



Explainable AI in drug discovery: self-interpretable graph neural network for molecular property prediction using concept whitening

Michela Proietti¹ · Alessio Ragno¹ · Biagio La Rosa¹ · Rino Ragno² · Roberto Capobianco^{1,3}

Received: 6 July 2022 / Revised: 20 March 2023 / Accepted: 17 July 2023 /
Published online: 31 October 2023
© The Author(s) 2023

Abstract

Molecular property prediction is a fundamental task in the field of drug discovery. Several works use graph neural networks to leverage molecular graph representations. Although they have been successfully applied in a variety of applications, their decision process is not transparent. In this work, we adapt concept whitening to graph neural networks. This approach is an explainability method used to build an inherently interpretable model, which allows identifying the concepts and consequently the structural parts of the molecules that are relevant for the output predictions. We test popular models on several benchmark datasets from MoleculeNet. Starting from previous work, we identify the most significant molecular properties to be used as concepts to perform classification. We show that the addition of concept whitening layers brings an improvement in both classification performance and interpretability. Finally, we provide several structural and conceptual explanations for the predictions.

Keywords Concept whitening · Drug discovery · Explainable artificial intelligence · Graph neural networks · QSAR

1 Introduction

Drug discovery (DD) is a process that consists in the identification of a new candidate drug that could be therapeutically useful in treating a pathological condition. An important step in the DD process is the development of quantitative structure-activity relationships (QSAR) models. These models allow understanding which structural properties of the tested molecules can be quantitatively correlated to the associated bioactivity, in order to eventually design more potent molecules as lead compounds to be developed as clinical candidates.

Editors: Paula Branco, Vitor Cerqueira, Carlos Soares, Luis Torgo.

Extended author information available on the last page of the article

Because of the complexity of this task, researchers have shown interest in the use of deep learning (DL) methods in DD, as they can provide accurate predictions while making the overall process less expensive and time-consuming. Moreover, several DL models are able to equal or also exceed the results of previously existing methods for drug discovery based on machine learning or classical QSAR (Lenselink et al., 2017). In particular, graph neural networks (GNNs) (Scarselli et al., 2009b) are widely used in DD (Wieder et al., 2020) because they exploit molecular graph representations, without requiring to represent molecules in other machine-readable formats, and make structure interpretation easier. Their ability to outperform traditional machine learning methods in the prediction of several molecular properties, such as hydrophobicity (Shang et al., 2021; Wang et al., 2019) and toxicity (Withnall et al., 2020; Xu et al., 2017) is mostly owed to their approximation capabilities (Scarselli et al., 2009a). Despite these promising results, it is in general difficult to understand the reasoning which leads to the models' output predictions.

This problem is addressed by explainable AI (XAI), which focuses on the development of interpretable AI systems. This crucial aspect needs to be investigated to bring machine learning closer to other scientific disciplines and increase the models' reliability. In general, explainability methods for GNNs are categorized as instance-based or model-based (Yuan et al., 2020). While the former provides example-specific explanations by identifying input features that are important for the output predictions, the latter tries to capture high-level insights into GNNs functioning. In this work, we focus on instance-level explanations. Many of the recently developed XAI approaches for GNNs propose post-hoc solutions, i.e., they provide explanations for already trained models. These methods often come from the generalization of techniques for convolutional neural networks (CNNs) (Pope et al., 2019; Schwarzenberg et al., 2019; Schnake et al., 2022), such as class activation mapping (Zhou et al., 2016), gradient-weighted class activation mapping (Selvaraju et al., 2019), and excitation backpropagation (Zhang et al., 2017). However, there is not much prior work on self-interpretable GNNs (Dai & Wang, 2021; Gui et al., 2022; Ragno et al., 2022), and in particular concept-based explanations for GNNs represent an unexplored research path.

In this work, we propose an adaptation of a widely used XAI approach for CNNs, called concept whitening (CW) (Chen et al., 2020), to graph data, in order to develop self-interpretable QSAR models for DD. In particular, we focus on spatial convolutional GNNs (Conv-GNNs) (Kipf & Welling, 2017), which can be seen as the CNNs' counterpart in the graph domain. Basically, CW consists of a module that can be added to a CNN to align the axes of the latent space with known concepts of interest. Its main application is in image recognition tasks, where Chen et al. (2020) show that CW increases the networks' interpretability while maintaining the same performance. Our idea is to use molecular descriptors as concepts in the CW module, in order to predict a specific molecular property. In this way, CW can benefit the task of QSAR by identifying how each concept contributes to the output prediction and consequently how a given property in a certain part of the molecule mostly contributes to modulating its biological behavior. To this aim, we analyze the importance of the concepts in each of the CW layers. As well as showing how inserting CW layers affects the models' performances, we also compare the results obtained using three different architectures: spatial Conv-GNNs (GCNs) (Kipf & Welling, 2017), graph attention networks (GATs) Veličković (2018), and graph isomorphism networks (GINs) (Xu et al., 2019). Moreover, as Chen et al. (2020) substitute CW layers to the batch normalization (BatchNorm) layers (Ioffe & Szegedy, 2015) of a pre-trained model, we additionally test whether using different types of normalization in the black-boxes then leads to higher performances after their substitution with CW layers. Specifically, we compare the results obtained using BatchNorm (Ioffe & Szegedy, 2015), instance normalization

(InstanceNorm) (Ulyanov et al., 2016), layer normalization (LayerNorm) (Ba et al., 2016) and graph normalization (GraphNorm) (Cai et al., 2021), respectively. We also introduce two new activation modes to be used within the CW layers in GNNs, which leverage top-k pooling (Knyazev et al., 2019). We perform experiments using several benchmark datasets from MoleculeNet (Wu et al., 2018). Finally, to understand which structural properties of the molecules are the most relevant for a given concept, we use the post-hoc explainability method GNNExplainer (Ying et al., 2019) on the concepts' activations.

In summary, the specific contributions of our work are the following:

- Adaptation of CW to Conv-GNNs, in order to obtain concept-based explanations for this type of networks, and analysis of the performances obtained using different architectures;
- Definition of two novel activation modes leveraging top-k pooling;
- Design of a way of interpreting QSAR models, by understanding which concepts, i.e., which properties of interest, mostly contribute to a specific type of activity of a given molecule and to which part of the molecules themselves to directly drive the chemical modifications.
- Comparisons between the performances obtained after the addition of the CW layers when using different types of normalization in the black-box models.

The code of this project is available at the following link <https://github.com/KRLGroup/Molecular-CW> and it is implemented in the www.3d-qsar.com portal (Ragno, 2019).

2 Related work

2.1 Graph neural networks

Graph neural networks (GNNs) (Scarselli et al., 2009b) represent a particular type of neural network developed to operate on graph-structured data. As highlighted by Scarselli et al. (2009a), GNNs can approximate up to any degree of precision any function preserving the unfolding equivalence, and most useful maps on graphs belong to this class of functions. Thanks to their approximation properties and due to the large availability of graph data coming from different scientific areas, GNNs are popular and used to solve several tasks in a wide variety of applications (Zhou et al., 2020).

