Graph Neural Networks for Drug Discovery: An Integrated Decision Support Pipeline

Conference Paper · October 2023

DOI: 10.1109/MetroXRAINE58569.2023.10405789

1 author:

Pietro Bongini
University of Siena
32 PUBLICATIONS **412** CITATIONS

SEE PROFILE

2023 IEEE International Conference on Metrology for eXtended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE) | 979-8-3503-0080-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/METROXRAINE58569.2023.10405789

Graph Neural Networks for Drug Discovery: an integrated decision support pipeline

1st Pietro Bongini
Department of Information Engineering and Mathematics
University of Siena
Via Roma, 56 - Siena, 53100, Italy
ORCID: 0000-0001-9074-0587

***Abstract*—Drug Discovery is a fundamental discipline that is needed to produce new pharmacological solutions to the ever-evolving challenges of healthcare systems. Obtaining a new drug is a long and costly process, often involving many institutions and companies. In recent years, many proposals of support systems based on deep learning have been devised, in order to integrate the traditional methods and cut down the costs, in terms of time and money, to obtain a new approved drug. These deep learning based methods can help in different stages of the process, from the proposal of drug candidates to the evaluation of how to deploy them on the market. In the first stages of the process, where the molecule gets designed and undergoes several chemical evaluations, many support tools based on Graph Neural Networks have been proposed. Graph Neural Networks are deep learning models and have been demonstrated to be universal approximators on graphs, a characteristic which represents a great advantage with respect to other deep learning methods when dealing with molecular graphs. Graph Neural Networks have been successfully employed in molecular graph generation, drug side–effect and polypharmacy prediction, and in many chemical classification and regression tasks. This makes them the ideal model to build a drug–discovery support pipeline which can assist in the first stages of the drug discovery process: designing the molecular structure of candidate drugs, evaluating their chemical characteristics, predicting their probable side–effects. This paper is intended as a proof of concept of this pipeline, providing its outline and highlighting the future challenges in this direction.**

***Index Terms*—Graph Neural Networks, Drug Discovery, AI-aided Drug Discovery, Neural Networks, Molecular Graph Generation.**

I. Introduction

Use of prescription drugs is increasing all around the World [1]. Drug Discovery is fundamental to develop new therapies, find new cures for difficult diseases, propose new molecules that treat health issues with less side–effects, and face the ever–evolving challenges of healthcare. The process of Drug Discovery is expensive in terms of time and money: it takes several years to transform a new molecule into a commercial drug and many obstacles can hinder the effort [2]. The development of a new drug from target identification to commercialization can be seen as a pipeline, with each step of the pipeline being a problem to be solved. With the discovery of each new drug, the discovery of future new drugs becomes a little more difficult. As a consequence, and also due to the increasing difficulty of the challenges tackled by drug research, the pipeline keeps becoming longer, more expensive, and filled with more obstacles [3].

The increasing capabilities of information technology have provided solutions to counter this trend (in particular, in the *in silico* phase). The collection of information technology methods developed to help drug discovery is known as Computer–Aided Drug Discovery [4]. Artificial Intelligence (AI) is certainly gaining space and weight on a daily basis [5]. The potential of AI, and in particular of Deep Learning (DL), is astonishingly vast and yet to be fully determined. DL has already provided many valuable methods that increase the efficiency of drug discovery efforts [6].

Graph Neural Networks (GNNs) were first invented in 2008 [7] and have since known a great development, with a steep increase in the last five years [8]. GNNs are universal approximators on graphs and, thanks to their capabilities [9], they are helping solve many graph–based problems in almost every field of Science [10], including Drug Discovery [11]. Various GNN models have been developed adapting to this great variety of scenarios. They can be classified into two broad families: Convolutional GNNs, such as Graph Convolution Networks (GCNs) [12], GraphSAGE [13], GraphNets [14], and GAT [15]; and Recurrent GNNs, such as the original model [7], Message Passing Neural Networks (MPNNs) [16], and Graph Isomorphism Networks (GINs) [17].

