Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Few-shot learning with transformers via graph embeddings for molecular property prediction

Luis H.M. Torres [*], Bernardete Ribeiro, Joel P. Arrais

Univ Coimbra, Centre for Informatics and Systems of the University of Coimbra, Department of Informatics Engineering, Coimbra, 3030-290, Portugal

ARTICLE INFO

Dataset link: https://github.com/ltorres97/FS-GNNTR

Keywords:
Few-shot learning
Transformer
Graph neural networks
Graph embedding
Molecular property prediction

ABSTRACT

Molecular property prediction is an essential task in drug discovery. Recently, deep neural networks have accelerated the discovery of compounds with improved molecular profiles for effective drug development. In particular, graph neural networks (GNNs) have played a pivotal role in identifying promising drug candidates with desirable molecular properties. However, it is common for only a few molecules to share the same set of properties, which presents a low-data problem unanswered by regular machine learning (ML) approaches. Transformer networks have also emerged as a promising solution to model the long-range dependence in molecular embeddings and achieve encouraging results across a wide range of molecular property prediction tasks. Nonetheless, these methods still require a large number of data points per task to achieve acceptable performance. In this study, we propose a few-shot GNN-Transformer architecture, FS-GNNTR to face the challenge of low-data in molecular property prediction. The proposed model accepts molecules in the form of molecular graphs to model the local spatial context of molecular graph embeddings while preserving the global information of deep representations. Furthermore, we introduce a two-module meta-learning framework to iteratively update model parameters across few-shot tasks and predict new molecular properties with limited available data. Finally, we conduct multiple experiments on small-sized biological datasets for molecular property prediction, Tox21 and SIDER, and our results demonstrate the superior performance of FS-GNNTR compared to simpler graph-based baselines. The code and data underlying this article are available in the repository, https://github.com/ltorres97/FS-GNNTR.

1. Introduction

Drug discovery is a complex and expensive operation that ranges from the discovery of innovative active ingredients to the selection of compounds with suitable *drug-like* properties. The objective is to generate a few candidate compounds ready to be tested, commercialized and administered to a patient. Despite the most recent scientific advances in pharmaceutical development, this can often be a lengthy, costly and unsuccessful venture. Once a molecule has been identified, the possibility of attrition and occurrence of side effects contribute to a high probability of failure in later stages of drug development (Hughes, Rees, Kalindjian, & Philpott, 2011; Waring et al., 2015).

In silico predictions of molecular properties such as chemical toxicity and adverse side effects can greatly improve the long and expensive process of developing new drugs. Computational methods identify potential issues before clinical trials, saving time, resources and providing more efficient and safe treatments for patients (Leelananda & Lindert, 2016). To speed up drug discovery, machine learning (ML) can be used to establish quantitative structure–activity relationships (QSAR) and characterize molecular profiles shared by multiple drug candidates. These models typically rely on fixed-sized chemical descriptors to select clinically relevant properties that can be used for appropriate drug selection (Paul et al., 2021). More recently, deep learning (DL) has emerged as a promising tool to generate expressive representations on large chemical data, which improve the efficiency of drug discovery efforts. In particular, DL methods are among the most used techniques in QSAR modeling able to generalize to non-linear systems of molecular property prediction (Gawehn, Hiss, & Schneider, 2016; Mayr, Klambauer, Unterthiner, & Hochreiter, 2016). However, molecular property prediction is a low-data problem since only a few compounds can pass virtual screening and be selected as potential drug candidates. Due to the limited amount of labeled information available in a large chemical space, DL models struggle to generalize to new molecular properties on small drug repositories. Thus, finding ways to effectively learn from a small number of labeled molecules for molecular property prediction remains a key challenge in drug discovery (Abbasi, Poso, Ghasemi, Amanlou, & Masoudi-Nejad, 2019; Ma et al., 2021).

* Corresponding author.
E-mail addresses: uc2015241578@student.uc.pt, luistorres@dei.uc.pt (L.H.M. Torres), bribeiro@dei.uc.pt (B. Ribeiro), jpa@dei.uc.pt (J.P. Arrais).

https://doi.org/10.1016/j.eswa.2023.120005
Received 27 October 2022; Received in revised form 19 March 2023; Accepted 28 March 2023
Available online 31 March 2023

Well-known benchmarks for drug discovery (e.g., Tox21, SIDER) include high-level molecular property measurements with only a few compounds sharing the same set of properties across several assay collections. This lack of supervised information prompts the need to explore models that quickly adapt to predict molecular properties with just a few data points (Deng et al., 2022; Wu et al., 2018).

Molecules can be described as heterogeneous graph structures, where atoms are represented as nodes and chemical bonds as edges. Node and edge features express the connected relations between close neighbors in a graph to promote molecular representation learning. Graph neural networks (GNNs) operate on molecular graphs to iteratively update node representations by neighborhood aggregation and compute comprehensive graph embeddings. Such graph-level representations can be used by a simple classifier to predict a given property or molecular label (Kearnes, McCloskey, Berndl, Pande, & Riley, 2016; Yang et al., 2019). Nonetheless, despite being effective at aggregating local information, GNNs struggle to capture the global aspects of node-edge connections important for molecule classification (Chen, Barzilay, & Jaakkola, 2019).

Contemporary Transformer architectures target at modeling these long-range dependencies to build deep representations from graph-level embeddings. These attention-based networks have been recently explored to preserve the global information of molecular embeddings and deliver accurate predictions of molecular properties (Maziarka & Tabor, 2019; Mialon, Chen, Selosse, & Mairal, 2021). The recent success of these emerging architectures has led to the development of a set of hybrid approaches to extract meaningful representations for several downstream applications. Such potential remains to be explored in molecular representation learning to predict the intrinsic properties of molecular structures.

Molecular property prediction involves the discovery of new chemical properties based on the molecule's structural features. This is a challenging task due to the existence of a vast chemical space and the lack of supervised experimental data available for training. However, ML and DL methods require a large amount of labeled data to create robust models to accurately predict molecular properties, which can be expensive and time-consuming to obtain. In this work, we propose a few-shot GNN-Transformer architecture, FS-GNNTR to explore the contextual information of molecular graph embeddings for molecular property prediction. To address the problem of low-data in molecular property discovery, we propose a few-shot meta-learning framework to iteratively update model parameters across few-shot tasks and enable fast adaptation to new molecular properties with limited available data. The main contributions of this study include:

1. A few-shot GNN-Transformer architecture that accepts compounds in the form of molecular graphs and explores the global-semantic structure of graph-level embeddings for molecular property prediction;
2. A few-shot learning strategy to effectively learn across few-shot tasks and generalize to task-specific molecular properties with just a few labeled molecules;
3. A two-module meta-learning framework to iteratively build task-transferable knowledge across tasks and adapt to downstream molecular property prediction tasks;
4. Experiments on multi-property prediction data, Tox21 and SIDER, to demonstrate that the proposed model consistently outperforms other graph-based methods.

2. Related work

2.1. Few-shot learning

Humans have an innate ability to learn new concepts with little or no previous demonstration. The idea is to systematically revisit previous learning iterations and define a promising strategy based on experience. Specifically, a few examples are needed to make the process faster and more efficient. This inductive-learning perspective gave rise to the few-shot learning (FSL) methods (Fei-Fei, Fergus, & Perona, 2006; Wang, Yao, Kwok, & Ni, 2020).

FSL is a meta-learning algorithm that aims to learn new representations across few-shot tasks to predict a new set of test tasks with limited available data (Sun, Liu, Chua, & Schiele, 2019). A meta-learner iteratively updates model parameters and generalizes to new experimental tasks from a limited amount of labeled data. Specifically, considering a set of training tasks t and task-specific data D_t (meta-training), the goal is to learn model parameters θ that generalize well across all learning tasks,

$$\theta' = argmin_{\theta} \sum_{t_i \sim \rho(t)} \mathcal{L}(D_{t_i}, \theta) \tag{1}$$

where $\rho(t)$ is a distribution of tasks, D_{t_i} is the data of task t_i and $\mathcal{L}$ the loss for a downstream task. Model parameters are updated to quickly adapt to the new task at hand. In the meta-testing phase, the meta-learner generalizes to unseen representations by mapping the initialized parameters θ' to new test tasks t_j with parameters

$$\theta'' = \mathcal{L}(D_{t_j}, \theta') \tag{2}$$

Typically, D_{t_j} contains just a few examples of each test task t_j.

In general, FSL is divided in two main classes of few-shot models: metric-based and optimization-based methods. Metric-based models adapt to new tasks by learning the similarity between a support set and a query. Optimization-based methods update model parameters across tasks to quickly adapt to new representations with just a few gradient steps (Finn, Abbeel, & Levine, 2017; Wang, Cheng, Yu, Guo, & Zhang, 2019). In this work, we apply an optimization-based FSL approach to face the challenge of low-data in molecular property prediction. The goal is to learn an optimized parameter configuration that generalizes well to new test tasks for molecular property prediction. The model is trained on the support set of each training task and is evaluated by computing the gradient and loss on a query. Then, the updated parameters are used to quickly adapt to new tasks and predict task-specific molecular properties using limited available data. Furthermore, FSL improves the efficiency of drug discovery efforts by reducing the number of molecules required for molecular property prediction, thereby reducing the overall cost and time required for drug development. By leveraging a FSL approach, it can be possible to identify potential drug candidates more efficiently, leading to improved outcomes in drug discovery.

In drug discovery, Altae-Tran, Ramsundar, Pappu, and Pande (2017) propose a FSL method to make meaningful predictions of molecular properties with a limited amount of labeled molecules. The proposed Iteratively Refined LSTM (IterRefLSTM) model uses graph convolution to aggregate node representations across tasks and predict task-specific molecular properties on few-shot data.

2.2. Graph representation learning

Conventional descriptors of molecular structures, such as chemical fingerprints, encode the spatial arrangement of atoms and bonds in a fixed-size feature vector. These representations have limited expressive power and oversimplify the complexity of chemical interactions and stereochemical attributes (Butler, Davies, Cartwright, Isayev, & Walsh, 2018).

GNNs have improved molecular representations to learn neural embeddings by graph representation learning. Gilmer, Schoenholz, Riley, Vinyals, and Dahl (2017) firstly introduce GNNs as powerful message-passing frameworks useful to predict molecular properties. GNNs are able to raise the expressive power by representing molecules as a set of node and edge features. These methods aggregate local information of close neighbors in a graph to update node representations and learn comprehensive graph embeddings.

