GeNNius: An ultrafast drug-target interaction inference method based on graph neural networks

Uxía Veleiro[1], Jesús de la Fuente[1,2,3], Guillermo Serrano[1,2], Marija Pizurica[4], Mikel Casals[2], Antonio Pineda-Lucena[1], Silve Vicent[1], Idoia Ochoa[2], Olivier Gevaert[4,*], and Mikel Hernaez[1,*]

[1] CIMA University of Navarra, IdiSNA, Pamplona, 31008, Spain
[2] TECNUN, University of Navarra, San Sebastian, 20016, Spain
[3] Center for Data Science, New York University, New York, 10012, USA
[4] Stanford Center for Biomedical Informatics Research, Dept of Medicine and Dept. Biomedical Data Science, Stanford University, Stanford, 94305, USA

Abstract. Drug-target interaction (DTI) prediction is a relevant but challenging task in the drug repurposing field. In-silico approaches have drawn particular attention as they can reduce associated costs and time commitment of traditional methodologies. Yet, current state-of-the-art methods present several limitations: existing DTI prediction approaches are computationally expensive, thereby hindering the ability to use large networks and exploit available datasets, and the generalization to unseen datasets of DTI prediction methods remains unexplored, which could potentially improve the development processes of DTI inferring approaches in terms of accuracy and robustness. In this work, we introduce GeNNius (Graph Embedding Neural Network Interaction Uncovering System), a Graph Neural Network (GNN)-based method that outperforms state-of-the-art models in terms of both accuracy and time efficiency across a variety of datasets. We also demonstrated its prediction power to uncover new interactions by evaluating not previously known DTIs for each dataset. We further assessed the generalization capability of GeNNius by training and testing it on different datasets, showing that this framework can potentially improve the DTI prediction task by training on large datasets and testing on smaller ones. Finally, we investigated qualitatively the embeddings generated by GeNNius, revealing that the GNN encoder maintains biological information after the graph convolutions while diffusing this information through nodes, eventually distinguishing protein families in the node embedding space.

Keywords: drug repurposing · drug-target interaction · bioinformatics.

1 Introduction

The process of identifying new drugs to treat a specific disease can be simplified by seeking a chemical compound that modulates a pharmacological target implicated in that disease, with the goal of altering its biological activity. Even though different biological entities can be chosen as targets, such as RNA or

proteins, the latter are the most common pharmacological targets [24]. Targeting proteins allows the modulation of many biological processes implicated in maintaining health and potentially preventing or treating diseases. For example, drugs targeting metabolic enzymes can alter how cells process nutrients [4].

Although high-throughput wet-lab techniques were developed to accelerate drug discovery pipelines, these approaches are costly and time-consuming [2]. Computational methods have arisen as promising tools to reduce the time and resources required to bring new treatments to market. The field of drug repurposing involves predicting novel drug-target interactions (DTIs) that will ultimately enable the discovery of new uses for already approved drugs [22].

Specific to DTI prediction, several different machine learning architectures have been proposed in recent years. However, most of these technologies do not consider the global view of how proteins and drugs are connected, which could be informative towards the discovery of novel relationships. To allow for modeling the network topology, recent works have been proposed to represent DTI data as a graph [18,21]. Specifically, DTIs can be modeled as a heterogeneous graph connecting drugs and proteins (both represented as nodes) based on recorded interactions in wet-lab experiments (edges). The DTI prediction model is then trained to predict whether a drug has the potential to interact with a protein.

Advances in machine learning for graphs have highlighted Graph Neural Networks (GNNs) as a powerful tool to model these complex networks. The defining characteristic of a GNN is that it uses a form of neural message passing, where at each iteration the hidden embeddings of the nodes are updated [9]. Recently, entire libraries have been developed to work with GNNs. Special mention should be made to PyTorch Geometric (PyG), a geometric deep learning library built on top of PyTorch [5]. Among other functions and layers, PyG implements the SAGEConv layer, which corresponds to the GraphSAGE operator that was originally designed to allow the training of GNNs in large networks [10]. SAGEConv simultaneously learns the topological structure of the neighborhood of each node, as well as the distribution of the features of the nodes in the neighborhood.

In this work, we present a novel DTI prediction method, termed GeNNius (Graph Embedding Neural Network Interaction Uncovering System), built upon SAGEConv layers followed by a neural network (NN)-based classifier. GeNNius outperforms state-of-the-art (SOTA) methods across several datasets, not only in the evaluation metrics but also in execution time. Additionally, we evaluated the ability of GeNNius to predict unseen DTI interactions, yielding promising results, and we assessed its generalization capability by training in one dataset and testing in a different one. Finally, we analyzed qualitatively how drug and protein features are combined during the GNN encoder, revealing that it maintains biological information while diffusing this information through nodes, eventually distinguishing protein families in the node embeddings.

Overall, the results of our evaluation provide strong support for the effectiveness of GeNNius, and introduce relevant guidelines to build GNN-based drug repurposing approaches.

2 Materials and Methods

2.1 Methods

Model architecture GeNNius architecture is composed of a Graph Neural Network (GNN) encoder that generates node embeddings and a Neural Network (NN)-based classifier that aims to learn the existence of an edge (i.e., an interaction) given the concatenation of a drug and protein node embeddings (Figure 1).

In GNNs, nodes in the graph exchange messages with their neighbors to update their feature representation, which is formulated with two fundamental functions: the message and the update functions.[31]:

$$\mathbf{m}_v^k = \sum_{u \in N(v)} M_k(\mathbf{h}_v^{k-1}, \mathbf{h}_u^{k-1}, e_{vu}). \tag{1}$$

$$\mathbf{h}_v^k = U_k(\mathbf{h}_v^{k-1}, \mathbf{m}_v^k), \tag{2}$$

where $k \in \{1, ..., K\}$ represents the layer, $\mathbf{m}_v$ the aggregated message vector for node v, $N(v)$ the neighbor nodes of v, and $\mathbf{h}_v \in \mathcal{R}^d$ the node v embedding, of dimension d. $M_k(\mathbf{h}_v, \mathbf{h}_u, e_{vu})$ defines the message between node v and its neighbor node u, which depends on the edge information e_{vu}. Finally, U_k is the node update function, which combines aggregated messages from the node's neighbors with the node's own representation.

GeNNius's encoder is composed of four SAGEConv layers, which are responsible for generating network-preserving node embeddings $\mathbf{h} \in \mathcal{R}^d$ ($d = 17$ in our case) by aggregating information from the embeddings of each node's local neighborhood. Thus, in GeNNius, the embedding of node v at SAGEConv layer k is given by:

$$\mathbf{h}_v^k = f(\mathbf{W}_0^k \mathbf{h}_v^{k-1} + \text{AGG}(\mathbf{W}_1^k \{\mathbf{h}_u^{k-1}, u \in \mathcal{N}(v)\})), \tag{3}$$

where f is the activation function (Tanh in our case) and AGG represents the aggregation function (SUM in our case). $\mathbf{W}_0^k$ and $\mathbf{W}_1^k$ are the learnable weight matrices; since we are working with heterogeneous graphs, where a drug is only connected to proteins and vice versa, if $\mathbf{W}_0^1 \in \mathcal{R}^{d \times d_P}$ then $\mathbf{W}_1^1 \in \mathcal{R}^{d \times d_N}$, or the other way around, being d_P (d_N) the initial dimension of proteins (drugs) node features. For $k > 1$, both matrices have dimension $d \times d$.

The NN-based classifier is composed of two dense layers, both using ReLu as the activation function, followed by the output layer, which is composed of a single neuron with a sigmoid activation function. The input to the classifier is a vector of dimension $2d$ (corresponding to the concatenation of a drug and protein embeddings), and the output is the estimated probability of having an interaction (positive edge).

GeNNius architecture (depicted in Figure 1) and hyperparameters were chosen through a grid search with ten independent runs, using different types and number of GNN layers, different embedding dimension d, activation functions, aggregation functions, and different number of heads for layers with attention. This approach helped us to fine-tune the model (see Supplementary Material (SM), Section 1 for a detailed description of the process and hyperparameters).

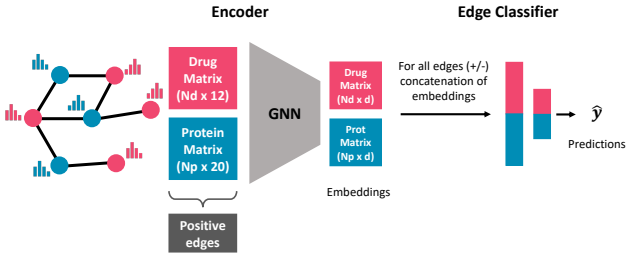


Fig. 1: Schematic of GeNNius architecture. GeNNius inputs a graph containing drug (red) and protein (blue) nodes, where N_d and N_p represent the number of drugs and proteins, respectively. First, a GNN generates node embeddings of dimension $d = 17$. Second, a NN-based classifier aims at learning the existence of an edge given a set of concatenations of drug and protein embeddings.

