mathematics

Article

Drug Side Effect Prediction with Deep Learning Molecular Embedding in a Graph-of-Graphs Domain

Niccolò Pancino [1,2,*,†], Yohann Perron [3,†], Pietro Bongini [1,2,4,*] and Franco Scarselli [1]

1 Department of Information Engineering and Mathematics (DIISM), University of Siena, 53100 Siena, Italy
2 Department of Information Engineering, University of Florence, Via S. Marta 3, 50139 Florence, Italy
3 Ecole Polytechnique, Institut Polytechnique de Paris, Rte de Saclay, 91120 Palaiseau, France
4 Department of Computer Science, University of Pisa, Largo B. Pontecorvo 3, 56127 Pisa, Italy
* Correspondence: niccolo.pancino@unifi.it (N.P.); pietro.bongini@unifi.it (P.B.)
† These authors contributed equally to this work.

Abstract: Drug side effects (DSEs), or adverse drug reactions (ADRs), constitute an important health risk, given the approximately 197,000 annual DSE deaths in Europe alone. Therefore, during the drug development process, DSE detection is of utmost importance, and the occurrence of ADRs prevents many candidate molecules from going through clinical trials. Thus, early prediction of DSEs has the potential to massively reduce drug development times and costs. In this work, data are represented in a non-euclidean manner, in the form of a graph-of-graphs domain. In such a domain, structures of molecule are represented by molecular graphs, each of which becomes a node in the higher-level graph. In the latter, nodes stand for drugs and genes, and arcs represent their relationships. This relational nature represents an important novelty for the DSE prediction task, and it is directly used during the prediction. For this purpose, the MolecularGNN model is proposed. This new classifier is based on graph neural networks, a connectionist model capable of processing data in the form of graphs. The approach represents an improvement over a previous method, called DruGNN, as it is also capable of extracting information from the graph-based molecular structures, producing a task-based neural fingerprint (NF) of the molecule which is adapted to the specific task. The architecture has been compared with other GNN models in terms of performance, showing that the proposed approach is very promising.

Keywords: graph neural networks; drug side effect prediction; node classification; deep learning; transductive learning

MSC: 68T07

check for updates

Citation: Pancino, N.; Perron, Y.; Bongini, P.; Scarselli, F. Drug Side Effect Prediction with Deep Learning Molecular Embedding in a Graph-of-Graphs Domain. *Mathematics* **2022**, *10*, 4550. https://doi.org/10.3390/math10234550

Academic Editor: Jianping Gou

Received: 3 November 2022
Accepted: 27 November 2022
Published: 1 December 2022

1. Introduction

Drug side effects (DSEs), also known as adverse drug reactions (ADRs), are unwanted and undesirable effects that are possibly related to a drug, and they have become a major modern healthcare concern, particularly given the increasing complexity of therapeutics; thus, they have a high impact on the costs of the public health care system and drug discovery processes [1,2]. They can arise when a drug is used either as indicated, or as a consequence of improper dosages or multiple drug interactions, ranging from minor problems such as a runny nose to life-threatening events, such as a heart attacks or liver damage: although fatal ADRs are rare once a drug is launched on the market, they are estimated to be one of the highest causes of death in the United States [3].

As prescription drug use is increasing [4], DSEs are becoming a central problem for pharmaceutical companies, since their occurrence is one of the most important reasons for candidate drug elimination during clinical trials, accounting for around 30% of failures [2,5] and preventing candidate molecules from being selected as commercial drugs.

The development of computational tools that can predict adverse effects might thus reduce the attrition rate, avoiding health risks for participants and cutting drug development costs.

In general, DSE computational prediction methods range from simple predictors to machine learning (ML) techniques such as support vector machines or multilayer perceptrons (MLPs), and to more complex models based on random forests or deep learning [6–8]. However, most of the DSE prediction techniques are still limited by the reliance on euclidean data [9,10], which prevents models from considering relational information in the task, since molecular data must undergo preprocessing to be encoded into vectors, with an inevitable loss of possibly useful information. In this context, usually a molecule structure is encoded in a computer–readable format such as the SMILE string or International Chemical Identifier (InChI) keys, which are strings of symbols of variable length, used by computer software to facilitate the search for molecular information in databases and for creating two or three-dimensional models. Although these formats represent molecules in their entirety, in drug discovery or in drug design applications, fixed–sized encoding strategies are quite useful, as they are better suited for computational methods. The most popular are the molecular fingerprints (MFs), mainly used for assessing similarities between molecules in virtual screening (VS) processes and chemical analysis. MFs represent molecules with fixed-size vectors of real numbers, encoding chemical properties of the molecules and their structural and sub-structural features. Fingerprints of this kind are the results of mostly hand-crafted algorithms based on sub-graph structure detection [11]. However, in recent years, many machine learning-based approaches offered an alternative strategy: neural fingerprints (NF), which are obtained by training neural networks on a specific task [12,13]; therefore, the choice of machine learning architecture for the fingerprint production becomes crucial in order to obtain a fingerprint with the least loss of information.

Only recently, a method named DruGNN was proposed to overcome the preprocessing limit by using graph neural networks (GNNs) [14], although it still relies on preprocessed MFs. GNNs are powerful connectionist models for graph-structured data processing, which have become practical tools for any problem involving graphs, thanks to their capability of processing relational data directly in graph form and calculating an output at each node or edge, with minimal loss of information [15]. Since the seminal work [16,17], many GNN models have been proposed, such as graph convolution networks (GCN) [18], GraphSAGE [19], and Graph attention networks [20]. GNNs have been successfully applied to a wide variety of tasks, spreading from computer vision [21–23] and social and recommendation systems [24–26] to biological and chemical tasks in protein-related problems [27–30] and drug–discovery applications [31,32].

Following the approach proposed in DruGNN [14], the data of interest consist of a heterogeneous graph including two types of nodes (drugs and genes) and three types of edges (drug—gene, drug—drug, and gene—gene relationships). Formally, the task is a node-focused, multi-class, multi-label classification problem: namely, the GNN model is trained to predict the DSEs associated with the drug nodes. The method proposed in this paper aims to overcome both the aforementioned limitations by processing directly graph-structured data and by generating NFs at learning time.

In this context, the structure of a generic drug can be represented as a single graph, consisting of nodes representing atoms and linked by chemical bonds, making any GNN-based model a proper choice for this task. Indeed, since GNNs can handle graph-structured molecules directly and learn how to encode them during the training process, they are ideal to overcome the limits in DSE prediction and in the calculation of fingerprints, by adapting the molecule representation to the task they are being trained for—thus capturing structural and relational information relevant for the specific task, which is something traditional fingerprints and euclidean-based ML models cannot do.

In the presented approach, by exploiting the structural representations of drugs, a richer learning domain is obtained, which is composed of a graph of graphs: each graph corresponds to its molecular graph at the lower level and to a node of the knowledge graph at the higher level. In the latter, drugs are linked to each other by similarity and are linked to gene nodes according to their interactions. A new GNN classifier is exploited for a single-DSE prediction called MolecularGNN that processes such heterogeneous and composed graph-structured data. Notice that the use of a graph-of-graphs domain is an interesting research topic by itself, as GNNs have been applied in very few cases to such a peculiar domain in the literature [33–36]. Moreover, in such a context, the GNN model is supposed to automatically extract a neural fingerprint (NF) from each molecule. Furthermore, a mixed inductive–transductive learning scheme [37] has been adopted, which takes advantage of the knowledge on DSEs of some drugs in order to predict DSEs on novel drug molecules. The performances of the model have been measured in terms of AUC and F1-score and compared with those of other GNN models, showing that the proposed approach is very promising.