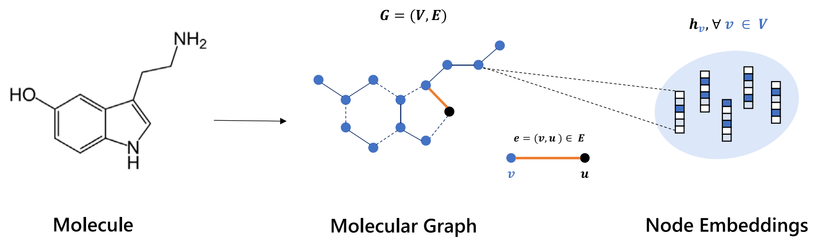


Fig. 1. Graph representation of a serotonin molecule mapped from a molecular graph $G_i \in G$ to a reduced space of vectorial embeddings $h_v \in H$.

Graph convolutional networks (GCN) are well-established GNNs in drug discovery applications. GCNs co-evolve close neighbors in the same receptive field to discover graph embedding representations. This operation resembles the application of convolutional filters on the aggregated nodes to obtain more abstract and compressed representations across GNN layers (Defferrard, Bresson, & Vandergheynst, 2016; Xu, Jegelka, Hu, & Leskovec, 2019). The improvement of GCNs led to the development of top-performing GNNs. GraphSAGE adapts graph convolution to an optimized learning procedure that uses small node-centric batches. This inductive framework aggregates node and edge features to predict unseen graph representations (Hamilton, Ying, & Leskovec, 2017). Graph attention networks (GAT) are also an extension of GCN models that perform neighborhood aggregation based on attention scores (Veličković et al., 2018). Nevertheless, graph isomorphism networks (GIN) provide the best performance results by generalizing the Weisfeiler–Lehman (WL) test to capture the relevant parts of a node's neighborhood (Xu et al., 2019).

GNNs are explored by Hu et al. (2019) to propose a novel pre-training strategy to enhance the prediction of molecular properties. This method learns local information from molecular graph embeddings to generalize to new molecular properties on few-shot data. Building on this approach, Guo et al. (2021) propose a graph-based meta-learning framework to predict new molecular properties across tasks. GNNs are optimized to meet several self-supervised objectives, such as bond reconstruction or atom type prediction.

Although GNNs are able to capture local dependencies through neighborhood aggregation, they may not retain global information. Currently, Transformer networks have proved to model such global patterns to preserve the structural information of molecular graphs (Ying et al., 2021). Thus, many efforts have been made to explore these emerging architectures in order to improve the modeling of contextual information in graph-level embeddings.

2.3. Vision transformer networks

Transformers are a class of attention networks, proposed by Vaswani et al. (2017), to solve Natural Language Processing (NLP) challenges related to machine translation and sequence-to-sequence prediction. These large models are composed by self-attention layers followed by feed-forward (FF) networks to provide a strong and effective method to better capture the global patterns in sequential data.

Vision Transformers extend the concept of standard Transformer networks to treat an input as a series of non-overlapping visual tokens called patches. Dosovitskiy et al. (2020) propose this novel architecture to outperform CNNs and address the problem of image classification in computer vision. Image inputs are split into a sequence of patch embeddings mapped into the Transformer dimension and propagated across multi-head self-attention (MSA) layers to model the long-range dependencies relevant for image classification (Touvron, Cord, Sablayrolles, Synnaeve, & Jégou, 2021). The recent success of vision Transformers gave rise to a set of hybrid approaches to combine the attention component with other neural network modules (Srinivas et al., 2021). As a result, vision Transformers can classify the output feature maps of a neural network module, which are propagated as a set of visual tokens to learn globally connected patterns across self-attention layers.

In molecular property prediction, the practical use of vision Transformers to learn expressive representations of molecular embeddings remains largely unexplored. The presented work proposes a vision Transformer architecture to compute deep representations of molecular graph embeddings across few-shot tasks and predict task-specific molecular properties using limited available data.

3. Methodology

3.1. Graph neural network module

Molecules can be described by molecular graphs where each atom is represented by a node, and each edge represents a chemical bond between atoms. Molecular graphs can be formally defined by $G = (V, E)$, where V is the set of nodes v and E is the set of edges e. An edge is defined by $e = (v, u)$, where v and u are neighboring nodes connected in a close neighborhood $N(v)$ (see Fig. 1).

GNNs convert graph representations of a node $v \in V$ into node embeddings h_v by neighborhood aggregation. Node and edge features are iteratively updated to compute graph embedding h_G representations. Such graph-level embeddings can serve as input to a simple classifier to predict a given property or molecular label.

In this section, we introduce a GNN as a graph embedding module to learn a non-linear function f that maps a molecular graph G to a graph embedding h_G.

$$f : G \mapsto h_G \tag{3}$$

GNNs operate on graph-structured data to generate graph-level embeddings by neighborhood aggregation. An AGGREGATE step iteratively updates node representations by neighborhood aggregation to learn a vectorial embedding h_v for each node $v \in V$. A COMBINE step combines neighboring nodes $u \in N(v)$ with previous representations of a node h_v^{l-1} to compute the updated embeddings h_v. After l iterations, embedded representations of a node h_v capture the contextual information within a l-hop neighborhood (Kim & Ye, 2020; Wu et al., 2019).

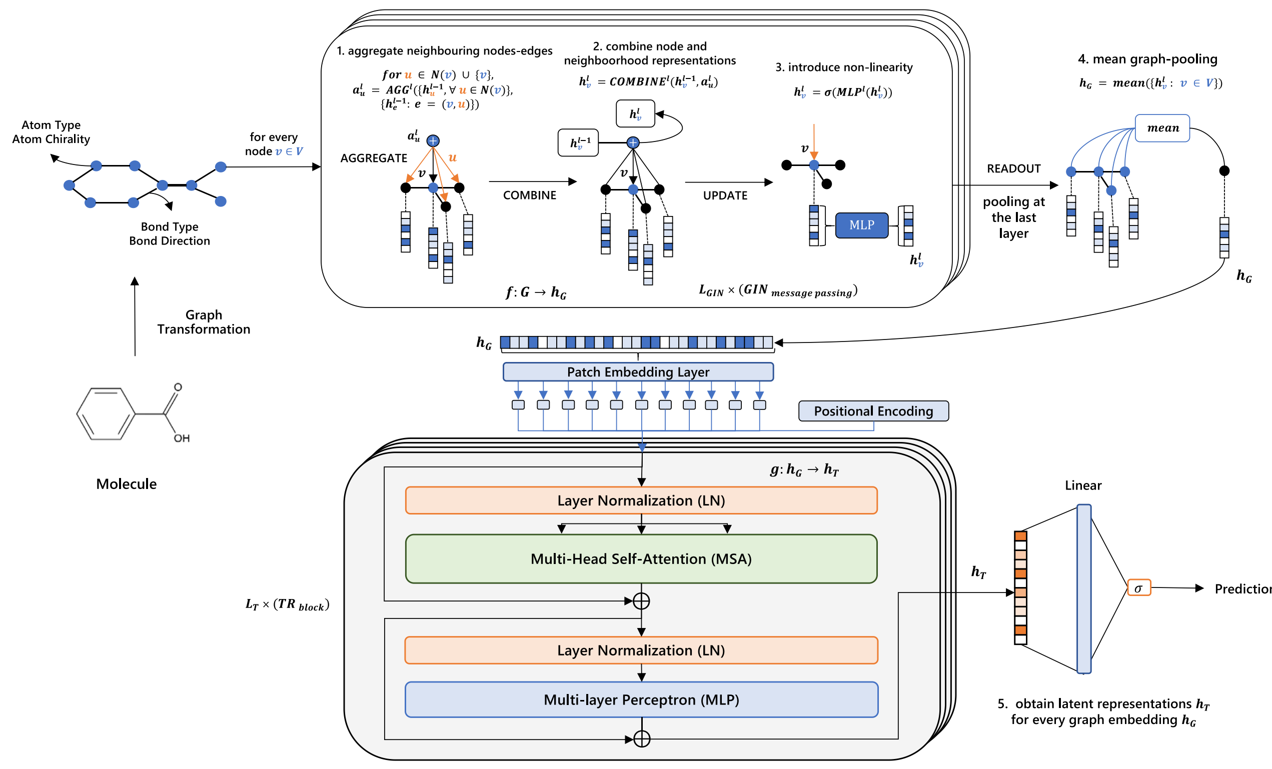
In this work, we perform node aggregation using a graph isomorphism network (GIN) (Xu et al., 2019) with $L_{GIN} = 5$ message-passing layers to obtain graph-level embeddings h_G. The proposed GIN module computes both AGGREGATE and COMBINE steps as the sum of node and edge features (Hu et al., 2019). After neighborhood aggregation, the embedded representations h_v on the lth layer are given by the AGGREGATE and UPDATE steps

$$a_u^l = AGGREGATE^l(\{h_u^{l-1} \forall u \in N(v)\}, \{h_e^{l-1} : e = (v, u)\}) \tag{4}$$

$$h_v^l = ReLU(MLP^l(COMBINE^l(h_v^{l-1}, a_u^l))) \tag{5}$$

where a is the node-edge aggregate for iteration l, h_v^l is the embedded representation of a node, h_u^l are the vectorial embeddings of neighboring nodes $u \in N(v)$, and h_e^l is the edge embedding between nodes u and $v \in V$. A multi-layer perceptron (MLP) performs the UPDATE operation followed by ReLU neural activation. More specifically, these operations are given by

$$h_v^l = ReLU(MLP^l(\sum_{u \in N(v) \cup v} h_u^{l-1} + \sum_{e=(v,u) : u \in N(v) \cup v} h_e^{l-1})). \tag{6}$$



1. aggregate neighbouring nodes-edges
2. combine node and neighboorhood representations
3. introduce non-linearity
4. mean graph-pooling
Atom Type
Atom Chirality
Bond Type
Bond Direction
for every node $v \in V$
AGGREGATE
COMBINE
UPDATE
READOUT
pooling at the last layer
MLP
mean
$f: G \to h_G$
$L_{GIN} \times (GIN_{message\ passing})$
Graph Transformation
Molecule
Patch Embedding Layer
Positional Encoding
$g: h_G \to h_T$
Layer Normalization (LN)
Multi-Head Self-Attention (MSA)
Layer Normalization (LN)
Multi-layer Perceptron (MLP)
$L_T \times (TR_{block})$
Linear
Prediction
5. obtain latent representations h_T for every graph embedding h_G

Fig. 2. Graphical depiction of the proposed model architecture, FS-GNNTR.

