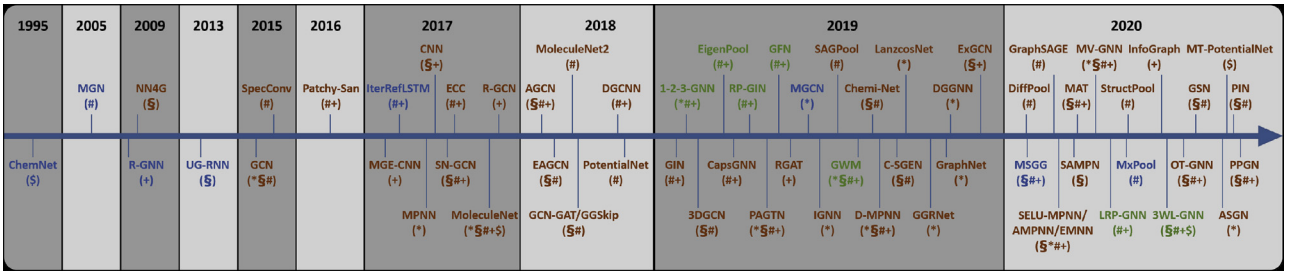


Fig. 2. Difference between recurrent (Rec-GNN) and convolution (Conv-GNN) based graph neural networks. Rec-GNN apply the same set of weights ($W_1 = W_2$) until a convergence criterion is met, whereas Conv-GNNs apply different weights at each iteration.



Drug Discovery Today: Technologies

Fig. 3. Timeline of GNNs for molecular property prediction with the corresponding categories used in this survey. * stands for the **quantum mechanical** property category, § is the **physicochemical** category, **#** is the **biophysics** category, **+** is the **biological effects** category and **$** stands for a collection of **datasets**

bigger deviations from this general approach, which can be seen in Table 2.

The advantage of spectral methods is that they are capable of capturing global patterns [52] in graphs whereas spatial Conv-GNNs only achieve that by either stacking several different aggregation layers [28] on top of each other or by using additional architectures like attention mechanisms [19]. Nevertheless, the main disadvantage of spectral methods is that $\Theta_{spectral}$ depends on the number of nodes n as well as the graph structure encoded in V, which makes it difficult to apply learned filters on graphs with different structures ([10] in supplementary material).

Spatial Conv-GNNs, on the other hand, work by adding up the adjacent local neighborhood of a node. Within this Conv-GNN variant, we distinguish two different approaches: The first one can be summarized under the name graph convolution networks (GCN [23]), whereas the second one is defined as message passing neural networks (MPNN [5]). The main difference between the two is that GCN in general comprises two phases; the aggregation and the readout phase, whereas the MPNN variant includes a message passing function $M^{(t)_{w_i}}(\cdot)$ – basically the aggregation function in GCN – an additional update function $U_{(t)_{w_k}}(\cdot)$ and the readout function $O_{w_j}(\cdot)$. Both perform the same type aggregation but call it differently. Overall the aggregation or message passing function involves multiplying the feature matrix $X \in \mathbb{R}^{n \times d}$ with different forms of the graph's A defined as

$$\begin{aligned} H^t &= \rho(AH^{(t-1)}W^{(t)}) \\ H^t &= \rho(\tilde{A}H^{(t-1)}W^{(t)}) \\ H^t &= \rho(\tilde{D}^{-1}\tilde{A}H^{(t-1)}W^{(t)}) \\ H^t &= \rho(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(t-1)}W^{(t)}) \end{aligned} \tag{7}$$

$H^{(t)}$ is the updated hidden feature matrix and $W^{(t)}$ is the weight matrix at the current time step. $\tilde{A}$ is the adjacency matrix of the graph combined with its identity matrix. Each of the equations in Eq. (7) is adding another complexity level. The first one is the most basic spatial Conv-GNN approach, which simply sums over the local neighborhood and updates the corresponding features. The second one does the same but also includes a self-loop. The third equation additionally includes a row-wise normalization of A and the fourth and last equation also utilize a symmetric normalization.