Model configuration The model was trained with the Adam optimizer [15] and a learning rate of 0.01. We use a loss that combines the sigmoid of the output layer and the binary cross entropy in a single function. Given a dataset divided into batches of size N, the loss l_n for sample n in the batch is computed as follows:

$$l_n = -\left[y_n \cdot \log \sigma\left(x_n\right) + \left(1 - y_n\right) \cdot \log\left(1 - \sigma\left(x_n\right)\right)\right], \tag{4}$$

where $y_n \in \{0, 1\}$ is the associated label for sample n, $\hat{y}_n = \sigma(x_n)$ the estimated probability of the sample belonging to the positive class (i.e., existence of an interaction), and x_n the output of the last linear layer (before the activation function). The final batch loss L is then computed as the average of $(l_1, \ldots, l_N)$. Finally, a dropout of 0.2 is used at the encoder stage (the GNN) to address potential collinearities of node features [29] (dropout rate chosen through hyperparameter-tuning, see SM, Section 1).

The model is implemented with early stopping, calculated on validation data, with a minimum of 40 training epochs. The latter is especially useful for small datasets where an early stop may occur during the first epochs, eventually causing underfitting. The model was built with the latest version of PyTorch Geometric (2.3.0), with PyTorch 2.0.0-cuda11.7, and the following packages: pyg-lib (0.2.0), torch-scatter (2.1.1) and torch-sparse (0.6.17). GeNNius code (incl. Dockerfile) is available at https://github.com/ubioinformat/GeNNius.

Model training and evaluation In the standard setting in which a single dataset is used to evaluate model performance, the input graph is randomly split into a 70:10:20 ratio for train, validation, and testing, respectively, via the random link split function of PyG. This function also randomly selects the negative edges needed for training and testing the NN-based classifier in a 1:1 positive/negative ratio. The training set requires further shuffling of positive and negative edges. Only 70% of training edges are used for training the encoder, while the rest are kept apart for the edge prediction part (i.e., the edge classifier).

To assess the performance of the models in the edge classification task on test data, we use the area under the Receiver Operating Characteristic curve (AUROC), as well as the area under the precision-recall curve (AUPRC), both widely used for evaluating DTI prediction models. We refer to SM Section 2 for a more extended description of these metrics.

Node features Due to the different nature of drugs and proteins, we choose a vastly different set and dimension of features for drug and protein nodes. The protein node features are encoded as a 20-dimensional vector, accounting for the 20 different amino acids, where each feature indicates the proportion of the corresponding amino acid in the protein sequence associated to the node. Drug node features are well-known molecular descriptors, calculated with RDKit [16] from their SMILES. Specifically, the 12 selected features for drug nodes are: LogP value, molecular weight, topological polar surface area, and the number of H-bond acceptors, H-bond donors, heteroatoms, rotatable bonds, rings and aromatic rings, NHs and OHs, Ns and Os, heavy atoms, and valence electrons. While some of the above-mentioned features are related, model learning and performance is not expected to deteriorate as a dropout layer was introduced to reduce the potential effect of features' collinearity (the correlation matrices are provided in SM Section 3). While other node features could be considered, such as protein pre-computed embeddings, training the model with those features showed almost no increase in performance. Moreover, these embeddings were not available for all proteins, hindering the model's generalization capabilities, specially when trained in small networks (SM Section 4).

Related work In order to benchmark our proposed method GeNNius, we focus on the latest DTI prediction models that have been shown to outperform previously developed models in their respective publications.

- **DTINet** [18] considers a heterogeneous graph with four node types (drugs, proteins, side effects and diseases) and six edge types (DTIs, protein-protein and drug-drug interactions, drug-disease, protein-disease, and drug-side-effect associations, plus a calculation of drug/protein similarities). After compact feature learning on each network drugs/proteins, it calculates the best projection of one space onto another using a matrix completion method, and then infers interactions according to the proximity criterion.
- **EEG-DTI** [21] considers a heterogeneous network similar to DTINet (see above). The model first generates a low-dimensional embedding for drugs and proteins with three Graph Convolutional Networks (GCN) layers. Then, it concatenates them for drugs and proteins separately, and calculates the inner product to get the protein-drug score.
- **HyperAttentionDTI** [35]. This method only requires the SMILES string for drugs and the amino acid sequence for proteins. Then, it embeds each character of the different sequences into vectors. The model is based on the attention mechanism and Convolutional Neural Networks (CNNs).

– **Moltrans** [12]. As HyperAttentionDTI above, it needs the SMILES for drugs and amino acid sequences for proteins. Then, it makes use of unlabeled data to decompose both drug and nodes into high-quality substructures, to later create an augmented embedding using transformers. The model is able to identify which substructures are contributing more to the DTI.

2.2 Materials

Datasets In this work we selected various datasets that have been widely used for DTI prediction tasks:

– **DrugBank** [30]. Drug-Target interactions collected from DrugBank Database Release 5.1.9. Its first release was in 2006, although it has had significant upgrades during the following years.
– **BioSNAP** [19]. Dataset created by Stanford Biomedical Network Dataset Collection. It contains proteins targeted by drugs on the U.S. market from DrugBank release 5.0.0 using MINER [26].
– **BindingDB** [17]. Database that consists of measured binding affinities, focusing on protein interactions with small molecules. The binarization of the dataset was done by considering interactions positive if their K_d was lower than 30. Data downloaded from Therapeutics Data Commons (TDC) [11].
– **Davis** [1]. Dataset of kinase inhibitors with kinases covering >80% of the human catalytic protein kinome. The binarization of the dataset has been done considering as positive interactions with a K_d lower than 30. Data downloaded from Therapeutics Data Commons (TDC) [11].
– **Yamanishi et al.** [33]. It is composed of four subsets of different protein families: Enzymes (E), Ion-Channels (IC), G-protein-coupled receptors (GPCR) and nuclear receptors (NR). Yamanishi dataset has been considered the golden standard dataset for DTI prediction and has been used in several published models [36,21]. DTIs in this dataset come from KEGG BRITE [14], BRENDA [25], SuperTarget [8] and DrugBank.

Note that the above-mentioned datasets, with the exception of BindingDB and Davis, contain only positive samples, i.e., positive links in the network. Nevertheless, when choosing negative samples, we performed random subsampling to have a balanced dataset prior to training the model. Datasets statistics are summarized in Table 1. These datasets were released in different years, and thus some drug-target interactions can be shared across datasets (See SM Section 5).

Dataset configuration for inferring unknown positives DTI datasets contain information from diverse sources, have been released in different years, and may be curated in various ways. As a result, negatively labeled edges in one dataset may be reported as positive in other datasets. We evaluate these unknown positive edges for each dataset to asses if GeNNius can predict them (see SM Section 5 for details on the number of these DTIs). Importantly, we ensured that testing edges do not appear as negatives during training to asses

Table 1: Dataset Statistics.

	DrugBank	BIOSNAP	BindingDB	Davis	E	Yamanishi GPCR	IC	NR
Number of drugs	6823	4499	3084	59	444	222	210	53
Number of proteins	4652	2113	718	218	660	94	203	25
Total number of nodes	11475	6612	3802	277	1104	316	413	78
Total number of edges	23708	13838	5937	673	2920	634	1471	86
Sparsity (%)	0.07	0.15	0.27	5.52	1.01	3.13	3.57	6.94
# Connected components	412	174	231	1	44	18	3	10
Avg degree drug nodes	3.47	3.08	1.93	11.41	6.58	2.86	7.00	1.62
Avg degree protein nodes	5.10	6.55	8.27	3.09	4.42	6.74	7.25	3.44

how well GeNNius predicts these specific interactions; we repeated the process ten independent times, enabling us to investigate the variability of the prediction depending on training edges, which is often not reported in DTI models.

Data leakage prevention during evaluation on unseen datasets Contrary to previously proposed models, we assess the generalization capability of GeNNius by training it on one dataset and testing it on another.

Let us consider two nodes that are present both in the training and test datasets. There are four possible scenarios for an edge connecting these nodes. A positive edge in both datasets is a clear example of data leakage from the train to the test set, as we already informed the model about that positive edge during training. Hence those repeated DTIs are removed during training. On the other hand, edges that appear in one dataset but not on the other one are kept. Keeping the negative edges in the training data makes sense from a usability perspective since a non-reported DTI in a given dataset does not necessarily mean that that pair does not interact, and we aim to test GeNNius' capabilities under this general scenario. Further, a negative edge may be shared in both datasets; however, since negative edges are randomly selected when generating the training and testing sets, the probability of picking the same edge in both datasets is very low. As an illustrative example, when training in DrugBank and testing in NR, the probability of selecting the same negative edge is approximately $3e^{-6}$.