At the last message-passing iteration, node embeddings h_v are used to compute a graph-level representation h_G. A READOUT mean-pooling operation averages node embeddings h_v^L to obtain a graph embedding h_G,

$$h_G = mean(\{h_v^L : v \in V\}) \tag{7}$$

Molecules are described by node and edge features including atom number (AN), atom chirality (AC), bond type (BT) and bond direction (BD). These molecular attributes specify structural aspects such as the spatial relations between atoms and bonds or the chemical interaction behavior of neighboring nodes. The original node and edge inputs for aggregation can be formalized by (h_v^0, h_e^0), which are described by these molecular attributes with $h_v^0 = \{v_{AN}, v_{AC}\}$ and $h_e^0 = \{e_{BT}, e_{BD}\}$.

The GIN module is pre-trained with graph-based pre-training strategies proposed by Hu et al. (2019) to achieve better parameter initialization. In this setup, we consider 5 message-passing layers and an embedding size of 300.

The proposed model architecture is depicted in Fig. 2.[1]

3.2. Transformer encoder module

In this work, we explore a Transformer encoder architecture with $L_T = 5$ blocks to build a molecular property prediction module. As stated in previous sections, molecular graphs G are used to compute graph embedding representations h_G by neighborhood aggregation.

[1] For graph operations, the nodes being operated on are displayed in blue, and neighboring nodes shown in black. AGGREGATE, COMBINE and UPDATE operations are shown for a single node $v \in V$ and performed on all nodes in the graph, simultaneously. A GIN model with 5 message-passing layers updates node and edge embeddings (h_v, h_e) by neighborhood aggregation. A pooling operation (READOUT) pools node embeddings h_v into graph embeddings h_G. A Transformer encoder computes deep representations h_T. Blue color squares denote different values of graph embeddings. Orange color squares denote different values of deep representations.

The following step is to map graph embeddings h_G to deep vectorial representations h_T using a Transformer network g,

$$g : h_G \mapsto h_T \tag{8}$$

The standard Transformer encoder treats an input as a one-dimensional (1D) sequence of token embeddings. The proposed module acts as a vision Transformer (Beyer, Zhai, & Kolesnikov, 2022; Dosovitskiy et al., 2020) that accepts graph embeddings h_G in the form of 1D feature vectors with an embedding size of 300. Such embeddings are converted into a sequence of 1D patches in a space of dimension $D = N \times (P^2.C)$ with N the number of patches, C the number of channels and P the size of each patch. First, the Transformer receives input embeddings x and converts them into a sequence of patches x_p,

$$p(x) = [x_p^1, x_p^2, \dots, x_p^N] \tag{9}$$

where x_p^i represents the ith patch vector. Patch embeddings are obtained by linear projections of individual patches x_p into the Transformer dimension D. This process of tokenization converts input embeddings in visual tokens defined by

$$T(x) = p(x).K \tag{10}$$

where $K \in \mathbb{R}^{(P^2C)\times D}$ is a learnable linear projection for tokens. To retain position information, sinusoidal positional embeddings PE are added to the original patch embeddings. This positional encoding introduces a position-invariant trigonometric signal to locate different elements of the token embedding sequences.

$$PE_{(pos,2i)} = sin(\frac{pos}{10000^{\frac{2i}{D}}}) \tag{11}$$

$$PE_{(pos,2i+1)} = cos(\frac{pos}{10000^{\frac{2i}{D}}}) \tag{12}$$

At this point, the Transformer encoder accepts graph embedding features h_G and converts them into $N = (\lfloor \frac{300}{P} \rfloor)^2$ patches of size P. Patch embeddings are obtained by projecting individual patches in the latent space to build deep representations h_T.

Transformer blocks propagate token embeddings h_T across MSA layers. MSA takes three inputs: queries, keys, and values (q, k, v) stacked into matrices (Q, K, V) to optimize the dot-product attention operation. The self-attention mechanism computes the dot-product between each query in Q and all keys in K and applies a softmax function to calculate the attention weights for each value in V

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}})V. \tag{13}$$

These attention scores are used to weight the values and combine them to produce the output. Since we consider multiple projection heads H in MSA layers, the output attention score is given by

$$MSA(Q, K, V) = CONCAT(head_1, \dots, head_H)W \tag{14}$$

$$head_j = Attention(QW_j^Q, KW_j^K, VW_j^V) \tag{15}$$

with (W_j^Q, W_j^K, W_j^V) the projection matrices obtained by the projection of (Q, K, V) for each head j. Transformer blocks include a MSA layer followed by a MLP. MSA and MLP are preceded by layer normalization (LN) and residual connections. Considering 1D patch sequences x_p, deep representations h_T propagated across Transformer blocks l can be formulated as

$$h_T^0 = [x_p^1 K, x_p^2 K, x_p^3 K, \dots, x_p^N K] + K_{pos} \tag{16}$$

$$h_T^{l*} = MSA(LN(h_T^{l-1})) + h_T^{l-1} \tag{17}$$

$$h_T^l = MLP(LN(h_T^{l*})) + h_T^{l*} \tag{18}$$

$$y = LN(h_T^{L_T}) \tag{19}$$

where $l = \{1, \dots, L_T\}$, h_T^l are latent vectors, K are the linear projections of patches and positional encoders with $K \in \mathbb{R}^{(P^2C) \times D}$ and $K_{pos} \in \mathbb{R}^{(N+1) \times D}$ and y is the output vector.

A single linear head followed by sigmoid activation uses the output of the last Transformer block to compute the prediction of a molecular property label (condensed in a value $\in \{0, 1\}$).

Further details of the Transformer architecture are described in Appendix A and in the Supplementary Material. A graphical depiction of the Transformer module is shown in Fig. 3.

The proposed Transformer encoder goes beyond the local inductive bias of GNNs to interpret graph embeddings as a sequence of patches. Patch embeddings are sparsely viewed to compute deep representations as comprehensive descriptors expressing the long-range depedencies relevant for molecule classification.

3.3. Two-module meta-learning framework

In this work, we propose a two-module meta-learning framework to predict the molecular properties of a small amount of labeled molecules. This novel framework consists of two neural network modules: a graph neural network (GNN) and a Transformer network (TR) module. Both modules are trained to iteratively update model parameters across few-shot tasks (meta-training) using a task-specific support set for training and a disjoint query set for evaluation. The updated parameters are used to generalize to new representations in the test data (meta-testing).

Here, we focus on predicting task-specific molecular properties on small biological datasets, Tox21 and SIDER, so that $\{f_\theta(G), g_{\theta^*}(h_G)\} : M \Rightarrow \{0, 1\} \in Y$, where M is the space of all molecular graph structures G, h_G are the output graph embeddings from a GNN f_θ, g_{θ^*} is the Transformer network (TR), and Y is the molecular property labels. A GIN f_θ with model parameters θ and a Transformer g_{θ^*} with parameters θ^* are trained across few-shot tasks $t \in \{1 \dots, T\}$. For each task, meta-models f_θ and g_{θ^*} are trained on a task-specific support set S_t of molecular graphs $G_{S_{t_i}}$ and evaluated on a query set Q_t of molecular graphs $G_{Q_{t_i}}$.

In meta-training, a support set of size k is randomly sampled to serve as an input to the GNN-Transformer and compute the support losses $\mathcal{L}_t^{GNN}$, $\mathcal{L}_t^{TR}$ for each task $t \in \{1 \dots, n_{train}\}$. Support losses are then used to iteratively update model parameters $\theta \to \theta'$, $\theta^* \to \theta^{*'}$. Both meta-models compute the query losses $\mathcal{L}_t^{GNN'}$, $\mathcal{L}_t^{TR'}$ using the remaining n samples for that task. In meta-training, to update model parameters, we apply a few gradient steps

$$\theta_t = \theta - \alpha \nabla_\theta \mathcal{L}_t^{GNN}(\theta) \tag{20}$$

$$\theta_t^* = \theta^* - \alpha^* \nabla_{\theta^*} \mathcal{L}_t^{TR}(\theta^*) \tag{21}$$

where α and α^* are the size of the steps for the gradient descent updates. In meta-testing, a support set of k examples is randomly sampled for a new test task to initialize model parameters using the updated parameters from meta-training $\theta \to \theta'$, $\theta^* \to \theta^{*'}$ for each task $t \in \{n_{train} + 1 \dots, T\}$. Then, both meta-models apply the updated parameters to predict new molecular properties on a query set with the remaining n samples for that task.

In this framework, data is divided into a set of training and testing tasks. During meta-training, a set of training tasks are performed. For each task, a random support set of size (k_+, k_-) (with $k+$ positive and k_- negative samples) is used for training, and a query set is used for evaluation. Specifically, we compute the gradient of the loss with respect to model parameters using just a few examples from that task and update model parameters such that it performs well on the query set of this task. The updated parameters are later used to initialize model parameters for the next training task. In meta-testing, the updated parameters obtained from meta-training are used to initialize model parameters and generalize to new test tasks. The idea is to use the updated parameters in a way that makes it easy to adapt to the query set of these new tasks with limited available data.

Few-shot tasks consist of a random support set with a set of positive samples k_+ and negative samples k_- provided for training. The remaining data points for that task are used as a query set for evaluation. Thus, model performance is evaluated separately for each test task using a random support set of size (k_+, k_-). As reported in Section 4.3, model performance is measured on the query set of each test task. In this case, we report the performance results for $(5+, 5-)$ and $(10+, 10-)$ random support sets in 5-shot and 10-shot experiments, respectively.

The meta-learning framework is depicted in Fig. 4 and the meta-learning algorithm is also displayed below. More details about model training and implementation are provided in Supplementary material.