GNNs core building blocks are the message passing layers, which are responsible for combining the node and edge information into the node embeddings. This is done by iteratively aggregating the information of each node with its neighbors' one, thus obtaining a new embedding that is used to update the representation of each node. Overall, GNNs can be divided in convolutional (Conv-GNNs) and recurrent. In this work, we just focus on the former. Conv-GNNs are in turn categorized in spectral (Bruna et al., 2014; Deferrard et al., 2016) and spatial (Kipf & Welling, 2017). In this work, we focus on spatial Conv-GNNs, and more specifically on the graph convolutional network proposed by Kipf and Welling (2017), to which we refer as GCN, and on graph attention networks (GATs) Veličković (2018), and graph isomorphism networks (GINs) (Xu et al., 2019), which differ in the aggregate and update functions. GCNs are characterized by the propagation rule, first proposed by Kipf and Welling (2017), that expresses the spatial convolution for a node. Differently from GCNs, where all neighboring nodes are assumed to contribute equally

to the update of a given node i , GATs (Veličković, 2018) exploit graph attentional layers to produce attention scores that represent the importance of the features of each neighboring node j to node i . Finally, Xu et al. (2019) design GIN to maximize the representational power. In fact, they show that the proposed network has the same discriminative power as the Weisfeiler–Lehman graph isomorphism test, which is used to evaluate the non-isomorphism of two graphs. In GIN, node embeddings are aggregated through a sum operator, with the neighboring nodes contributing equally to the update of the central node. However, the latter is weighted by a learnable parameter ϵ , and a multi-layer perceptron (MLP) is added after the aggregation of the neighbors' features has been performed.

2.2 GNNs in drug discovery

DL methods and, in particular, GNN-based methods are promising for addressing several tasks in DD (Kim et al., 2021).

Many works use such approaches to predict drug-target interactions, in order to identify lead compounds with a higher potency starting from hit compounds. For instance, Gilmer et al. (2017) specifically develop message passing neural networks (MPNNs) to achieve state-of-the-art performance on molecular property prediction. Similarly, Hamilton et al. (2017) propose GraphSAGE, a novel framework that generates node embeddings by aggregating information from local neighborhoods, and demonstrate its ability to generalize to unseen graphs in protein-protein interaction prediction. In particular, the use of GCNs, GATs and GINs in this work is motivated by their success in the prediction of different molecular properties (Wieder et al., 2020), such as toxicity (Chen et al., 2021; Hu et al., 2020; Peng et al., 2020; Wieder et al., 2020), hydrophobicity, blood-brain barrier permeability (Hu et al., 2020), and solvation free energy (Hu et al., 2020; Wang et al., 2019).

Another important application of GNNs in DD is the prediction of drug side effects (Bongini et al., 2022), which are generally caused by complex biological processes related to many factors, including drug structure and protein-protein interactions. Similarly, GNNs are used to predict polypharmacy effects (Deac et al., 2019; Zitnik et al., 2018) arising from the combined use of different drugs. The use of DL models is crucial in this case, as it allows drugs screening before the clinical trials and potentially leads to the identification of undesired effects that could still be unknown when the drug is on the market.

Finally, GNNs are widely used in de novo drug design in order to generate novel molecules with desired properties (Bongini et al., 2021; Li et al., 2018; Lim et al., 2019).

2.3 Drug discovery with XAI

In general, XAI methods can be categorized into post-hoc approaches, which produce explanations for trained neural networks by looking at their outputs and parameters, and self-explaining approaches, which consist in defining inherently interpretable models (Jiménez-Luna et al., 2020).

Feature attribution methods are post-hoc explanation approaches that determine the relevance of every input feature for the final prediction and they have been widely used in DD. For example, McCloskey et al. (2019) employ gradient-based attribution (Sundararajan et al., 2017) to detect fragment pharmacophores relevant for ligand binding. However, the study also shows that the models can still learn spurious correlations. Pope et al. (2019) adapt gradient-based feature attribution, more specifically gradient-weighted class activation mapping (Selvaraju et al., 2019) and excitation backpropagation (Zhang et al., 2017), to identify relevant

functional groups in adverse effect predictions. Ishida et al. (2019) use gradient-based feature attribution methods, such as integrated gradients (Sundararajan et al., 2017), together with GCNs for retrosynthetic reaction predictions and identify the atoms involved in each reaction step. Additionally, (Rodríguez-Pérez & Bajorath, 2019, 2020) use Shapley additive explanations (SHAP) (Lundberg & Lee, 2017) to interpret relevant features for compound potency and multitarget activity prediction. Among the post-hoc approaches specifically developed for GNNs, GNNExplainer (Ying et al., 2019) learns soft masks for edges and nodes features to find the crucial subgraphs and features to explain the predictions. Tested on a dataset for the classification of the molecules' mutagenic effect on *Salmonella typhimurium* (Debnath et al., 1992), GNNExplainer is able to identify several known mutagenic functional groups as relevant.

Since post-hoc explanations are not directly linked to the decision flow of GNNs, they can be biased and misrepresent the true explanations (Dai & Wang, 2021). For this reason, some studies focus on the development of self-explaining models. For instance, Dai and Wang (2021) propose a new framework to achieve explainable node classification by finding the K-nearest labeled nodes for each unlabeled node. While most current methods aim at explaining graph nodes, edges, or features, Gui et al. (2022) identify and exploit the most important message flows to provide explanations. None of these methods has ever been adopted in DD.

Finally, in the context of CNNs, there are several works studying how a predefined concept influences the internal representation of the hidden units of the networks. For instance, Kim et al. (2018) have introduced a method, called testing with concept activation vectors, that uses directional gradients to measure to what extent a user-defined concept influences a certain classification outcome. In a successive work, Ghorbani et al. (2019) propose a novel technique, called automated concept-based explanation, that aggregates related local image segments across diverse data to automatically extract visual concepts and is able to identify human-friendly concepts relevant to the network's output predictions. However, both these methods consider specific concepts for each class, ignoring the fact that some of them may be shared by different categories. In this way, the same concept might be represented differently across classes, and this could negatively affect the concept's importance for classification. To address this issue, Fang et al. (2020) present a novel visual concept mining algorithm. This approach comprises two main components: a potential concept generator to discover concepts by automatically searching and grouping important pixels via saliency map calculation; a visual concept extractor to learn the similarity and diversity of the concepts among different classes and quantify their correlation and unique contribution to each class.

Although they are useful, these post-hoc concept-based methods are based on the assumption that the representatives of different concepts lie in separate portions of the latent space, and this property may not hold (Chen et al., 2020). This difficulty is mitigated with CW, since it directly forces the network to produce latent representations (in the case of graphs, node, edge, and consequently graph embeddings) that allow discriminating samples based on their belonging to certain concept classes.

3 Methods

In this section, we first highlight the main similarities and differences between GCNs, GATs and GINs, and between the normalization types we use. Successively, we present CW and the strategy we adopt to adapt it to GNNs, introducing two new activation modes for updating the gradients within the CW layer. Finally, we illustrate the

method for identifying the structural parts of the molecules that are relevant to a certain concept.

3.1 Graph neural networks

Graph neural networks are based on the functional mechanism of message passing, which is responsible for the generation of new node embeddings through the iterative aggregation and update of the nodes and edges information. In this section, we describe in more detail the architectures used in this work, namely GCNs, GATs, and GINs. A hidden layer of a GCN can be written as a nonlinear function f :

$$H^{(l+1)} = f(H^{(l)}, A) \quad (1)$$

that takes as input the graph's adjacency matrix \mathbf{A} and the latent node features \mathbf{H} for some layer l . A simple layer-wise propagation rule for a GCN can therefore be written as:

$$f(H^{(l)}, A) = \sigma(AH^{(l)}W^{(l)}) \quad (2)$$

where \mathbf{W} is a weight matrix for the l -th neural network layer and σ is a nonlinear activation function. By multiplying the weight matrix with the adjacency matrix, all the feature vectors of the immediate neighbours are aggregated for every node, but self-loops are not included, which means that the feature vector of the node itself is not considered. This issue is solved by Kipf and Welling (2017), who define a generic l -th layer of a GCN as:

$$H^{(l+1)} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}) \quad (3)$$

where $H^{(l)}$ are the hidden features of the l -th layer, $W^{(l)}$ are its learnable parameters, and $\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$ is the symmetrically-normalized adjacency matrix with $\tilde{A} = A + I$ being the adjacency matrix taking into account also the presence of self-loops, so that each node in the graph also includes its own features in the next representation, and \tilde{D} is the degree matrix of \tilde{A} . By writing Eq. (3) in vector form, we get:

$$h_i^{(l+1)} = \sigma\left(\sum_j \frac{1}{c_{ij}} h_j^{(l)} W^{(l)}\right) \quad (4)$$

where j is the index of the neighboring node of node i and c_{ij} is a normalization constant for the edge connecting nodes i and j , which is obtained using the symmetrically normalized adjacency matrix $D^{-\frac{1}{2}}A^{-\frac{1}{2}}$. At this point, by choosing an appropriate non-linearity and initializing the weight matrix such that it is orthogonal, this update rule becomes stable, also thanks to the normalization with c_{ij} .

