A Deep Learning Approach to the Prediction of Drug Side–Effects on Molecular Graphs

Pietro Bongini, Elisa Messori, Niccolò Pancino, and Monica Bianchini

Abstract—Predicting drug side effects before they occur is a critical task for keeping the number of drug–related hospitalizations low and for improving drug discovery processes. Automatic predictors of side–effects generally are not able to process the structure of the drug, resulting in a loss of information. Graph neural networks have seen great success in recent years, thanks to their ability of exploiting the information conveyed by the graph structure and labels. These models have been used in a wide variety of biological applications, among which the prediction of drug side–effects on a large knowledge graph. Exploiting the molecular graph encoding the structure of the drug represents a novel approach, in which the problem is formulated as a multi–class multi–label graph–focused classification. We developed a methodology to carry out this task, using recurrent Graph Neural Networks, and building a dataset from freely accessible and well established data sources. The results show that our method has an improved classification capability, under many parameters and metrics, with respect to previously available predictors. The method is not ready for clinical tests yet, as the specificity is still below the preliminary 25% threshold. Future efforts will aim at improving this aspect.

Index Terms—Drug side–Effect prediction, drug side–effects, graph classification, graph neural networks, molecular graphs.

I. INTRODUCTION

Drug discovery is a fundamental but expensive process to make new pharmaceutical products available for healthcare [1]. Detecting and identifying Drug Side–Effects (DSEs) is mandatory to ensure that only safe drugs enter the market. DSEs have high costs for public health [2], and cause a significant number of hospitalizations every year [3], a constantly increasing trend, also due to the growing use of prescription drugs [4]. Predicting DSEs automatically in silico, before submitting drug candidates to clinical trials, would represent a fundamental improvement for drug discovery processes, cutting their costs in terms of time and money [5].

Automatic DSE predictors have traditionally relied on Euclidean data representations [5] or drug similarity [6]. In the last decade, we have seen an evolution towards Machine Learning (ML), with methods based on Random Forests [7], Support Vector Machines [8], and Clustering [9]. Substantial improvements have been brought by the use of Deep Learning (DL) techniques, which integrate heterogeneous data sources [10]. Indeed, the number and variety of features to be used for prediction have steadily increased, as DSEs are complex biological phenomena involving many metabolic and genetic mechanisms [11].

Since their introduction, Graph Neural Networks (GNNs) [12] have represented a very powerful model [13] for the prediction [14] and generation [15] of graph–structured data, with a wide variety of applications in the biological domain [16]. Their capability of processing relational data directly in graph form, with little information loss and high flexibility [17], allows GNNs to be successfully applied to an enormous variety of different tasks involving graph–structured data [18]. As a consequence, an ever increasing number of models have been developed to improve the field and to adapt the base theory [19] to the various scenarios, like Graph Convolution Networks (GCNs) [20], spectral GCNs [21][22], GraphSAGE [23], GraphNets [24], Message–Passing Neural Networks [25], and Graph Attention Networks (GATs) [26], just to name the most important. The whole GNN family has been classified into categories in order to better navigate through the configurations and to better study their properties from a mathematical point of view. This latter idea has lead to interesting ways of measuring their power in order to maximize the theoretical capabilities of future models, either using unfolding trees [17] or Weisfeiler–Lehman tests [27].

GNNs have also been applied to a related but very different task than the one addressed in this paper: polypharmacy effect prediction, in which the goal is to determine whether two compounds can trigger adverse reactions when taken together. In this framework, two main approaches exist: the first predicts the interactions as edges in a knowledge graph that conveys metabolomics and interactomics information [28], while the second exploits a GAT–based graph co–attention mechanism on the two molecular graphs of each pair of compounds, training the co–attention mechanism to estimate the likelihood of a polypharmacy effect between the two drugs [29].

In this paper, the DSE prediction is addressed as a multi–class multi–label classification problem. Indeed, multi–class problems are all those problems in which more than two classes are defined. Usually, though, a multi–class problem requires that each example is assigned to exactly one class. In a multi–class multi–label problem, instead, an example may be assigned to any number of classes between zero and C, where C is the total...
number of classes defined by the problem. Thus, solving such a problem is more akin to solving C binary classification problems in parallel, rather than solving a "standard" multi-class problem on C classes. Since each drug typically has an arbitrary number of side-effects greater than one, drug side-effect prediction is naturally modeled as a multi-class multi-label problem. This is also consistent with the previous literature on drug side-effect prediction [5, 9, 10, 30]. As the drug structure can be efficiently encoded by a molecular graph, we exploit GNNs to learn and automatically predict DSEs based on the drug structure only. This substantially differentiates the methodology from the only other GNN-based DSE predictor we are aware of: DruGNN, which predicted DSEs on a large knowledge graph integrating drug features, gene features, gene–gene interactions, drug–gene interactions and drug–drug similarities [31]. While being capable of better integrating information from heterogeneous domains, thanks to the properties of Composite Graph Neural Networks (CGNNs), DruGNN is not capable of exploiting the full molecular information, because the drug structure is encoded as a fingerprint vector. Using SMILES, like other non-graph based methodologies do, also implies a loss of information. Molecular graphs instead retain the full amount of structural information that can be associated to each drug compound.