3.4. Loss function

The loss for both GNN and Transformer modules, $\mathcal{L}^{GNN}$ and $\mathcal{L}^{TR}$ is the binary cross-entropy loss over the predictions y' and the molecular property labels y with k the number of samples,

$$\mathcal{L} = -\frac{1}{k} \sum_{i=1}^{k} y_i \; log(y_i') + (1 - y_i) \; log(1 - y_i') \tag{22}$$

However, the datasets used (Tox21 and SIDER) are heavily imbalanced in terms of classes. This means that when training the model, the learning will become biased towards the majority class. To address this issue, a weighted version of the original learning objective is considered. Here, we introduce a weighted binary cross-entropy loss to take into account the distribution of each class to greatly penalize failed predictions on rare-class instances. The binary cross-entropy loss defines a weight c for the minority class as the ratio between positive and negative samples,

$$\mathcal{L} = -\frac{1}{k} \sum_{i=1}^{k} c \; y_i \; log(y_i') + (1 - y_i) \; log(1 - y_i') \tag{23}$$

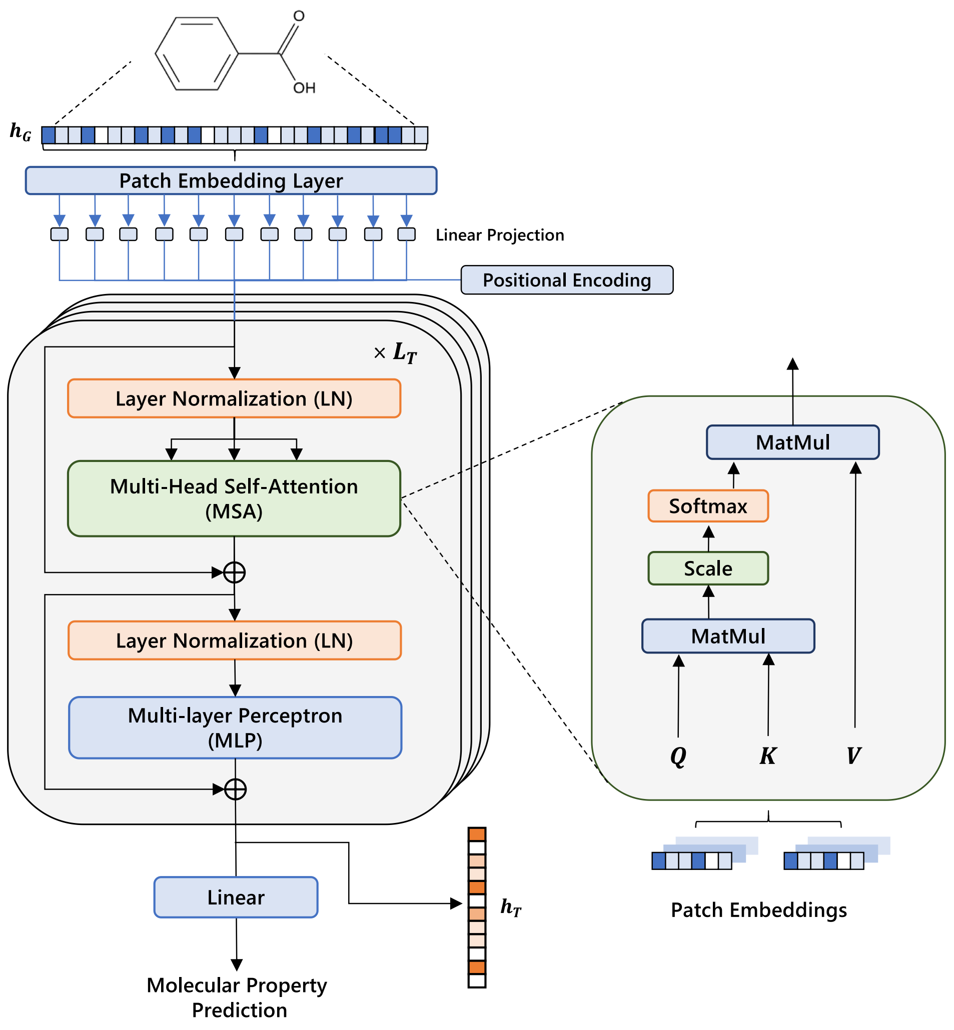


Fig. 3. Graphical representation of the Transformer encoder module.

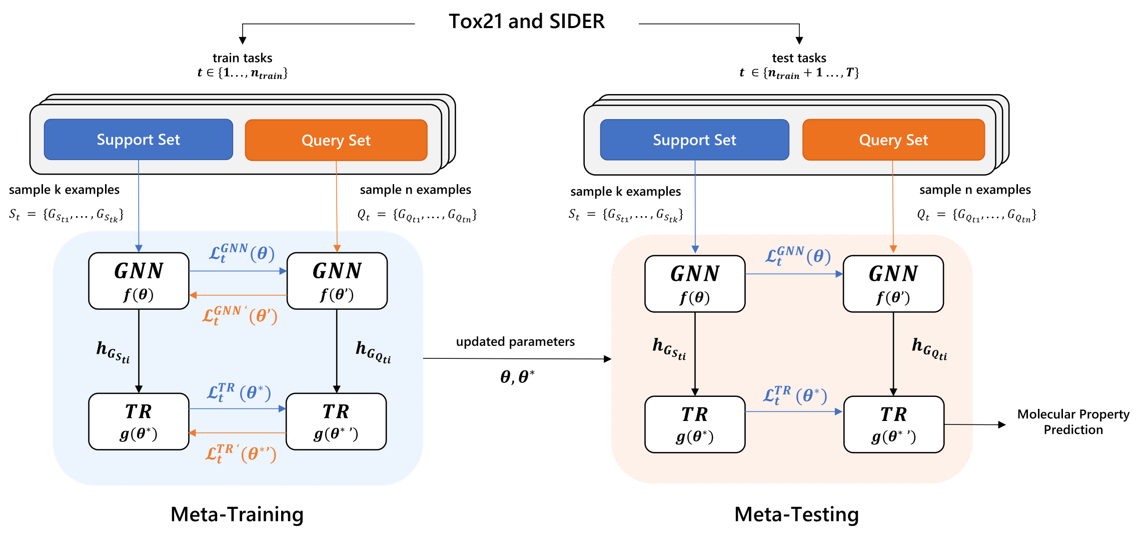

Tox21 and SIDER
train tasks
$t \in \{1 \ldots, n_{train}\}$
test tasks
$t \in \{n_{train} + 1 \ldots, T\}$
Support Set
Query Set
sample k examples
$S_t = \{G_{S_{t1}}, \ldots, G_{S_{tk}}\}$
sample n examples
$Q_t = \{G_{Q_{t1}}, \ldots, G_{Q_{tn}}\}$
GNN $f(\theta)$
GNN $f(\theta')$
$\mathcal{L}_t^{GNN}(\theta)$
$\mathcal{L}_t^{GNN'}(\theta')$
$h_{G_{S_{ti}}}$
$h_{G_{Q_{ti}}}$
TR $g(\theta^*)$
TR $g(\theta^{*\prime})$
$\mathcal{L}_t^{TR}(\theta^*)$
$\mathcal{L}_t^{TR'}(\theta^{*\prime})$
updated parameters
θ, θ^*
Molecular Property Prediction
Meta-Training
Meta-Testing

Fig. 4. Graphical schema of the proposed meta-learning framework for molecular property prediction.

Algorithm 1 : FS-GNNTR

```
Require: Support data: $(S_t, Y_t)$, Query data: $(Q_t, Y_t')$, $\alpha, \beta, \alpha^*, \beta^*$: update
  step sizes
  $\theta \leftarrow$ Pre-trained GNN
  while not done do
      Sample a batch of tasks $t \sim \rho(T)$
      for all $t$ do
          Sample $k$ examples $\{G_{S_{t_1}}, G_{S_{t_2}}, ...G_{S_{t_k}}\} \in S_t$
          for $i = 1$ to $k$ do
              $y_{t_i}, h_{G_{S_{t_i}}} = \text{GNN}(G_{S_{t_i}}, \theta)$
          end for
          $\mathcal{L}_t^{GNN} \leftarrow \{y_{t_1}, y_{t_2}, ..., y_{t_k}\}$
          for $i = 1$ to $k$ do
              $y_{t_i}, h_{T_{S_{t_i}}} = \text{TR}(h_{G_{S_{t_i}}}, \theta^*)$
          end for
          $\mathcal{L}_t^{TR} \leftarrow \{y_{t_1}, y_{t_2}, ..., y_{t_k}\}$
          $\theta' = \theta - \alpha \nabla \mathcal{L}_t^{GNN}$
          $\theta^{*'} = \theta^* - \alpha^* \nabla \mathcal{L}_t^{TR}$
          Sample $n$ examples $\{G_{Q_{t_1}}, G_{Q_{t_2}}, ...G_{Q_{t_n}}\} \in Q_t$
          for $j = 1$ to $n$ do
              $y'_{t_j}, h_{G_{Q_{t_j}}} = \text{GNN}(G_{Q_{t_j}}, \theta')$
          end for
          $\mathcal{L}_t^{GNN'} \leftarrow \{y'_{t_1}, y'_{t_2}, ..., y'_{t_n}\}$
          for $j = 1$ to $n$ do
              $y'_{t_j}, h_{T_{Q_{t_j}}} = \text{TR}(h_{G_{Q_{t_j}}}, \theta^{*'})$
          end for
          $\mathcal{L}_t^{TR'} \leftarrow \{y'_{t_1}, y'_{t_2}, ..., y'_{t_n}\}$
      end for
      $\theta \leftarrow \theta \beta \nabla_\theta \sum_{t \sim \rho(T)} \mathcal{L}_i^{GNN'}$
      $\theta^* \leftarrow \theta^* \beta^* \nabla_{\theta^*} \sum_{t \sim \rho(T)} \mathcal{L}_i^{TR'}$
  end while
```

In this case, since we have multiple tasks with different positive–negative ratios, the weight c is determined by exploring multiple values in a reasonable range. We select a value of $c = 25$ for Tox21 and $c = 1$ for SIDER due to task variability.

4. Experiments

In this study, we investigate the ability to predict molecular properties using two small biological datasets: Tox21 and SIDER. For each dataset, we conduct multiple experiments using different train-test splits.

Tox21 is a publicly available database that contains high-throughput screening results for 12 different toxic effects in 7831 molecules. These effects are measured across 12 biological targets including nuclear receptors and stress response pathways (Mayr et al., 2016). Specifically, for a total of 12 tasks we consider 9 tasks for training and 3 for testing, with each task indicating whether a given property is active or inactive for a particular molecule. In this case, tasks {SR-HSE, SR-MMP, and SR-p53} are selected for evaluation (see Table 1).

SIDER includes information on 1427 marketed drugs as well as adverse drug reactions (ADRs). The objective is to predict whether a compound triggers a side effect on 27 organ classes. Data is collected from several publications and public repositories, including drug-target interactions, drug-side effect classification, or qualitative measurements of side-effect frequency (Kuhn, Letunic, Jensen, & Bork, 2016). To evaluate the performance on SIDER, information is divided into 27 tasks – 21 for training and 6 for testing – covering all 27 organ classes. Tasks {R.U.D.: "Renal and urinary disorders", P.P.P.C.: "Pregnancy, puerperium and perinatal conditions", E.L.D.: "Ear and labyrinth disorders", C.D.: "Cardiac disorders", N.S.D.: "Nervous system disorders", I.P.P.C.: "Injury, poisoning and procedural complications"} are selected for evaluation (see Table 2).

Table 1
Tox21 comprises qualitative toxicity measurements related to 12 biological targets.

Tox21 benchmark				
Task Type	#Tasks	#Train Tasks	#Test Tasks	#Compounds
Classification	12	9	3	7831

Table 2
SIDER includes a database of marketed medicines grouped into 27 different system organ classes.