The rest of this paper is organized as follows: Section 2 describes the dataset, the data sources, and the GNN model, from both a theoretical and an architectural point of view, with a focus on the fingerprint generator sub-model. Section 3 presents the obtained results and discusses their relevance and meaning.

Finally, Section 4 draws conclusions on this work, summarizing the main obtained results and deriving some future perspectives.

2. Materials and Methods

Over the past ten years, graph neural networks (GNNs) have shown significant and consistent growth: their primary strength is the ability to directly and efficiently process graph-structured data [17], calculating an output at each node or edge, or on any subset of them. Considering structural relationships is a great improvement with respect to euclidean models, as data do not need to be preprocessed and embedded into vectors of real numbers. In this work, the original GNN model [16], and in particular its most recent implementation [38], is exploited to build the DSE predictor, as described in the following.

2.1. The GNN Model

A graph is a non-linear data structure composed of a collection of nodes and edges. Formally, it is defined as a tuple $G = (N, E)$, where N is the set of nodes and $E \subseteq N \times N$ is the set of edges. Nodes represent objects or entities, and relationships between them are represented by edges. Nodes and edges can be associated with values or vectors of values; describing their attributes are defined as a node feature vector x_n for each node $n \in N$ and an edge feature vector $e_{n,m}$ for each edge $(n, m) \in E$, respectively. A GNN assigns a state s_n to each node $n \in N$ and updates it iteratively by sending messages through the edges connecting n to its neighbors $Ne(n)$. The GNN theoretical model is fully described by two parametric functions, f_w and g_w, which, respectively, regulate the state updating process and the output calculation. GNNs create an encoding network, a recurrent neural network architecture which replicates the topology of the input graph, by means of two MLP units, one implementing the state transition function f_w at each node and the other implementing the output function g_w (on targeted nodes or edges).

The GNN replicates the MLP units on each node of the input graph and implements a recurrent algorithm for exchanging useful information between nodes and their neighbors, for a pre-set number of iterations T or until the graph dynamics reaches a stable equilibrium point at time $t \leq T$. A feed-forward network-like architecture is then generated, known as an unfolding network, in which each layer represents an iteration of the implemented algorithm and therefore contains copies of all the elements of the encoding network, on which are based all the connections between the various layers. The information associated with each node can thus be propagated through the whole graph in a sufficient number of iterations; then an output on nodes, edges, or whole graphs—depending on the problem under analysis—is produced by the MLP implementing g_w. For a node-focused problem, such as the one described in this paper, the state transition function f_w and the output function g_w are defined in Equations (1) and (2).

$$s_n^{t+1} = f_w(s_n^t,\ x_n, \varphi(s_{Ne(n)}^t,\ x_{Ne(n)},\ e_{Ne(n),n})) \tag{1}$$

$$o_n = g_w(s_n^T,\ x_n) \tag{2}$$

In particular, $\varphi(\cdot)$ in Equation (1), is the aggregating function and defines how the messages coming from the neighborhood $Ne(n)$ of a node $n \in N$ are aggregated.

GNNs can be extended to the heterogeneous graph-structured data domain, where multiple types of nodes coexist, as they represent different kinds of objects and may have different numbers and types of features. The GNN model for heterogeneous graphs is known as a composite graph neural network (CGNN). In this setting, the CGNN learning process is based on multiple MLPs for the nodes state updating process, one for each type of nodes. In this way, each node type is associated with a unique MLP which learns a unique state transition function. The rest of the learning process is exactly the same as the homogeneous case. The CGNN learning process scheme on a heterogeneous graph is depicted in Figure 1.

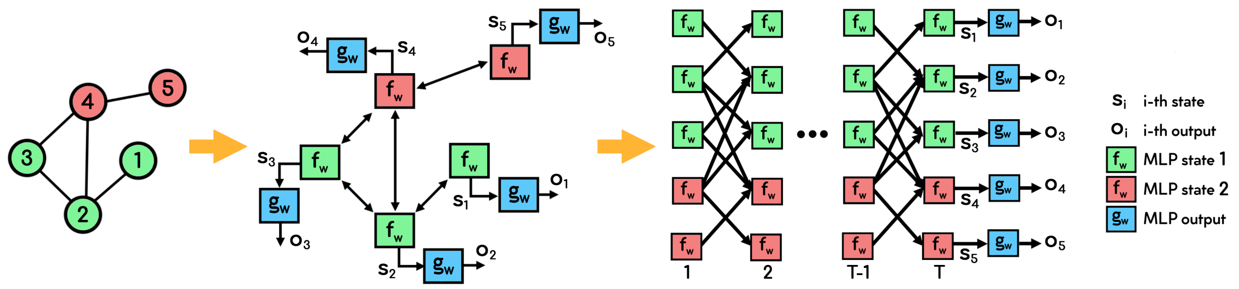

Figure 1. Scheme of a CGNN learning process on a heterogeneous graph. From the left: The input heterogeneous input graph, where blue and red circles represent two types of nodes, and connections belong to a single undirected edge type. The encoding network, generated by means of the MLPs implementing the state transition functions f_w (red and green rectangles, one for each node type) and the output function g_w (blue rectangles). The unfolding network, in which the information can flow from the input to the output, passing through the aforementioned MLPs and the graph connections defining this feed-forward neural network. Note that for a non-composite GNN, learning on a homogeneous graph, the scheme is the same, with the only MLP state 1 considered in this setting, since only one type of node is admitted. This figure was taken from [38].

2.2. The MolecularGNN Model

To improve the performance by exploiting information from both the molecule structures and the graph, a new GNN-based model, MolecularGNN, is proposed. This model is an improvement of DruGNN, as it is also capable of extracting information directly from the graph-based molecular structure representation, by processing it with a GNN sub-model in order to produce a NF which considers both node features and relational data, thereby improving the overall predictions.

An overview of the architecture and processing scheme is given in Figure 2. The MolecularGNN model is composed of two GNN sub-models:

- A molecular embedding module: A GNN-based architecture fed with the graph representations of the molecular structures—extracted from the SMILES strings describing the drug molecules—to produce a task-based NF, which is concatenated to the drug node features and used by the following sub-model to predict DSEs on drug nodes; this approach should encourage the molecular embedding sub-model to extract information, which is not captured by the MF used in the original work, as it remains the same overall training procedure. In this case, since the NF is produced by a ML model, it is dynamically adapted during the learning procedure for the DSE prediction task to maximize the performance of the general model.

- DruGNN: The model is similar the one introduced in [14]. This network is a recurrent GNN model [16] which exploits an inductive–transductive learning scheme to predict DSEs on drug nodes. More detail about its implementation and its hyperparameters can be found in the original paper [14]. The main difference with the original model is the fingerprint used as part of the drug node features, since in this work it is not a pre-calculated or standard MF, but a NF produced by the molecular embedding sub-model.