GNNs have been frequently employed in biomedical applications, with increasing success in many problems inspired by the real World. Just to make a few examples, they were employed in the prediction of mutagenicity [18], classification of counter–HIV activity [19], and the creation of a support network for caregivers [20]. We use the most recent implementation of the original GNN model: GNN–keras [21], which has been applied to the prediction of protein–protein interactions [22], the prediction of drug side–effects [23], [24], and the generation of molecular graphs for drug discovery purposes [25].

This work is a proof of concept of a GNN–based pipeline for AI–assisted drug discovery, focused on the early *in silico* stages. The objective of the system is to provide a pool of promising drug candidates, some of which could become future drug molecules. Each of the steps will be discussed together with its objectives and the GNN methodology to realize it. To the best of our knowledge, this is the first attempt to create one such pipeline based on GNNs. Moreover, the

979-8-3503-0080-2/23/$31.00 ©2023 IEEE

only similar AI–based pipelines proposed so far are sketches based on generative models, such as the ones proposed for MoFlow [26] and Chemformer [27]. MoFlow is an invertible–flow molecular generator, while Chemformer is a SMILES–based generator based on transformers. The functions of these pipelines are often limited to the generation of molecules and the analysis of their synthesizability. The rest of the paper is organized as follows: Section II describes the pipeline; Section III sketches the implementation, explains the level of progress of each step and the expected path to realize what is left; Section IV gives an outline of the expected data flow; Finally, Section V draws the conclusions.

II. Methodology

The purpose of this work is to design a GNN–based pipeline for AI–assisted drug discovery. This is not intended as a way to fully automatize the first stages of drug discovery, it is instead a tool to help experts speed–up the process and increase the availability of molecular structures among which the future drug candidates could be selected. The pipeline will be structured in four main steps, each of which will be analyzed in a dedicated subsection:

- Molecular Graph Generation (Subsection II-A),
- Sinthesizability Evaluation (Subsection II-B),
- Drug Side–Effect Prediction (Subsection II-C),
- Evaluation of the Molecular Properties (Subsection II-D).

Figure 1 gives a graphical outline of the described pipeline. Please notice that implementing each step with GNNs is not mandatory. Yet, the use of GNNs throughout the whole pipeline is the main strong point of this proposal, thanks to their capabilities in dealing with relational data. Moreover, there is no need to reformat data between steps, as all the modules can naturally process the molecular graphs.

A. Molecular Graph Generation

Drug Design is the phase where the molecular structure of a candidate drug is outlined. The structure is naturally described by a molecular graph. Since this requires a relevant amount of time and money for each candidate molecule, computational methods are increasingly employed to help designing the molecular graphs. Deep learning methods are very suited for this task as they can be programmed to generate molecules. It is even possible to give the desired characteristics in input to the model [28].

Molecular graph generators have known a rapid development thanks to the introduction of deep learning. This is a complex task, with discrete processes to be taken into account when designing algorithms and models. The first deep learning techniques for molecular structure generation were based on one–dimensional convolutional networks or recurrent neural networks [29]. Variational AutoEncoders (VAEs) can instead generate the molecules in one single shot [30]. While the early models processed the SMILES strings that describe the structure, newer ones have focused on the SMILES grammar [31], and later on the molecular graphs [32].

On one hand, sequential methods can either work on the sequence of steps or on the sequence of decisions. The former are mainly based on recurrent neural networks [33], while the latter are based on reinforcement learning [34] or GAN–like mechanisms [35]. On the other hand, early VAEs generated the SMILES string of the molecule, while advanced models generate the molecular graph. For instance, GraphVAE generates the adjacency, node feature, and edge feature matrices [32]. Recently, these models have been hybridized with MPNNs, introducing some message passing in the encoder/decoder to better capture structural information [36].

We developed a method based on GNNs for the generation of molecular graphs, called MG^2N^2. For a more detailed description of the model and its capabilities, please see the corresponding publication [25].

B. Synthesizability Evaluation

Synthesizability is key in determining which generated molecules could become drug candidates. Compounds for which the synthesis is too complex and expensive or which cannot be synthesized at all cannot obviously become commercial drugs. In this scope, it is useful to analyze the reactions that are needed and to know the list of reactants involved. This analysis also allows to determine how difficult a molecule is to synthesize. With traditional Drug Discovery techniques, this step is embedded in the drug design process: the molecules are usually thought of as variants of known molecules, and their synthesis process is consequently a modification of the synthesis of the known similar molecule. With the development of automatic methods for molecular structure generation, that can output many thousands of unknown molecular structures, it has become necessary to develop tools that can evaluate the synthesizability of these generated molecules. Given the quantity of compounds produced, the tools for their evaluation also need to be fully automatic.