SIDER benchmark				
Task Type	#Tasks	#Train Tasks	#Test Tasks	#Compounds
Classification	27	21	6	1427

Molecules are represented using SMILES (Simplified Molecular-Input Line-Entry System) strings converted into molecular graphs using the RDKit.Chem library in Python (Landrum, 2021). RDKit was used to compute basic features including atom-type, atom chirality, bond type and bond direction for atoms and chemical bonds in a given molecule. This set of features formed the initial set of atom and bond features fed into GNN layers.

4.1. Baselines

In this work, we compare the proposed model with three graph-based baselines

1. Pre-trained GIN. GIN uses the WL test to focus on key parts of a node's neighborhood;
2. Pre-trained GCN: A GCN includes a convolutional component to promote node aggregation by co-evolving graph features in the same receptive field similar to CNN layers;
3. Pre-trained GraphSAGE: GraphSAGE is an inductive method that considers a node-centric small batch training procedure to aggregate node embeddings for unseen graph features.

All graph-based methods are pre-trained with models of Hu et al. (2019) to obtain better parameter initialization and improve generalization. A meta-learning framework is applied to all baselines to achieve comparable results.

4.2. Evaluation

In Section 4.3, we evaluate the ability of our proposed model to perform the binary classification across few-shots tasks on Tox21 and SIDER. For each test task, we randomly select a support set of k samples, and use the remaining n data points as a query set for evaluation.

Performance results are evaluated on the query set of each test task. In this work, we conduct 5-shot and 10-shot experiments with random support sets of size $(5+, 5-)$ and $(10+, 10-)$, respectively. Each experiment is repeated 30 times, using different random support sets each time, to obtain a reliable and robust estimate of the model's performance.

In Tables 3 and 4, we present the mean and standard deviation of ROC-AUC scores for 5-shot and 10-shot experiments with $(5+, 5-)$ and $(10+, 10-)$ random support sets. These results show the average results across 30 experiments for each test task on Tox21 and SIDER.

Table 3
Average ROC-AUC scores for binary classification with 5-shots on benchmark datasets Tox21 and SIDER.

Dataset	Task	GIN	GCN	GraphSAGE	FS-GNNTR (GIN+TR)	$\triangle\,(AUC)$
5-shot (5+, 5−)						
Tox21	SR-HSE	0.6131 ± 0.0077	0.6517 ± 0.0243	0.6253 ± 0.0270	**0.7724 ± 0.0020**	+0.1207
	SR-MMP	0.5779 ± 0.0073	0.6654 ± 0.0143	0.6426 ± 0.0264	**0.7860 ± 0.0009**	+0.1206
	SR-p53	0.5888 ± 0.0118	0.6257 ± 0.0207	0.6002 ± 0.0328	**0.7299 ± 0.0026**	+0.1042
	Average	0.5933	0.6476	0.6227	**0.7628**	+0.1152
SIDER	R.U.D.	0.6987 ± 0.0105	0.6024 ± 0.0068	0.6333 ± 0.0095	**0.7157 ± 0.0016**	+0.0017
	P.P.P.C.	**0.7689 ± 0.0077**	0.7203 ± 0.0124	0.7248 ± 0.0078	0.7307 ± 0.0024	−0.0382
	E.L.D.	0.7011 ± 0.0080	0.6231 ± 0.0104	0.6476 ± 0.0098	**0.7326 ± 0.0014**	+0.0315
	C.D.	0.6886 ± 0.0123	0.5983 ± 0.0091	0.6305 ± 0.0093	**0.7355 ± 0.0025**	+0.0469
	N.S.D.	0.6536 ± 0.0107	0.5937 ± 0.0190	0.5922 ± 0.0096	**0.6627 ± 0.0045**	+0.0091
	I.P.P.C.	0.7290 ± 0.0082	0.6455 ± 0.0178	0.6798 ± 0.0130	**0.7398 ± 0.0019**	+0.0108
	Average	0.7067	0.6306	0.6514	**0.7195**	+0.0128

Table 4
Average ROC-AUC scores for binary classification with 10-shots on benchmark datasets Tox21 and SIDER.

Dataset	Task	GIN	GCN	GraphSAGE	FS-GNNTR(GIN+TR)	$\triangle\,(AUC)$
10-shot (10+, 10−)						
Tox21	SR-HSE	0.6533 ± 0.0118	0.6509 ± 0.0226	0.6661 ± 0.0262	**0.7802 ± 0.0020**	+0.1141
	SR-MMP	0.6240 ± 0.0108	0.6601 ± 0.0228	0.6910 ± 0.0236	**0.7964 ± 0.0017**	+0.1054
	SR-p53	0.6280 ± 0.0120	0.6298 ± 0.0109	0.6346 ± 0.0249	**0.7468 ± 0.0021**	+0.1122
	Average	0.6351	0.6469	0.6639	**0.7745**	+0.1106
SIDER	R.U.D.	0.6893 ± 0.0034	0.5963 ± 0.0104	0.6361 ± 0.0060	**0.7181 ± 0.0025**	+0.0288
	P.P.P.C.	**0.7786 ± 0.0077**	0.7094 ± 0.0176	0.7362 ± 0.0091	0.7566 ± 0.0043	−0.0220
	E.L.D.	0.7090 ± 0.0027	0.6137 ± 0.0157	0.6486 ± 0.0133	**0.7259 ± 0.0025**	+0.0169
	C.D.	0.6760 ± 0.0062	0.5908 ± 0.0099	0.6326 ± 0.0076	**0.7244 ± 0.0023**	+0.0454
	N.S.D.	**0.6593 ± 0.0093**	0.5789 ± 0.0128	0.5972 ± 0.0207	0.6537 ± 0.0050	−0.0056
	I.P.P.C.	0.7399 ± 0.0064	0.6418 ± 0.0117	0.6910 ± 0.0084	**0.7484 ± 0.0019**	+0.0085
	Average	0.7086	0.6218	0.6570	**0.7212**	+0.0126

4.3. Results and discussion

In this study, we present a novel architecture called FS-GNNTR, a few-shot GNN-Transformer to accurately predict molecular properties using a limited amount of labeled data. The proposed model leverages deep representations of molecular graph embeddings to outperform simpler graph-based methods in downstream molecular property prediction tasks. To evaluate the effectiveness of our approach, we conduct multiple experiments on small data collections for chemical toxicity and side effect prediction, Tox21 and SIDER. These datasets include high-variance observations of highly uncertain indicators, which pose a challenge for few-shot learning. Thus, the goal is to perform few-shot classification and learn new classes of molecular properties for these small drug repositories.

A two-module meta-learning framework iteratively updates model parameters to predict molecular properties with limited available data. Knowledge is transferred across few-shot tasks to provide strong boosts in low-data learning and promote fast adaptation to new experimental tasks similar or marginally identical to those found in training. The experiments go further and show that FS-GNNTR takes a step forward in modeling the global-semantic structure of molecular graph embeddings. A GNN module learns local patterns by converting molecular graphs into graph embeddings using neighborhood aggregation. A Transformer module exploits the global information of these vectorial embeddings across attention layers to infer task-specific properties, such as toxicity or drug side effects.

The results in Tables 3 and 4 report the average ROC-AUC scores obtained across 30 experiments with 30 (5+, 5−) and (10+, 10−) random support sets, respectively. ROC-AUC scores in Table 3 demonstrate that FS-GNNTR outperforms the best graph-based baselines on Tox21 for all test tasks and for 5 test tasks on SIDER. For (5+, 5−) random support sets, we report an average improvement of +11.52% and +1.28% for Tox21 and SIDER, respectively. ROC-AUC scores in Table 4 confirm these results for (10+, 10−) random support sets. The proposed model outperforms the best baseline method for all test tasks on Tox21 and 4 test tasks on SIDER. In 10-shot experiments, ROC-AUC scores report an average overall improvement of +11.06% and +1.26% for Tox21 and SIDER, respectively. Boxplots 5 and 6 show the average performance results across 30 experiments on Tox21 and SIDER. The dashed line denotes the average ROC-AUC score across all test tasks using FS-GNNTR for both data collections. As illustrated by these results, the proposed model demonstrates a superior performance for both assay collections. Thus, for 5-shot and 10-shot experiments, FS-GNNTR strongly dominates other simpler graph-based baselines. In this case, the standard deviations reported by the proposed model also indicate a smaller variance, which ensures a stable performance through more robust results.

For both data collections, there are notable differences in performance among few-shot tasks. SIDER includes a set of high-variance observations of high-level molecular property measurements, which leads to unstable performances and volatile predictions due to task variability and a small number of samples per task. On the other hand, Tox21 has a large number of data points per task and greater similarity among test tasks, resulting in higher ROC-AUC scores and lower variances with larger support sets. On the contrary, graph-based baselines produce volatile predictions for individual tasks. This means that they may generalize well in some cases, but collapse for the vast majority of tasks.

In Section 4.5, t-SNE visualizations compare deep representations and graph-level embeddings for molecular property prediction. We compare t-SNE cluster plots to demonstrate that deep representations perform better in mapping positive and negative samples in the reduced space for task-specific molecular properties.

The results obtained by FS-GNNTR on Tox21 and SIDER demonstrate the effectiveness of the proposed multi-task meta-learning framework and its potential to be applied in the field of quantitative structure–activity relationship (QSAR) modeling. The proposed model allows the learning in an environment with little supervised information and high class imbalance without negatively impacting the predictive performance. In addition, FS-GNNTR can capture the complex interactions

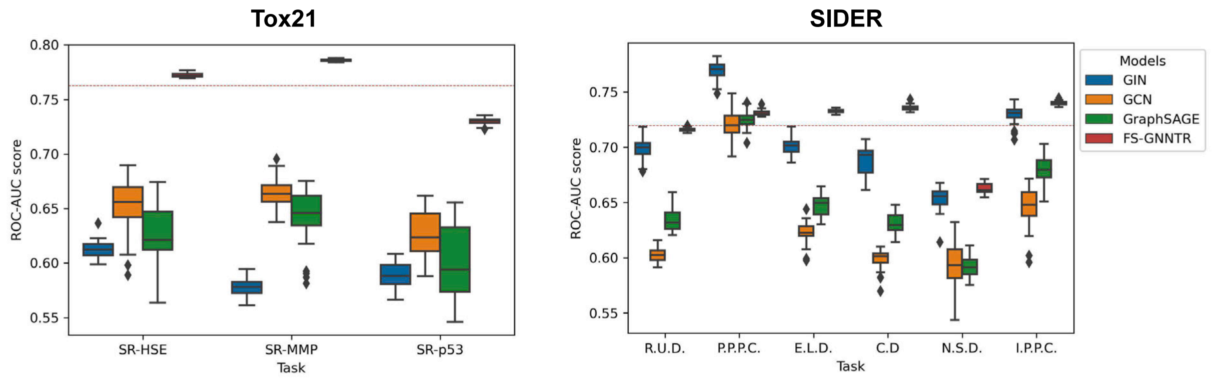


Fig. 5. Distribution of ROC-AUC scores for 5-shot experiments on Tox21 and SIDER.