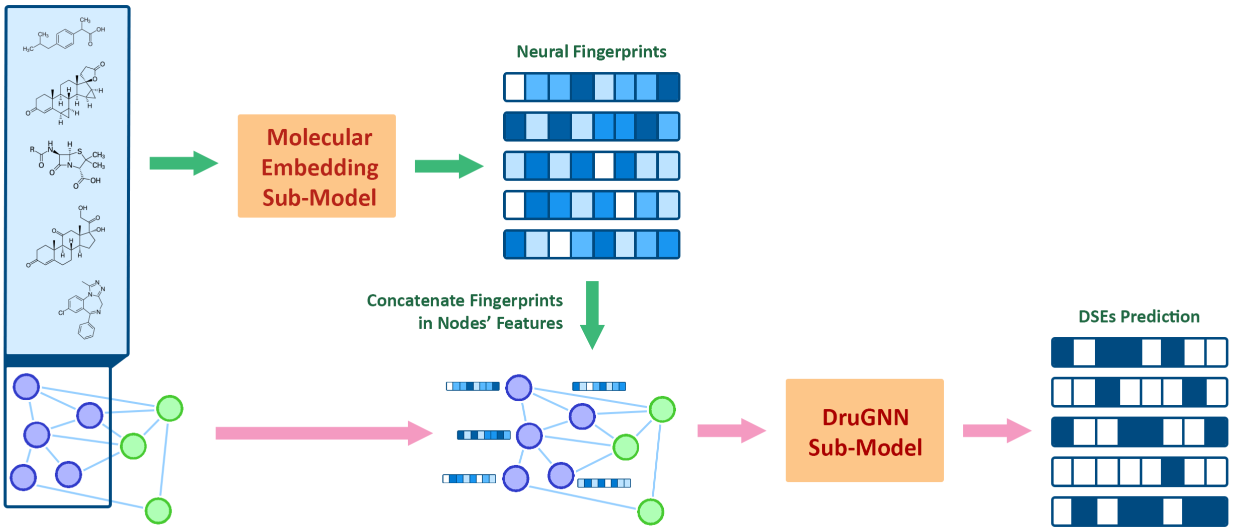
Molecular Embedding Sub-Model
Neural Fingerprints
Concatenate Fingerprints in Nodes' Features
DruGNN Sub-Model
DSEs Prediction

Figure 2. MolecularGNN architecture and processing scheme. Molecular graph structures—representing a single drug node each—are processed by the Molecular Embedding sub-model to produce NFs which are then concatenated to the drug node features vectors. The enriched graph-structured dataset is then processed by the DruGNN module to produce predictions on drug node DSEs. A possible new drug molecule can be added to the dataset by calculating its possible relationships with other nodes in the graph and extrapolating its graph representation from its SMILES code, and the related DSEs can be predicted by just applying the pre-trained model to the new dataset, composed of the one on which the model has been trained, and enriched by the new information derived from the new drug molecule.

To ensure that the information extracted from the molecule complement the information contained in the DruGNN graph, the two aforementioned modules are jointly trained as a single model. The model was implemented using the GNNkeras library [38] which is publicly available on GitHub https://github.com/NickDrake117/GNNkeras, accessed on 1 November 2022. Please refer to the GNNkeras paper [38] and the Data Availability section for more details about GNN and DruGNN models and implementations.

2.3. Dataset Description

The dataset for the DSE prediction task was the one constructed in [14]. Please refer to the Data Availability Statement section for data and code availability. It consists of a single heterogeneous graph where genes and drugs are represented as nodes, connected by three different sets of edges, with relationships based on gene–gene interactions, drug–gene interactions, and drug–drug similarity. In particular, gene node features are retrieved from the BioMart database [39] and consist of chromosome and strand location, percentage of GC content, and a one-hot vector representing the molecular function ontology term, clustered from molecular function ontology [40]; instead, drug nodes are described by a set of chemico-physical molecular descriptors provided by the PubChem database [41], such as molecular weight, polar surface area, xlogp coefficient, heavy atom and rotatable bonds count, number of hydrogen bond donors, and acceptors.

MFs were extracted from the SMILES strings describing each molecular structure of the drugs, using the rdkit python library [11]. MFs were used to define the drug—drug relationships, based on the drug similarity measured in terms of Tanimoto similarity [42], the most common measure of similarity between molecules. Drug–gene relationships are based on drug-to-protein interactions (DPI) extracted from the STITCH database [43], one of the most extensive and up-to-date DPI databases available online, and Biomart, allowing us to retrieve a mapping between drugs and genes whose products are the proteins the drugs interact with. Finally, gene–gene relationships are retrieved from the Human Reference Interactome (HuRI) [44] and from a gene-to-protein mapping provided by Biomart. The labels for the drug nodes were extracted from the SIDER database [45] and consist of 360 single DSEs, retrieved from medical documents, with no information about the concentration at which the side effects appear: the final task is then a multi-label, multi-class, node-focused classification. In particular, belonging to the positive class for a generic drug means that it produces the particular side effect, corresponding to the value 1 in the related position of the target vector. Note that each drug can cause multiple side effects. The original dataset construction, with all the source databases and preprocessing operations, is sketched in Figure 3.

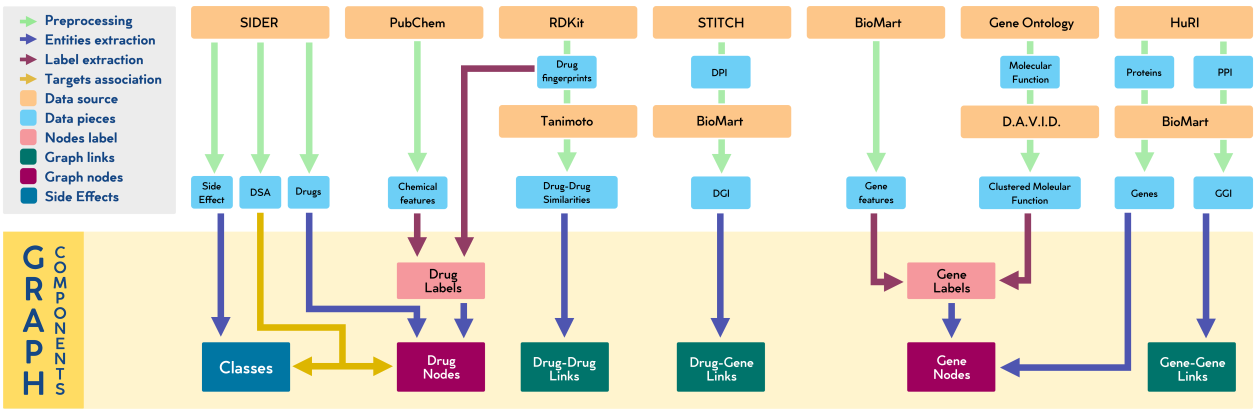

Figure 3. Sketch of the DruGNN original dataset's construction. The information flows from the orange and cyan rectangles representing data sources and data pieces to the final graph representation, in which purple and pink rectangles represent graph nodes and their features, green rectangles stand for the edges between nodes and blue rectangles represent the labels of a subset of the drug nodes.