The main contributions of this paper are as follows.

- A novel dataset of molecular graphs is introduced: it can be used for the prediction of DSEs with any predictor model that accepts the drug structure in input; molecular graphs can also be enriched with relevant chemical features of the compound;
- GNN–MGSEP (Graph Neural Network — Molecular Graph Side–Effect Predictor), a GNN–based model for the prediction of DSEs is introduced and validated on the dataset presented above; the problem is tackled as a graph–focused multi-class multi-label classification, where DSEs represent the class labels of the molecular graphs;
- Usability of GNN–MGSEP is discussed, and potential applications of the method in the real–world are introduced.

The rest of the paper is organized as follows. Section II explains the methodology, describes the GNN model used for carrying out the predictions, and defines the experimental setup; Section III presents and details the relevant experimental results for the validation of the method; Section IV discusses the significance of the results, shows the prediction performance in comparison with other available methods, and describes the usability of our approach; Section V draws conclusions on the work presented, discusses its impact and applicability, and introduces possible future research.

II. MATERIALS AND METHODS

In the following, the experimental methodology used in this work is described in detail. The definition of the Graph Neural Network model is reported, as well as its usage in this paper, the dataset characteristics and its building procedure. Table I keeps track of all the notation employed throughout the paper to define and use these concepts.

| Notations | Descriptions |
|-----------|--------------|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
molecule (there is no direct dependence on the individual nodes of the graph, which contribute equally to the output).

The GNN shares the same topology of the input graph and, at each node \( n \), a state \( x_n \), and (possibly) the relative output \( o_n \), are calculated (see Fig. 1). State transition functions and local output functions, indicated with \( f_w \) and \( g_w \), respectively, are both computed via Multi-Layer Perceptrons (MLPs). In particular, for the state network, we employed the SELU (Scaled Exponential Linear Unit) function:

\[
\text{SELU}(x) = \begin{cases} 
\lambda x & \text{if } x > 0 \\
\lambda \alpha (e^x - 1) & \text{if } x < 0
\end{cases}
\]

with the default parameters proposed in [32], with \( \lambda = 1.05070998 \) and \( \alpha = 1.67326324 \) and the LeCun Normal initialization for the weights, while, in the output network, the sigmoid function and the Glorot normal initialization were used.

Our implementation is based on GNNKeras [33], a flexible tool which allows the construction of a large subclass of GNNs. The model iteratively calculates the state update as:

\[
x_n^t = f_w(x_n^{t-1}, l_n, a \sum_{m \in \mathcal{N}(n)} (x_m^{t-1}, l_m, e_{m,n}))
\]

where the state of node \( n \) at iteration \( t \), \( x_n^t \), is described as a function \( f_w \) of the state and label of the node \( n \) itself and the element-wise aggregation of the incoming messages from its neighbors, each of which is the concatenation of the state at the previous iteration \( t - 1 \), of the label, and the arc connecting the neighbor to node \( n \). \( \mathcal{N}(n) \) represents the set of all the neighbors of node \( n \). Finally, two aggregations are possible: average or sum of the neighboring information. The hyperparameter \( a \) allows to select the aggregation type by assuming value 1 or \( \frac{1}{|\mathcal{N}(n)|} \).

The local output function, \( g_w \), is computed at each node based only on the label of the node and on its state, after convergence or a predefined number of iterations \( K \). Finally, the whole output function is calculated by:

\[
y_G = \frac{1}{|\mathcal{N}_{\text{out}}|} \sum_{n \in \mathcal{N}_{\text{out}}} g_w(x_n^K, l_n)
\]

where the set \( \mathcal{N}_{\text{out}} \) contains all the output nodes — in our case all the nodes of the graph, \( \mathcal{N}_{\text{out}} = \mathcal{N} \). Since this is a multi-class multi-label problem, the output MLP of the GNN has a dedicated output unit for each class (side-effect) and all the side-effects of a drug are predicted in parallel. This configuration naturally ensures that the side-effects of a drug are modeled independently of each other. In other words, we do not exploit knowledge on a subset of side-effects of a drug to infer another subset of its side-effects; the prediction is solely based on the drug structural features.

B. Data Collection and Pre-Processing

The only data required in this framework is the drug chemical structure and some known drug side-effect associations. To retrieve such data, we used the public database SIDER [34], which contains data on 1430 drugs, 5880 Adverse Drug Reactions (ADRs), and 140,064 drug–ADR pairs [35]. We referred to the “stereo” version of the STITCH compound identifier [36], since it can be used as a key on the public database PubChem [37].

Before retrieving a graph representation of drug chemical structures, the SIDER database needed to undergo a preprocessing procedure in order to filter out duplicates of drug–ADR pairs, which occur inevitably due to the presence of both lowest level terms (LLTs) and primary terms (PTs) for side-effects: we filtered out all associations referred to LLT side-effects, so that our dataset was made only of drug–ADR pairs in which side-effects are expressed via a PT. In such a
Fig. 2. (a) The number of associated DSEs per drug follows a non uniform skewed distribution. Most drugs have few DSEs, while few drugs are associated to a large number of DSEs. This causes an imbalance in the class distributions which can lead to a bias in the model. (b) The number of drugs associated to each specific DSE. Most DSEs are associated with few drugs.