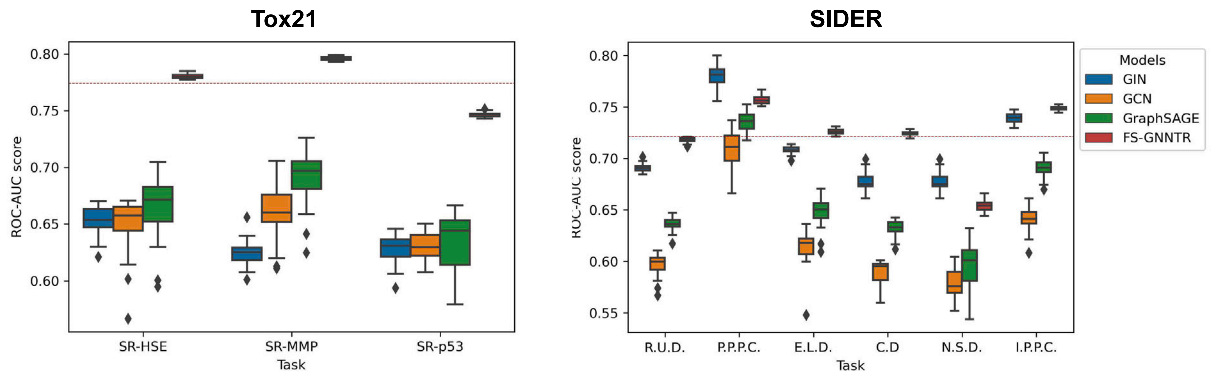


Fig. 6. Distribution of ROC-AUC scores for 10-shot experiments on Tox21 and SIDER.

between atoms and chemical bonds to predict molecular properties based on the graph embedding representation of the molecule. This method could be used to prioritize and exclude compounds with the highest predicted activity for multiple task-specific properties, such as chemical toxicity and adverse side effects, reducing the time and cost associated with experimental testing. Thus, FS-GNNTR could act as an efficient QSAR method to predict molecular properties with limited available data and facilitate drug discovery by providing a faster and more effective way of identifying potential *drug-like* candidates.

In Supplementary material, further details about the experiments conducted and additional experimental results are included to provide a more thorough evaluation of the proposed model and demonstrate its potential for application in drug discovery.

4.4. Statistical significance analysis of experimental results

In this work, ROC-AUC (Receiver Operating Characteristic — Area Under the Curve) scores evaluate the performance of our proposed model and graph-based baselines. The ROC-AUC score is a widely used metric to measure the ability of a classifier to distinguish between positive and negative classes. However, simply reporting the ROC-AUC score alone may not be sufficient to draw conclusions about the significance of the performance difference between different classifiers.

To address this issue, we perform a statistical significance analysis to determine whether the difference in the ROC-AUC scores between the proposed model and graph-based baselines is statistically significant or merely due to chance. The analysis involved comparing the ROC-AUC results obtained by different methods using a statistical significance test. The p-value results indicate the probability of observing such a difference by chance alone, and a threshold level of significance ($\alpha = 0.05$) is used to determine whether the difference is statistically significant. The first step was to compute a normality test to determine if ROC-AUC scores for each pair of performance results are normally distributed. In this case, we used a normality test provided by the SciPy library (Virtanen et al., 2020) based on the D'Agostino-Pearson omnibus test (D'Agostino, 1971; Pearson, D'"agostino, & Bowman, 1977), which combines skewness and kurtosis measurements to provide a p-value that indicates the likelihood that the data is normally distributed. The null hypothesis for the test is that the data is normally distributed. If the p-value is less than the chosen significance level ($p < 0.05$), then we can reject the null hypothesis and conclude that the data is not normally distributed. Otherwise, we fail to reject the null hypothesis, indicating that the ROC-AUC results are more likely to follow a normal distribution.

Next, we assessed the descriptive statistic of the normality test to evaluate whether the variance among both distributions was the same. In this case, we found a difference in variance between both distributions for all model results. Finally, to test the statistical significance between each pair of results, we used a modified version of the Student t-test if both distributions are more likely to be normal. This version of the t-test, known as Welch's t-test, is used when there is unequal variance between the two distributions being compared. If the distributions are unlikely to follow a normal distribution, we apply the Mann–Whitney U non-parametric test. To perform the test of statistical significance, p-values are calculated considering the hypothesis:

H0: Performance results are likely drawn from the same distribution;
H1: Performance results are likely drawn from different distributions (reject H0).

The calculated p-values are used to assess the statistical significance of the mean differences between the two distributions, considering a significance level of $\alpha = 0.05$. If the p-value <= 0.05, we reject the null hypothesis (H0) and conclude that there is evidence to support the alternative hypothesis (H1), indicating that the observed result is

Table 5
p-value results of the statistical significance test for 5-shot and 10-shot experiments.

Dataset	Task	FS-GNNTR		
		GIN	GCN	GraphSAGE
5-shot (5+, 5−)				
Tox21	SR-HSE	5.4407e−44	2.2156e−22	1.9080e−23
	SR-MMP	4.5348e−45	6.8112e−29	2.9991e−11
	SR-p53	2.4787e−35	9.6780e−23	1.4226e−19
SIDER	R.U.D.	7.0254e−10	2.7545e−40	4.4689e−30
	P.P.P.C.	2.9579e−11	1.2358e−05	1.2379e−04
	E.L.D.	6.2469e−20	2.7335e−32	7.5022e−30
	C.D.	3.0066e−11	3.0085e−11	3.0085e−11
	N.S.D.	5.5939e−05	2.4577e−19	5.3074e−33
	I.P.P.C.	8.0535e−10	3.0085e−11	1.0358e−21
10-shot (10+, 10−)				
Tox21	SR-HSE	7.2158e−33	3.0141e−11	3.0122e−11
	SR-MMP	6.4439e−38	1.3237e−24	3.0047e−11
	SR-p53	3.0028e−11	2.5496e−33	3.7494e−21
SIDER	R.U.D.	2.9897e−11	3.0029e−11	2.999e−11
	P.P.P.C.	1.2421e−17	1.5644e−15	5.9330e−14
	E.L.D.	1.2471e−32	3.0104e−11	3.0104e−11
	C.D.	3.0047e−11	2.9878e−11	3.0009e−11
	N.S.D.	5.4650e−03	9.1187e−28	9.0707e−16
	I.P.P.C.	4.0458e−08	1.1261e−30	1.3199e−27

statistically significant. In Table 5, we report the results of the significance analysis of the ROC-AUC scores obtained across 30 experiments by the proposed model with respect to the graph-based baselines for 5-shot and 10-shot experiments. The results of the statistical analysis show that all p-values calculated are less than the significance level, leading us to reject the null hypothesis and conclude that the ROC-AUC scores obtained by the proposed model are statistically significant when compared with standard graph-based baselines.

In summary, the statistical significance analysis provides a reliable method to compare the performance of few-shot models, enhance the validity of our findings, and offer a more robust evidence to support the conclusions. This approach allowed us to determine whether the differences in performance between models are statistically significant and make informed decisions about the efficacy of the proposed model.

4.5. Experiments with t-SNE embedding visualizations

In this section, we compare t-SNE visualizations (Maaten & Hinton, 2008) of deep representations extracted by the proposed model with graph embeddings obtained by GNN baselines. The results achieved show that the proposed model performs better in discriminating both positive and negative samples in molecular property prediction.

t-distributed stochastic neighborhood embedding (t-SNE) is a non-linear dimensionality reduction technique that operates by retaining the local aspects of data in a low-dimensional space. t-SNE maps closely placed datapoints to contiguous regions in the reduced space by modeling them as a t-distributed distribution. The goal is to find an optimal embedding, so that neighbors in a d-dimensional space are found close together in the embedded space. This is achieved by minimizing the Kullback–Leibler (KL) divergence between such distributions.

t-SNE visualizations are very sensitive to a perplexity parameter. Perplexity translates the balance between the local and global aspects of data. It essentially defines the number of points to include in a t-distribution to find a neighborhood for each datapoint. Small perplexity values indicate local connections between datapoints. Higher values of perplexity denote a global sense of geometry in high-dimensional spaces. To comply with both views, we set the perplexity parameter value to 30.

In Figs. 7 and 8, t-SNE visualizations of deep representations generated by the proposed model are shown for molecular property tasks on Tox21 and SIDER, respectively. Fig. 9 compares t-SNE visualizations of graph embeddings and deep representations. Blue dots denote negative samples. Orange dots represent positive samples.

Tox21 tasks: {SR-HSE, SR-MMP, and SR-p53} in Fig. 7 are depicted in scatter plots (a), (b) and (c), respectively. SIDER tasks: {R.U.D., P.P.P.C., E.L.D., C.D., N.S.D., I.P.P.C.} in Fig. 8 are depicted in scatter plots (a), (b), (c), (d), (e) and (f), respectively.

In Fig. 7, t-SNE visualizations indicate that FS-GNNTR performs well in discriminating positive and negative samples. Clusters of positive datapoints (orange dots) are found closer to each other, progressively separating from negative datapoints (blue dots). In the scatter plot (b), positive samples are located in the right side separated from the negative samples, to form two well-defined clusters of molecules.

Similarly, in Fig. 8, our model outperforms simpler graph-based baselines in discriminating positive and negative samples for the SIDER dataset. Clusters of positive samples gradually separate from negative samples to form groups of molecules closely related to each other. Additionally, the t-SNE visualizations outline an irregular shape to express the complex interaction patterns shared by deep representations.

In Fig. 9, the graph-based baselines show sparsely located datapoints, making it difficult to identify well-defined groups of compounds. On the other hand, deep representations extracted by our proposed model show clusters of negative samples closely located in the low-dimensional space, gradually separating from positive samples, outperforming the graph-based baselines.

5. Conclusion

In this paper, we propose a few-shot GNN-Transformer architecture, FS-GNNTR to face the problem of low-data in molecular property prediction. It is demonstrated that FS-GNNTR outperforms simpler graph-based methods on small data collections such as Tox21 and SIDER. Both datasets contain a set of high-level observations and out-of-distribution samples which may lead to unstable performances and unreliable predictions. In this low-data scenario, few-shot models produce superior results and provide a more robust solution for small biological datasets.