The aggregation and readout of a GCN in its simplest form can be defined as

$$\begin{aligned} h_u^{(t)} &= \textstyle\sum_{v \in N(u)} \rho(M_{w_i}^{(t-1)}(h_v^{(t-1)})) \\ \hat{y} &= O_{w_j}(\textstyle\sum_{v \in G} h_v^{(t)}) \end{aligned} \tag{8}$$

where $M_{w_i}^{(t-1)}(\cdot)$ is the aggregation and $O_{w_j}(\cdot)$ is the output function leading to the prediction $\hat{y}$. The simplest MPNN [5] can be defined with one addition as

$$\begin{aligned} m_u^{(t)} &= \textstyle\sum_{v \in N(u)} \rho(M_{w_i}^{(t-1)}(h_v^{(t-1)})) \\ h_u^{(t)} &= U_{w_k}^{(t-1)}([h_u^{(t-1)}, m_u^{(t-1)}]) \\ \hat{y} &= O_{w_j}\left(\textstyle\sum_{v \in G} h_v^{(t)}\right) \end{aligned} \tag{9}$$

where $m_u^{(t)}$ is the message coming from node u neighbors and $U_{w_k}^{(t-1)}$ is the update function, which is the main addition.

Distinct architectures

This section introduces several additions to the basic GNNs like different pooling strategies and skip-connections, which can be applied to any kind of GNN or architecturally distinct GNNs that should get highlighted separately and do not fully fit into the Conv-, Rec-GNN classification scheme described above.

Additions

Skip-connections are additions that take the input vector $h_u^{(t-1)}$ of $layer^{(t-1)}$ and concatenate it (or any other kind of multiplication) with the output vector $h_u^{(t)}$ – so they simply skip one layer at a time. By doing so, they introduce an alternative flow path for the gradient during the back-propagation step. This can be beneficial for the convergence of the model by mitigating the vanishing gradient problem [38,12].

Pooling, on the other hand, is a non-linear approach for down-sampling features [35] and can be done in different ways. Applying simple functions like taking the mean, maxi-

mum or sum in order to reduce the dimension of the feature vector can often lead to poor performance([6] in supplementary material). This is especially striking in smaller graphs as single nodes might have a strong influence on its overall property and with a pooling function that increases or decreases the information of that node, it can lead to poor performance([11] in supplementary material). This is why using different pooling strategies have been developed especially for molecular graphs – ranging from relational pooling (RP [7] in supplementary material) to graph Fourier transformations approaches (EigenPool [53]).

Attention mechanisms are another important addition to almost any GNN architecture (they can also be used as pooling operations [10] in supplementary material). By applying attention mechanisms, GNNs are capable of giving particular nodes or edges a higher weighting during the aggregation and therefore a higher impact on the predicted value. This means that it learns which kind of node [17], edge [43] or substructure [37] should have a higher impact for the current task at hand. In general there are several different types of attention mechanisms for graphs, but most of them calculate the normalized attention score $\alpha_{u,v}^{(t)}$ as follows

$$\begin{aligned} s_{u,v}^{(t)} &= \rho(F_w^{(t)}([h_u, h_v])) \\ \alpha_{u,v}^{(t)} &= \frac{\exp(s_{u,v})}{\sum_{v^* \in N(u)} \exp(s_{u,v^*})} \\ h_u^{(t+1)} &= \rho(\textstyle\sum_{v \in N(u)} \alpha_{(u,v)}^{(t)} h_v^{(t)}) \end{aligned} \tag{10}$$

where $s_{u,v}^{(t)}$ is the unnormalized attention score calculated by the function $F_w^{(t)}(\cdot)$. $h_u^{(t+1)}$ is then updated with its corresponding attention value $\alpha_{(u,v)}^{(t)}$ and its neighbour features – similar to GCN aggregation operations. The calculation of $\alpha_{(u,v)}^{(t)}$ is basically done via a softmax function and its values sum up to one, which can be interpreted as probabilities.

Super-nodes (SN) – also called virtual nodes (VN) in this review – are nodes that are not part of the molecular graph but can serve as an auxiliary module ([11] in supplementary material) which gathers information over the whole graph. This is especially helpful in the prediction of molecular properties that rely on the global structure of the graph. One way of doing this is by introducing a super node that is connected to all other nodes via directed edges, which does not affect the local propagation [36]. A more active approach has been applied by [11] in supplementary material for molecular property prediction. They added a SN that actively transmits information via longer distances using a MPNN with a gate and attention mechanism.

Distinct GNNs

Two distinct GNNs that were used for molecular property predictions are the transformer architecture and Weisfeiler–Lehman networks.