We performed five independent training runs on each dataset, i.e., randomly selecting each time a different set of edges for training the model. Next, for each trained model, we performed five independent testing runs. We report the average and standard deviation of the AUROC and AUPRC metrics, of the test set, across the total 25 runs per training-testing dataset pair.

Protein and Drug Annotation Protein family and enzyme annotation was retrieved from the ChEMBL database (release 31), as its family hierarchy is manually curated and according to commonly used nomenclature [6]. Drug chemical annotation was generated using ClassyFire [3].

Hardware All simulations were performed on a server with 64 intel xeon gold 6130 2.1Ghz cores with 754Gb of RAM and a NVIDIA GeForce RTX 3080, driver version 515.43.04, with cuda 11.7. version

3 Results

3.1 GeNNius outperforms state-of-the-art methods

The proposed model was run on the eight selected datasets with five independent runs. The resulting AUROC and AUPRC metrics on the test sets across all datasets, as well as running times (corresponding to train, validation and test), are presented in Figure 2 (see also SM Section 6 where we included AUPRC results). GeNNius returned AUROC and AUPCR performance close to 1 (>0.9) for large datasets, and while smaller datasets reported worse results, they are still compelling (>0.8 in almost all runs). NR, being the smallest one, achieved the worst results (>0.7). Additionally, the large datasets showed stable results, with a low standard deviation, across the five independent runs. Further, the model execution time was ultrafast for all datasets (less than a minute). Note that the time variance in the large datasets is due to early stopping.

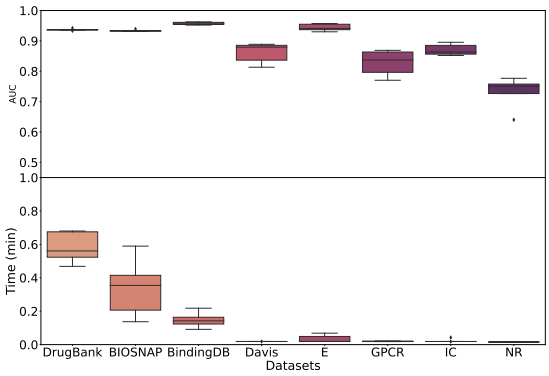


Fig. 2: Boxplots of GeNNius for five independent runs using the eight selected datasets. **Upper**. AUROC results. **Lower**. Time results in minutes.

Next, we compared the performance of GeNNius with previously proposed methods. Table 2 shows the performance results of GeNNius and the SOTA methods for both DrugBank and BIOSNAP, the largest standard DTI datasets. We focus on these datasets as they better characterize the current size of testable available drugs. GeNNius outperformed all benchmarked methods in terms of AUROC and AUPRC. Importantly, the execution time is significantly reduced, even when executed without GPU (see SM Section 7). Previous methods' running time was in the order of tens of minutes (except DTINet, which took 4.23 min), while GeNNius took less than 0.6 minutes to perform the training, validation, and testing. The closest performance in AUROC and AUPRC to GeNNius was achieved by EEG-DTI. However, EGG-DTI took four orders of magnitude more time to run (917.39 min versus 0.58 min in DrugBank). Finally, we also compared GeNNius to off-the-shelf machine learning baselines Logistic Regression (LR)

Table 2: Benchmarking results of GeNNius against four SOTA DTI methods and machine learning baselines, for BIOSNAP and DrugBank datasets. Best values are highlighted in bold, excluding baseline results. AUROC/AUPRC results correspond to test set, execution time correspond to train/validation/test. SOTA models were run in their default configuration, i.e., Moltrans correspond to 5 independent runs, while DTINet and EEG-DTI to a 10-Fold Cross Validation, and HyperAttentionDTI to 10-times repeated 5-fold Cross-Validation.

	BIOSNAP			DrugBank		
Method	AUROC	AUPRC	Time (min)	AUROC	AUPRC	Time (min)
DTINet	0.8557 ± 0.0011	0.8856 ± 0.0009	4.23	0.8154 ± 0.0004	0.8569 ± 0.0005	7.99
HyperAttentionDTI	0.8616 ± 0.0026	0.7716 ± 0.0627	66.57	0.8624 ± 0.0034	0.7756 ± 0.0456	610.45
Moltrans	0.7921 ± 0.0084	0.6452 ± 0.0037	43.35	0.7982 ± 0.0079	0.6622 ± 0.0053	122.09
EEG-DTI	0.9021 ± 0.0094	0.9046 ± 0.0098	41.39	0.8886 ± 0.0049	0.8795 ± 0.0066	917.39
GeNNius	**0.9340** ± 0.0032	**0.9349** ± 0.0021	**0.34**	**0.9371** ± 0.0033	**0.9392** ± 0.0041	**0.58**
Logistic Regression	0.6173 ± 0.0026	0.5731 ± 0.0020	0.02	0.6196 ± 0.0048	0.5747 ± 0.0035	0.06
Random Forest	0.7910 ± 0.0050	0.7519 ± 0.0056	0.05	0.7698 ± 0.0032	0.7212 ± 0.0031	0.09

and Random Forest (RF), to assess the actual improvement in accuracy using the same features (see SM Section 8 for further details). Comparing our model with LR and RF, we observed an increase in AUROC of 31.75% and 16.73%, respectively, indicating that GeNNius is superior due to its architecture: it not only uses node features but also incorporates network's topology.

3.2 GeNNius prediction capabilities for inferring previously unreported drug-target interactions

To analyze the capability of GeNNius to detect unknown interactions, we first identified those target-protein pairs lacking an edge in one dataset (negative label) but connected in the other datasets (positive label). Then, we assessed whether GeNNius was able to annotate these edges as positive. We trained the model ensuring that the edges for testing were not seen during the training process and repeated the process ten times. More detailed in Methods (Section 2.2).

The ratio of correctly predicted edges for each dataset is presented in Figure 3. When trained with large datasets, GeNNius returned good prediction capabilities, detecting more than 80% of edges in almost all cases. It is worth noting that with DrugBank it successfully predicted more than 90% of these edges.Further, when using Yamanishi datasets (E, GPCR, IC, and NR), GeNNius returned satisfactory results, predicting 70% of DTIs on average across different runs, although with higher variability than when using large datasets. This suggests that training on a small dataset hinders the inference of new interactions, as the random choice of edges for training has larger impact on the predictive power in these cases. We note that the observed outliers could be due to a non-informative random selection of training edges. Finally, the Davis dataset yielded significantly worst results than the other datasets. At first sight, this behavior could be due to the origin of the Davis dataset, as it is generated from affinity experiments. However, BindingDB, which is also generated from affinity data, does not yield such low performance. Hence, this may indicate

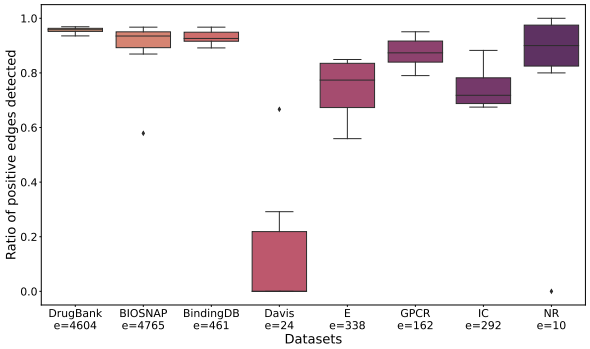


Fig. 3: Boxplot of the ratio of correctly identified positive edges in 10 independent runs. Note that e is the number of edges to be evaluated.

that the problem comes from the significant difference in the topology of Davis: it is the only dataset formed as a uniquely connected network (see Table 1).

3.3 GeNNius generalization capabilities

We evaluated GeNNius performance when training and testing on different datasets. In order to ensure that there is no data leakage that might oversimplify the prediction task, DTIs that were common to train and test datasets were discarded prior to applying the model (see Methods, Section 2.2).

AUROC results are presented in Figure 4 (AUPRC results are similar, see SM Section 9), where each entry of the heatmap shows the performance of GeNNius on the row dataset when trained on the column dataset. The reported values correspond to 25 runs, where statistical deviation in AUROC and AUPRC arise from the random selection of edges.

In general terms, GeNNius returned compelling results in its generalization capabilities; however, there was a strong dependence on the training dataset. GeNNius reported the best generalization capabilities when trained on larger datasets, such as DrugBank, BIOSNAP, and E. On the other hand, when the model is trained on the smallest dataset, NR, it cannot generalize, resulting in lower AUROC/AUPRC values compared to others (whiter colors in the NR column). Additionally, despite the Davis dataset being similar in size to other Yamanishi datasets, it returned the second-to-worst results for both training and testing. As mentioned previously, Davis' topology is different from the rest of the networks. In addition, Davis and BindingDB, unlike other datasets, come from affinity experiments. However, the latter seems to perform similarly, albeit slightly worse, than DrugBank when used for training.