A two-module meta-learning framework optimizes model parameters across tasks to promote fast adaptation to new molecular properties on few-shot data. The experiments conducted demonstrate the predictive power and robustness of the proposed model over standard graph-based methods on multi-property prediction data. As shown in Section 4.3, FS-GNNTR outperforms the best baseline method for 5-shot experiments with an average improvement in ROC-AUC of $+11.52\%$ and $+1.28\%$ for Tox21 and SIDER, respectively. It is also reported an average overall improvement in ROC-AUC for 10-shot experiments. The small variances reported for both assay collections provide a stable performance across few-shot tasks.

The results achieved by FS-GNNTR on Tox21 and SIDER provide evidence supporting the hypothesis that the proposed multi-task meta-learning framework can be beneficial for QSAR modeling since it allows learning in an environment with little information and class imbalance without negatively affecting the predictive performance. As a result, FS-GNNTR could serve as an efficient QSAR method to select a series of compounds as potential drug candidates synthesized and tested for task-specific molecular properties including chemical toxicity and adverse side effects. The chemical structures and activity data of compounds could be used to train the proposed QSAR method and predict the molecular properties of new compounds with similar structures to those found in the training collection. Thus, the proposed framework has significant potential to advance drug discovery and assist in the identification of promising *drug-like* candidates.

Ultimately, we posit that the proposed model is able to capture both local and global information of molecular graph embeddings across few-shot tasks with limited available data and has the potential to improve the process of drug discovery through the accurate prediction of molecular properties in the early stages of drug development.

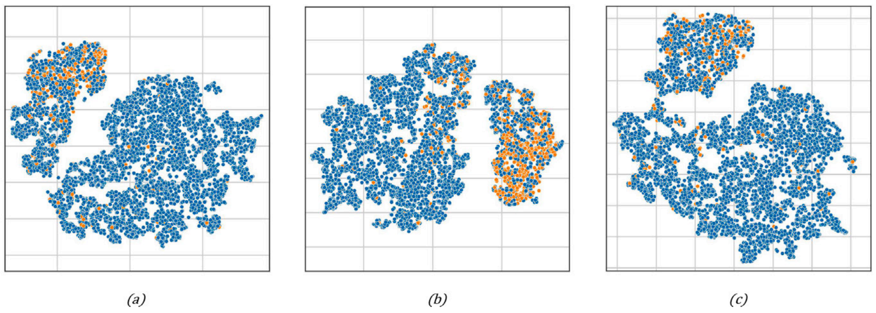


Fig. 7. t-SNE visualizations of deep representations h_T generated by FS-GNNTR with (5+, 5−) random support sets on Tox21.

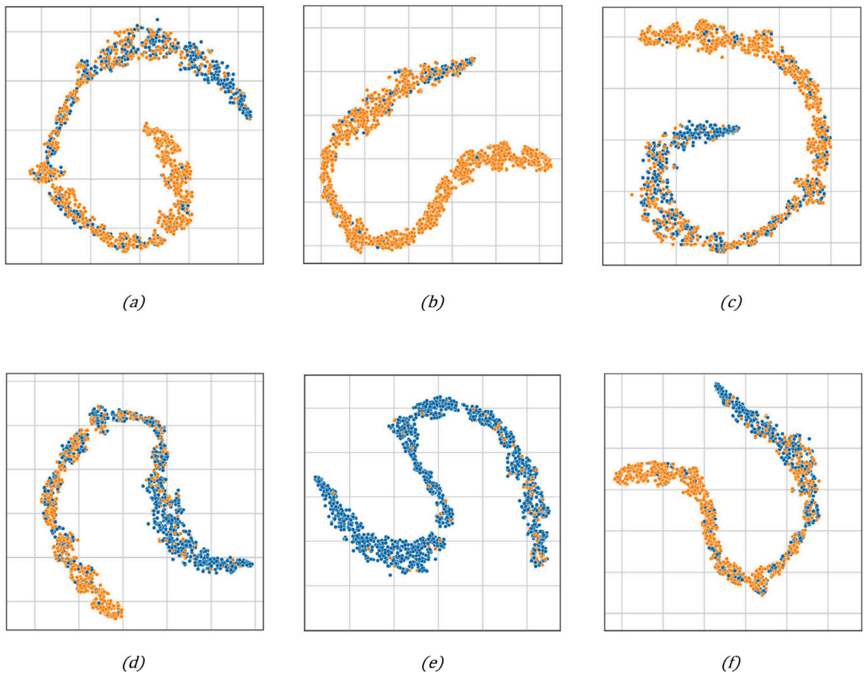


Fig. 8. t-SNE visualizations of deep representations h_T generated by FS-GNNTR with (5+, 5−) random support sets on SIDER.

Data linking

All the documentation and code scripts to reproduce the results are available in the repository, https://github.com/ltorres97/FS-GNNTR, to facilitate further experimentation.

CRediT authorship contribution statement

Luis H.M. Torres: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Bernardete Ribeiro:** Writing – review & editing, Supervision. **Joel P. Arrais:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The code and data underlying this article can be found in the repository https://github.com/ltorres97/FS-GNNTR.

Acknowledgments

This work is funded by the FCT — Foundation for Science and Technology, Portugal, I.P./MCTES through national funds (PIDDAC), within the scope of CISUC R&D Unit - UIDB/00326/2020 or project code UIDP/00326/2020.

Appendix A

In this Appendix section, we provide details regarding model hyper-parameters and the data train-test split used for both Tox21 and SIDER.

A.1. GNN-transformer hyper-parameters

We consider GIN as the embedding module in our experiments. The pre-trained GIN models of Hu et al. (2019) are used to achieve better parameter initialization. The proposed GIN includes 5 message-passing layers and an embedding dimension of 300.

The Transformer encoder module has the hyper-parameters displayed on Table 6.

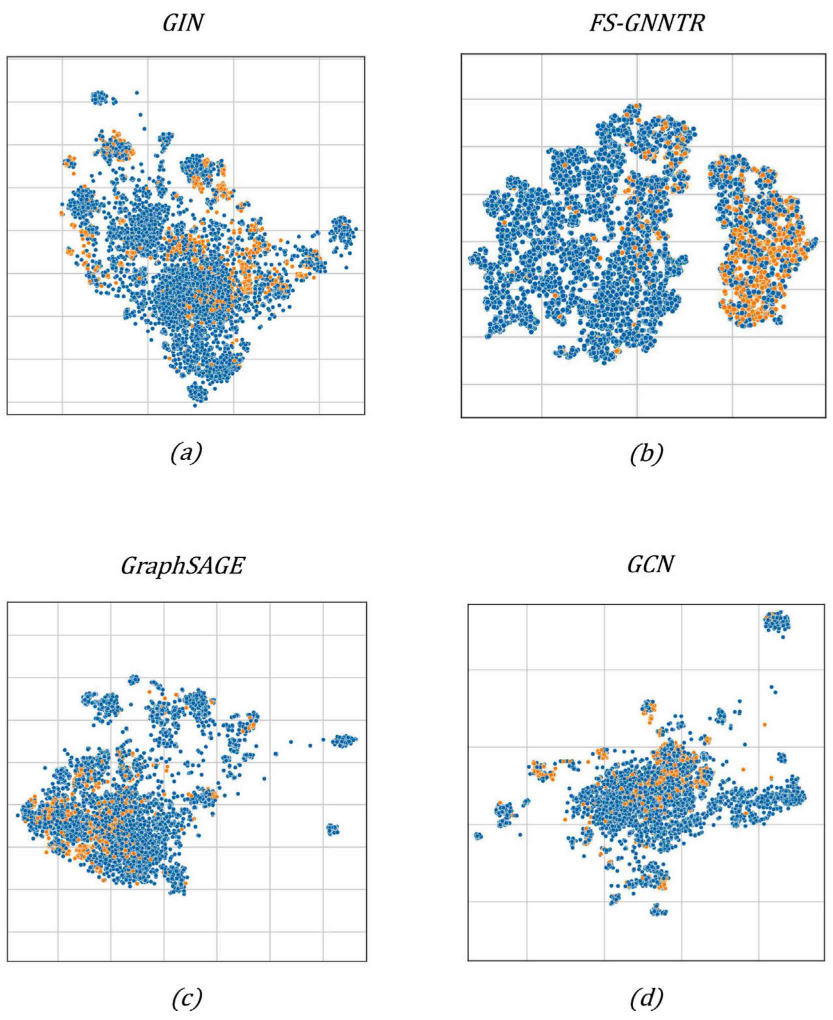


Fig. 9. t-SNE visualizations of graph embeddings generated by GraphSAGE, GIN, GCN and deep representations (FS-GNNTR) for the Tox21 SR-MMP task with $(5+, 5-)$ random support sets.

Table 6
Main hyper-parameters of the Transformer encoder.

Hyper-parameter	Value
input dim	300
patch size (P)	30
number of patches (N)	100
number of attention heads	5
linear transform dim (D)	128
MLP dim	256
number of blocks	5
batch size	10
pooling	mean pool
learning rate	$1e^{-5}$
learning rate decay	polinomial
optimizer	Adam

A.2. Details of model training and implementation

All models were implemented in PyTorch version 1.10.1 with CUDA version 11.3 and Python 3.7. RDKit libraries were used to compute basic features including atom-type, atom chirality, bond type and bond direction for atoms and chemical bonds in a given molecule. This set of features formed the initial set of atom and bond features fed into GNN layers. The SciPy library was used to calculate the p-value results and perform the analysis of statistical significance of performance results.

Few-shot models were trained across $(n_{train} \times epochs)$ episodes with n_{train} number of training tasks and $epochs$ number of epochs. In most cases, the models stopped improving significantly after 1000 epochs. In this work, we do not mainly focus on hyper-parameter optimization, especially for model baselines. Thus, we did not put an extensive effort into optimizing model hyper-parameters, leaving this task for future work. More specifically, we consider a learning rate of $1e^{-5}$ and an update step of 5 for training and 10 for testing.

All Python libraries and code scripts used to reproduce the results are available in the repository, https://github.com/ltorres97/FS-GNNTR, to facilitate further experimentation.

A.3. Train-test split

Tox21 train-test split: Tasks {NR-AR, NR-ARLBD, NR-AhR, NR-Aromatase, NR-ER, NR-ER-LBD, NRPPAR-gamma, SR-ARE, SR-ATAD5} were used for training. Tasks {SR-HSE, SR-MMP, and SR-p53} were used for evaluation.