The **transformer** architecture introduced by [15] in supplementary material is strictly speaking not a GNN in the classical sense as it requires a sequence as input. Nevertheless, both publications cited in this review ([8,9] in supplementary material) circumvent this issue by introducing workarounds. Overall, the main contribution of the transformer network is its positional encoding in combination with a multi-head self-attention mechanism. The latter one is basically an extension of the above described attention mechanism and can be formulated with one addition as

$$h_u^{(t+1)} = \left[\rho \left(\sum_{v \in N(u)} \alpha_{(u,v)}^{t,k} h_v^{t,k} \right) \right]_k \tag{11}$$

where k is the number of heads that are being calculated within each iteration and then concatenated to get the final $h_u^{(t+1)}$.

Weisfeiler–Lehman networks (WL-GNNs) are a variant of neural networks that try to address the question of graph isomorphism. They are concerned with GNN and their expressiveness to distinguish between different types of graph structures in order to determine whether they are topologically the same or not. Overall these networks try to approach this problem by reformulating GNN by incorporating the WL hierarchy and be at least as expressive as the Weisfeiler–Lehman graph isomorphism test (WL-test [16] in supplementary material). The WL-test is based on iterative updating and recoloring of the nodes in the target graph till a stable equilibrium is reached. When two graphs have the same color, they are supposed to be isomorphic – this is however not always the case. The WL-test is quite similar to the standard MPNN ([6] in supplementary material). Nevertheless with respect to distinguishing non-isomorphic graphs, GNNs are at best similar, but not more powerful than the 1-WL-test ([4] in supplementary material). In order to get more expressive GNNs, different approaches were proposed. The reason for putting them into a distinct section was that it is especially important for molecular property prediction that the methods used are in theory able to distinguish between certain isomorphic graphs. Moreover, most standard GNNs such as vanilla MPNN or GraphSAGE are incapable to do so ([6] in supplementary material). [5] in supplementary material points out that GNN with more expressiveness does not necessarily lead to better results or are computational very expensive.

Molecular property prediction

For this review we mainly applied the classification scheme of [9] but extended it with a fifth category as well as several molecular properties with their corresponding datasets. These categories are defined by their general level of complexity. Nevertheless, several ambiguities between the different molecular properties and their assigned categories exist and, in several cases, one could argue for or against the carried out assignment. As stated in the disclaimer above, we do not

seek to find exact categorizations without any ambiguities for the different datasets but we want to provide a general overview over molecular property prediction using GNNs.

Table 3 shows the different types of general categories as well as their associated molecular properties in combination with the corresponding datasets, the type of task (regression or classification) and the GNN that has been used to predict them. Overall we included 20 different molecular properties split among 48 different datasets.

The first category is loosely based on quantum-mechanic properties and contains three molecular property sections namely coordinates, energies and partial charges comprising six datasets. The prediction of energies with 13 different GNN architectures makes the majority of this category, which results from the easy access to the QM7-QM9 datasets. Moreover, most of the networks within the QM category can be found in the Conv-GNN category – especially the MPNN approach with more than 6 out of 14 GNN architectures.

The physicochemical property category comprises 10 molecular properties, where aqueous solubility is the dominant property regarding available datasets. Other properties include polar surface area, bioavailability, octanol solubility, metabolic stability, boiling and melting point, hydrophobicity, solvation free energy, passive membrane permeability and blood brain permeability. For the prediction of aqueous solubility, 16 different GNN architectures are listed which is followed by hydrophobicity with 13 GNN and solvation free energy and blood-brain permeability prediction with 10 and 11 unique architectures, respectively. The majority of networks in the physicochemical property category are Conv-GNNs. More than 13 out of 21 GNN architectures in this category are based on the GCN approach. This is being followed by the MPNN approach with 8 distinct architectures.

The biophysics category covers three molecular properties – affinity, efficacy and activity. Activity is a very vague category and leaves a lot of room for interpretation regarding the nine different datasets included.

Architecture wise, this category includes almost all GNNs shown in Table 2 with a total number of 58 distinct networks. Most of them are from the Conv-GNN category with 21 architectures alone from the Conv-GNN GCN and another 8 from the MPNN approach. Rec-GNNs account for 7 different architectures. The NCI1 dataset was used with 25 different architectures and is therefore the most used one followed by the HIV dataset with 21 architectures.