### TABLE II

**GROUPING OF THE ELEMENTS EMPLOYED IN THIS WORK**

| Element group | Element | Element group | Element |
|---------------|--------|---------------|--------|
| 1             | C      | 9             | Br     |
| 2             | N      | 10            | Na, K, Li |
| 3             | O      | 11            | Ca, Mg, Ba, Sr |
| 4             | S, Se  | 12            | Co, Te, Mn, Fe |
| 5             | F      | 13            | Au, Ag, Pt, Zn |
| 6             | P      | 14            | B, Ge, In, Ti |
| 7             | Cl     | 15            | La, Gd |

way, we are not working with duplicate pairs and the learning procedure is not negatively affected by unnecessary data. For the same reason, we applied a constraint on the side-effect occurrences and we decided to filter out ADRs with less than 5 occurrences in the dataset. Through these procedures, we went from having almost 309,000 associations to about 159,000 associations, with a significant reduction of side-effect multiplicity (from the original 4,251 to 2,055 side-effects with a number of occurrences equal to or greater than 5). In a further phase, we obtained a dataset of 157,000 associations, following the removal of 33 compounds who lacked any intramolecular bond and, therefore, could not be converted into a molecular graph with our current methodology. Moreover, Experiment B1 (see Section III) was performed on a version of the dataset with an additional filter, i.e. by removing all drugs that were associated to either less than 5 or more than 400 side-effects; the distribution of the number of drugs associated with DSEs is reported in Figs. 2.

Since we noticed that chemical elements do not have a uniform distribution in our dataset, we developed a grouping system so that similar elements could be viewed as the same by the GNN model, based on their physico-chemical properties (see Table II). This is fundamental because some elements have very few occurrences, meaning that the network cannot possibly learn to manage them as a standalone node type. Moreover, filling the network with too many node types would make the learning problem more difficult.

### C. Molecular Graphs

The idea behind molecular graph representation is to map the atoms and bonds that make up a molecule into sets of nodes and edges. In fact, describing the chemical structures of molecules using graphs is very natural and allows minimizing the loss of information that would be inevitable if molecular data were represented with vector encoding, disregarding the spatial arrangements and existing covalent bonds among the atoms.

However, in order to retrieve the graph representation of a specific molecule, some intermediate steps were necessary. First of all, by using PubChemPy, we retrieved the SMILES (Simplified Molecular Input Line Entry Specification) [38] string associated to the compound.¹ Such a string is a specification for unambiguously describing the chemical structure of the molecule using a short ASCII string. SMILES strings can be imported by most molecule editors for conversion back into two- or three-dimensional models of the molecules. To this aim, we have used the RDKit library² to transform the SMILE string into an RWMol (i.e. an editable molecule class defined in the RDKit). The RWMol was subsequently exploited to build a NetworkX graph of the molecular structure. Finally, this latter

---

¹ In terms of a graph–based computational procedure, SMILES is a string obtained by collecting the node symbols encountered in a depth–first traversal of a chemical graph. The chemical graph is first trimmed to remove hydrogen atoms; then, cycles are broken to turn them into spanning trees. Where cycles have been broken, numeric suffix labels are included to indicate the connected nodes. Parentheses are used to indicate branching points on the trees.

² RDKit: Open–Source Cheminformatics Software, by Greg Landrum. URL: https://www.rdkit.org/
graph was converted into a GraphObject, a Python object defined
specifically to be used as a structured graph representation for
GNNKeras [33].

In particular, the graph representation used in this work is
made of three components:

- the node matrix, where rows represent the chemical ele-
  ment (or the chemical element group, as specified in
  Table II) described by the specific node; the following
general rule applies to the node matrix:

\[ n_{ij} = \begin{cases} 1 & \text{if the } i - \text{th node belongs to} \\ & \text{the } j - \text{th element group} \\ 0 & \text{otherwise} \end{cases} \]

- the edge list, made of arrays of length 6 that describe
each edge based on the nodes it connects, along with
a label indicating the chemical bond it represents; more
specifically, an edge label is composed of:

\[ \{ n_h, n_k, b_1, b_2, b_3, b_4 \} \]

where \( n_h \) and \( n_k \) are the initial and final nodes, respectively,
while the remaining entries encode (with a one–hot rep-
resentation) the bond type (namely single, double, triple,
aromatic);

- the target vector, needed to carry out supervised learning;
it consists in a binary vector of 2055 entries (i.e., one entry
per side–effect) such that

\[ t_i = \begin{cases} 1 & \text{if the drug can cause the } i - \text{th side–effect} \\ 0 & \text{otherwise} \end{cases} \]

The data structures just described completely define a graph
that can be processed by the GNNKeras software.