SIDER train-test split: Tasks {H.D.: “Hepatobiliary disorders”, M.N.D.: ”Metabolism and nutrition disorders”, P.I: “Product issues”, E.D.: ”Eye disorders”, I.M.C.T.D.: “Investigations, musculoskeletal and connective tissue disorders”, G.D.: ”Gastrointestinal disorders”, S.C”. “Social circumstances”, I.S.D: “Immune system disorders”, R.S.B.D.: ”Reproductive system and breast disorders”, N.B.M.U.: “Neoplasms benign, malignant and unspecified (incl cysts and polyps)”, G.D.A.C: ”General disorders and administration site conditions”, E.D.: “Endocrine disorders”, S.M.P: ”Surgical and medical procedures”, V.D: “Vascular disorders”, B.L.S.D: ”Blood and lymphatic system disorders”, S.S.T.D.: “Skin and subcutaneous tissue disorders”, C.F.G.T.D.: ”Congenital, familial and genetic disorders”, “I.I”.: “Infections and infestations”, R.T.M.D.: ”Respiratory, thoracic and mediastinal disorders”, P.D.: “Psychiatric disorders”} were used for training. Tasks {R.U.D.: “Renal and urinary disorders”, P.P.P.C.: ”Pregnancy, puerperium and perinatal conditions”, E.L.D.: “Ear and labyrinth disorders”, C.D.: ”Cardiac disorders”, N.S.D.: “Nervous system disorders”, I.P.P.C.: ”Injury, poisoning and procedural complications”} were used for evaluation.

Appendix B. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.eswa.2023.120005. In this study, we also provide supplementary material for our research work. In this document, we include additional experimental results and details regarding model training and optimization to support and enhance the findings presented in this paper.

References

Abbasi, K., Poso, A., Ghasemi, J., Amanlou, M., & Masoudi-Nejad, A. (2019). Deep transferable compound representation across domains and tasks for low data drug discovery. *Journal of Chemical Information and Modeling*, 4528–4539. http://dx.doi.org/10.1021/acs.jcim.9b00626.

Altae-Tran, H., Ramsundar, B., Pappu, A. S., & Pande, V. (2017). Low data drug discovery with one-shot learning. *ACS Central Science*, *3*, 283–293. http://dx.doi.org/10.1021/acscentsci.6b00367.

Beyer, L., Zhai, X., & Kolesnikov, A. (2022). Better plain ViT baselines for ImageNet-1k. http://dx.doi.org/10.48550/arxiv.2205.01580, URL: https://arxiv.org/abs/2205.01580.

Butler, K. T., Davies, D. W., Cartwright, H., Isayev, O., & Walsh, A. (2018). Machine learning for molecular and materials science. *Nature*, *559*, 547–555. http://dx.doi.org/10.1038/s41586-018-0337-2.

Chen, B., Barzilay, R., & Jaakkola, T. (2019). Path-augmented graph transformer network. http://dx.doi.org/10.48550/arxiv.1905.12712, URL: http://arxiv.org/abs/1905.12712.

D’Agostino, R. B. (1971). An omnibus test of normality for moderate and large size samples. *Biometrika*, *58*, 341–348. http://dx.doi.org/10.1093/biomet/58.2.341.

Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems* (pp. 3844–3852). Neural information processing systems foundation, http://dx.doi.org/10.48550/arXiv.1606.09375.

Deng, Y., Qiu, Y., Xu, X., Liu, S., Zhang, Z., Zhu, S., et al. (2022). META-DDIE: predicting drug–drug interaction events with few-shot learning. *Briefings in Bioinformatics*, *23*, http://dx.doi.org/10.1093/bib/bbab514.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., et al. (2020). An image is worth 16 × 16 words: Transformers for image recognition at scale. http://dx.doi.org/10.48550/arxiv.2010.11929, URL: https://arxiv.org/abs/2010.11929.

Fei-Fei, L., Fergus, R., & Perona, P. (2006). One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*, http://dx.doi.org/10.1109/TPAMI.2006.79.

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *34th international conference on machine learning, vol. 3* (pp. 1856–1868). International Machine Learning Society (IMLS), http://dx.doi.org/10.48550/arXiv.1703.03400.

Gawehn, E., Hiss, J. A., & Schneider, G. (2016). Deep learning in drug discovery. *Molecular Informatics*, *35*(1), 3–14. http://dx.doi.org/10.1002/minf.201501008.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. *3*, In *34th international conference on machine learning, vol. 3* (pp. 2053–2070). International Machine Learning Society (IMLS), http://dx.doi.org/10.48550/arXiv.1704.01212.

Guo, Z., Zhang, C., Yu, W., Herr, J., Wiest, O., Jiang, M., et al. (2021). Few-shot graph learning for molecular property prediction. In *The web conference 2021 - Proceedings of the world wide web conference*. http://dx.doi.org/10.1145/3442381.3450112.

Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, *2017-December*, 1025–1035. http://dx.doi.org/10.48550/arXiv.1706.02216.

Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., et al. (2019). Strategies for pre-training graph neural networks. http://dx.doi.org/10.48550/arxiv.1905.12265, URL: https://arxiv.org/abs/1905.12265.

Hughes, J. P., Rees, S. S., Kalindjian, S. B., & Philpott, K. L. (2011). Principles of early drug discovery. *British Journal of Pharmacology*, *162*(6), 1239–1249. http://dx.doi.org/10.1111/j.1476-5381.2010.01127.x.

Kearnes, S., McCloskey, K., Berndl, M., Pande, V., & Riley, P. (2016). Molecular graph convolutions: moving beyond fingerprints. *Journal of Computer-Aided Molecular Design*, *30*(8), http://dx.doi.org/10.1007/s10822-016-9938-8.

Kim, B. H., & Ye, J. C. (2020). Understanding graph isomorphism network for rs-fMRI functional connectivity analysis. *Frontiers in Neuroscience*, *14*, http://dx.doi.org/10.3389/fnins.2020.00630.

Kuhn, M., Letunic, I., Jensen, L. J., & Bork, P. (2016). The SIDER database of drugs and side effects. *Nucleic Acids Research*, *44*, D1075–D1079. http://dx.doi.org/10.1093/nar/gkv1075.

Landrum, G. (2021). RDKit: Open-source cheminformatics software. URL: http://www.rdkit.org/.

Leelananda, S. P., & Lindert, S. (2016). Computational methods in drug discovery. *Beilstein Journal of Organic Chemistry*, *12*, 2694–2718. http://dx.doi.org/10.3762/bjoc.12.267.

Ma, J., Fong, S. H., Luo, Y., Bakkenist, C. J., Shen, J. P., Mourragui, S., et al. (2021). Few-shot learning creates predictive models of drug response that translate from high-throughput screens to individual patients. *Nature Cancer*, *2*, 233–244. http://dx.doi.org/10.1038/s43018-020-00169-2.

Maaten, L. V. D., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*.

Mayr, A., Klambauer, G., Unterthiner, T., & Hochreiter, S. (2016). DeepTox: Toxicity prediction using deep learning. *Frontiers in Environmental Science*, *3*, http://dx.doi.org/10.3389/fenvs.2015.00080.

Maziarka, Ł., & Tabor, J. (2019). Molecule-augmented attention transformer. In *NeurIPS workshop on machine learning and the physical sciences*. http://dx.doi.org/10.48550/arXiv.2002.08264.

Mialon, G., Chen, D., Selosse, M., & Mairal, J. (2021). GraphiT: Encoding graph structure in transformers. http://dx.doi.org/10.48550/arxiv.2106.05667, URL: https://arxiv.org/abs/2106.05667.

Paul, D., Sanap, G., Shenoy, S., Kalyane, D., Kalia, K., & Tekade, R. K. (2021). Artificial intelligence in drug discovery and development. *Drug Discovery Today*, *26*(1), 80–93. http://dx.doi.org/10.1016/j.drudis.2020.10.010.

Pearson, E. S., D’”agostino, R. B., & Bowman, K. O. (1977). Tests for departure from normality: Comparison of powers. *Biometrika*, *64*, 231–246. http://dx.doi.org/10.1093/biomet/64.2.231.

Srinivas, A., Lin, T. Y., Parmar, N., Shlens, J., Abbeel, P., & Vaswani, A. (2021). Bottleneck transformers for visual recognition. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition* (pp. 16514–16524). IEEE Computer Society, http://dx.doi.org/10.1109/CVPR46437.2021.01625.

Sun, Q., Liu, Y., Chua, T. S., & Schiele, B. (2019). Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition* (pp. 403–412). http://dx.doi.org/10.1109/CVPR.2019.00049, arXiv:1812.02391.

Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., & Jégou, H. (2021). Going deeper with image transformers. In *Proceedings of the IEEE international conference on computer vision* (pp. 32–42). Institute of Electrical and Electronics Engineers Inc., http://dx.doi.org/10.1109/ICCV48922.2021.00010.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *2017-Decem*, http://dx.doi.org/10.48550/arXiv.1706.03762.

Veličković, P., Casanova, A., Liò, P., Cucurull, G., Romero, A., & Bengio, Y. (2018). Graph attention networks. In *6th international conference on learning representations, ICLR 2018 - conference track proceedings*. http://dx.doi.org/10.1007/978-3-031-01587-8_7.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*, 261–272. http://dx.doi.org/10.1038/s41592-019-0686-2.

Wang, D., Cheng, Y., Yu, M., Guo, X., & Zhang, T. (2019). A hybrid approach with optimization and metric-based meta-learner for few-shot learning. http://dx.doi.org/10.48550/arxiv.1904.03014, URL: https://arxiv.org/abs/1904.03014.

Wang, Y., Yao, Q., Kwok, J. T., & Ni, L. M. (2020). Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys*, *53*(3), http://dx.doi.org/10.1145/3386252, arXiv:1904.05046.

Waring, M. J., Arrowsmith, J., Leach, A. R., Leeson, P. D., Mandrell, S., Owen, R. M., et al. (2015). An analysis of the attrition of drug candidates from four major pharmaceutical companies. *Nature Reviews Drug Discovery*, *14*(7), 475–486. http://dx.doi.org/10.1038/nrd4609.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2019). A comprehensive survey on graph neural networks. http://dx.doi.org/10.1109/tnnls.2020.2978386, URL: https://arxiv.org/abs/1901.00596.

Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., et al. (2018). MoleculeNet: A benchmark for molecular machine learning. *Chemical Science*, *9*(2), http://dx.doi.org/10.1039/c7sc02664a.

Xu, K., Jegelka, S., Hu, W., & Leskovec, J. (2019). How powerful are graph neural networks? In *7th international conference on learning representations*. http://dx.doi.org/10.48550/arXiv.1810.00826.

Yang, K., Swanson, K., Jin, W., Coley, C., Eiden, P., Gao, H., et al. (2019). Analyzing learned molecular representations for property prediction. *Journal of Chemical Information and Modeling*, *59*(8), 3370–3388. http://dx.doi.org/10.1021/acs.jcim.9b00237, arXiv:1904.01561.

Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., et al. (2021). Do transformers really perform bad for graph representation? http://dx.doi.org/10.48550/arxiv.2106.05234, URL: https://arxiv.org/abs/2106.05234.