The biological effect category includes three molecular property sub-categories, namely side effects, toxicity and ADMET. Toxicity is the category with six datasets. In this category, the Tox21 and MUTAG datasets are the ones which have been used in combination with 24 GNN architectures. ClinTox was used by 12 followed by ToxCast and the PTC dataset with both 11 different architectures. Overall 35 different architectures have been used of which 22 were from the Conv-GNN GCN variant, which have been applied throughout all dataset. The second most used variant is the MPNN with 8 architectures used in 6 datasets closely followed by the spectral GCN and Rec-GNNs with both 7 different architectures.

Conclusion

Graph neural networks have seen an immense acceleration in the field of drug discovery – especially for the prediction of molecular properties. In this survey, we reviewed 63 different publications, categorized 80 different GNNs approaches according to their underlying architectures and gave a comprehensive overview over which of the 20 molecular property categories, split into 48 datasets, have been predicted with the reviewed GNN setups.

Reference note. Due to limitations in the number of allowed citations we continue the list of references in the supplementary material section (while fully recognizing and agreeing with [17] in supplementary material).

Conflict of interest

There are no conflicts of interest or disclosures associated with this manuscript.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.ddtec.2020.11.009.

References

[1] Yang K, Swanson K, Jin W, Coley C, Eiden P, Gao H, et al. Analyzing learned molecular representations for property prediction. J Chem Inform Model 2019;59(8):3370–88. http://dx.doi.org/10.1021/acs.jcim.9b00237.
[2] Merkwirth C, Lengauer T. Automatic generation of complementary descriptors with molecular graph networks. J Chem Inform Model 2005;45(5):1159–68. http://dx.doi.org/10.1021/ci049613b.
[3] Young T, Hazarika D, Poria S, Cambria E. Recent trends in deep learning based natural language processing [review article]. IEEE Comput Intell Mag 2018;13(3):55–75. http://dx.doi.org/10.1109/MCI.2018.2840738.
[4] Bhamare D, Suryawanshi P. Review on reliable pattern recognition with machine learning techniques. Fuzzy Inform Eng 2018;10(3):362–77. http://dx.doi.org/10.1080/16168658.2019.1611030.
[5] Gilmer J, Schoenholz SS, Riley PF, Vinyals O, Dahl GE. Neural message passing for quantum chemistry; 2017. http://dx.doi.org/10.1002/nme.2457.
[6] Wu Z, Pan S, Chen F, Long G, Zhang C, Yu PS. A comprehensive survey on graph neural networks; 2019, arXiv:1901.00596.
[7] Zhou K, Dong Y, Lee WS, Hooi B, Xu H, Feng J. Effective training strategies for deep graph neural networks; 2020;p. 1–26, arXiv:2006.07107.
[8] Mayr A, Klambauer G, Unterthiner T, Steijaert M, Wegner JK, Ceulemans H, et al. Large-scale comparison of machine learning methods for drug target prediction on ChEMBL. Chem Sci 2018;9(24):5441–51. http://dx.doi.org/10.1039/c8sc00148k.
[9] Wu Z, Ramsundar B, Feinberg EN, Gomes J, Geniesse C, Pappu AS, et al. MoleculeNet: a benchmark for molecular machine learning; 2018, arXiv:1703.00564v3 [cs. LG].