This problem can be addressed with two main approaches. On one hand, generative methods can be built in an explainable way [37], naturally following the possible reaction paths, and keeping track of the reactants involved in the synthesis [38]. Another similar idea is to explore the tree of known chemical reactions, in which nodes represent reactants and edges represent reactions [39]. This can also be done by applying a Bayesian Optimization method on the graph of known reactions [40]. On the other hand, synthesizability can be determined after the generation, for instance with ad–hoc heuristic methods [41]. This approach has been explored also with GNNs, proposing a retrosynthesis simulator [42]. In principle, any post–hoc synthesizability module can be used in this phase.

C. Drug Side–Effect Prediction

Drug Side–Effects (DSEs) cause millions of hospitalizations and about 200,000 deaths each year [43]. These numbers are constantly increasing, following the trend of prescription drug use [1]. Drug side–effects are also a major obstacle to the discovery of new drugs, because they can prevent promising molecules from becoming valuable cures. This typically

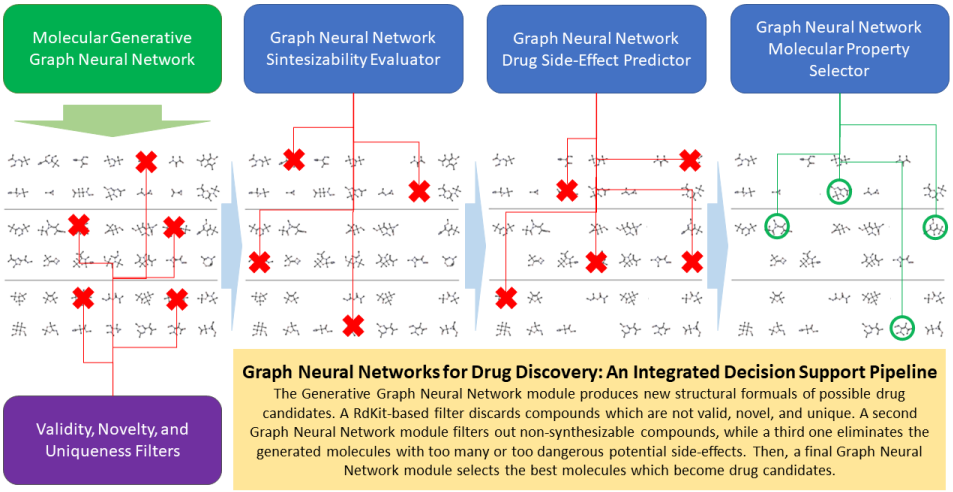

Fig. 1. Graphical outline of the pipeline. The main steps are four, each addressed with a GNN–based method. An additional filtering step, based on RdKit, is used as a first filter for generated molecules.

happens during the last stages of the process, after years of work and investments [44]. A reliable method for the automatic prediction of DSEs would identify the molecules with dangerous DSE profiles *in silico* before performing studies and trials on them. Additionally, an efficient DSE predictor is fundamental for screening the compounds generated by an AI method. In our pipeline, a DSE predictor should filter out all the compounds with too many or too dangerous DSEs. This module actually works only on those compounds which have been identified as synthesizable by the previous module.

Side–effect predictors have been developed starting from relatively simple methods based on Euclidean data [45], to then integrate drug similarity measures [46], and an increasing variety of heterogeneous data sources. The real turning point has been the application of Machine Learning (ML) techniques [47], [48]. Predictors have become more accurate by combining data from different sources and exploiting methods capable of processing them, like Random Forests (RFs) [49] and, finally, deep MLPs [50]. DSEs are complex biological phenomena, involving many different entities such as human metabolism, protein–protein interactions, drug–protein interactions, and the characteristics of each of these entities. We developed a first method for DSE prediction, called DruGNN [23], which integrates data from seven sources in one graph. Drug nodes are classified according to their DSEs. The findings of this work have led to the development of two other approaches based on GNNs. More details are reported in Section III.