In GATs, self-attention is performed by applying an attention function, a which produces attention scores:

$$e_{ij} = a(\mathbf{W}h_i, \mathbf{W}h_j) \quad (5)$$

that represents the importance of the features of node j to node i . Successively, the coefficients are normalized for every choice of j with the softmax function to facilitate comparisons between coefficients of different nodes, thus obtaining:

$$\alpha_{ij} = \text{softmax}_j(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k \in N_i} \exp(e_{ik})} \quad (6)$$

These normalized attention coefficients are used to compute a linear combination of the nodes features, to obtain the new embedding for each node. A non-linearity is also usually applied, thus getting the following update rule:

$$h_i^{(l+1)} = \sigma \left(\sum_{j \in N_i} \alpha_{ij} h_j^{(l)} W^{(l)} \right) \quad (7)$$

Finally, in GINs the aggregation of the features of neighbouring nodes and the update of each node's embedding is implemented as follows:

$$h_i^{(l+1)} = MLP^{(l+1)} \left((1 + \epsilon^{(l+1)}) \cdot h_i^{(l)} + \sum_{j \in N_i} h_j^{(l)} \right) \quad (8)$$

where $\epsilon^{(l+1)}$ is a learnable parameter and MLP is a multi-layer perceptron with non-linearity.

Since molecules that are tested as potential candidate drugs are usually small, GNNs that are adopted in drug discovery are generally made up of just a few convolutional layers, that are used to generate the latent representation of the nodes. GNNs can provide node-, edge-, and graph-level predictions, but in this work we will just focus on graph classification. To this aim, a readout layer is needed to obtain a representation for the whole graph and it is followed by a final classification layer to obtain predictions.

3.1.1 Normalization layers for GNNs

Cai et al. (2021) analyzes the effectiveness of different types of normalization for GNNs, by adapting some existing methods, including BatchNorm, LayerNorm, and InstanceNorm, to the graph domain, and proposing a novel one, called GraphNorm.

Given a set of samples x_1, \dots, x_n , a normalization operation shifts each x_i by the mean μ , and scales them down by the standard deviation σ : $x_i \rightarrow \gamma \frac{x_i - \mu}{\sigma} + \beta$, with γ and β being learnable parameters. What differs among the various normalization methods is the set of feature values the normalization is applied to. BatchNorm normalizes all values in a given feature dimension across the nodes of all graphs in the batch. LayerNorm, instead, normalizes values across different dimensions of each node. InstanceNorm normalizes values across all nodes for each individual graph. Finally, GraphNorm adds a learnable parameter α , multiplying μ , to automatically control which proportion of the mean should be kept in the shift operation.

3.2 Concept whitening

CW is a module inserted into a neural network that aligns the axes of the latent space with known concepts of interest and facilitates their extraction. Thus, CW allows learning an inherently interpretable model, since it can show how a concept is represented at a given layer of the network. CW works similarly to batch whitening (Huang et al., 2018b), as it decorrelates and normalizes each axis of the latent space, thus transforming the post-convolution latent space so that the covariance matrix between channels is the identity. However, CW also provides an extra step involving a rotation matrix used to match the concepts to the axes of the

latent space. This matrix is optimized using Cayley-transform-based curvilinear search algorithms (Wen & Yin, 2013).

More formally, suppose that $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n \in \mathcal{X}$ are dataset samples and $y_1, y_2, \dots, y_n \in \mathcal{Y}$ are the corresponding labels. A deep neural network classifier $f : \mathcal{X} \rightarrow \mathcal{Y}$ can be split into two parts, namely a feature extractor $\Phi : \mathcal{X} \rightarrow \mathcal{Z}$ with parameters θ , and a classifier $g : \mathcal{Z} \rightarrow \mathcal{Y}$ with parameters ω . Then, $\mathbf{z} = \Phi(\mathbf{x}; \theta)$ is the latent representation of \mathbf{x} , and $f(\mathbf{x}) = g(\Phi(\mathbf{x}; \theta); \omega)$ is the predicted label. For each $k \in [1, K]$, with K being the number of concepts, we need to define an auxiliary dataset \mathbf{X}_{c_k} . Exploiting CW, we want to simultaneously learn Φ and g , such that the classifier gives accurate predictions and the j th dimension z_j of the latent representation \mathbf{z} aligns with concept c_j . By doing this, samples in \mathbf{X}_{c_j} should have larger values of z_j than other samples.

Now, let $\mathbf{Z}_{d \times n}$ be the latent representation matrix of n samples, where each column $\mathbf{z}_i \in \mathbb{R}^d$ contains the latent features of the i th samples. The CW module consists of two parts, a whitening and an orthogonal transformation. The whitening transformation decorrelates and standardizes the data, and it is defined as:

$$\psi(\mathbf{Z}) = \mathbf{W}(\mathbf{Z} - \mu \mathbf{1}_{n \times 1}^T) \tag{9}$$

where $\mu = \frac{1}{n} \sum_{i=1}^n \mathbf{z}_i$ is the sample mean and $\mathbf{W}_{d \times d}$ is the whitening matrix such that $\mathbf{W}^T \mathbf{W} = \Sigma^{-1}$, with $\Sigma_{d \times d} = \frac{1}{n} (\mathbf{Z} - \mu \mathbf{1}^T)(\mathbf{Z} - \mu \mathbf{1}^T)^T$ being the covariance matrix. The whitening matrix is computed as in zero-phase component analysis (ZCA) (Huang et al., 2018a):

$$\mathbf{Z} = \mathbf{D} \Lambda^{-\frac{1}{2}} \mathbf{D}^T \tag{10}$$

where $\Lambda_{d \times d}$ and $\mathbf{D}_{d \times d}$ are respectively the eigenvalue diagonal matrix and the eigenvector matrix given by the eigenvalue decomposition of the covariance matrix, $\Sigma = \mathbf{D} \Lambda \mathbf{D}^T$.