III. EXPERIMENTAL RESULTS

A. Experimental Setting

Our task is the DSE prediction framed as a graph–focused
problem with multiple classes and a multi–label setting, as
each drug can cause multiple side–effects among the considered
DSEs.

For this purpose, a binary cross–entropy loss function is
used, although it is typically exploited for binary classification
problems, where the goal is to predict one of two mutually
exclusive classes (e.g., yes or no, true or false). However, in a
multi–label and multi–class classification problem, where there
are multiple classes and each data point can belong to more
than one class, the binary cross–entropy loss function can be
reformulated as

\[ \mathcal{L}_{BCE} = - \frac{1}{M|G|} \sum_j \sum_i |y_{ij}| \log(p_{ij}) \]  

(1)

where \( M \) is the number of DSE classes, \( y_{ij} \) is the label — 1 if the
\( i - \text{th} \) molecular graph represents a drug showing the \( j - \text{th} \) DSE
and 0 otherwise —, and \( p_{ij} = p(y_{ij}) \) is the predicted probability
of having the \( j - \text{th} \) DSE for all graphs in \( G \).

Using the binary cross–entropy defined in (1), each of the pre-
dicted DSEs is compared to the actual class output, calculating
an average score that penalizes the probabilities based on the
distance from the expected value. In this case, each DSE class
is considered independently from the others.

The loss function was optimized via the Adam (Adaptive
moment estimation) optimizer with its default parameters, which
has proved to be highly efficient in terms of memory consump-
tion, and appropriate for problems with noisy and/or sparse
gradients [39].

The performance presented in the following are the result
of a 5–fold cross–validation procedure on GNN–MGSEP. The
following settings differ from each other on various parameters,
such as number of epochs, batch size and stopping criteria.
These differences are presented and compared in Table III. Various
metrics were employed in order to evaluate different aspects of
the model’s performance, like the binary accuracy, which com-
putes the percentage of predicted values matching with actual
values. When using such metrics, entries of target arrays are con-
sidered independent from each other, by performing an element–
wise comparison between the predicted array and the desired
output. Despite its relevance, binary accuracy does not provide
enough information about the performance of the network, due
to the nature of the target distribution. In fact, each target array
consists of 2055 entries and each chemical compound in the
filtered dataset causes \( \approx 97 \) side–effects on average \(^3\), which
accounts for only 4.71% of the total number of target array entries:
as a consequence, a high level of binary accuracy is not neces-
sarily an indication of good network performance, since it could
be reached even in the case in which the network predictions
consisted of vectors full of zeros. For this reason, we employed
also the Area Under ROC Curve (AUC) and the Area Under
Precision Recall Curve (AUPR): they are obtained by plotting
the true positive rate against the false positive rate and by plot-
ing the positive predicted values against the true positive rate,
respectively.

B. Results

Table IV shows the results obtained in each experiment: Exp.
B1, which was carried out after a further filtering of the data,
provided the best performance in terms of binary accuracy and
AUC, while Exp. A resulted in a better AUPR.

\(^3\) More precisely, the computation of the mean results in 96.744 side–effects
per compound.

---

| Parameter | Exp. A | Exp. B | Exp. B1 | Exp. C |
|-----------|--------|--------|---------|--------|
| Batch size| 32     | 32     | 32      | 16     |
| Threshold loss| 0.15 | 0.15 | 0.14 | 0.14 |
| Epochs | 8000 | 10000 | 10000 | 7500 |
| Patience | 2000 | 2000 | 2000 | 10000 |

Exp. B and B1 were carried out using the same parameters but different datasets (see section II).
The best three experiments underwent a further analysis: Table V reports the percentage of positive predicted values, negative predicted values, specificity and sensitivity of experiments A, B and B1. Moreover, the average confusion matrix obtained in experiment B1 is shown in Fig. 3. The other experiments yielded very similar results, therefore we just report B1 as an example. For each class C, the matrix counts how many times the drugs which have class C among their target labels were classified correctly as C or misclassified, and how many times drugs without class C among their target labels were classified correctly (without C) or misclassified predicting C on them.

Summing up the results for all the classes, averaging the results over the five runs and approximating to the nearest integer value, we obtained the confusion matrix in Fig. 3. It is worth noting how different the positive predicted value and the specificity are, compared to the negative predicted value and sensitivity. In particular, a low specificity suggests that the imbalance between positive class and negative class introduces a bias in the learning procedure. Moreover, the high standard deviation measured for specificity suggests that sometimes the model is capable of managing this bias better, depending on the weight initialization and consequent learning path. This suggests that future work can focus on stabilizing the model on this aspect, in order to improve its specificity.