[10] Yang K, Swanson K, Jin W, Coley C, Eiden P, Gao H, et al. Analyzing learned molecular representations for property prediction. J Chem Inform Model 2019;59(8):3370–88. http://dx.doi.org/10.1021/acs.jcim.9b00237.
[11] Liao R, Zhao Z, Urtasun R, Zemel RS. LanczosNet: multi-scale deep graph convo-lutional networks. 7th international conference on learning representations ICLR 2019;1–18, arXiv:1901.014842019v2.
[12] Shindo H, Matsumoto Y. Gated graph recursive neural networks for molecular property prediction; 2019, arXiv:1909.00259.
[13] Shang C, Liu Q, Chen K-S, Sun J, Lu J, Yi J, et al. Edge attention-based multi-relational graph convolutional networks; 2018, arXiv:1802.04944.
[14] Wang X, Li Z, Jiang M, Wang S, Zhang S, Wei Z. Molecule property prediction based on spatial graph embedding. J Chem Inform Model 2019;59(9):3817–28. http://dx.doi.org/10.1021/acs.jcim.9b00410.
[15] Bécigneul G, Ganea O-E, Chen B, Barzilay R, Jaakkola T. Optimal transport graph neural networks; 2020, arXiv:2006.04804.
[16] Xu Y, Pei J, Lai L. Deep learning based regression and multiclass models for acute oral toxicity prediction with automatic chemical feature extraction. J Chem Inform Model 2017;57(11):2672–85. http://dx.doi.org/10.1021/acs.jcim.7b00244.
[17] Withnall M, Lindelöf E, Engkvist O, Chen H. Building attention and edge message passing neural networks for bioactivity and physical-chemical property prediction. J Cheminformatics 2020;12(1). http://dx.doi.org/10.1186/s13321-019-0407-y.
[18] Yuan H, Ji S. Structpool: structured graph pooling via conditional random fields. International conference on learning representations 2020, https://openreview.net/forum?id=BJxg_hVtwH.
[19] Hu W, Liu B, Gomes J, Zitnik M, Liang P, Pande VS, et al. Pre-training graph neural networks; 2019, arXiv:1905.12265.
[20] Hamilton WL, Ying R, Leskovec J. Inductive representation learning on large graphs. Advances in neural information processing systems 2017-December (Nips) 2017;1025–35, arXiv:1706.02216.
[21] Errica F, Podda M, Bacciu D, Micheli A. A fair comparison of graph neural networks for graph classification; 2019;p. 1–14, arXiv:1912.09893.
[22] Kireev DB, Chemnet:. A novel neural network based method for graph/property mapping. J Chem Inform Comput Sci 1995;35(2):175–80. http://dx.doi.org/10.1021/ci00024a001.
[23] Duvenaud D, Maclaurin D, Aguilera-Iparraguirre J, Gómez-Bombarelli R, Hirzel T, Aspuru-Guzik A, et al. Convolutional networks on graphs for learning molecular fingerprints. Advances in neural information processing systems 2015-January 2015;2224–32, arXiv:1509.09292.
[24] Micheli A. Neural network for graphs: a contextual constructive approach. IEEE Trans Neural Netw 2009;20(3):498–511. http://dx.doi.org/10.1109/TNN.2008.2010350.
[25] Coley CW, Barzilay R, Green WH, Jaakkola TS, Jensen KF. Convolutional embedding of attributed molecular graphs for physical property prediction. J Chem Inform Model 2017;57(8):1757–72. http://dx.doi.org/10.1021/acs.jcim.6b00601.
[26] Chen T, Bian S, Sun Y. Are powerful graph neural nets necessary? A dissection on graph classification; 2019, arXiv:1905.04579.
[27] Liang Y, Zhang Y, Gao D, Xu Q. MxPool: multiplex pooling for hierarchical graph representation learning; 2020, arXiv:2004.06846v1.
[28] Zhang M, Cui Z, Neumann M, Chen Y. An end-to-end deep learning architecture for graph classification. 32nd AAAI conference on artificial intelligence 2018;2018:4438–45.
[29] Simonovsky M, Komodakis N. Dynamic edge-conditioned filters in convolutional neural networks on graphs; 2017, arXiv:1704.02901.
[30] Altae-Tran H, Ramsundar B, Pappu AS, Pande V. Low data drug discovery with one-shot learning. ACS Central Sci 2017;3(4):283–93. http://dx.doi.org/10.1021/acscentsci.6b00367.
[31] Cho H, Choi IS. Three-dimensionally embedded graph convolutional network (3DGCN) for molecule interpretation; 2018;p. 1–39. http://dx.doi.org/10.1002/cmdc.201900458.
[32] Sun F-Y, Hoffmann J, Verma V, Tang J. Infograph: unsupervised and semi-supervised graph-level representation learning via mutual information maximization; 2020, arXiv:1908.01000.