We also found that, for smaller networks, our method obtains better results when trained on large datasets and tested on smaller ones compared to when trained and tested on the same small dataset. For instance, GeNNius obtained an AUROC of 0.86 when trained on DrugBank and tested on NR (lower left

corner of heatmap), while it achieved an AUROC of 0.73, using NR for training and testing (Section 3.1). This suggests that training on large networks helps the model learn and generalize to unseen and smaller datasets.

In addition, to assess how much these results depend on the node features, we compared them with a random forest model that has no information on network topology. RF showed incapability to generalize, contrasting with the results obtained when training and testing on the same dataset (Table 2). The presented results indicate that GeNNius is capable of generalizing by employing both features and the network's topology (see SM Section 9).

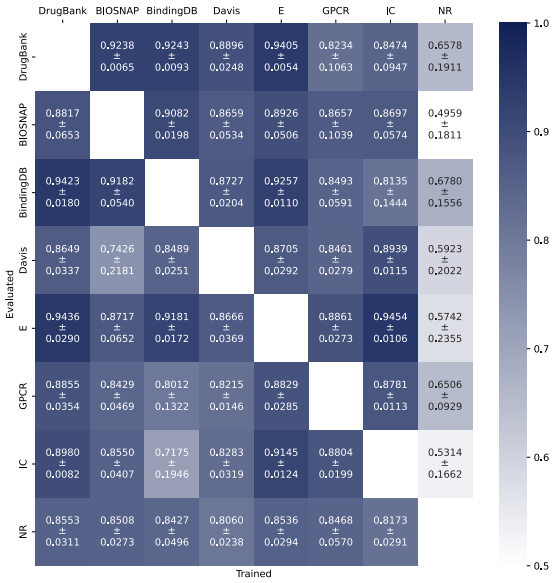


Fig. 4: Performance of GeNNius in terms of AUROC when training in one dataset (column) and testing in other (row). Train datasets do not contain positive edges that appear in the test dataset.

3.4 GeNNius encoder preserves biological information in edges and diffuses it in nodes

To qualitatively interpret the generated embeddings by GeNNius's GNN encoder, we computed the t-SNE of both the input features and the computed embeddings for all nodes and DTI edges. We focused on the DrugBank dataset, since as shown in previous sections, it reports one of the best AUROC/AUPRC results (Sections 3.1 and 3.3) and yields one of the lowest variability during DTI evaluation (Section 3.2). We aimed at shedding some light on whether the embed-

dings generated by GeNNius carry meaningful biological information beyond the ability to uncover new DTIs.

Firstly, we observe that the edge space with the input features contains information about drug chemical categories and protein families (see SM Section 10 Figures 7a,7b,7c). Using the generated embeddings instead, we observe that despite the new shapes, the biological information is conserved after graph convolutions, i.e., we can still distinguish groups by drug chemical classification but especially by protein families (see SM Section 10 Figures 7d, 7e, 7f).

Secondly, when analyzing the nodes, we found that node input features contain almost no information about protein families, i.e., nodes do not form groups by protein families, conversely to drug nodes grouped by chemical categories (see SM Section 10 Figures 8a, 8b, 8c). The next emerging question is whether the GNN encoder diffuses the biological information such that the embedding of nodes reflects it. The grouping of drug nodes concerning their chemical classification spread after applying the encoder; this is an awaited result, as we desire drugs in a DTI prediction model to be promiscuous (SM Section 10 Figure 8d). However, protein node embeddings displayed better identifiable groups than before (SM Section 10 Figure 8e). Protein families, such as membrane receptors and ion channels, revealed some grouping at the top of the figure, despite not being evident. Moreover, enzymes gathered in separate groups and, further, upon its annotation, we found a more clear grouping, e.g., kinases formed a small group on the right of the t-SNE (SM Section 10 Figure 8f).

Ultimately, the encoder maintains biological information in edge space while spreading biological information through nodes, such as protein family classification in protein nodes and sub-classification of enzymes.

4 Conclusion

We introduced a novel Drug-Target Interaction (DTI) model, termed GeNNius, composed of a GNN encoder followed by an NN-edge classifier. GeNNius outperformed state-of-the-art models in terms of AUROC and AUPRC while being several orders of magnitude faster. Further, we showed that the generalization capabilities of GeNNius and demonstrated its ability to infer previously unreported drug-target interactions. In addition, we showed that GeNNius GNN encoder exploits both node features and graph topology to maintain biological information in edge space while spreading biological information through nodes. Ultimately, GeNNius 's ability to generalize and predict novel DTIs reveals its suitability for drug repurposing. Additionally, its remarkable speed is key in its usability as it enables fast validation of multiple drug-target pairs.

Acknowledgements This work was supported by the following grants: DoD of the US - CDMR Programs [W81XWH-20-1-0262], Ramon y Cajal contracts [MCIN/AEI RYC2021-033127-I] [RYC2019-028578-I], DeepCTC [MCIN/AEI TED2021-131300B-I00], Gipuzkoa Fellows [2022-FELL-000003-01], the Spanish MCIN (PID2021-126718OA-I00), Fulbright Predoctoral Research Program [PS00342367], and FEDER/MCIN - AEI (PID2020-116344-RB-100/MCIN/AEI/10.13039/501100011033).

References

1. Davis, M.I., Hunt, J.P., Herrgard, S., Ciceri, P., Wodicka, L.M., Pallares, G., Hocker, M., Treiber, D.K., Zarrinkar, P.P.: Comprehensive analysis of kinase inhibitor selectivity. Nature biotechnology **29**(11), 1046–1051 (2011)
2. DiMasi, J.A., Grabowski, H.G., Hansen, R.W.: Innovation in the pharmaceutical industry: New estimates of r&d costs. Journal of Health Economics **47**, 20–33 (2016). https://doi.org/https://doi.org/10.1016/j.jhealeco.2016.01.012, https://www.sciencedirect.com/science/article/pii/S0167629616000291
3. Djoumbou Feunang, Y., Eisner, R., Knox, C., Chepelev, L., Hastings, J., Owen, G., Fahy, E., Steinbeck, C., Subramanian, S., Bolton, E., Greiner, R., Wishart, D.S.: Classyfire: automated chemical classification with a comprehensive, computable taxonomy. Journal of Cheminformatics **8**(1), 61 (2016)
4. Duggan, B.M., Marko, D.M., Muzaffar, R., Chan, D.Y., Schertzer, J.D.: Kinase inhibitors for cancer alter metabolism, blood glucose, and insulin. J Endocrinol **256**(2) (Feb 2023). https://doi.org/10.1530/JOE-22-0212
5. Fey, M., Lenssen, J.E.: Fast graph representation learning with PyTorch Geometric. In: RLGM Workshop at ICLR (2019)
6. Gaulton, A., Hersey, A., Nowotka, M., Bento, A.P., Chambers, J., Mendez, D., Mutowo, P., Atkinson, F., Bellis, L.J., Cibrián-Uhalte, E., Davies, M., Dedman, N., Karlsson, A., Magariños, M.P., Overington, J.P., Papadatos, G., Smit, I., Leach, A.R.: The ChEMBL database in 2017. Nucleic Acids Research **45**(D1), D945–D954 (11 2016). https://doi.org/10.1093/nar/gkw1074, https://doi.org/10.1093/nar/gkw1074
7. Grover, A.: node2vec: Scalable feature learning for networks (2016). https://doi.org/10.48550/ARXIV.1607.00653, https://arxiv.org/abs/1607.00653
8. Günther, S., Kuhn, M., Dunkel, M., Campillos, M., Senger, C., Petsalaki, E., Ahmed, J., Urdiales, E.G., Gewiess, A., Jensen, L.J., Schneider, R., Skoblo, R., Russell, R.B., Bourne, P.E., Bork, P., Preissner, R.: Supertarget and matador: resources for exploring drug-target relationships. Nucleic Acids Res **36**(Database issue), D919 (Jan 2008)
9. Hamilton, W.L.: Graph representation learning. Synthesis Lectures on Artificial Intelligence and Machine Learning **14**(3), 1–159 (2020)
10. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. CoRR **abs/1706.02216** (2017)
11. Huang, K., Fu, T., Gao, W., Zhao, Y., Roohani, Y., Leskovec, J., Coley, C.W., Xiao, C., Sun, J., Zitnik, M.: Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development (2021). https://doi.org/10.48550/ARXIV.2102.09548, https://arxiv.org/abs/2102.09548
12. Huang, K., Xiao, C., Glass, L.M., Sun, J.: MolTrans: Molecular Interaction Transformer for drug–target interaction prediction. Bioinformatics **37**(6), 830–836 (10 2020)
13. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S.A.A., Ballard, A.J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A.W., Kavukcuoglu, K., Kohli, P., Hassabis, D.: Highly accurate protein structure prediction with alphafold. Nature **596**(7873), 583–589 (2021)