C. Experiments on the OFFSIDES Dataset

Usually, side-effects are determined during the last phases of the experimentation on a drug candidate, and the results of the necessary trials have a high influence on the decision about it becoming a new commercial drug or not. Yet, it is almost impossible to determine the entire spectrum of the possible adverse reactions of a drug before it enters the market. This happens because even the largest clinical trial cannot account for all the use cases the drug will undergo during its post-marketing phase. Polypharmacy effects are particularly difficult to detect and predict [28, 29], but this is beyond the scope of this work. Even sticking to single-drug adverse reactions, the rarest side-effects are often discovered and reported during the post-marketing phase. This has been the motivation behind the development of in-silico methods for the prediction of the additional/post-marketing side-effects of drugs whose side-effects are known. Of course, these models rely on the correlation between side-effects to predict new possible DSEs based on the known ones. This approach is different from ours (we have the goal of predicting side-effects of new drug candidates, the side-effects of which are unknown), yet complementary. While our method and similar ones can be applied in the early stages of the drug discovery pipeline (we rely only on the molecular structure to make predictions), post-marketing predictors can

### TABLE IV

| Metric           | Exp. A | Exp. B | Exp. B1 | Exp. C |
|------------------|--------|--------|---------|--------|
| Binary accuracy (%) | 95.16 ± 0.42 | 95.13 ± 0.43 | 95.25 ± 0.57 | 94.94 ± 0.34 |
| AUC (%)          | 86.13 ± 0.46 | 86.11 ± 0.95 | 86.73 ± 0.55 | 85.86 ± 0.33 |
| AUPR (%)         | 29.13 ± 2.22 | 28.83 ± 1.84 | 28.54 ± 3.54 | 26.82 ± 1.67 |

### TABLE V

| Metric                          | Exp. A | Exp. B | Exp. B1 | Exp. C |
|---------------------------------|--------|--------|---------|--------|
| Positive predicted value (%)    | 47.56 ± 25.84 | 45.61 ± 24.22 | 45.12 ± 24.01 |        |
| Negative predicted value (%)    | 95.89 ± 4.44    | 96.00 ± 4.33  | 96.34 ± 4.56  |        |
| Specificity (%)                 | 21.32 ± 14.59   | 23.82 ± 13.52 | 20.86 ± 14.02 |        |
| Sensitivity (%)                 | 99.05 ± 1.36    | 98.91 ± 1.45  | 99.86 ± 2.36  |        |

This was calculated removing predictions that yielded no association between 19 molecules and the side-effects.

### TABLE VI

| Metric                          | Exp. A | Exp. B | Exp. B1 | Exp. C |
|---------------------------------|--------|--------|---------|--------|
| Most frequent DSEs (%)          | 69.31 ± 11.28 | 85.39 ± 15.11 | 83.61 ± 15.61 |        |
| Least frequent DSEs (%)         | 1.50 ± 3.22    | 2.97 ± 4.52   | 5.68 ± 7.29   |        |
| Overall average (%)             | 11.18 ± 19.19  | 12.06 ± 21.71 | 10.90 ± 19.33 |        |

An analysis focused on the side-effects revealed that the relative frequency of each adverse reaction highly influences the ability of the model in detecting cases of positive associations regarding such side-effects. Table VI shows such results, by considering the 10 most frequent and less frequent DSEs in our datasets and reporting the ratio between true positive predictions and the number of occurrences of such adverse reactions. The difference in the “detectability” of side-effects based on their number of occurrences in the dataset is clearly shown.
be applied in the last stages of the same pipeline (and even use the side–effects predicted before to formulate their predictions).

Datasets of adverse reactions reported in the post–marketing phase exist and are of course used by these methods to measure their performance in this task. One of the most important such datasets is OFFSIDES [40], composed of 3,206,558 drug–DSE association entries. To see how many of these post–marketing side–effects our model could predict from scratch (based only on the drug structure), we downloaded and processed the OFFSIDES dataset. The dataset contains entries relative to 2,744 substances. We filtered out 386,844 entries about 625 substances which:

- are not molecules (f.i. grapefruit extract);
- lack a thorough definition (i.e. their Pubchem ID cannot be determined);
- are composed of a single atom (f.i. Copper, CID:23978);
- are vaccines or food (f.i. haemophilus b conjugate vaccine, cranberry juice);
- are homeopathic preparations, or in any case associated to pseudoscientific theories or traditional medicine practices without scientific bases (f.i. inulin, cystine);
- are very toxic (f.i. carbon monoxide);
- cannot be associated to a drug activity (f.i. kerosene, water).

We also deleted 123,372 entries for which the side–effect ID was not convertible to the SIDER standard and 1,386,879 entries relative to meddra LLT and other side–effects which our model had not been trained for. The remaining 1,409,463 entries relative to 2,068 drugs were submitted to our model for a test. The model was trained on our full dataset (see Section II), using the hyperparameters of experiment B1 (see Table III), and tested on this OFFSIDES extract. The test yielded very promising results, with a binary accuracy of 72.84% ± 0.27% over ten full training–test runs. On the same set of experiments we measured an AUC of 66.12% ± 0.22% and a AUPR of 45.08% ± 0.32%. Of course, the results are not comparable with those obtained on our dataset, but given the nature of the OFFSIDES dataset, they highlight the possibility of efficiently predicting even side–effects reported in the post–marketing phase, based only on the drug structure, and therefore with information available in the early stages of the drug discovery pipeline (which translates to years of advantage).