The molecules' SMILES strings were also used to obtain graph-based representations of the drug molecules, with which the molecular embedding module was fed during training time. Even if the graph representation of a drug as a molecule may seem natural, it can still be extremely complex: a simple way to represent them is to set each atom as a node (except for hydrogen atoms) and each bond between them as an edge in the graph, both associated with feature vectors, as the GNN model [16] supports features on both the edges and the nodes. Since some atoms are quite rare in the dataset—for instance, arsenic is present in only one molecule—they are grouped in accordance with their chemical families which are encoded by a 1-hot vector: in total, 18 classes of atoms have been identified, as shown in Figure 4.

Moreover, atom feature vectors are enriched by additional chemical information, such as the degree of the atom (number of electrons involved in chemical bonds), the number of hydrogen atoms linked to them, the number of radical electrons around the atom (which constitute a rare feature, as they introduce instability), and their formal charges.

Edge features describe the type (simple, double, triple, or aromatic) of bond the edge represents, encoded by a 1-hot vector: this is the only feature used on edges, as other important quantities should be deductible from the atoms on both sides of the edge, and from their neighborhood.

The DruGNN dataset consists of a single graph composed of 9222 nodes (of which 1341 are drug nodes and 7881 are gene nodes) and 331,623 edges (of which 12,002 are gene–gene interaction links, 314,369 are DPIs, and 5252 are drug–drug similarities). At a high-resolution level, drug nodes are represented as a single homogeneous graph each, where nodes represent atoms and edges represent chemical bonds. In the original dataset, only side effects present in more than 100 drugs were considered for the task. However, since the dataset contains many DSEs occurring in a very limited number of cases—some of them even in less than 10 drugs—only the 280 most frequent DSEs, out of the 360 originally described in [14], were considered in the present study. The final dataset, represented by a single graph of graphs, was then fed to the model, for a multi-label, multi-class classification task.

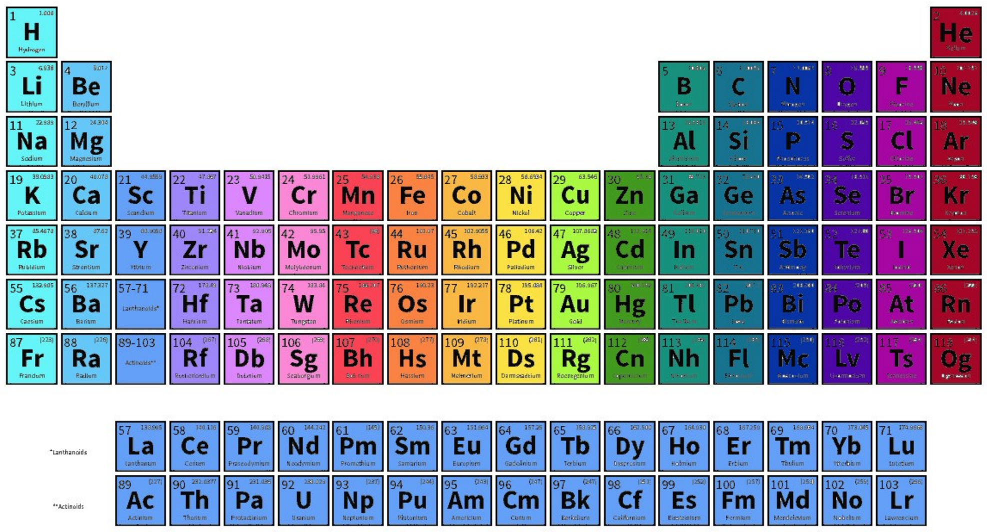

Figure 4. Grouping of atom according to their chemical family. Each color represents a single atom class.

2.4. Experimental Setup

GNNs and CGNNs can be trained by either inductive and/or transductive learning [37] thanks to the natural way the information flows across the graph. Inductive learning is the common approach: the training set contains target values for some nodes, which are used to update the network parameters, so that the GNN will produce the desired output. In transductive learning, the target values are used as node attributes and the GNN makes the information flow through the graph. Inductive and transductive learning can be also applied together. In this case, for a subset of the training nodes—called transductive nodes—the feature vectors are enriched with their targets, to be explicitly adopted in the diffusion process, whereas for other nodes—called inductive nodes—the targets are used in the error function [37].

Mixed inductive–transductive learning in DSEs has been already used in [14] with great success: instead of predicting DSEs on all the drug nodes based only on their "inductive" features, the labels of some of the already known drugs can be exploited to improve the model's prediction on the remaining inductive drug nodes, as the DSE information coming from the transductive drug nodes is considered by the model in the prediction procedure. An overview of the batch generation for the mixed inductive–transductive strategy is depicted in Figure 5.

A grid-search procedure has been carried out based on the hyperband algorithm [46], together with an early stopping callback, monitoring the F1-score on a validation set. The hyperparameters search space is summarized in Table 1.

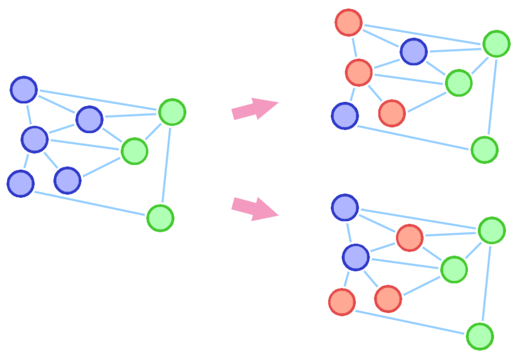

Figure 5. An example of two-batch generation for the mixed inductive–transductive learning setting. From the left: the original graph, with no transductive nodes; the generated batches, with both inductive and transductive nodes. Blue, red, and green nodes represent inductive drug nodes, transductive drug nodes, and gene nodes, respectively. Note that the transduction is performed only on drug nodes, since no prediction is performed on nodes belonging to a different type.

Table 1. Hyperparameter search space for MolecularGNN. M = molecular embedding sub-model; D = DruGNN sub-model. In bold is the final architecture of the MolecularGNN model.

Hyperparameter	Range
M number of layers in net state	$[1, 2, \mathbf{3}, 4]$
M number of layers in net output	$[1, \mathbf{2}, 3]$
M batch normalization layer before net state	$[\mathbf{True}, \mathit{False}]$
M dropout in net state	$[\mathbf{0.0}, 0.1]$
M L2 regularization	$[0.0, \mathbf{0.001}, 0.01, 0.1]$
M activation function	$[\mathbf{relu}, \mathit{tanh}, \mathit{selu}]$
M state dimension	$[50, 70, 100, 120, \mathbf{150}]$
M max iteration	$[2, 3, \mathbf{4}, 5, 6, 7]$
D number of layers in net state	$[1, \mathbf{2}, 3, 4]$
D number of layers in net output	$[1, \mathbf{2}, 3]$
D batch normalization layer before net state	$[\mathbf{True}, \mathit{False}]$
D dropout in net state	$[0.0, \mathbf{0.1}]$
D L2 regularization	$[\mathbf{0.0}, 0.001, 0.01, 0.1]$
D activation function	$[\mathbf{relu}, \mathit{tanh}, \mathit{selu}]$
D state dimension	$[\mathbf{50}, 70, 100, 120]$
D max iteration	$[2, 3, \mathbf{4}, 5, 6, 7]$

Moreover, other three strategies were adopted during the learning procedure to deal with the data distribution imbalance:

- Using binary focal crossentropy instead of simple binary crossentropy;
- Not considering the DSEs with less than 100 positive occurrences, as described in Section 2.3;
- Using a weighting scheme to encourage better prediction of rarer side effects.