Once the latent space has been mean-centered and decorrelated, samples are rotated in their latent space so that those that are related to concept c_j , \mathbf{X}_{c_j} , are highly activated on the j th axis. In particular, the orthogonal matrix $\mathbf{Q}_{d \times d}$, whose column \mathbf{q}_j is the j th axis, is obtained by optimizing the following objective:

$$\max_{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_k} \sum_{j=1}^k \frac{1}{n_j} \mathbf{q}_j^T \psi(\mathbf{Z}_{c_j}) \mathbf{1}_{n_j \times 1} \quad \text{s.t.} \quad \mathbf{Q}^T \mathbf{Q} = \mathbf{I}_d \tag{11}$$

where \mathbf{Z}_{c_j} is a $d \times n_j$ matrix denoting the latent representation of \mathbf{X}_{c_j} . During training, two different objectives are optimized alternately. The main objective is the one related to the classification accuracy:

$$\min_{\theta, \omega, \mathbf{W}, \mu} \frac{1}{n} \sum_{i=1}^n l(g(\mathbf{Q}^T \psi(\Phi(\mathbf{x}_i; \theta); \mathbf{W}, \mu); \omega), y_i) \tag{12}$$

where Φ and ψ are the layers before and after the CW module, parametrized by θ and ω respectively. The actual CW module is represented by $\mathbf{Q}^T \psi$, while l is any differentiable loss, e.g., cross-entropy loss. The second objective is the concept alignment loss:

$$\max_{\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_k} \sum_{j=1}^k \frac{1}{n_j} \sum_{\mathbf{x}_i^{c_j} \in \mathbf{X}_{c_j}} \mathbf{q}_j^T \psi(\Phi(\mathbf{x}_i^{c_j}; \theta); \mathbf{W}, \mu) \quad \text{s.t.} \quad \mathbf{Q}^T \mathbf{Q} = \mathbf{I}_d \tag{13}$$

\mathbf{Q} is fixed while training for the main objective, while the other parameters are fixed when training for \mathbf{Q} . The second optimization problem with the orthogonality constraint is solved by gradient-based approaches on the Stiefel manifold. At each step t in which the second objective is handled, the orthogonal matrix \mathbf{Q} is updated by the Cayley transform:

$$\mathbf{Q}^{(t+1)} = \left(I + \frac{\eta}{2} \mathbf{A} \right)^{-1} \mathbf{Q}^{(t+1)} = \left(I + \frac{\eta}{2} \mathbf{A} \right) \mathbf{Q}^{(t)} \quad (14)$$

where $\mathbf{A} = \mathbf{G}(\mathbf{Q}^{(t)})^T - \mathbf{Q}^{(t)}\mathbf{G}^T$ is a skew-symmetric matrix, \mathbf{G} is the gradient of the loss function and η is the learning rate. The optimization procedure is also accelerated by curvilinear search at each step (Wen & Yin, 2013). Since in (14) the stationary points are reached when $\mathbf{A} = \mathbf{0}$, there are multiple solutions that lie in a high-dimensional space, so the stationary points are likely to be saddle points. To address this issue, stochastic gradient descent (SGD) is used, and momentum has been exploited to accelerate and stabilize the training.

Concerning implementation details, it is important to highlight that each channel within one layer is used to represent a specific concept. This is done by reshaping the output of each convolutional layer $Z_{n \times d \times h \times w}$ into a matrix $Z_{d \times (hwn)}$, with d being the number of channels. In this way, after performing CW, we still have a matrix with shape $d \times (hwn)$ and, by reshaping it to $n \times d \times h \times w$, we obtain that each feature map in the resulting tensor now represents whether an important concept is detected at each location in the image for the considered layer. Finally, an activation value for each $h \times w$ feature map is computed considering different *activation modes*:

- *mean* computes the mean of all feature map values;
- *max* takes the maximum over all the feature map values;
- *pos_mean* computes the mean of all positive feature map values;
- *max_pool* calculates the mean of the down-sampled feature map obtained by max pooling. This is the activation mode that was used in the experiments in (Chen et al., 2020) since it is able to capture both high-level and low-level concepts.

3.3 Adaptation of CW to Conv-GNNs

Concept-based explanation methods are particularly suited for interpreting QSAR models because they allow leveraging domain-specific knowledge by focusing on molecular properties that are known to affect bioactivity. Due to the lack of any such type of approaches for graph data, we adapt CW to Conv-GNNs. In particular, we follow the procedure of Chen et al. (2020), using the CW module to replace the BatchNorm layer straight after a convolutional one.

The mathematical formulation of the whitening transformation and the optimization problem that allows finding the orthogonal matrix described in Sect. 3.2 remains unaltered while applying them to GNNs. Consequently, the basic functioning of the CW module is identical to the case in which it is added to a CNN. However, the data is completely different, and therefore the input shapes of the transformations, i.e., the output shapes of the convolutional layers, change accordingly. In fact, while the output of a 2D convolutional layer has shape $n \times d \times h \times w$, with n , d , h , w being respectively the batch size, the dimension of the latent space, and the image height and width, the output of a graph convolutional layer has shape $N \times d$, where N is the number of nodes and d is again the dimension of the latent space. These represent the shapes of the tensors that are given as input to the concept

whitening layer. As explained in Sect. 3.2, in the case of CNNs, each feature map obtained from the convolution of one filter is reshaped so that, for each filter, the output of a convolutional layer $Z_{n \times d \times h \times w}$ is reshaped into a matrix $Z_{(g \times d) \times (hwn)}$, where g is the number of channels used to represent each concept and hwn corresponds to the total number of features. Since one channel is used to represent each concept, this reduces to $Z_{d \times (hwn)}$. This operation can be easily transferred to the graph domain, by changing the dimension of the output of each convolutional layer from $Z_{N \times d}$ to $Z_{g \times d \times N}$. The subsequent computations remain exactly the same as before, and after the CW layer, each channel represents a specific concept. We clarify that the CW module necessarily needs to be inserted in place of a normalization layer, after a convolutional layer. The input shape to the CW module is therefore the output shape of the convolutional layer. Since the shape of the transformations that compose the CW module is inferred from the shape of the input data, the module is able to automatically adapt to varying inputs in the case of Conv-GNNs. Similarly, if we change the dimension of the convolutional layer, the shape of the transformations in the CW module will change based on the new shape of the convolutional output.

The only aspect that cannot be directly adapted to GNNs concerns one of the activation modes we presented in Sect. 3.2: *max_pool*. In fact, this consists in performing 2D pooling and unpooling operations, which cannot be performed on graph data. In this work, we propose two alternative activation modes that exploit the top-k pooling operator presented by Gao and Ji (2019). Instead of clustering “similar” nodes, top-k pooling propagates only part of the input, that is not uniformly sampled from the input itself. By specifying a pooling ratio, $k \in (0, 1]$, we can select just some parts of the input graph, by keeping just $[kN]$ nodes out of the initial N . This selection is done by projecting node features onto the direction of a trainable vector, \mathbf{p} , and keeping the nodes with the highest projection scores. In fact, the scalar projection of the i -th node’s feature vector, \mathbf{x}_i , on \mathbf{p} , computed as $y_i = \frac{\mathbf{x}_i \mathbf{p}}{\mathbf{p}}$, indicates how much information is retained after the projection onto \mathbf{p} and we want to preserve as much information as possible. To make \mathbf{p} learnable by back-propagation, the projection scores are used as gating values to control how much information to keep from the retained nodes. This last property of top-k pooling was exploited to actually develop two different activation modes, called *topk_pool* and *weighted_topk_pool*. The first one is obtained by just computing the mean of the node embeddings of the down-sampled graph. The second one, instead, is obtained by making a weighted average of the node embeddings of the down-sampled graph, using as weights the projection scores returned by top-k pooling.

3.3.1 Structural information related to concept activations

Together with providing conceptual explanations thanks to the adaptation of the CW module, we analyze the concept-structure relationship by combining CW with a XAI post-hoc method such as GNNExplainer. More generally, being the CW layer differentiable, any post-hoc method can be used to understand the structural motifs that modulate the concept activation. In this way, we can additionally obtain structural explanations. The use of a post-hoc method for this further step is necessary because some of the molecular properties that we use as concepts are “abstract”. With this adjective, we mean that it is not possible to directly derive the input attributions from the concept’s value. Consequently, we use GNNExplainer to extract the portion of the molecular graphs that most contributes to the concept latent value. GNNExplainer, in fact, is a perturbation based method that optimizes

an edge mask such that the mutual information between the GNN's output and the distribution of possible sub-graphs is maximized.