IV. DISCUSSION

A. Relevance of the Results

The experimental results described in Section III demonstrate that the DSE prediction task can be effectively carried out exploiting just the drug structure (the molecular graph) as it is done by GNN–MGSEP. A similar task had been carried out with GNN (DruGNN) by exploiting a large knowledge graph containing as much as seven main information resources: drug structural fingerprints, drug chemical properties, gene molecular function ontology, genomic information, gene–gene interactions, drug–drug similarity and drug–gene interactions [31]. The setup presented in this work is much simpler, with only drug structures needed for the prediction, yet the molecular graphs convey structural information which is very important to determine the drug functionality. This results in a simpler yet very efficient prediction framework as highlighted by the metrics.

For performance comparison, we will also take into account other predictors not based on GNNs, such as:

- Pauwels [30], which is a good structure–based baseline as it predicts DSEs based on the Sparse Canonical Correlation Analysis (SCCA) of structural fingerprints only;
- DrugClust [9], which is a good predictor based on clustering and Gene Expression (GEX) data;
- DeepSide [10] that represents a more complex predictor based on deep learning and integrates heterogeneous data from different sources.

These methods together with DruGNN constitute a very good set of models allowing to evaluate the capabilities of GNN–MGSEP in comparison to what has been achieved so far. Of course, a direct comparison is not possible as all of these methods use different data types and have been therefore trained and tested on datasets of different nature. Yet, once assessed the differences on the types of data used, the size of the datasets, and the sets of DSEs which are predicted in each case, they allow to evaluate the placement of our method with respect to other predictors.

The number of drugs taken into account, the number of predicted side–effects, as well as the information used and the method behind each predictor are described in Table VII. As demonstrated by DeepSide and DruGNN, integrating an increasing amount of heterogeneous information has been the key for improving DSE predictors so far. In particular, the former was one of the first DL approaches to the problem, while the latter introduced the use of GNNs for analyzing the knowledge graph of DSEs. GNNs though can also be exploited to analyze the structure of each molecule, since molecules are naturally represented by graphs. Moreover, molecular graphs convey the full structural information of the molecule more efficiently with respect to the structural fingerprints, which are widespread in this field. This results in comparable performance between DeepSide, DruGNN and our GNN–MGSEP, which uses a much simpler load of information consisting only of the molecular graph of each compound, therefore representing a very easy to use predictor with respect to the other two.

Some predictors use some side–effects of a drug as features to predict other side–effects. On one hand, this guarantees an information advantage that allows to reach better performance. On the other hand, this is not realistic for a new drug or drug candidate, the side–effects of which are unknown. This means that predictors of this category have a different objective with respect to ours: predicting the unreported side–effects of known drugs instead of detecting the side–effects of new drugs. It is also worth noting that similar methods exist which make predictions on unobserved/post–marketing DSEs of a drug based on its documented side–effects and other information, such as:

- Li [41], which uses Inductive Matrix Completion (IMC) to associate drugs to side–effects based on their side–effects and on the drug–protein interactions. The results of this particular method are not reported because they are not
TABLE VII
COMPARISON OF DATA AND METHODOLOGY FOR EACH PREDICTOR

| Predictor       | Num. Drugs | Num. DSEs | Method       | Data Types |
|-----------------|------------|-----------|--------------|------------|
| Paweles [30]    | 888        | 1,385     | SCCA         | SF         |
| DrugClust [9]   | 1,080      | 2,260     | Clustering   | DF+DPI+GEX |
| DeepSide [10]   | 791        | 1,042     | MLP          | GEX+GO+SF  |
| DruGNN [31]     | 1,341      | 360       | CGNN         | CF+SF+GO+PPI+DF+DSE |
| GNN–MGSEP       | 1,397      | 2,055     | GNN          | MG         |

SF stands for structural fingerprints, MG for molecular graphs, GO for gene ontology data, CF for chemical features, DPI for drug–protein interactions, DSS for drug–drug similarity, PPI for protein–protein interactions.

TABLE VIII
COMPARISON OF DATA AND METHODOLOGY FOR EACH POST-MARKETING PREDICTOR

| Predictor       | Num. Drugs | Num. DSEs | Method       | Data Types |
|-----------------|------------|-----------|--------------|------------|
| Li [41]         | 394        | 461       | IMC          | DFI+DSE    |
| Cami [42]       | 809        | 852       | LR           | DSE        |
| Zhang [43]      | 1,430      | 6,742     | MFRS         | SF+DSE     |
| Galeano [44]    | 1,525      | 2,050     | RS           | DSE        |
| GSEM [45]       | 565        | 904       | MCRS         | SF+BF+DSE  |

TABLE IX
COMPARISON OF PREDICTION PERFORMANCE WITH RESPECT TO THE OTHER DESCRIBED METHODOLOGIES

| Predictor       | Binary Accuracy | AUC | AUPR |
|-----------------|-----------------|-----|------|
| Cami [42]       | -               | 87.60% | - |
| Zhang [43]      | -               | 94.85% | 41.27% |
| Galeano [44]    | -               | 95.20% | 34.20% |
| GSEM [45]       | -               | 74.60% | - |
| Paweles [30]    | -               | 89.32% | - |
| DrugClust [9]   | -               | 91.38% | 33.36% |
| DeepSide [10]   | -               | 87.70% | - |
| DruGNN [31]     | 86.30%          | -   | - |
| GNN–MGSEP       | 95.25%          | 86.73% | 29.13% |

*DSEs are not modeled independently.