D. Evaluation of Molecular Properties

Properties of the selected molecules can be tuned to the objectives of the single studies. There exist two principal ways of obtaining the molecular characteristics desired by the drug discovery experts for a given task. On one hand, molecules can be generated conditionally, training the generator to create structures based on features given as input. This can be done by expanding our previous work on MG^{2}N^2: A variant of the model will be developed that takes molecular features in input and learns to generate molecules based on them. Another interesting idea would be to develop an autoencoder–like model that uses the desired features both as input of the model and as a reference to calculate the loss. This can be done by calculating the same characteristics on the generated molecule and computing the loss as the difference between them and the desired values. On the other hand, the molecules with the best characteristics can be selected from a pool of previously generated compounds, coming from a model without input conditioning. In this setup, a GNN could be trained to select the best molecules given the objective. In some cases, this step could also be done by simply calculating the chemical features of the generated molecules with the RDKit package and selecting the molecules which are closer to the objective.

III. Realization

The proof of concept of the pipeline sketched in Section II has already been partially realized, with some preliminary results. We will account for the current state of realization in the present Section, which will also draw a sketch of how to implement the remaining parts, with ideas for future work and integration of the pipeline steps. For the implementation of every GNN model placed into the pipeline, we use the GNN–keras framework [21], which allows to easily define and train a GNN model in few lines of code, managing the dataset with ad–hoc structures and optimized processing functions. A sketch of how a graph is processed by the recurrent GNN is shown in Figure 2.

The GNN processes the input graph by replicating its structure in the architecture. This is done by using MLPs as building blocks. A state updating network is replicated on every graph node, obtaining an encoding network, which is replicated once for every state updating iteration, allowing for message passing

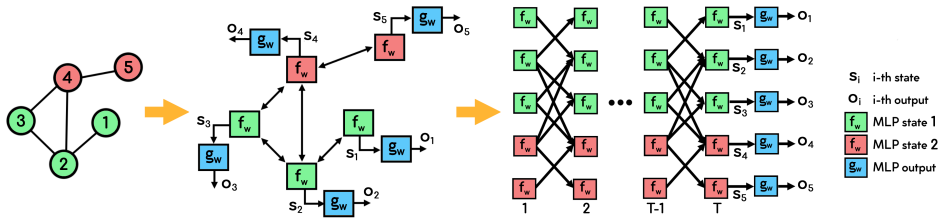

Fig. 2. Sketch of how the GNN processes an input graph. A state updating MLP is placed upon each of the nodes. If the graph is heterogeneous, a different MLP is used for each node type. Another MLP acts as the output network, calculating the output based on the node states. This figure is taken from [21].

to occur. This produces the unfolded encoding network, on which backpropagation works. An output network calculates the output of the GNN based on the node states after the last iteration, where needed. The output can be calculated on nodes, edges, or whole graphs depending on the problem. If the input graphs are heterogeneous, a variant of the model, called Composite Graph Neural Network (CGNN) can be employed. In this case, we have a different state updating network for each node type. The different MLPs can exchange information during the message passing process thanks to the common dimension of the node states.

Molecular graph generation has been addressed by various models, cited in Section II-A, most of which can be inserted in the first spot of our pipeline regardless of the steps that will follow. In particular, we will cover the realization proposed in MG^2N^2 [25]. We chose this model because it is the only one (to the best of our knowledge) exclusively based on recurrent GNNs developed so far. It is composed of three GNN modules, that generate the molecule with a sequential algorithm. The first GNN generates nodes, deciding if they exist and, in case, their type. The nodes are expanded (generating their neighbors) following the order in which they were generated, with a breadth–first approach. The second GNN decides the type of edge (chemical bond) connecting the newly generated neighbor to the atom node being expanded. The third GNN generates loops between the newly generated atom and the rest of the graph. All the modules are equipped with a Gumbel softmax output layer to learn and apply a stochastic behaviour. This algorithm has been proven to be very competitive for drug–like molecule generation on QM9 [51] and Zinc [52], showing that GNNs can be efficiently used to generate novel, valid, and original compounds [25]. A first filtering step can be applied immediately after the generation, using RdKit, deleting the compounds which are not chemically valid, are already known, or represent duplicates of already generated molecules. The main limitation of this approach is that it cannot be driven towards a specific objective, an issue which can be solved by implementing a conditional version of the model, which would take a vector of chemical properties in input.