14. Kanehisa, M., Goto, S., Hattori, M., Aoki-Kinoshita, K.F., Itoh, M., Kawashima, S., Katayama, T., Araki, M., Hirakawa, M.: From genomics to chemical genomics: new developments in kegg. Nucleic Acids Res **34**(Database issue), D354–7 (Jan 2006)
15. Kingma, D.P.: Adam: A method for stochastic optimization (2017)
16. Landrum, G., Tosco, P., Kelley, B., Sriniker, G., Gedeck: Rdkit: Open-source cheminformatics. (2022), https://www.rdkit.org
17. Liu, T., Lin, Y., Wen, X., Jorissen, R.N., Gilson, M.K.: Bindingdb: a web-accessible database of experimentally determined protein–ligand binding affinities. Nucleic acids research **35**(suppl_1), D198–D201 (2007)
18. Luo, Y., Zhao, X., Zhou, J., Yang, J., Zhang, Y., Kuang, W., Peng, J., Chen, L., Zeng, J.: A network integration approach for drug-target interaction prediction and computational drug repositioning from heterogeneous information. Nat Commun **8**(1), 573 (2017)
19. Marinka Zitnik, Rok Sosič, S.M., Leskovec, J.: BioSNAP Datasets: Stanford biomedical network dataset collection (2018)
20. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: An Imperative Style, High-Performance Deep Learning Library. Curran Associates Inc., Red Hook, NY, USA (2019)
21. Peng, J., Wang, Y., Guan, J., Li, J., Han, R., Hao, J., Wei, Z., Shang, X.: An end-to-end heterogeneous graph representation learning-based framework for drug–target interaction prediction. Briefings in Bioinformatics **22**(5) (2021)
22. Pushpakom, S., Iorio, F., Eyers, P.A., Escott, K.J., Hopper, S., Wells, A., Doig, A., Guilliams, T., Latimer, J., McNamee, C., Norris, A., Sanseau, P., Cavalla, D., Pirmohamed, M.: Drug repurposing: progress, challenges and recommendations. Nature Reviews Drug Discovery **18**(1), 41–58 (2019). https://doi.org/10.1038/nrd.2018.168, https://doi.org/10.1038/nrd.2018.168
23. Que, Z., Loo, M., Luk, W.: Reconfigurable acceleration of graph neural networks for jet identification in particle physics. In: 2022 IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS) (2022). https://doi.org/10.1109/AICAS54282.2022.9869941
24. Santos, R., Ursu, O., Gaulton, A., Bento, A.P., Donadi, R.S., Bologa, C.G., Karlsson, A., Al-Lazikani, B., Hersey, A., Oprea, T.I., Overington, J.P.: A comprehensive map of molecular drug targets. Nature Reviews Drug Discovery **16**(1), 19–34 (2017). https://doi.org/10.1038/nrd.2016.230, https://doi.org/10.1038/nrd.2016.230
25. Schomburg, I., Chang, A., Ebeling, C., Gremse, M., Heldt, C., Huhn, G., Schomburg, D.: Brenda, the enzyme database: updates and major new developments. Nucleic Acids Res **32**(Database issue), D431 (Jan 2004)
26. Stanford-SNAP-Group: Miner: Gigascale multimodal biological network. GitHub Repository (2017)
27. Varadi, M., Anyango, S., Deshpande, M., Nair, S., Natassia, C., Yordanova, G., Yuan, D., Stroe, O., Wood, G., Laydon, A., Žídek, A., Green, T., Tunyasuvunakool, K., Petersen, S., Jumper, J., Clancy, E., Green, R., Vora, A., Lutfi, M., Figurnov, M., Cowie, A., Hobbs, N., Kohli, P., Kleywegt, G., Birney, E., Hassabis, D., Velankar, S.: AlphaFold Protein Structure Database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. Nucleic Acids Research **50**(D1), D439–D444 (11 2021)

28. Verma, N., Qu, X., Trozzi, F., Elsaied, M., Karki, N., Tao, Y., Zoltowski, B., Larson, E.C., Kraka, E.: Ssnet: A deep learning approach for protein-ligand interaction prediction. International Journal of Molecular Sciences (2021). https://doi.org/10.3390/ijms22031392, https://www.mdpi.com/1422-0067/22/3/1392
29. Wager, S., Wang, S., Liang, P.: Dropout training as adaptive regularization (2013)
30. Wishart, D.S., Knox, C., Guo, A.C., Shrivastava, S., Hassanali, M., Stothard, P., Chang, Z., Woolsey, J.: Drugbank: a comprehensive resource for in silico drug discovery and exploration. Nucleic acids research (2006)
31. Wu, L., Cui, P., Pei, J., Zhao, L.: Graph Neural Networks: Foundations, Frontiers, and Applications. Springer Singapore, Singapore (2022)
32. Xu, C., Huang, H., Ying, X., Gao, J., Li, Z., Zhang, P., Xiao, J., Zhang, J., Luo, J.: Hgnn: Hierarchical graph neural network for predicting the classification of price-limit-hitting stocks. Information Sciences **607**, 783–798 (2022). https://doi.org/https://doi.org/10.1016/j.ins.2022.06.010, https://www.sciencedirect.com/science/article/pii/S0020025522005928
33. Yamanishi, Y., Araki, M., Gutteridge, A., Honda, W., Kanehisa, M.: Prediction of drug–target interaction networks from the integration of chemical and genomic spaces. Bioinformatics (2008)
34. Yi, H.C., You, Z.H., Huang, D.S., Kwoh, C.K.: Graph representation learning in bioinformatics: trends, methods and applications. Briefings in Bioinformatics **23**(1) (09 2021)
35. Zhao, Q., Zhao, H., Zheng, K., Wang, J.: HyperAttentionDTI: improving drug–protein interaction prediction by sequence-based deep learning with attention mechanism. Bioinformatics **38**(3), 655–662 (10 2021)
36. Zong, N., Wong, R.S.N., Yu, Y., Wen, A., Huang, M., Li, N.: Drug–target prediction utilizing heterogeneous bio-linked network embeddings. Briefings in Bioinformatics **22**(1), 568 (12 2019). https://doi.org/10.1093/bib/bbz147

Supplementary Material

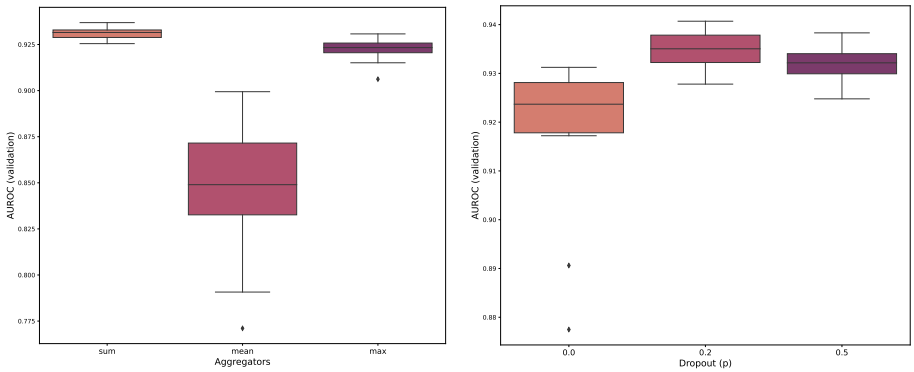
1 Hyperparameter tuning for GeNNius's GNN encoder

We performed a grid search to asses which hyperparameters are more effective for the DTI prediction task. The grid search was performed using the DrugBank dataset, for being the largest dataset available, under the idea that it may be easier for the model to generalize. The grid search tested several hyperparameters, repeating ten times each hyperparameter configuration. In all cases the dataset was randomly split into train, validation and test sets with a 70/10/20 ratio. For selecting the best-performing architecture, we used the validation set to prevent overfitting to the test set, the latter used to report results in the main text. The script used to perform the hyperparameter tuning is provided in the GitHub repository (https://github.com/ubioinformat/GeNNius), as well as a CSV file with the obtained results.

We evaluated the following hyperparameters:

- Type of layer: SAGEConv, GraphConv, GATConv, Tranformerconv. For further information about the layers and their implementation visit the PyG webpage (https://pytorch-geometric.readthedocs.io/en/latest/modules/nn.html). Note even if there are several layers implemented in PyG, not all can be used in the scenario of an heterogeneous bipartite network.
- Number of layers: from one to five, as with more than 4 layers the AUROC/AUPRC decreased considerably due to the over smoothing problem.
- Embedding dimension: from 6 to 24.
- Dropout rate: 0, 0.2, and 0.5.
- Number of heads for those models with attention (GATConv, Transformerconv): 1, 2, 4.
- Aggregators for SAGEConv and GraphConv layers: sum, mean, max.
- Activation functions: ReLu and Tanh.