4 Experiments

In this section, after introducing the datasets and the concepts we use, we report our experimental setup to make results reproducible. Then, we start our analysis by comparing the classification performances obtained using GCN, GAT and GIN. In particular, we show how using different activation modes within the CW layers leads to different results. Moreover, we compare the performances obtained using different types of normalization in the black-box models. Subsequently, we study how concepts are represented within the CW layers and analyze the concept-structure relationships. Finally, we compute quantitative metrics to evaluate the improvement in interpretability.

4.1 Datasets

We train and test all the different architectures on four molecule datasets for graph classification. Molecules are represented as graphs, in which nodes and edges represent atoms and bonds, respectively. The first dataset we used is BBBP (Martins et al., 2012), which addresses blood-brain barrier penetration. This is a crucial aspect in the development of drugs targeting the central nervous system. The second one is BACE (Subramanian et al., 2016), which has as target beta-secretase 1. This protein is essential for the generation of beta-amyloid peptide in neural tissue, a component of amyloid plaques widely believed to be involved in the development of Alzheimer's disease. The third one is ClinTox (Wu et al., 2018), which compares drugs approved by the FDA and drugs that have failed clinical trials for toxicity reasons. The last one is the HIV dataset,¹ which contains 40,000 compounds, tested for their ability to inhibit HIV replication, that are associated to binary labels indicating whether they are active or inactive.

4.2 Concept selection

As previously mentioned, we select the concepts by taking into account the molecular properties that are known to be relevant for each of the studied tasks. More specifically, we build each concept's dataset by keeping all the molecules that present a value for the corresponding molecular property within a predefined interval. In order to select the most appropriate concepts, we follow the work of Sakiyama et al. (2021) and Subramanian et al. (2016) for BBBP and BACE, respectively. For ClinTox we select the concepts among the ones used for BBBP and BACE that led to the greatest improvement when used alone within the CW layers. For the HIV dataset, instead, we follow the work of Sirois et al. (2005) and Kiralj and Ferreira (2003). Table 1 reports the concepts selected for each dataset, together with the corresponding threshold values. For more details on the meaning of the concepts and the threshold selection, please refer to Appendix A.

¹ AIDS Antiviral Screen Data, <http://wiki.nci.nih.gov/display/NCIDTPdata/AIDS+Antiviral+Screen+Data>, accessed 27 Sept, 2017.

Table 1 Concepts selected for each tested dataset with the chosen threshold

Dataset	Concept	Threshold
BBBP	Quantitative estimate of drug-likeness (QED)	> 0.6244
	Topological polar surface area (TPSA)	> 70.73
	Logarithm of the octanol-water partition coefficient (LogP)	< 5.0000
	Number of N and O atoms (NOCount)	< 5
	Number of heteroatoms (n_heteroatoms)	< 6
BACE	Quantitative estimate of drug-likeness (QED)	> 0.5221
	Topological polar surface area (TPSA)	> 95.19
	Molecular weight (mol_weight)	< 500
	Number of H-bond acceptors (HBA)	< 10
	Number of H-bond donors (HBD)	< 5
ClinTox	Quantitative estimate of drug-likeness (QED)	> 0.5335
	Molecular weight (mol_weight)	< 500
	Number of H-bond acceptors (HBA)	< 10
	Logarithm of the octanol-water partition coefficient (LogP)	< 5.0000
	Number of heteroatoms (n_heteroatoms)	< 7
HIV	Number of H-bond acceptors (HBA)	< 10
	Number of H-bond donors (HBD)	< 5
	Logarithm of the octanol-water partition coefficient (LogP)	< 2.9810
	Number of double bonds (nDoubleBonds)	< 2.41
	Number of oxygen atoms (nO)	< 5

Molecules are required to have a value for each concept that is lower than the specified threshold, with the exception of QED and TPSA, for which the concept datasets contain all the compounds with a value that is greater than the threshold

4.3 Experimental setup

All the code was developed in PyTorch (Paszke et al., 2019), using PyTorch Geometric library (Fey & Lenssen, 2019) (PyG). Each model used in the experiments is made up of three convolutional layers, which can be *GCNConv*, *GATConv*, or *GINConv* layers, with 128 units and a dense layer with 128 units. In *GATConv* multi-head attention layers, the number of attention heads is set to 2. Instead, the MLP in *GINConv* layers (see Eq. (8)) is designed as a single linear layer with the input and output dimensions equal to the number of units in the previous and following convolutional layers, respectively. The convolutional and dense layers have ReLU activation function. Additionally, there is a normalization layer after each convolutional layer, which can be a BatchNorm, LayerNorm, InstanceNorm, or GraphNorm layer. The CW layers are substituted to these normalization layers. Finally, the readout layer is represented by a global sum pooling in the case of BBBP and HIV and a global max pooling in the case of BACE and ClinTox, and it is followed by a simple dense layer that performs the classification.

For all the black-box models, we have implemented early stopping with a patience of 20 epochs. After the addition of CW, we found out that training the models for a maximum of 50 epochs with early stopping with patience 5 allows obtaining the best performance. Each experiment was run 15 times to compare different seeds for statistical

Table 2 Hyperparameters used to train the models on the three considered datasets

Hyperparameter	BBBP	BACE	ClinTox	HIV
Batch size	64	128	128	64
Learning rate	0.001	0.01	0.01	0.001
Weight decay	5×10^{-4}	0.0	0.0	5×10^{-4}
# Epochs	200	400	400	200
Early stopping patience ^a	20/5	20/5	20/5	20/5
Conv. hidden units	128	128	128	128
Conv. activation	ReLU	ReLU	ReLU	ReLU
Dense hidden units	128	128	128	128
Dense activation	ReLU	ReLU	ReLU	ReLU
Readout ^b	sum	max	max	sum

^aThe first value refers to the training of the black-box model, while the second one is used while tuning the CW layers in the pre-trained models

^bAll the layers are global pooling layers

Table 3 Performance on BBBP over 15 runs before and after implementing early stopping (ES), together with the epoch at which on average early stopping occurs

Model	ROC-AUC—No ES	ROC-AUC—With ES	ES epoch
GCN	0.86 ± 0.03	0.88 ± 0.03	37
GAT	0.87 ± 0.04	0.89 ± 0.04	38
GIN	0.87 ± 0.04	0.88 ± 0.03	36

significance. In Table 2, we summarize the values of the hyperparameters that were used for BBBP, BACE, ClinTox, and HIV, respectively.

4.4 Hyperparameters tuning

In order to choose the number of epochs and to decide whether or not to add early stopping, we have performed hyperparameters tuning. In Table 3 we report the performance of the network on the validation set before and after implementing early stopping on BBBP dataset. We also report the epoch at which on average early stopping occurs. For all the architectures, early stopping occurs pretty soon during training. Moreover, by training the models for the number of epochs specified in Table 2 without early stopping the performances decrease, meaning that the models are overfitting. After training the black-boxes, we tried to fine-tune the models with CW layers for just one additional epoch. However, this led to very low results, both in terms of accuracy and ROC-AUC, and that is why we set the maximum number of epochs to 50 but, following the same procedure we have just described, we set early stopping patience to 5.