The performance of each model is reported in Table IX. It is worth noting that the predictors in the upper half of the table solve a different problem (post-marketing prediction) with respect to GNN–MGSEP, while the predictors in the second half are directly comparable with our method. Moreover, each predictor makes use of different data and predicts different sets of side-effects on different sets of drugs, as reported in Table VII and in Table VIII.

B. Usability

The usability of GNN–MGSEP is very easy thanks to the low amount of information it needs to accurately predict the occurrence of side-effects: for estimating the probable DSEs of a new drug, it will be sufficient to submit its molecular graph to the model. There is no need to retrain the model every time that a new drug is introduced in the dataset. Yet, when a significant amount of new drugs becomes available together with their labels (known occurrence of side-effects) a retraining will improve the model performance for future predictions. Our model is lightweight and training it does not require heavy amounts of resources: in our experiments we just used a commercial laptop even without a GPU. Once the model is trained, obtaining a prediction with GNN–MGSEP is even more lightweight as there is no need to load the whole dataset of molecular graphs. Therefore, our model can be used as a simple screening service to predict the occurrence of side-effects on massive amounts of molecular graphs, in the very early stages of a drug discovery pipeline.

C. Future Developments

In the future, the model can be further developed in multiple directions. On the one hand, introducing a larger amount of drug examples could improve performance of GNN–MGSEP while retaining the same simplicity and lightweight style. On the other hand, the model can be refined by integrating heterogeneous data as it is the case for DeepSide and DruGNN. The more straightforward addition that could be...
made is constituted by the chemical features of drugs, that can be retrieved from PubChem and integrated inside the molecular graph. Other data, describing drug–protein interactions, metabolomics, gene expression, and ontologies, could be integrated as well, though this would imply a rethinking of the model to attach these pieces of information to a molecular graph.

An integration with other DL tools thought for drug discovery is also possible. As GNN–MGESEP is ideal for screening huge amounts of molecular graphs, it represents a very good model for processing the output of molecular graph generators based on DL, such as ChemV AE [46], JTVAE [47], CCG-VAE [48], GraphVAE [49], MolGAN [50], or the GNN–based MG2N2 [15]. These methods can in fact produce massive amounts of possible drug candidates, but often lack the ability of evaluating the possible DSEs of the generated compounds. A three–step chain can also be devised, in which the graph generator constitutes the first step, aimed at producing a large pool of possible drug candidates. The drug candidates could then be screened for their drug–likeness, retaining only compounds with a high QED score [51] or druggability score, which can be estimated with various methods, including deep learning predictors [52], [53], [54]. Finally, GNN–MGSEP could screen the selected compounds, filtering out those with too many or too dangerous side–effects.

V. CONCLUSION

This paper presented GNN–MGSEP, a new model for the prediction of drug side–effects based on the molecular graph that describes the drug structure. A dataset of molecular graphs and associated side–effects was built in order to train and test the model. The experimental results show that the model is capable of very good performance on the task of drug side–effect prediction. Exploiting only the molecular graph, it is able to obtain comparable performance, and in some cases even better performance, with respect to the state of the art methods in such task, which need large loads of information from heterogeneous sources to formulate their predictions — even though a direct comparison is not possible due to the different nature of the data used by each predictor. The usability of GNN–MGSEP was discussed, highlighting the ease of use and lightweight training procedure of the model. Future directions of research are very promising, including the possibility of refining the predictions by integrating more data, and using GNN–MGSEP in a pipeline fully based on deep learning. In this latter framework, a graph generator outputs huge amounts of molecular graphs of possible drug candidates, which are subsequently screened for their drug–likeness using a dedicated model, and then for their side–effects using GNN–MGSEP.

ACKNOWLEDGMENT

The authors would really like to thank Professor Franco Scarselli for the very wise insights on Graph Neural Networks he gave to them when talking about this project.