The best model (i.e., hyperparameter configuration) was selected as the one with the highest average AUROC (and AUPRC) on the validation set across the ten independent runs. While the variation of some hyperparameters, such as the embedding dimension, did not affect AUROC results, others as the aggregator function of the convolutional layer and the dropout rate did. Figures 1a, and 1b show the obtained results as a function of the layer type, the aggregator function, and the dropout rate, respectively. The SUM aggregator function provided the highest AUROC values, and the dropout rate of 0.2 enhances the model's learning compared to higher values (0.5) and to no dropout at all.
The selected hyperparameters for GeNNius's GNN architecture are summarized in Table 3.



(a) Aggregators. (b) Dropout rate.

Fig. 1: Boxplot of the AUROC values obtained on the validation set with GeNNius selected architecture but varing two different hyperparameters. (a) AUROC boxplot as a function of the aggregator function., and (b) Boxplots of the AUROC as a function of the dropout rate.

Table 3: Summary of GeNNius hyperparameters.

Parameter type	Value
Layer type	SAGEConv
Number of hidden layer	4
Activation function	Tanh
Aggregation type	Sum
Embedding dimension	17
Learning rate	0.01
Dropout	0.2

2 Evaluation metrics

To assess the performance of the models in the classification task, we used the area under the ROC (Receiver Operating Characteristic) curve (AUROC), as well as the area under the precision-recall curve (AUPRC), both widely used for evaluating DTI prediction models. The ROC curve depicts the false positive rate (FPR) versus the true positive rate (TPR), defined as:

$$TPR = \frac{TP}{TP + FN}; \quad FPR = \frac{FP}{FP + TN}, \tag{5}$$

where TP, FN, and FP stand for true positives, false negatives, and false positives, respectively. The precision-recall (PR) curve plots instead the precision (P) versus the recall (R), defined as follows:

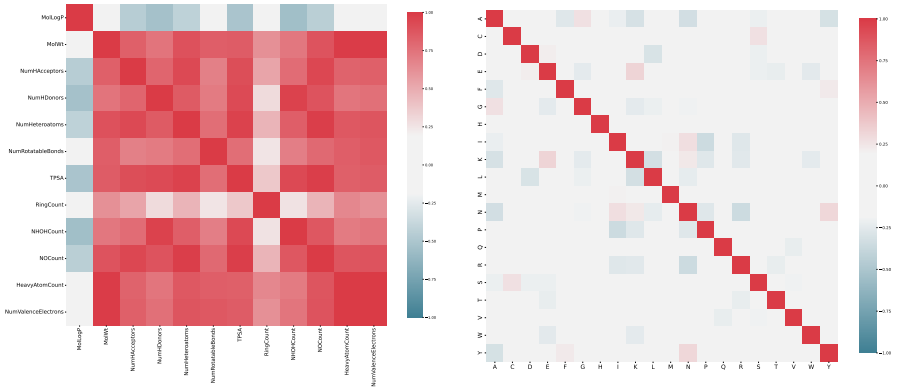
$$P = \frac{TP}{TP + FP}; \quad R = TPR = \frac{TP}{TP + FN}. \tag{6}$$

Recall that for a given input corresponding to the information of a drug and protein node, the classifier outputs a number between 0 and 1, indicating the probability of existence of and edge between the two nodes (i.e., a positive outcome). By modifying the threshold for which a decision between negative and positive outcomes is made, we can generate the ROC and PR curves.

The AUROC and AUPRC are then given by the area under the ROC or PR curves, respectively, and ranges between 0 and 1, with 0.5 corresponding to a random classifier and 1 to a perfect one. This value helps to evaluate the reliability and confidence of the models. Intuitively, positive (negative) inputs should produce high (low) probabilities. Sorting the inputs by their probability should therefore result in positive samples appearing before negative ones. In other words, a high (low) threshold should produce few FPs (FNs). Hence, the considered AUROC and AUPRC metrics show how well the model separates both classes.

3 Node Features correlation

Correlation maps of drug node features for DrugBank dataset in Figure 2a. For completeness, Figure 2b presents the correlation matrix of protein features, also for DrugBank.



(a) Drug features correlation matrix

(b) Protein features correlation matrix

Fig. 2: Correlation of node features in DrugBank. (a) Drug nodes feature correlation and (b) protein node features correlation.

4 Using pre-computed protein embeddings

To test other protein features that could be more standard in other tasks or informative to the model, we trained and tested the model using pre-computed embeddings of protein sequences, specifically embeddings retrieved from the UniProt database, corresponding to those generated using the ProtT5 model (https://www.uniprot.org/help/embeddings).

The use of those features, with the same architecture selected with GeNNius, did not increase considerably the resulting AUROC (see Table 4). Further, it seems that for smaller models such as NR the model is overfitting; for 2 out of 5 runs, NR returned 1.000 in AUROC. Moreover, in terms of generalization seemed to hamper the model. Firstly, smaller networks such as NR pre-computed embeddings were not available at Uniprot, leading to a network with not enough edges for training. Secondly, the generalization results for the rest of the available datasets were worse for some datasets (see heatmap in Figure 3). These results validate the use of amino acid ratio for the initial protein features, being also faster to generate, and available for all proteins.

Table 4: Results with protein pre-computed embedding as protein node features for the selected datasets.

Dataset	AUROC	AUPRC	Time
DrugBank	0.9304 ± 0.0072	0.9329 ± 0.0070	0.15 ± 0.09
BIOSNAP	0.9263 ± 0.0061	0.9306 ± 0.0070	0.10 ± 0.01
BindingDB	0.9475 ± 0.0094	0.9445 ± 0.0110	0.04 ± 0.02
Davis	0.6325 ± 0.0561	0.6554 ± 0.0806	0.02 ± 0.00
E	0.9382 ± 0.0060	0.9278 ± 0.0080	0.03 ± 0.01
GPCR	0.8212 ± 0.0163	0.8257 ± 0.0240	0.02 ± 0.00
IC	0.8609 ± 0.0107	0.8506 ± 0.0145	0.02 ± 0.01
NR	0.8625 ± 0.1556	0.8767 ± 0.1323	0.02 ± 0.00

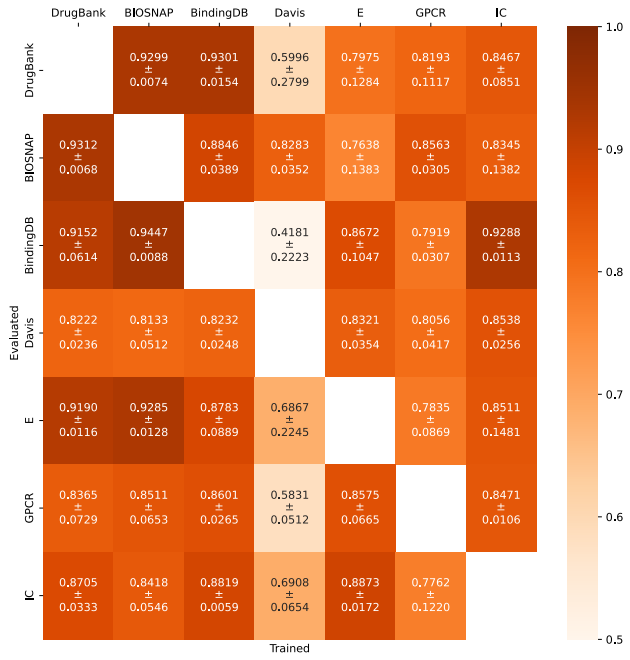
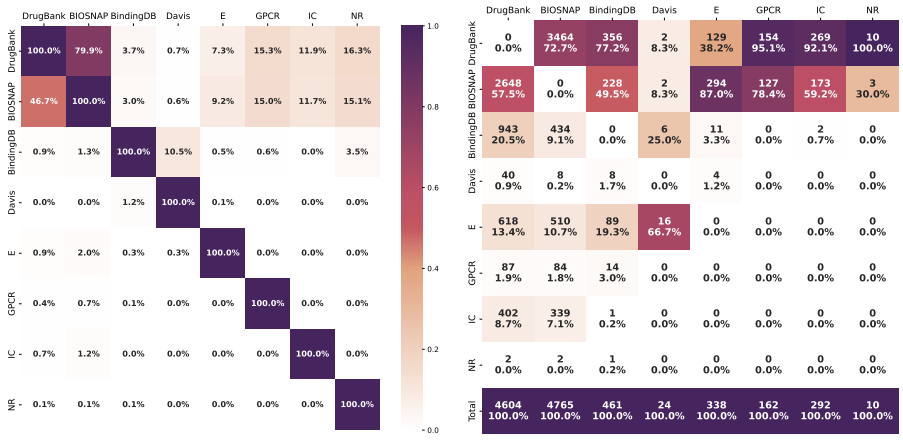


Fig. 3: Performance of GENNIUS in terms of AUROC using pre-computed embeddings as initial features ($d = 17$ for GNN encoder) when training in one dataset (column) and testing in other (row). Train datasets do not contain positive edges that appear in the testing dataset. Set-up similar to that explained in Main Section 2.2.