4.5 Classification performances

In this section, we verify how the substitution of the BatchNorm layers of a pre-trained model with CW layers affects its performance. In Table 4, we present the mean ROC-AUC values obtained using GCN, GAT and GIN on BBBP, BACE, ClinTox, and HIV,

Table 4 Mean ROC-AUC over 15 runs before and after the addition of the three concept whitening layers for BBBP, BACE, ClinTox, and Tox21

Model	BBBP	BACE	ClinTox	HIV
Random forest	0.79 ± 0.01	0.79 ± 0.01	0.74 ± 0.01	0.65 ± 0.01
MLP	0.87 ± 0.01	0.84 ± 0.01	0.75 ± 0.05	0.84 ± 0.01
MPNN	0.88 ± 0.02	0.70 ± 0.03	0.72 ± 0.02	0.75 ± 0.01
GCN	0.88 ± 0.03	0.86 ± 0.02	0.86 ± 0.08	0.79 ± 0.03
– <i>weighted_topk_pool</i>	0.81 ± 0.06	0.93 ± 0.02	0.92 ± 0.10	0.69 ± 0.03
– <i>topk_pool</i>	0.87 ± 0.03	0.92 ± 0.01	0.92 ± 0.08	0.70 ± 0.02
– <i>pos_mean</i>	0.89 ± 0.03	0.92 ± 0.02	0.95 ± 0.02	0.77 ± 0.02
– <i>max</i>	0.91 ± 0.02	0.92 ± 0.02	0.93 ± 0.05	0.79 ± 0.02
– <i>mean</i>	0.88 ± 0.03	0.92 ± 0.03	0.95 ± 0.03	0.72 ± 0.02
GAT	0.89 ± 0.03	0.86 ± 0.02	0.84 ± 0.08	0.80 ± 0.03
– <i>weighted_topk_pool</i>	0.89 ± 0.03	0.93 ± 0.02	0.90 ± 0.04	0.74 ± 0.04
– <i>topk_pool</i>	0.92 ± 0.03	0.94 ± 0.02	0.90 ± 0.04	0.76 ± 0.02
– <i>pos_mean</i>	0.91 ± 0.02	0.91 ± 0.02	0.88 ± 0.04	0.80 ± 0.02
– <i>max</i>	0.97 ± 0.02	0.93 ± 0.01	0.88 ± 0.05	0.80 ± 0.02
– <i>mean</i>	0.92 ± 0.05	0.93 ± 0.02	0.90 ± 0.03	0.75 ± 0.02
GIN	0.89 ± 0.02	0.87 ± 0.03	0.75 ± 0.11	0.78 ± 0.03
– <i>weighted_topk_pool</i>	0.85 ± 0.03	0.92 ± 0.02	0.77 ± 0.08	0.71 ± 0.03
– <i>topk_pool</i>	0.84 ± 0.04	0.82 ± 0.02	0.73 ± 0.13	0.70 ± 0.03
– <i>pos_mean</i>	0.90 ± 0.02	0.92 ± 0.02	0.69 ± 0.12	0.79 ± 0.02
– <i>max</i>	0.91 ± 0.02	0.91 ± 0.02	0.74 ± 0.11	0.79 ± 0.02
– <i>mean</i>	0.88 ± 0.03	0.92 ± 0.02	0.70 ± 0.15	0.71 ± 0.02

The results obtained using different activation modes have been reported. We write in bold the results of the best model for each architecture

respectively. In particular, we compare the performances obtained by the black-box models with BatchNorm and their interpretable versions, while using different activation modes. For each dataset, there is at least one activation mode within the CW layers that allows to equal or, in most cases, improve the ROC-AUC with respect to their black-box versions for all the architectures, as shown in Fig. 1. In particular, *max* mode is the one that guarantees the best results for all the architectures on BBBP and HIV, and the highest accuracy and ROC-AUC values are obtained with GAT architecture. On the contrary, the activation mode that gave the worst results for both GCN and GAT on BBBP and HIV is *weighted_topk_pool*, which is instead the best one in the case of GCN architecture on BACE and GIN architecture on ClinTox. Despite the good performance of GCN on BACE, the best results on this dataset were obtained with GAT and *topk_pool* activation mode. The best results on ClinTox were obtained using GCN architecture, with *mean* or *pos_mean* activation modes leading to similar performances. We impute the higher performances of the interpretable models to the ability of the CW layers to force the node embeddings produced by the models to represent relevant information in terms of the nodes' belonging to each concept's class, which attributes such representations a greater discriminative power.

Another important consideration is that the activation modes we propose, namely *topk_pool* and *weighted_topk_pool*, allow obtaining comparable performances with respect to the already existing ones, even improving them in some cases.

Table 5 Mean ROC-AUC over 15 runs on all tested datasets before and after the addition of the three concept whitening layers using different normalization layers in the baseline models corresponding to the best-performing interpretable models

Normalization type	BBBP	BACE	ClinTox	HIV
BatchNorm	0.89 ± 0.03	0.86 ± 0.02	0.86 ± 0.08	0.80 ± 0.03
– <i>weighted_topk_pool</i>	0.89 ± 0.03	0.93 ± 0.02	0.92 ± 0.10	0.74 ± 0.04
– <i>topk_pool</i>	0.92 ± 0.03	0.94 ± 0.02	0.92 ± 0.08	0.76 ± 0.02
– <i>pos_mean</i>	0.91 ± 0.02	0.91 ± 0.02	0.95 ± 0.02	0.80 ± 0.02
– <i>max</i>	0.97 ± 0.02	0.93 ± 0.01	0.93 ± 0.05	0.80 ± 0.02
– <i>mean</i>	0.92 ± 0.05	0.93 ± 0.02	0.95 ± 0.03	0.75 ± 0.02
LayerNorm	0.88 ± 0.03	0.88 ± 0.03	0.83 ± 0.03	0.71 ± 0.04
– <i>weighted_topk_pool</i>	0.88 ± 0.03	0.86 ± 0.02	0.73 ± 0.13	0.74 ± 0.02
– <i>topk_pool</i>	0.89 ± 0.03	0.85 ± 0.03	0.71 ± 0.09	0.75 ± 0.02
– <i>pos_mean</i>	0.90 ± 0.02	0.87 ± 0.02	0.76 ± 0.06	0.79 ± 0.03
– <i>max</i>	0.88 ± 0.03	0.86 ± 0.03	0.71 ± 0.09	0.81 ± 0.02
– <i>mean</i>	0.90 ± 0.03	0.86 ± 0.03	0.74 ± 0.11	0.75 ± 0.02
InstanceNorm	0.60 ± 0.12	0.69 ± 0.07	0.58 ± 0.06	0.61 ± 0.05
– <i>weighted_topk_pool</i>	0.87 ± 0.05	0.86 ± 0.02	0.91 ± 0.05	0.74 ± 0.03
– <i>topk_pool</i>	0.87 ± 0.03	0.94 ± 0.02	0.92 ± 0.02	0.77 ± 0.03
– <i>pos_mean</i>	0.90 ± 0.03	0.94 ± 0.03	0.92 ± 0.04	0.80 ± 0.02
– <i>max</i>	0.92 ± 0.03	0.94 ± 0.02	0.92 ± 0.05	0.81 ± 0.02
– <i>mean</i>	0.87 ± 0.04	0.94 ± 0.01	0.92 ± 0.04	0.76 ± 0.02
GraphNorm	0.59 ± 0.20	0.76 ± 0.05	0.56 ± 0.09	0.60 ± 0.05
– <i>weighted_topk_pool</i>	0.86 ± 0.04	0.92 ± 0.02	0.82 ± 0.08	0.73 ± 0.02
– <i>topk_pool</i>	0.85 ± 0.05	0.92 ± 0.02	0.71 ± 0.09	0.74 ± 0.03
– <i>pos_mean</i>	0.87 ± 0.05	0.91 ± 0.01	0.86 ± 0.09	0.78 ± 0.06
– <i>max</i>	0.89 ± 0.03	0.92 ± 0.02	0.82 ± 0.10	0.80 ± 0.02
– <i>mean</i>	0.87 ± 0.05	0.90 ± 0.04	0.88 ± 0.07	0.75 ± 0.02

The results obtained using different activation modes have been reported. For each normalization type, we write in bold the results obtained with the activation mode that leads to greatest overall improvement in both ROC-AUC and accuracy

The table we have just described also contains the average performance of three known baselines: a random forest, a multi-layer perceptron (MLP) and a MPNN. Overall, the results obtained with these models are comparable or, in most cases, lower than those reached by our black-box models and always worse than those obtained with our interpretable models. Just in the case of the HIV dataset, MLP performs particularly well, overcoming the performance of the best of our interpretable models.