REFERENCES

[1] M. L. Billingsley, “Druggable targets and targeted drugs: Enhancing the development of new therapeutics,” Pharmacology, vol. 82, no. 4, pp. 239–244, 2008.
[2] F. R. Ernst and A. J. Grizzle, “Drug–related morbidity and mortality: Updating the cost–of–illness model,” J. Amer. Pharmaceut. Assoc. (1996), vol. 41, no. 2, pp. 192–199, 2001.
[3] H. Khalil and C. Huang, “Adverse drug reactions in primary care: A scoping review,” BMC Health Serv. Res., vol. 20, no. 1, pp. 1–13, 2020.
[4] 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.
[5] S. Mizutani, E. Pauwels, V. Stoven, S. Goto, and Y. Yamashiti, “Relating drug–protein interaction network with drug side effects;” Bioinformatics, vol. 28, no. 18, pp. i522–i528, 2012.
[6] W. Zhang, Y. Chen, S. Tu, F. Liu, and Q. Qu, “Drug side effect prediction through linear neighborhoods and multiple data source integration;” in Proc. IEEE Int. Conf. Bioinf. Biomed., 2016, pp. 427–434.
[7] 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;” Pharmacogenomics J., vol. 21, no. 2, pp. 673–681, 2021.
[8] I. Shaked, M. A. Oberhardt, N. Atlas, R. Sharan, and E. Ruppin, “Metabolic network prediction of drug side effects;” Cellyst., vol. 2, no. 3, pp. 209–213, 2016.
[9] G. M. Dimitri and P. Liò, “DrugClust: A machine learning approach for drugs side effects prediction;” Comput. Biol. Chem., vol. 68, pp. 204–210, 2017.
[10] O. C. Uner, R. G. Cimbis, O. Tastan, and A. E. Cicek, “DeepSide: A deep learning framework for drug side effect prediction;” 2019, doi: https://doi.org/10.1101/843029.
[11] D. Saigusa, N. Matsukawa, E. Hishinuma, and S. Koshiba, “Identification of biomarkers to diagnose diseases and find adverse drug reactions by metabolomics;” Drug Metab. Pharmacokinetics, vol. 37, 2021, Art. no. 100373.
[12] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfaridini, “The graph neural network model;” IEEE Trans. Neural Netw., vol. 20, no. 1, pp. 61–80, Jan. 2009.
[13] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, “A comprehensive survey on graph neural networks;” IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 1, pp. 4–24, Jan. 2021.
[14] N. Pancino et al., “Graph neural networks for the prediction of protein–protein interfaces;” in Proc. ESANN, 2020, pp. 127–132.
[15] P. Bongini, M. Bianchini, and F. Scarselli, “Molecular generative graph neural networks for drug discovery;” Neurocomputing, vol. 450, pp. 243–252, 2021.
[16] P. Bongini, N. Pancino, F. Scarselli, and M. Bianchini, BioGNN: How Graph Neural Networks Can Solve Biological Problems, ch. 11. Cham, Switzerland: Springer, 2022, pp. 211–231, doi: 10.1007/978-3-031-11154-9_11.
[17] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfaridini, “Computational capabilities of graph neural networks;” IEEE Trans. Neurol. Netw., vol. 20, no. 1, pp. 81–102, Jan. 2009.
[18] J. Zhou et al., “Graph neural networks: A review of methods and applications;” AI Open, vol. 1, pp. 57–81, 2020.
[19] F. Scarselli, S. L. Yong, M. Gori, M. Hagenbuchner, A. C. Tsoi, and M. Maggini, “Graph neural networks for ranking web pages;” in Proc. IEEE/WIC/ACM Int. Conf. Web Intell., 2005, pp. 666–672.
[20] T. N. Kipf and M. Welling, “Semi–supervised classification with graph convolutional networks;” in Proc. 5th Int. Conf. Learn. Representations, 2017.
[21] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, “Spectral networks and deep locally connected networks on graphs;” in Proc. 2nd Int. Conf. Learn. Representations, 2014.
[22] M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering;” in Proc. Adv. Neural Inf. Process. Syst., 2016, pp. 3844–3852.
[23] W. Hamilton, Z. Ying, and J. Leskovec, “Inductive representation learning on large graphs;” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 1024–1034.
[24] P. W. Battaglia et al., “Relational inductive biases, deep learning, and graph networks;” 2018, arXiv:1806.01261.
[25] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, “Neural message passing for quantum chemistry;” in Proc. 34th Int. Conf. Mach. Learn., 2017, vol. 70, pp. 1263–1272.
M. Simonovsky and N. Komodakis, “GraphVAE: Towards generation of small graphs using variational autoencoders,” in Proc. 31st Int. Conf. Neural Inf. Process. Systems, pp. 412–422, 2018.

N. De Cao and T. Kipf, “On theoretical foundations and applications of deep generative models,” in Proc. Int. Conf. Mach. Learn., 2018.

G. R. Bickerton, G. V. Paolini, J. Besnard, S. Muresan, and A. L. Hopkins, “Quantifying the chemical beauty of drugs,” Nature Chem., vol. 4, no. 2, pp. 90–98, 2012.

M. Skalic, A. Varela-Rial, J. Jiménez, G. Martínez-Rosell, and G. De Fabritiis, “LigVoxel: Improving binding pockets using 3D-convolutional neural networks,” Bioinformatics, vol. 35, no. 2, pp. 243–250, 2019.

H. A. Hussein, A. Borrel, C. Geneix, M. Petitjean, L. Regad, and A.-C. Camproux, “PockDrug–server: A new web server for predicting pocket druguggability on holo and apo proteins,” Nucleic Acids Res., vol. 43, no. W1, pp. W436–W442, 2015.

P. Bongini, N. Nicolici, and M. Bianchini, “Glycine-induced formation and druguggability score prediction of protein surface pockets,” J. Bioinf. Comput. Biol., vol. 17, no. 05, 2019, Art. no. 1950026.