[33] Xinyi Z, Chen L. Capsule graph neural network. International conference on learning representations no 2018 2019;1–16, https://openreview.net/forum?.id=Byl8BnRcYm.
[34] Niepert M, Ahmad M, Kutzkov K. Learning convolutional neural networks for graphs. 33rd international conference on machine learning vol 4 2016;2958–67. arXiv:1605.05273.
[35] Chen Z, Chen L, Villar S, Bruna J. Can graph neural networks count substructures?; 2020, arXiv:2002.04025.
[36] Li J, Cai D, He X. Learning graph-level representation for drug discovery; 2017, arXiv:1709.03741.
[37] Meng M, Wei Z, Li Z, Jiang M, Bian Y. Property prediction of molecules in graph convolutional neural network expansion. Proceedings of the IEEE international conference on software engineering and service sciences ICSESS 2019-October 2019;263–6. http://dx.doi.org/10.1109/ICSESS47205.2019.9040723.
[38] Ryu S, Lim J, Hong SH, Kim WY. Deeply learning molecular structure-property relationships using attention- and gate-augmented graph convolutional network; 2018. http://dx.doi.org/10.1039/b000000x/been.
[39] Feinberg EN, Sur D, Wu Z, Husic BE, Mai H, Li Y, et al. PotentialNet for molecular property prediction. ACS Central Sci 2018;4(11):1520–30. http://dx.doi.org/10.1021/acscentsci.8b00507.
[40] Feinberg EN, Joshi E, Pande VS, Cheng AC. Improvement in ADMET prediction with multitask deep featurization. J Med Chem 2020;63(16):8835–48. http://dx.doi.org/10.1021/acs.jmedchem.9b02187.
[41] Maron H, Ben-Hamu H, Serviansky H, Lipman Y. Provably powerful graph networks; 2019, arXiv:1905.11136.
[42] Ma H, Bian Y, Rong Y, Huang W, Xu T, Xie W, et al. Multi-view graph neural networks for molecular property prediction; 2020, arXiv:2005.13607.
[43] Tang B, Kramer ST, Fang M, Qiu Y, Wu Z, Xu D. A self-attention based message passing neural network for predicting molecular lipophilicity and aqueous solubility. J Cheminformatics 2020;12(1):1–9. http://dx.doi.org/10.1186/s13321-020-0414-z.
[44] Hao Z, Lu C, Huang Z, Wang H, Hu Z, Liu Q, et al. ASGN: an active semi-supervised graph neural network for molecular property prediction; 2007, arXiv:2007.03196v1.
[45] Wang Y, Fass J, Stern CD, Luo K, Chodera J. Graph nets for partial charge prediction; 2019, arXiv:1909.07903.
[46] Mansimov E, Mahmood O, Kang S, Cho K. Molecular geometry prediction using a deep generative graph neural network. Sci Rep 2019;9(1):1–13. http://dx.doi.org/10.1038/s41598-019-56773-5.
[47] Schlichtkrull M, Kipf TN, Bloem P, van den Berg R, Titov I, Welling M. Modeling relational data with graph convolutional networks. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics) 10843 LNCS (1) 2018;593–607. http://dx.doi.org/10.1007/978-3-319-93417-4_38.
[48] Bouritsas G, Frasca F, Zafeiriou S, Bronstein MM. Improving graph neural network expressivity via subgraph isomorphism counting; 2020, arXiv:2006.09252.
[49] Hu W, Fey M, Zitnik M, Dong Y, Ren H, Liu B, et al. Open graph benchmark: datasets for machine learning on graphs; 2020, arXiv:2005.00687v4 [cs.LG].
[50] Henaff M, Bruna J, LeCun Y. Deep convolutional networks on graph-structured data; 2015, arXiv:1506.05163.
[51] Busbridge D, Sherburn D, Cavallo P, Hammerla NY. Relational graph attention networks; 2019;p. 1–21, arXiv:1904.05811.
[52] Li R, Wang S, Zhu F, Huang J. Adaptive graph convolutional neural networks. 32nd AAAI conference on artificial intelligence 2018;2018:3546–53, arXiv:1801.03226.
[53] Ma Y, Wang S, Aggarwal CC, Tang J. Graph convolutional networks with eigenpooling; 2019, arXiv:1904.13107.
[54] Lusci A, Pollastri G, Baldi P. Deep architectures and deep learning in chemoinformatics: the prediction of aqueous solubility for drug-like molecules. J Chem Inform Model 2013;53(7):1563–75. http://dx.doi.org/10.1021/ci400187y.
[55] Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G. The graph neural network model. IEEE Trans Neural Netw 2009;20(1):61–80. http://dx.doi.org/10.1109/TNN.2008.2005605.