The synthesizability evaluator is in the first stages of realization. We provide just a sketch of two possible ways of implementing this idea with GNNs. The simpler method consists in estimating a synthesizability score over the molecular graph, as a graph–focused regression task. The synthesizability score can be supervised with the number of reactions needed to produce a compound, which can be weighted according to the difficulty of the reactions and/or the rarity of reactants. Another more complex but also more precise way of assessing synthesizability would be to identify the reaction chain that allows to produce the compound. This has the advantage of explaining the feasibility of the production in a very precise way, while coming with a higher computational cost. We can work on a graph with two types of nodes: chemical reactions and compounds. Incoming arcs go from reactants to the reactions in which they are required, with a label indicating the quantity. Outgoing arcs exit reaction nodes and land into molecule nodes, with a label indicating the amount produced. The synthesizability evaluation can be performed by adding the generated compound to the graph and predicting if a reaction exists linking it to any existing compound, generating the reaction node and all its incoming and outgoing arcs. If such reaction exists, finding the chain of reactions that allows to produce the compound would be as easy as finding the shortest path over the modified graph. This can be refined further by also taking into account the amounts of reactants needed and the difficulty/cost of every reaction.

The prediction of DSEs can be performed with methods similar to those already existing, including GNN–based methods [23], [24], though they need to be improved in order to reach the required level of precision and recall. DruGNN [23] builds a graph composed of two types of nodes: drugs and genes. Their interactions are modeled by edges. Similarity edges are also inserted between drug nodes. Gene features are inserted in the gene node labels. Drug features, including a structural fingerprint vector, are inserted in the drug node labels. The DSE prediction is then modeled as a multi–class multi–label classification task on drug nodes. Molecular–GNN [24] replicates the same approach, but uses an additional GNN module to estimate neural fingerprints taylored to the task at hand (the two GNNs are trained in cascade). This allows to improve the *Precision at* 90% *Recall* by more than 25% with respect to DruGNN. Compounds can also be screened for polypharmacy effects (side–effects occurring when two or more drugs are taken together), two GNN–based models exist: one estimates the polypharamcy effects with a link prediction model on a graph relating drugs with genes [53], while the other uses a graph co–attention mechanism on the two graphs of a pair of compounds in order to determine if they interact in harmful ways [54].

The last step consists in selecting the best compounds for the

single projects. The realization of this step could imply using GNNs but this is not mandatory. Moreover, the methodology for this step will be determined at the time of system deployment and will vary according to the objective. It is therefore out of the scope of this paper to give a set of methodology guidelines for this last step of the pipeline.

IV. Data flow

Given the current volume of compounds that can be generated by MG^2N^2, the output of the first step of the pipeline can be estimated around one million molecules per day, using a Nvidia Titan-RTX GPU. This already takes into account the computational time dedicated to the first filtering step, carried out with RdKit. This computational burden is mainly determined by the comparison of generated molecules with a library of known compounds to determine their novelty, and by the screening for duplicate molecules. This filter takes considerably more time than the proper generative process, and the cost increases as the number of already generated molecules to screen against increases. As a consequence, with large amounts of molecules added to the library each day, the volume of compounds produced will start to decrease rapidly. Yet, the generation of some millions of molecules will already provide a good starting point for all the following steps, decreasing the demand for new batches of generated material at an even higher rate.

Starting from the reference one million compounds produced in one day, and given the current VUN score of MG^2N^2 (the fraction of generated molecules which are valid, unique, and novel), which is equal to 0.454 [25], the first filtering step will likely delete around 54.6 % of the generated compounds. This means that the second step of the pipeline will have to process around 454,000 molecules per day.