In Table 5, we compare the results obtained in terms of ROC-AUC by training the black-box model corresponding to the best-performing architecture in the previous experiments (GAT for BBBP, BACE, and HIV, and GCN for ClinTox) with different types of normalization. The best black-box model is the one with BatchNorm layers for all datasets. Moreover, BatchNorm is the type of normalization that allowed obtaining the best results after the addition of CW both in terms of accuracy and ROC-AUC on BBBP and ClinTox. However, this is not true for BACE, on which InstanceNorm and BatchNorm guaranteed the best performances, and HIV, on which LayerNorm and InstanceNorm gave the best results.

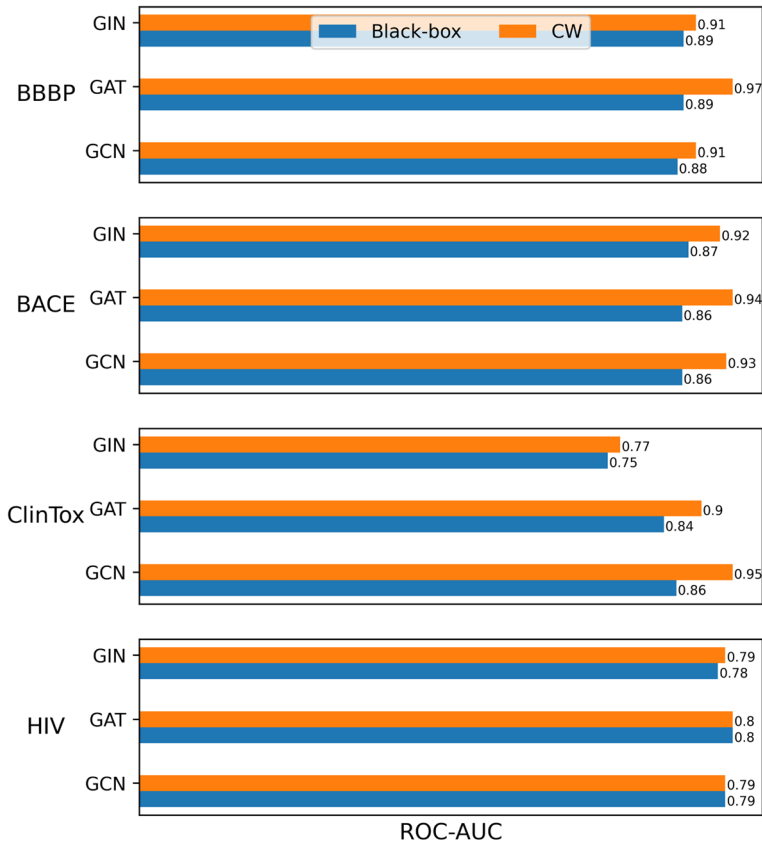


Fig. 1 Comparison between the performance of each black-box model and the best corresponding interpretable model. The ROC-AUC values obtained with CW are equal or higher than those obtained with the black-boxes

In Appendix B, we also report the accuracy values for all the experiments we have just described, both comparing different architectures in Table 8 and different normalization types in Table 9. In general, accuracies are in accordance with ROC-AUC values.

4.6 Concept representations

Since the main purpose of CW is to provide an easy way to understand which type of features are captured at a certain level of the network and to what extent each concept is relevant for the final prediction, it is useful to analyze the different importance of each concept at each layer of the network. In Fig. 2, we report the concepts importance at each of the CW layers for BBBP dataset. The contribution of each concept is computed as the sum of the positive directional derivatives of the gradients along the channel representing that concept. There are significant differences across layers. At layer 0, all importance scores are below 0.5, with the exception of logP, which is slightly above the threshold, and they are all quite similar across concepts. At layer 1, we start to notice a greater differentiation among concepts, and the importance score for NOCount reaches

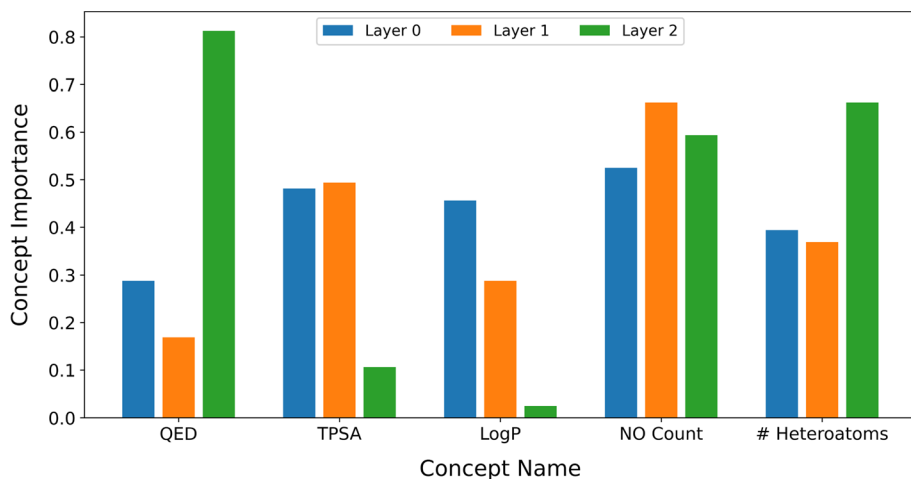


Fig. 2 Concepts importance measured at the three CW layers

0.6. Finally, the importance scores become well-separated at layer 2. The most relevant concept is QED, followed by # Heteroatoms and NOCount. On the contrary, TPSA and logP seem to be quite irrelevant. These results are in accordance with the observations made by Sakiyama et al. (2021). In fact, QED, NOCount, and # Heteroatoms are the only concepts among the five we selected that appear in all the descriptor sets studied by Sakiyama et al. (2021) resulting in the best performance in BBBP classification using a DL model. Finally, it is important to notice that the importance of the last two concepts, NOCount and # Heteroatoms, is usually in agreement. This is due to the fact that the two concepts are strongly related, since oxygen and nitrogen atoms are heteroatoms themselves.

To better understand what type of information is captured at different depths, we computed the percentile rank for the activation values for the concepts QED and NOCount at the three CW layers in the best-performing model on each dataset. Figure 3 shows an example from BBBP dataset. The trajectory confirms that the network first learns atom-level information, thus giving a higher percentile rank for the activations on the concept NOCount. Going deeper in the network, larger neighborhoods are considered, and therefore it is able to encode graph-level information, giving a higher percentile rank for the concept QED, which estimates the drug-likeness.