Focal cross-entropy was first introduced in [47] for object detection applications. The only difference from the standard cross-entropy loss is the introduction of a factor $(1-p)^\gamma$ for positive example and p^γ for negative examples, as defined in Equation (3):

$$Loss(p, y) = \begin{cases} -(1-p)^\gamma \log(p), & \text{if } y = 1 \\ -(p)^\gamma \log(1-p), & \text{otherwise} \end{cases} \quad (3)$$

From a practical point of view, it results in a loss which penalizes big errors and gives little to no importance to small ones in predictions. Despite its early development purpose, it can be effectively applied on any classification task where the model has to learn from unbalanced datasets, as the additional factor encourages the model to take some risks by predicting the minority class more often. Effects of γ on accuracy and F1-score metrics in the grid search procedure are shown in Figure 6.

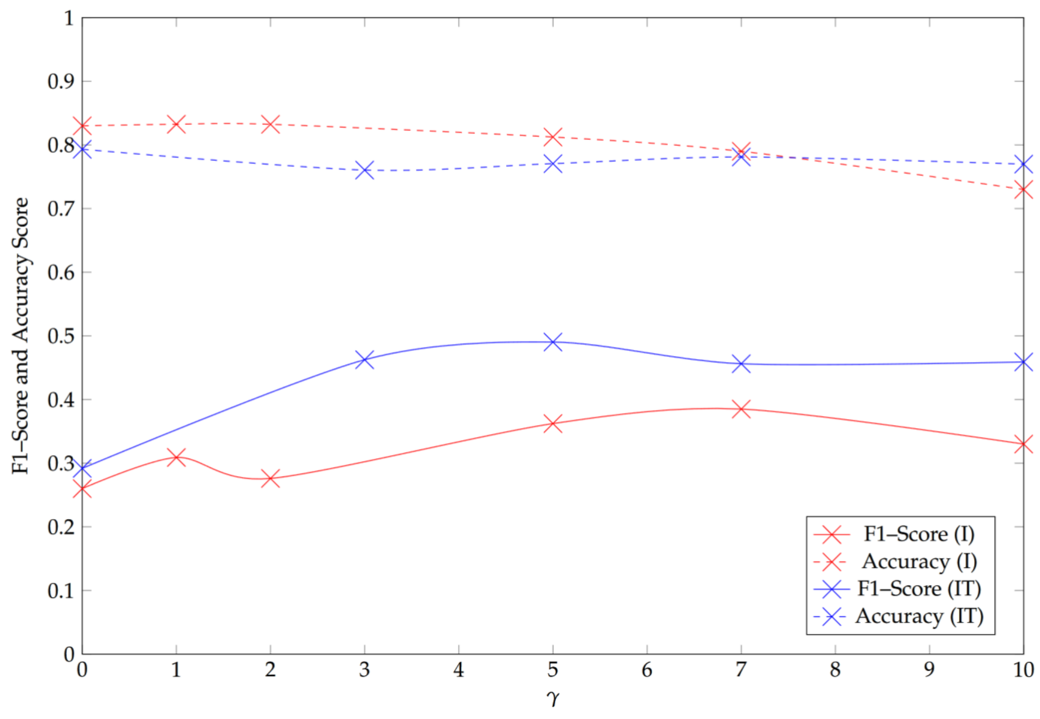

Figure 6. Effects of γ on the accuracy and F1-score metrics. I = only inductive learning approach; IT = mixed inductive–transductive learning approach.

In the dataset, some DSEs occur in only 100 cases, and others in more than 1000, suggesting that the difficulty of prediction for a given DSE is somehow related to the number of positive examples. To tackle this disparity, the model should give more relevance to rarer DSEs when predicting class probabilities: the ideal behavior should be for the network to have close to uniform predicting power over the whole drug set, as in perfectly balanced datasets. To achieve this, a weighting scheme inversely proportional to the number of positive examples has been used, as defined in Equation (4):

$$w = \left(\frac{100}{n_{positive}} \right)^{\mu} \tag{4}$$

where w is the weight given to a DSE with the $n_{positive}$ positive example, and μ is a geometric coefficient which determines the weight of this weighting process (no weighting for $\mu = 0$ and inversely proportional weighting for $\mu = 1$).

All the experiments have been carried out on a Linux-operated machine, equipped with an Intel® Core™ i9–10920X CPU @ 3.50 GHz (12 cores/24 threads), 128 GB DDR4 memory, and two NVIDIA Titan RTX with 24 GB VRAM GDDR6 each. A single graphics card was able to carry out an instance of the learning procedure and the related evaluation of the model, leading to an average of about 5 h and 30 min per experiment.

3. Results and Discussion

MolecularGNN has been compared with the DruGNN model using accuracy, area under the ROC curve (AUC), micro F1-score (F1-score), micro-precision (precision), micro-recall (recall), and precision at recall of 90% (PatR). The results are presented in Table 2.

Table 2. Comparison of MolecularGNN and DruGNN.

Configuration	Accuracy	AUC	F1-Score	Precision	Recall	PatR
MolecularGNN	0.7746	0.6846	0.4716	0.4425	0.5104	0.2591
DruGNN	0.862	0.7715	0.268	0.5934	0.1859	0.2002

MolecularGNN performed significantly better in terms of F1-score compared to the DruGNN model: indeed, the main improvement of MolecularGNN over DruGNN was in the recall, with an improvement of 0.32, which compensated a drop in precision of 0.15, resulting in an almost doubled F1-score. The PatR metric is particularly interesting, as it well simulates a real world scenario: in this scope, the ideal model (i.e., the one with the maximum score) should correctly classify 90% of the side effects, to make clinical tests to filter out the false positives. This metric shows a notable increase in 0.6. The threshold of 90% is still a somewhat low value for such a use case; however, it constitutes a robust metric for predictors. Some metrics, such as the accuracy, do not improve, resulting in a drop in performance of about 10 percentage points: that is in line with the expectations, as the dataset is characterized by an imbalanced class distribution—very common in most real-life classification problems or datasets—thus making the F1-score a better evaluation metric, since it gives a better measure of the incorrectly classified cases than the accuracy metric.

An ablation study was carried out to assess the importance of the contributions of the different changes in the learning procedure and in the dataset. The results are shown in Table 3 as the means of 10 learning procedures with different initialization parameters.

Table 3. Ablation study of MolecularGNN. Values are shown as the means (std) of the results of 10 learning procedures each. In red are the best result for each metrics. Baseline is the MolecularGNN model architecture, as described in Table 1.