5 Edge analysis between datasets

In order to asses the similarities and dissimilarities between datasets, we generated four different heatmaps summarizing the percentage of shared edges (Figure 4a), and the amount of edges not reported as positive in one dataset but that were found to be positive in others (Figure 4b).



(a) Heatmap of repeated edges. (b) Heatmap of negative edges.

Fig. 4: Datasets comparison statistics. (a) heatmap of repeated edges, where each entry represents the percentage of edges that the dataset in the column shares with the one in the row. (b) Heatmap of negative edges. Each entry represents first the number of negative edges from the dataset in the column that has been registered as positives in the one in the row, and second, the percentage of edges per dataset compared to the total number to evaluate, numbers that correspond to those in Main Section 3.2. Note that some edges may be repeated, for that reason the total number may be lower than the sum of edges per dataset.

6 Additional results

Table 5 provides the average and standard deviation of the AUROC and AUPRC metrics obtained by GeNNius on the different datasets (always in the test set) across five independent runs. The running time corresponds to training, validation and testing. The same results are represented in main text Figure 2.

Table 5: Results corresponding to main text Figure 2.

Dataset	AUROC	AUPRC	Time (min)
DrugBank	0.9371 ± 0.0033	0.9392 ± 0.0041	0.58 ± 0.09
BIOSNAP	0.9339 ± 0.0032	0.9349 ± 0.0021	0.34 ± 0.18
BingingDB	0.9576 ± 0.0045	0.9552 ± 0.0019	0.15 ± 0.05
Davis	0.8607 ± 0.0338	0.8596 ± 0.0238	0.02 ± 0.00
E	0.9440 ± 0.0117	0.9321 ± 0.0190	0.03 ± 0.02
GPCR	0.8273 ± 0.0428	0.8189 ± 0.0540	0.02 ± 0.00
IC	0.8704 ± 0.0189	0.8557 ± 0.0260	0.02 ± 0.01
NR	0.7304 ± 0.0536	0.7207 ± 0.0378	0.02 ± 0.00

7 Running time in CPU

GeNNius was also run without GPU to check the difference in time with respect to GPU (see main text Table 6). Reported running times correspond to train, test and validation.

Table 6: Time results (average and std) in minutes when running the model with CPU for all considered datasets. Presented results correspond to an average of 5 runs.

	Time	
Dataset	Average	std
DrugBank	0.9188	0.3297
BIOSNAP	0.6690	0.2445
BindingDB	0.1911	0.0845
Davis	0.0188	0.0001
E	0.0534	0.0295
GPCR	0.0188	0.0001
IC	0.0240	0.0068
NR	0.0159	0.0002

8 Baseline models

To assess whether the node features or the model itself (which includes information on network topology) play an essential role, i.e., if the obtained performance metrics are mostly due to the used features, we used two baseline models that use only node features for prediction: Logistic Regression (LR) and Random Forest (RF).

For hyperparameter tuning, we used the DrugBank dataset, for being the largest dataset used in this work. The models were implemented with the sklearn Python package. Both baselines take edges (concatenation of protein and drug features) as input to classify whether they are labeled as positive (interaction) or negative (not interaction). Those edges were selected by using the same function as in GeNNius, i.e., using the *RandomLinkSplit* function by PyG. Code available in the GitHub repository.

8.1 Logistic Regression

Logistic regression (LR) is a statistical model used for binary classification problems, where the goal is to predict the probability of a binary outcome. It estimates the relationship between the input and the output variable, using a logistic function to transform the input into a probability value between 0 and 1. The selected hyperparameters are presented in Table 7.

Table 7: Summary of LR hyperparameters

Parameter type	Value
C	3.16
Max_iter	100
Penalty	'l2'

8.2 Random Forest

Random forest (RF) is an ensemble model that trains multiple decision trees and combines their outputs to make predictions. Each decision tree is built using a random subset of features and training data, which helps to reduce overfitting and increase accuracy. The selected hyperparameters are presented in Table 8.

Table 8: Summary of RF hyperparameters

Parameter type	Value
criterion	gini
max_depth	10
max_features	auto
min_samples_leaf	6
min_samples_split	5
n_estimators	50

9 Additional generalization results

The generalization AUPRC results of GeNNius are shown in Figure 5. For completeness, we also tested the generalization ability when no network topology is used. The results when using the RF baseline model in terms of AUROC and AUPRC are depicted in Figures 6a and 6b, respectively. We selected the RF model as it exhibited better performance than the LR model (see main Table 2). Details on the RF model and hyperparameters are provided in Supplementary Section 8.

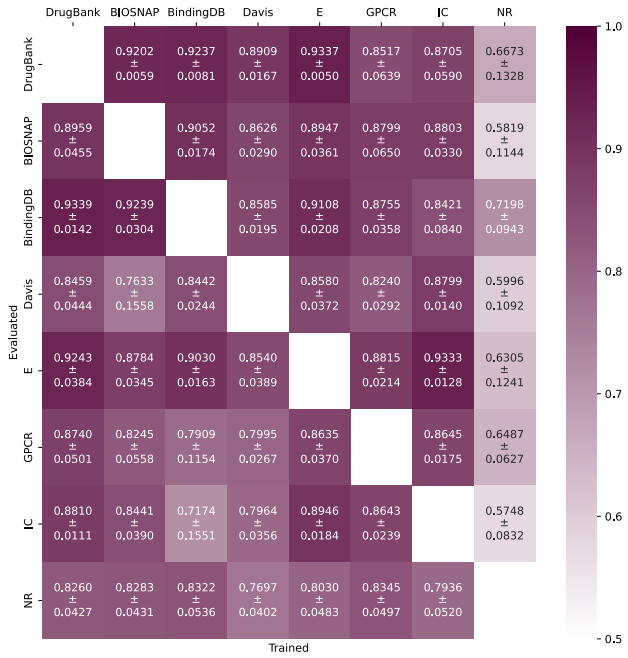
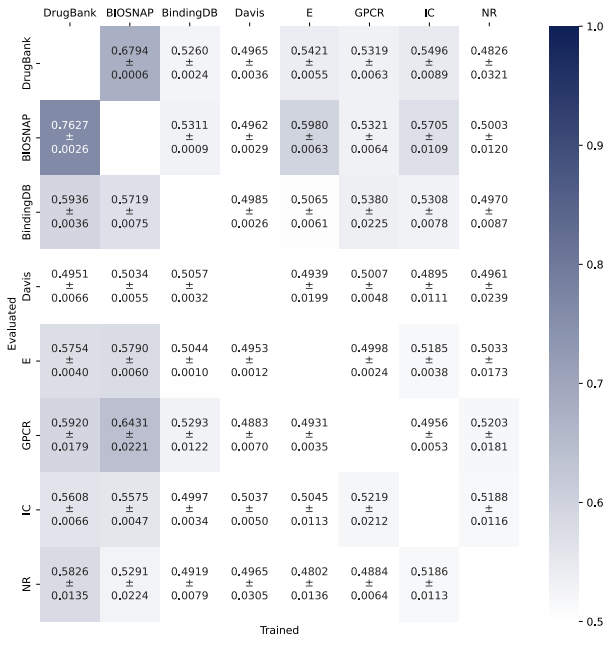
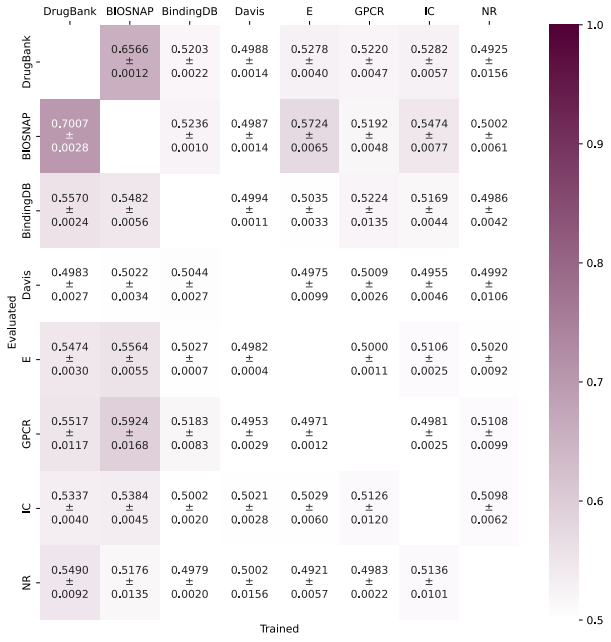


Fig. 5: Performance of GeNNius in terms of AUPRC when training in one dataset (column) and testing in other (row). Train datasets do not contain positive edges that appear in the testing dataset.