Since the synthesizability evaluator has not been realized yet, its actual computational costs and working times are not known. These will also depend on the algorithm applied and on the degree of precision required. Just estimating a synthesizability score would likely require a small computational power, while using a model that identifies the reactions needed to produce a compound would require a heavier load. Moreover, a precise estimation of the volume of compounds that will be eliminated is not available at the current stage of realization. We can expect the fraction of compounds surviving this stage to be something around 0.1 of the compounds in input. This means that the third step of the pipeline will receive around 45,400 molecules per day.

DSE prediction can easily work on large volumes of molecules. Once trained, both DruGNN and MolecularGNN can process about 2,000 compounds per minute on a commercial laptop. The polypharmacy prediction methods have slightly heavier yet comparable computational loads. Therefore, using both filters in cascade would imply a capacity of processing around 400 compounds per minute, or a total of 576,000 compounds per day. Using a more powerful machine would increase these volumes. As a consequence, this step is not critical to determine the data flow inside the pipeline.

V. Conclusions

We provided a sketch of a GNN–assisted pipeline for drug discovery, composed of four main steps: molecular graph generation, synthesizability evaluation, DSE prediction, selection of the best compounds. We have analyzed each of these problems, mentioning available methods and describing the current state of realization and/or future implementations. We can conclude that the pipeline could be ready for the first *in–silico* tests in the near future. This tool could provide a library of high–quality structures from which to select candidate drugs. In order to improve generation efficiency, MG^2N^2 can be tuned for conditional generation. Another interesting idea for future work is to develop a hierarchical version of this method: A first generator could design the general molecular structure as a graph in which each node corresponds to a group of atoms, with an embedding that describes its desired characteristics. A second generator could then add the details in each of these groups based on their embeddings, generating the single atoms and chemical bonds. Also the *in–silico* prediction of DSEs can be improved, retaining only generated compounds that have low probability of later failing clinical tests. Future work may focus on better integrating data from various sources, expanding the base knowledge of the predictor network. Finally, the last step of the pipeline, namely the selection of the best compounds for the task at hand, should be developed in close collaboration with drug discovery experts, based on their knowledge of the problem and their needs.

References

[1] E. D. Kantor, C. D. Rehm, J. S. Haas, A. T. Chan, and E. L. Giovannucci, "Trends in prescription drug use among adults in the United States from 1999–2012," *Jama*, vol. 314, no. 17, pp. 1818–1830, 2015.

[2] O. J. Wouters, M. McKee, and J. Luyten, "Estimated research and development investment needed to bring a new medicine to market, 2009-2018," *Jama*, vol. 323, no. 9, pp. 844–853, 2020.

[3] M. Dickson and J. P. Gagnon, "Key factors in the rising cost of new drug discovery and development," *Nature reviews Drug discovery*, vol. 3, no. 5, pp. 417–429, 2004.

[4] S. J. Y. Macalino, V. Gosu, S. Hong, and S. Choi, "Role of computer-aided drug design in modern drug discovery," *Archives of pharmacal research*, vol. 38, no. 9, pp. 1686–1701, 2015.

[5] J. S. Smith, A. E. Roitberg, and O. Isayev, "Transforming computational drug discovery with machine learning and ai," *ACS medicinal chemistry letters*, vol. 9, no. 11, pp. 1065–1069, 2018.

[6] C. F. Lipinski, V. G. Maltarollo, P. R. Oliveira, A. B. da Silva, and K. M. Honorio, "Advances and perspectives in applying deep learning for drug design and discovery," *Frontiers in Robotics and AI*, vol. 6, p. 108, 2019.

[7] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[8] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, 2020.

[9] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "Computational capabilities of graph neural networks," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 81–102, 2009.

[10] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[11] J. Kim, S. Park, D. Min, and W. Kim, "Comprehensive survey of recent drug discovery using deep learning," *International Journal of Molecular Sciences*, vol. 22, no. 18, p. 9983, 2021.

[12] T. N. Kipf and M. Welling, "Semi–supervised classification with graph convolutional networks," in *5th International Conference on Learning Representations, ICLR 2017*, 2017.

[13] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, pp. 1024–1034, 2017.

[14] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu, "Relational inductive biases, deep learning, and graph networks," 2018.

[15] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017.

[16] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proceedings of the 34th International Conference on Machine Learning*, vol. 70, pp. 1263–1272, 2017.