Configuration	Accuracy	AUC	F1-Score	Precision	Recall	PatR
Baseline	0.7746 (0.007)	0.6846 (0.014)	0.4716 (0.008)	0.4425 (0.012)	0.5104 (0.021)	0.2591 (0.004)
$\mu = 0$	−0.003 (0.008)	−0.001 (0.006)	−0.001 (0.004)	−0.006 (0.013)	0.002 (0.014)	−0.001 (0.003)
$\gamma = 0$	0.013 (0.012)	−0.040 (0.015)	−0.037 (0.008)	0.021 (0.024)	−0.097 (0.050)	−0.015 (0.005)
All 360 side effects	−0.002 (0.007)	−0.022 (0.030)	−0.013 (0.021)	−0.006 (0.010)	−0.016 (0.042)	−0.007 (0.008)
Only inductive	−0.021 (0.030)	−0.167 (0.001)	−0.138 (0.020)	−0.050 (0.040	−0.196 (0.043)	−0.046 (0.003)
Using fingerprint	0.019 (0.006)	−0.014 (0.005)	−0.034 (0.014)	0.038 (0.017)	−0.100 (0.032)	−0.002 (0.003)
No fingerprint	0.002 (0.007)	−0.042 (0.020)	−0.042 (0.010)	0.001 (0.019)	−0.079 (0.045)	−0.017 (0.005)

As already observed in [14], the mixed inductive–transductive learning approach described in Section 2 resulted in the biggest improvements over the basic model. The most notable results are the increases of 0.167 in the AUC, 0.138 in the F1-score, and 0.046 in the PatR. Those huge improvements show the importance of exploiting all the available information, including the actual labels of the neighboring drugs. The molecular embedding developed in Section 2.2 has an excellent effect on the F1-score (0.042) and on the PatR (0.017%), bringing the second-best improvement for both metrics, only behind the inductive–transductive learning approach. However, it also caused notable decreases in accuracy, AUC, and precision. This was probably the consequence of a mix of overfitting and the gained expressiveness, which was mainly directed toward the F1-score due to changes to the loss function. This was further reinforced by the simultaneous use of both molecular embedding and sklearn fingerprint, which did not increase accuracy but instead decreased the F1-score, the AUC, and the PatR.

The modifications to the loss function brought notable improvements: the use of focal cross-entropy described in Section 2.4 caused big increases in AUC (0.04), F1-score (0.04), and recall (0.10). It also caused the third-biggest increase in PatR, which is arguably the most important metric in the scope of real-world usage. All these came at the cost of slight decreases in accuracy and precision at a classification threshold equal to 0.5. Considering the massive increase in recall, the reduction in precision should be easily compensated by increasing the threshold. In contrast, the effect of weighting the loss based on the number of positive examples, as described in Section 2.4, is very small, but it is mostly beneficial, with the exception of the recall which showed a very slight improvement using $\mu = 0$ (no weighting).

Additionally, a comparison with previously developed methods for drug side effect prediction is provided in Table 4. It is worth noticing that each method was developed on different datasets and with different and often heterogeneous data types. Moreover, these differences led to different dataset sizes, as not all the data available online can be exploited in all the settings. More precisely, MolecularGNN was compared to: DruGNN [14]; Pauwels [48], which is a good baseline that calculates the sparse canonical correlations between the drug structure fingerprints and the side effects; DeepSide [49], which is the only other approach based on deep learning and exploits data concerning drug targets, structural fingerprints, and gene expression; DrugClust, which formulates the prediction as a clustering problem based on gene expression data [50]. Please keep in mind that the different nature of the predictors, the data used by each, and the number of examples mean that the comparison is purely qualitative. The comparison was made in terms of AUC metric, since this is the only metric that was measured for each of the predictors involved.

Table 4. Comparison of data and methodology for each predictor. NF stands for neural fingerprints, SF for structural fingerprints, MG for molecular graphs, GO for Gene Ontology data, CF for chemical features, DPI for drug–protein interactions, DDS for drug–drug similarity, PPI for protein–protein interactions.

Predictor	**AUC**	**Num. Drugs**	**Num. DSEs**	**Method**	**Data Types**
Pauwels [48]	89.32%	888	1385	SCCA	SF
DrugClust [50]	33.36%	1080	2260	Clustering	CF+DPI+GEX
DeepSide [49]	80.90%	791	1042	MLP	GEX+GO+SF
DruGNN [14]	77.15%	1341	360	CGNN	SF+CF+GO+PPI+DPI+DDS
MolecularGNN	68.46%	1384	360	CGNN+NF	MG+CF+GO+PPI+DPI+DDS

The presented method, with an unbalanced dataset, did not reach an AUC comparable to simpler predictors operating on less unbalanced datasets. The F1-score was the dominant metric in this case, and it was significantly better than the F1-score of DruGNN—the only other predictor trained on a dataset with a comparable level of unbalancedness between classes.

4. Conclusions

In this paper, an enhanced and innovative version of DruGNN was proposed: the new model, called MolecularGNN, is able to work directly on the molecular structure and to work on unbalanced datasets and a graph-of-graphs domain. The novelty of this model is represented by the generation of drug structural fingerprints, which can be adapted to the task the general model is trained for. In particular, a drug's molecular structure is processed by a GNN sub-model, producing a NF which is then used to enrich drug node features in the DruGNN graph dataset, which are eventually processed by the following GNN sub-model to predict the DSEs of drug nodes.

The new ML-based technique, trained with an inductive–transductive approach, together with the NF generation, results in an increase of 0.2 in the F1-score and 0.06 in precision at recall of 90% with respect to the original DruGNN model, though they can be further improved. The overall accuracy is not improved, but MolecularGNN succeeded in identifying more experimentally observed DSEs than DruGNN: this is an acceptable compromise, since this task is considered much more important than identifying DSEs which do not arise for a given drug, as a non predicted DSE could cause little harm or be potentially fatal. Although MolecularGNN is still limited, with an F1-score of less than 0.5, its precision at recall of 90% performance indicates that it could possibly be used as an aide during drug development processes.

Although a limitation of this work is represented by the fact that the model has been tested only on one dataset, please notice that the dataset construction itself is an important part of the methodology. Moreover, the dataset contains all the side effects for which all the data involved in the prediction are currently available. As a consequence, changing the dataset would also mean changing the data sources, and consequently, the problem's and model's definitions. For the same reason, it is common practice to test the models for DSE prediction only on the dataset for which they are conceived, as each model usually uses a unique combination of the many different pieces of information needed to model these complex biological phenomena.

Possible future developments in this context include a dynamic learning approach, by considering a dynamic topology of the graph, in which the connections between the drug nodes do not remain the same throughout the learning procedure, since they can be re-calculated on the basis of the similarity between the neural fingerprints generated at each epoch by the appropriate sub-model, possibly adding some constraints to respect a certain level of similarity given by the one obtained with the standard, preprocessed fingerprints.