(a) AUROC Heatmap Baseline RF



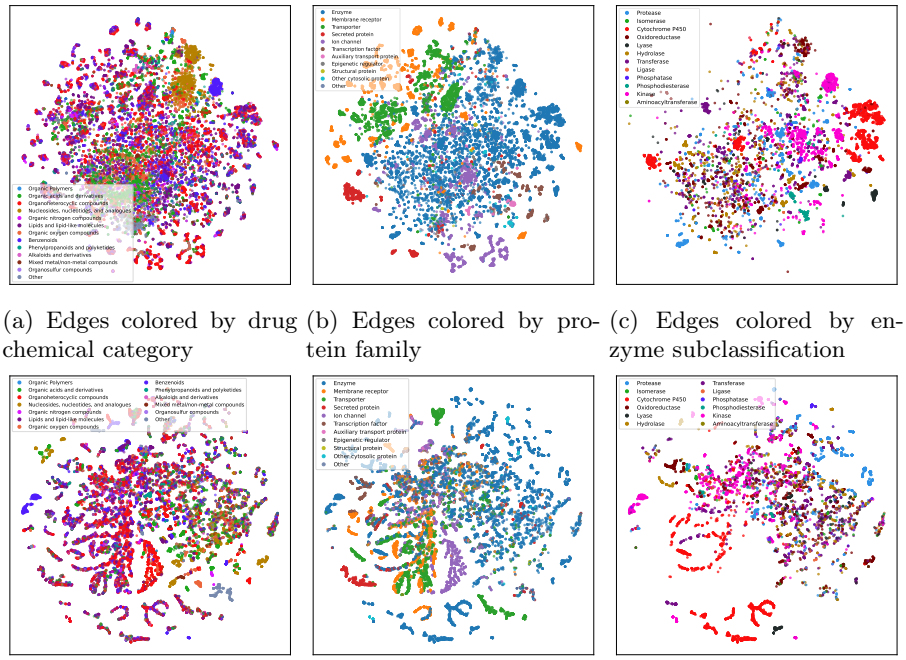
(b) AUPRC Heatmap Baseline RF

Fig. 6: Generalization performance of Random Forest in terms of (a) AUROC and (b) AUPRC (average and std), when training in one dataset (column) and testing in other (row). Train datasets do not contain positive edges that appear in the testing dataset. Results correspond to 25 runs, see methods in main article for further details on edge selection.

10 Edge and node embedding analysis

We wanted to qualitatively interpret how our GeNNius' encoder manages biological information in the network. For this, we used the DrugBank dataset; it is the largest dataset and also returned satisfactory results during all our experiments.

Firstly, we retrieved all edges in the network and concatenated the node features to get a view of the possible biological information stored in the network's topology. Then, with the resulting data, we performed a dimensionality reduction. The results indicate that the network contains information about protein families and drug classification. It is clear to distinguish edges annotated by drug classification grouping together (Figure 7a). It also happened for edges annotated by protein classification, such as groups of secreted enzymes (red), transporters (green), and (ion channels). These three examples of proteins show that the information on protein families in the network appears on the edges (Figure 7e). Similarly, this grouping appeared in the enzyme annotated Figure 7c. Next, we generated the t-SNE of the embedding space of edges to inspect whether the encoder maintains the biological information in this space and we verified that, despite the new shapes in the t-SNE, the biological information is conserved after graph convolutions, i.e., we can still distinguish groups by drug chemical classification but especially by protein families (see Figures 7d, 7e, 7f).



(a) Edges colored by drug chemical category (b) Edges colored by protein family (c) Edges colored by enzyme subclassification

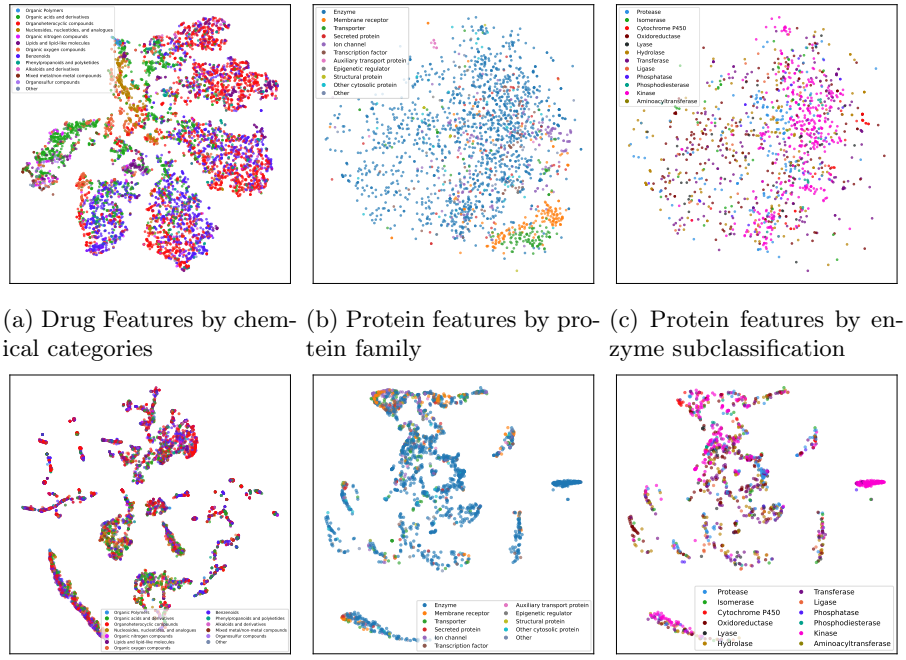
(d) edge embedding drug annot (e) edge embedding protein annot (f) edge embedding enzyme annot

Fig. 7: Edges. t-SNE paramteres for all t-SNEs: perplexity = 30, niter= 1000, selected metric was cosine with random initialization. For drugs, removed unclassified drugs, those who belong to "Other" group and those with less than 20 appearances together. For proteins removed unclassified proteins and those with less than 10 appearances.

Drug features grouped in the reduced space in a manner that resembled family branches or leaves, as expected due to the high relation between molecular descriptors and drug chemical categories (Figure 8a). Protein features, conversely, showed little to no evidence of containing information about their respective families. We could only distinguish a separation for proteins corresponding to transporters and membrane receptors; for those, this is an expected behavior, as those proteins evolved to 3D structures with membrane-bound functions, including more presence of hydrophobic amino acids (attach the protein in the lipid bilayer) and less of charged amino acids (avoid repulsion) (Figure 8b). Further, enzymes are soluble proteins that catalyze chemical reactions in the cell and contain specialized active sites. Hence, it may be expected an amino acid ratio with a higher proportion of polar or charged amino acids in the active site to interact with and stabilize, which may also depend on the type of enzyme. Some slight displacement of kinases, in pink, in Figure 8c) can be appreciated, but no strong grouping was found.

After applying GeNNius' encoder, we proceeded similar but plotting the embedded space of protein and drug nodes. We found that the grouping of drug nodes concerning their chemical classification spread after applying the encoder; this is a compelling result, as we desire drugs in a DTI prediction model to be promiscuous (8d). However, protein node embeddings displayed better identifiable groups (8e). Protein families, such as membrane receptors (orange) and ion channels (violet), revealed some grouping at the top of the Figure, despite not forming evident groups. Further, enzymes now gather in separate groups across the embedding space and, further, upon its annotation, we found a more clear group; kinases (fuchsia) formed a small group on the right of the plot; relatively similar, cytochrome P450 (red) appeared in a small group at the top; proteases (light blue) form a small group in the middle, and some oxidoreductases (brown) grouped at the left of the plot (8f).

Ultimately, the encoder maintains biological information in edge space while spreading biological information through nodes, such as protein family classification in protein nodes and sub-classification of enzymes.



(a) Drug Features by chemical categories (b) Protein features by protein family (c) Protein features by enzyme subclassification

(d) Drug embedding by chemical categories (e) Protein embedding by protein family (f) Protein embedding by enzyme subclassification

Fig. 8: Nodes. t-SNE paramteres: (a,b,c) perplexitiy = 45, niter= 850, using euclidean metric with random initialization. t-SNE parameters; (d,e,f,) perplexitiy = 30, number of iterations= 1000, using cosine metric and random initialization. For drugs, removed unclassified drugs, those who belong to "Other" group and those with less than 20 appearances together. For proteins removed unclassified proteins and those with less than 10 appearances.