[17] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," in *International Conference on Learning Representations*, 2018.

[18] F. M. Bianchi, D. Grattarola, L. Livi, and C. Alippi, "Hierarchical representation learning in graph neural networks with node decimation pooling," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[19] B. Wu, Y. Liu, B. Lang, and L. Huang, "Dgcnn: Disordered graph convolutional neural network based on the gaussian mixture model," *Neurocomputing*, vol. 321, pp. 346–356, 2018.

[20] F. Guerranti, M. Mannino, F. Baccini, P. Bongini, N. Pancino, A. Visibelli, and S. Marziali, "Caregivermatcher: graph neural networks for connecting caregivers of rare disease patients," *Procedia Computer Science*, vol. 192, pp. 1696–1704, 2021.

[21] N. Pancino, P. Bongini, F. Scarselli, and M. Bianchini, "Gnnkeras: A keras-based library for graph neural networks and homogeneous and heterogeneous graph processing," *SoftwareX*, vol. 18, p. 101061, 2022.

[22] N. Pancino, A. Rossi, G. Ciano, G. Giacomini, S. Bonechi, P. Andreini, F. Scarselli, M. Bianchini, and P. Bongini, "Graph neural networks for the prediction of protein–protein interfaces," in *ESANN*, pp. 127–132, 2020.

[23] P. Bongini, F. Scarselli, M. Bianchini, G. M. Dimitri, N. Pancino, and P. Liò, "Modular multi–source prediction of drug side–effects with DruGNN," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 20, no. 2, pp. 1211–1220, 2023.

[24] N. Pancino, Y. Perron, P. Bongini, and F. Scarselli, "Drug side effect prediction with deep learning molecular embedding in a graph–of–graphs domain," *Mathematics*, vol. 10, no. 23, 2022.

[25] P. Bongini, M. Bianchini, and F. Scarselli, "Molecular generative graph neural networks for drug discovery," *Neurocomputing*, vol. 450, pp. 242–252, 2021.

[26] Z. Chengxi and W. Fei, "Moflow: an invertible flow model for generating molecular graphs," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 617–626, 2020.

[27] R. Irwin, S. Dimitriadis, H. Jiazhen, and E. J. Bjerrum, "Chemformer: a pre-trained transformer for computational chemistry," *Machine Learning: Science and Technology*, vol. 3, no. 1, p. 015022, 2022.

[28] K. Seokho and C. Kyunghyun, "Conditional molecular design with deep generative models," *Journal of Chemical Information and Modeling*, vol. 59, no. 1, pp. 43–52, 2019. PMID: 30016587.

[29] X. Yang, J. Zhang, K. Yoshizoe, K. Terayama, and K. Tsuda, "Chemts: an efficient python library for de novo molecular generation," *Science and technology of advanced materials*, vol. 18, no. 1, pp. 972–976, 2017.

[30] R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik, "Automatic chemical design using a data-driven continuous representation of molecules," *ACS central science*, vol. 4, no. 2, pp. 268–276, 2018.

[31] M. J. Kusner, B. Paige, and J. M. Hernández-Lobato, "Grammar variational autoencoder," in *International Conference on Machine Learning*, pp. 1945–1954, PMLR, 2017.

[32] M. Simonovsky and N. Komodakis, "GraphVAE: Towards generation of small graphs using variational autoencoders," in *International Conference on Artificial Neural Networks*, pp. 412–422, Springer, 2018.

[33] R. Liao, Y. Li, Y. Song, S. Wang, W. Hamilton, D. K. Duvenaud, R. Urtasun, and R. Zemel, "Efficient graph generation with graph recurrent attention networks," in *Advances in Neural Information Processing Systems*, pp. 4255–4265, 2019.

[34] N. De Cao and T. Kipf, "MolGAN: An implicit generative model for small molecular graphs," 2018.

[35] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, and M. Guo, "GraphGAN: Graph representation learning with generative adversarial nets," in *Thirty-second AAAI conference on artificial intelligence*, pp. 2508–2515, 2018.

[36] D. Flam-Shepherd, T. Wu, and A. Aspuru-Guzik, "Graph deconvolutional generation," 2020.