The model presented in the paper also brings consistent improvement to the usability of the method, as it integrates two successful strategies for solving the same problem: exploiting the molecular graph to predict the side effects of the drug [51] and predicting the side effects on a wide graph that integrates heterogeneous information on the drug and the genes it interacts with [14]. These two strategies have been applied separately so far, but we combined them thanks to the ability of GNNs to process different forms of graph structured data, which is a very promising solution. This was demonstrated by the improvements in F1-score and PatR score shown in the results and will lead to better usability in future real-world scenarios. One of such scenarios, as already partially discussed for DruGNN [14], will consist of building a fully automated pipeline of deep learning-aided drug discovery, in which a molecular graph generator such as GraphVAE [52], JTVAE [53], or MG^2N^2 [54] could generate large quantities of potential drug candidates. After a first filtering step for discarding the compounds with low QED scores [55] or low druggability scores [56,57], MolecularGNN could be employed to screen out all the molecules with relevant probabilities of producing side effects. This pipeline could provide a large quantity of potential drug candidates with good drug-likeness and small side effect profiles, thereby constructing a reliable chemical space from which drug candidates can be drawn by experts, contributing to cutting the costs and difficulties of drug discovery research.

Author Contributions: Conceptualization, N.P., Y.P., P.B. and F.S.; methodology, N.P., Y.P. and P.B.; software, N.P., Y.P. and P.B.; validation, Y.P. and F.S.; formal analysis, F.S.; investigation, N.P., Y.P. and P.B.; resources, N.P., Y.P., P.B. and F.S.; data curation, Y.P. and P.B.; writing—original draft preparation, N.P. and Y.P.; writing—review and editing, N.P., P.B. and Y.P.; visualization, N.P. and Y.P.; supervision, F.S.; project administration, F.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available on GitHub https://github.com/YohannPerron/MolecularGNN, accessed on 1 November 2022 along with DruGNN python code https://github.com/PietroMSB/DrugSideEffects, accessed on 1 November 2022. Please refer to [14] for further information on the dataset and to [38] for the GNNkeras python code.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DSE	Drug Side Effect
ADR	Adverse Drug Reaction
DPI	Drug to Protein Interactions
MF	Molecular Fingerprint
NF	Neural Fingerprint
VS	Virtual Screening
ML	Machine Learning
DL	Deep Learning
MLP	Multilayer Perceptron
GNN	Graph Neural Network
CGNN	Composite Graph Neural Network
GCN	Graph Convolutional Network
AUC	Area Under the Curve
PatR	Precision at Recall at 90%
MDPI	Multidisciplinary Digital Publishing Institute

References

1. Khalil, H.; Huang, C. Adverse drug reactions in primary care: A scoping review. *BMC Health Serv. Res.* **2020**, *20*, 5. [CrossRef] [PubMed]
2. Billingsley, M.L. Druggable targets and targeted drugs: Enhancing the development of new therapeutics. *Pharmacology* **2008**, *82*, 239–244. [CrossRef] [PubMed]
3. Giacomini, K.M.; Krauss, R.M.; Roden, D.M.; Eichelbaum, M.; Hayden, M.R.; Nakamura, Y. When good drugs go bad. *Nature* **2007**, *446*, 975–977. [CrossRef] [PubMed]
4. Kantor, E.D.; Rehm, C.D.; Haas, J.S.; Chan, A.T.; Giovannucci, E.L. Trends in prescription drug use among adults in the United States from 1999 to 2012. *JAMA* **2015**, *314*, 1818–1830. [CrossRef]
5. Tatonetti, N.P.; Liu, T.; Altman, R.B. Predicting drug side-effects by chemical systems biology. *Genome Biol.* **2009**, *10*, 1–4. [CrossRef]
6. Shaked, I.; Oberhardt, M.A.; Atias, N.; Sharan, R.; Ruppin, E. Metabolic network prediction of drug side effects. *Cell Syst.* **2016**, *2*, 209–213. [CrossRef]
7. Lavecchia, A.; Di Giovanni, C. Virtual screening strategies in drug discovery: A critical review. *Curr. Med. Chem.* **2013**, *20*, 2839–2860. [CrossRef]
8. Zitnik, M.; Agrawal, M.; Leskovec, J. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics* **2018**, *34*, i457–i466. [CrossRef]
9. Mizutani, S.; Pauwels, E.; Stoven, V.; Goto, S.; Yamanishi, Y. Relating drug–protein interaction network with drug side effects. *Bioinformatics* **2012**, *28*, i522–i528. [CrossRef]
10. Zhang, W.; Chen, Y.; Tu, S.; Liu, F.; Qu, Q. Drug side effect prediction through linear neighborhoods and multiple data source integration. In Proceedings of the 2016 IEEE international conference on bioinformatics and biomedicine (BIBM), Shenzhen, China, 15–18 December 2016; pp. 427–434.
11. Landrum, G. Rdkit documentation. *Release* **2013**, *1*, 4.
12. Menke, J.; Koch, O. Using Domain-Specific Fingerprints Generated Through Neural Networks to Enhance Ligand-Based Virtual Screening. *J. Chem. Inf. Model.* **2021**, *61*, 664–675. [CrossRef]
13. Duvenaud, D.; Maclaurin, D.; Aguilera-Iparraguirre, J.; Gómez-Bombarelli, R.; Hirzel, T.; Aspuru-Guzik, A.; Adams, R.P. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In Proceedings of the Annual Conference on Neural Information Processing Systems 2015, Montreal, QC, Canada, 7–12 December 2015.
14. Bongini, P.; Pancino, N.; Dimitri, G.M.; Bianchini, M.; Scarselli, F.; Lio, P. Modular multi–source prediction of drug side–effects with DruGNN. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2022**. [CrossRef] [PubMed]
15. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Philip, S.Y. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4–24. [CrossRef]

16. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The graph neural network model. *IEEE Trans. Neural Netw.* **2008**, *20*, 61–80. [CrossRef] [PubMed]
17. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. Computational capabilities of graph neural networks. *IEEE Trans. Neural Netw.* **2008**, *20*, 81–102. [CrossRef] [PubMed]
18. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
19. Hamilton, W.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
20. Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *Stat* **2017**, *1050*, 20.
21. Wang, Y.; Kitani, K.; Weng, X. Joint object detection and multi-object tracking with graph neural networks. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; pp. 13708–13715.
22. Luo, A.; Li, X.; Yang, F.; Jiao, Z.; Cheng, H.; Lyu, S. Cascade graph neural networks for RGB-D salient object detection. In *Lecture Notes in Computer Science, Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020*; Springer: Cham, Switzerland, 2020; pp. 346–364.
23. Shi, W.; Rajkumar, R. Point-gnn: Graph neural network for 3d object detection in a point cloud. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 1711–1719.
24. Wang, Z.; Wei, W.; Cong, G.; Li, X.L.; Mao, X.L.; Qiu, M. Global context enhanced graph neural networks for session-based recommendation. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Xi'an, China, 25–30 July 2020; pp. 169–178.
25. Guerranti, F.; Mannino, M.; Baccini, F.; Bongini, P.; Pancino, N.; Visibelli, A.; Marziali, S. CaregiverMatcher: Graph neural networks for connecting caregivers of rare disease patients. *Procedia Comput. Sci.* **2021**, *192*, 1696–1704. [CrossRef]
26. Wu, C.; Wu, F.; Cao, Y.; Huang, Y.; Xie, X. Fedgnn: Federated graph neural network for privacy-preserving recommendation. *arXiv* **2021**, arXiv:2102.04925.
27. Lee, J.B.; Rossi, R.; Kong, X. Graph classification using structural attention. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1666–1674.
28. Bianchi, F.M.; Grattarola, D.; Livi, L.; Alippi, C. Hierarchical representation learning in graph neural networks with node decimation pooling. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *33*. [CrossRef]
29. Pancino, N.; Rossi, A.; Ciano, G.; Giacomini, G.; Bonechi, S.; Andreini, P.; Scarselli, F.; Bianchini, M.; Bongini, P. Graph Neural Networks for the Prediction of Protein-Protein Interfaces. In Proceedings of the ESANN, Bruges, Belgium, 2–4 October 2020; pp. 127–132.
30. Bongini, P.; Pancino, N.; Scarselli, F.; Bianchini, M. BioGNN: How Graph Neural Networks Can Solve Biological Problems. In *Artificial Intelligence and Machine Learning for Healthcare*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 211–231.
31. Smith, J.S.; Roitberg, A.E.; Isayev, O. Transforming computational drug discovery with machine learning and AI. *ACS Med. Chem. Lett.* **2018**, *9*, 1065–1069. [CrossRef]
32. Kim, J.; Park, S.; Min, D.; Kim, W. Comprehensive survey of recent drug discovery using deep learning. *Int. J. Mol. Sci.* **2021**, *22*, 9983. [CrossRef] [PubMed]
33. Wang, H.; Lian, D.; Zhang, Y.; Qin, L.; Lin, X. Gognn: Graph of graphs neural network for predicting structured entity interactions. *arXiv* **2020**, arXiv:2005.05537.
34. Harada, S.; Akita, H.; Tsubaki, M.; Baba, Y.; Takigawa, I.; Yamanishi, Y.; Kashima, H. Dual convolutional neural network for graph of graphs link prediction. *arXiv* **2018**, arXiv:1810.02080.
35. Wang, H.; Lian, D.; Liu, W.; Wen, D.; Chen, C.; Wang, X. Powerful graph of graphs neural network for structured entity analysis. *World Wide Web* **2022**, *25*, 609–629. [CrossRef]
36. Wang, Y.; Zhao, Y.; Shah, N.; Derr, T. Imbalanced Graph Classification via Graph-of-Graph Neural Networks. *arXiv* **2021**, arXiv:2112.00238.
37. Ciano, G.; Rossi, A.; Bianchini, M.; Scarselli, F. On inductive–transductive learning with graph neural networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 758–769. [CrossRef]
38. Pancino, N.; Bongini, P.; Scarselli, F.; Bianchini, M. GNNkeras: A Keras-based library for Graph Neural Networks and homogeneous and heterogeneous graph processing. *SoftwareX* **2022**, *18*, 101061. [CrossRef]
39. Smedley, D.; Haider, S.; Durinck, S.; Pandini, L.; Provero, P.; Allen, J.; Kasprzyk, A. The BioMart community portal: An innovative alternative to large, centralized data repositories. *Nucleic Acids Res.* **2015**, *43*, W589–W598. [CrossRef]
40. Ashburner, M.; Ball, C.A.; Blake, J.A.; Botstein, D.; Butler, H.; Cherry, J.M.; Davis, A.P.; Dolinski, K.; Dwight, S.S.; Eppig, J.T.; et al. Gene ontology: Tool for the unification of biology. *Nat. Genet.* **2000**, *25*, 25–29. [CrossRef] [PubMed]
41. Kim, S.; Chen, J.; Cheng, T.; Gindulyte, A.; He, J.; He, S.; Li, Q.; Shoemaker, B.A.; Thiessen, P.A.; Yu, B.; et al. PubChem in 2021: New data content and improved web interfaces. *Nucleic Acids Res.* **2021**, *49*, D1388–D1395. [CrossRef] [PubMed]
42. Tanimoto, T.T. IBM internal report. *Nov* **1957**, *17*, 1957.
43. Szklarczyk, D.; Santos, A.; von Mering, C.; Jensen, L.J.; Bork, P.; Kuhn, M. STITCH 5: Augmenting protein-chemical interaction networks with tissue and affinity data. *Nucleic Acids Res* **2015**, *44*, D380–D384. [CrossRef] [PubMed]
44. Luck, K.; Kim, D.K.; Lambourne, L.; Spirohn, K.; Begg, B.E.; Bian, W.; Brignall, R.; Cafarelli, T.; Campos-Laborie, F.J.; Charloteaux, B.; et al. A reference map of the human binary protein interactome. *Nature* **2020**, *580*, 402–408. [CrossRef]
45. Kuhn, M.; Letunic, I.; Jensen, L.J.; Bork, P. The SIDER database of drugs and side effects. *Nucleic Acids Res.* **2015**, *44*, D1075–D1079. [CrossRef]

46. Li, L.; Jamieson, K.; DeSalvo, G.; Rostamizadeh, A.; Talwalkar, A. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization. *J. Mach. Learn. Res.* **2018**, *18*, 6765–6816.
47. Lin, T.; Goyal, P.; Girshick, R.B.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2980–2988.
48. Pauwels, E.; Stoven, V.; Yamanishi, Y. Predicting drug side–effect profiles: A chemical fragment–based approach. *BMC Bioinform.* **2011**, *12*, 169. [CrossRef]
49. Uner, O.C.; Cinbis, R.G.; Tastan, O.; Cicek, A.E. DeepSide: A deep learning framework for drug side effect prediction. *Biorxiv* **2019**, 843029 . [CrossRef]
50. Dimitri, G.M.; Liò, P. DrugClust: A machine learning approach for drugs side effects prediction. *Comput. Biol. Chem.* **2017**, *68*, 204–210. [CrossRef]
51. Bongini, P.; Messori, E.; Pancino, N.; Bianchini, M. A Deep Learning Approach to the Prediction of Drug Side–Effects on Molecular Graphs. Under Review on Computers in Biology and Medicine.
52. Simonovsky, M.; Komodakis, N. Graphvae: Towards generation of small graphs using variational autoencoders. In Proceedings of the International conference on artificial neural networks, Rhodes, Greece, 4–7 October 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 412–422.
53. Jin, W.; Barzilay, R.; Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 2323–2332.
54. Bongini, P.; Bianchini, M.; Scarselli, F. Molecular generative graph neural networks for drug discovery. *Neurocomputing* **2021**, *450*, 242–252. [CrossRef]
55. Bickerton, G.R.; Paolini, G.V.; Besnard, J.; Muresan, S.; Hopkins, A.L. Quantifying the chemical beauty of drugs. *Nat. Chem.* **2012**, *4*, 90–98. [CrossRef] [PubMed]
56. Skalic, M.; Varela-Rial, A.; Jiménez, J.; Martínez-Rosell, G.; De Fabritiis, G. LigVoxel: Inpainting binding pockets using 3D-convolutional neural networks. *Bioinformatics* **2019**, *35*, 243–250. [CrossRef] [PubMed]
57. Bongini, P.; Niccolai, N.; Bianchini, M. Glycine-induced formation and druggability score prediction of protein surface pockets. *J. Bioinform. Comput. Biol.* **2019**, *17*, 1950026. [CrossRef]