[37] L. Oneto, N. Navarin, B. Biggio, F. Errica, A. Micheli, F. Scarselli, M. Bianchini, L. Demetrio, P. Bongini, A. Tacchella, and A. Sperduti, "Towards learning trustworthily, automatically, and with guarantees on graphs: An overview," *Neurocomputing*, vol. 493, pp. 217–243, 2022.

[38] J. Bradshaw, B. Paige, M. J. Kusner, M. Segler, and J. M. Hernández-Lobato, "A model to search for synthesizable molecules," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[39] J. Bradshaw, B. Paige, M. J. Kusner, M. Segler, and J. M. Hernández-Lobato, "Barking up the right tree: an approach to search over molecule synthesis dags," *Advances in neural information processing systems*, vol. 33, pp. 6852–6866, 2020.

[40] K. Korovina, S. Xu, K. Kandasamy, W. Neiswanger, B. Poczos, J. Schneider, and E. Xing, "Chembo: Bayesian optimization of small organic molecules with synthesizable recommendations," in *International Conference on Artificial Intelligence and Statistics*, pp. 3393–3403, PMLR, 2020.

[41] W. Gao and C. W. Coley, "The synthesizability of molecules proposed by generative models," *Journal of chemical information and modeling*, vol. 60, no. 12, pp. 5714–5723, 2020.

[42] C.-H. Liu, M. Korablyov, S. Jastrzebski, P. Włodarczyk-Pruszyński, Y. Bengio, and M. Segler, "Retrognn: fast estimation of synthesizability for virtual screening and de novo design by learning from slow retrosynthesis software," *Journal of Chemical Information and Modeling*, vol. 62, no. 10, pp. 2293–2300, 2022.

[43] H. Khalil and C. Huang, "Adverse drug reactions in primary care: A scoping review," *BMC health services research*, vol. 20, no. 1, pp. 1–13, 2020.

[44] M. L. Billingsley, "Druggable targets and targeted drugs: Enhancing the development of new therapeutics," *Pharmacology*, vol. 82, no. 4, pp. 239–244, 2008.

[45] S. Mizutani, E. Pauwels, V. Stoven, S. Goto, and Y. Yamanishi, "Relating drug–protein interaction network with drug side effects," *Bioinformatics*, vol. 28, no. 18, pp. i522–i528, 2012.

[46] W. Zhang, Y. Chen, S. Tu, F. Liu, and Q. Qu, "Drug side effect prediction through linear neighborhoods and multiple data source integration," in *2016 IEEE international conference on bioinformatics and biomedicine (BIBM)*, pp. 427–434, IEEE, 2016.

[47] G. M. Dimitri and P. Liò, "DrugClust: A machine learning approach for drugs side effects prediction," *Computational biology and chemistry*, vol. 68, pp. 204–210, 2017.

[48] E. Pauwels, V. Stoven, and Y. Yamanishi, "Predicting drug side–effect profiles: A chemical fragment–based approach," *BMC bioinformatics*, vol. 12, no. 1, pp. 1–13, 2011.

[49] A. Cakir, M. Tuncer, H. Taymaz-Nikerel, and O. Ulucan, "Side effect prediction based on drug–induced gene expression profiles and random forest with iterative feature selection," *The Pharmacogenomics Journal*, pp. 1–9, 2021.

[50] O. C. Uner, R. G. Cinbis, O. Tastan, and A. E. Cicek, "DeepSide: A deep learning framework for drug side effect prediction," 2019.

[51] R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. von Lilienfeld, "Quantum chemistry structures and properties of 134 kilo molecules," *Scientific Data*, vol. 1, 2014.

[52] J. J. Irwin and B. K. Shoichet, "Zinc – a free database of commercially available compounds for virtual screening," *Journal of chemical information and modeling*, vol. 45, no. 1, pp. 177–182, 2005.

[53] M. Zitnik, M. Agrawal, and J. Leskovec, "Modeling polypharmacy side effects with graph convolutional networks," *Bioinformatics*, vol. 34, no. 13, pp. i457–i466, 2018.

[54] A. Deac, Y.-H. Huang, P. Veličković, P. Liò, and J. Tang, "Drug–drug adverse effect prediction with graph co–attention," 2019.