Finally, we analyze normalized intra- and inter-concept similarities, as computed in the paper that presents CW (Chen et al., 2020). Specifically, intra-concept similarity for concept i is defined as:

$$d_{ii} = \frac{1}{n^2} \left(\sum_{j=1}^n \sum_{k=1}^n \frac{\mathbf{x}_{ij} \cdot \mathbf{x}_{ik}}{\|\mathbf{x}_{ij}\|_2 \|\mathbf{x}_{ik}\|_2} \right) \quad (15)$$

where n is the total number of samples belonging to concept i and \mathbf{x}_{ij} is the representation for sample j of concept i . Inter-concept similarity between two concepts p and q , instead, is computed as:

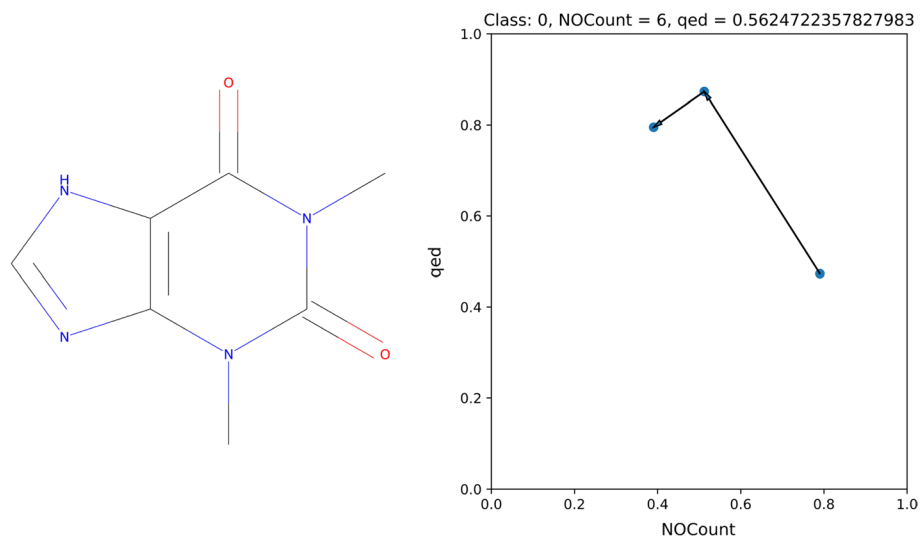


Fig. 3 Trajectory showing how the percentile rank for the activation values on the concepts QED and NOCount for the molecule on the left changes when the CW layers are inserted at different depths

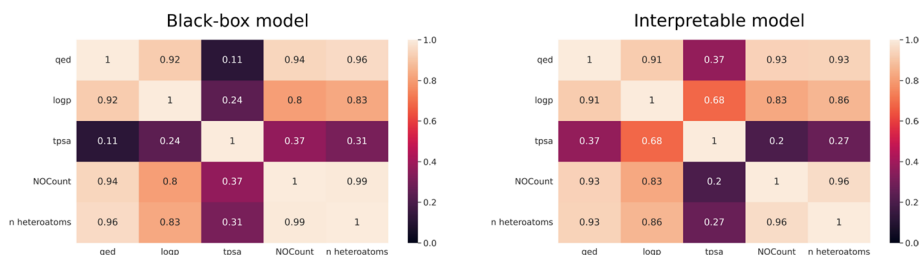


Fig. 4 Comparison between normalized intra-concept and inter-concept similarities, obtained using the best-performing interpretable model on BBBP (right) and its corresponding black-box (left). In the black-box model, the last convolutional layer is followed by a BatchNorm layer, while in the interpretable model it is followed by a CW module

$$d_{pq} = \frac{1}{nm} \left(\sum_{j=1}^n \sum_{k=1}^m \frac{\mathbf{x}_{pj} \cdot \mathbf{x}_{qk}}{\|\mathbf{x}_{pj}\|_2 \|\mathbf{x}_{qk}\|_2} \right) \quad (16)$$

where n and m are the number of samples belonging to concepts q and p , respectively. In the plot in Fig. 4, the value in cell (i, j) is obtained as follows:

$$Q_{ij} = \frac{d_{ij}}{\sqrt{d_{ii}d_{jj}}} \quad (17)$$

In Fig. 4, we notice that, in the BBBP dataset, models employing CW achieve greater separability between dissimilar concepts, more specifically NOCount and TPSA and #Heteroatoms and TPSA. At the same time, with CW the network has a greater ability to

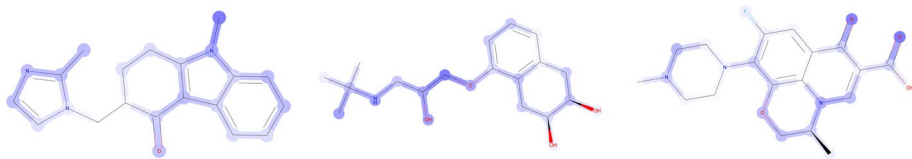


Fig. 5 Analysis of the concept-structure relationship concerning the number of heteroatoms through GNNExplainer

recognize the similarity between QED, TPSA, and LogP, and between NOCount and #Heteroatoms, which are indeed strictly related concepts.

4.7 Structural information related to concepts activations

Here, we show how to obtain a structural visualization of the concept activation using post-hoc explainability methods. In particular, we analyze the concept-structure relationship highlighted by GNNExplainer for the concept # Heteroatoms (Fig. 5) in the case of BBBP dataset.

It is interesting to observe that the atoms that more strongly influence the concept activation are indeed heteroatoms, most of the time oxygen and nitrogen atoms. These results suggest that the network is correctly identifying the structural sub-parts of the molecules that determine their belonging to a particular concept dataset.

These findings confirm that this method allows to understand the structural relationships of the molecules to descriptor-based concepts. Such a relationship could be exploited for the optimization of the molecules' biological activities in order to design novel drugs.

4.8 Improvement in interpretability

Since our aim is to propose self-interpretable QSAR models to be exploited in the field of drug discovery, we conceived interpretability as the ability to identify the structural parts of the molecules to which we need to drive chemical modifications to increase their potency. In this view, the ability to compute the importance of certain molecular properties (concepts) in order to obtain a desired type of biological activity and the possibility of using post-hoc approaches to identify the structural properties that are relevant for each concept represent by themselves an improvement in the interpretability of the models. By considering the definition of each concept, we can indeed make sure that post-hoc approaches are correctly highlighting the right atoms.

Additionally, we quantitatively evaluate the improvement in interpretability using *Fidelity+* (Pope et al., 2019). This metric is computed as the difference between the originally predicted probabilities and those obtained after masking out the important input features identified through GNNExplainer. This can be written as:

$$Fidelity+ = \frac{1}{N} \sum_{i=1}^N (f(G_i)_{y_i} - f(G_i^{1-m_i})_{y_i}) \quad (18)$$

where G_i is the i -th input graph and $f(\cdot)$ is the model to be evaluated. m_i is the importance map in which important input features correspond to 1 and all the others to 0.

Table 6 Concepts selected for each tested dataset with the chosen threshold

Model	BBBP	BACE	ClinTox	HIV
Black-box	0.24 ± 0.41	0.45 ± 0.45	-0.12 ± 0.25	0.004 ± 0.009
Interpretable	0.31 ± 0.38	0.45 ± 0.41	-0.06 ± 0.12	0.02 ± 0.19

Consequently, $G_i^{1-m_i}$ is the graph we obtain by masking out input features according to the complementary mask $1 - m_i$. Since QED and logP are abstract concepts, which cannot be directly associated to structural properties of the molecules, we need to use GNNExplainer to perform the input masking needed to compute fidelity scores.

For each dataset, we have computed the fidelity score for the best-performing interpretable GAT model and the corresponding black-box. The results are reported in Table 6. Overall, the greatest improvements in fidelity are registered for BBBP, HIV, and ClinTox. For the latter, we notice that even the interpretable model is identifying input properties whose removal brings an increase in the model's performance. This may be due to the fact that the selection of the concepts in the case of ClinTox has been performed by an experimental evaluation of those used for BBBP and BACE. Although the selected concepts proved to be useful in performing the prediction, more relevant concepts could be identified by domain experts in order to guarantee a positive fidelity score. Finally, in the case of BACE CW allows a reduction in standard deviation among fidelity scores.

5 Conclusions

This work proposes an adaptation of CW to spatial Conv-GNNs in order to develop inherently interpretable QSAR models. Thanks to CW, we obtain conceptual explanations, which allow identifying the concepts that mostly influence the output predictions by studying the evolution of the concept importance across layers. Additionally, we also provide structural explanations by combining CW with a post-hoc explanation method, namely GNNExplainer. In this way, we are able to identify the structural parts of the molecules that are relevant to a certain concept, i.e., we can retrieve input attributions. Based on them, we show an improvement in the fidelity score when using CW. We also report an improvement in terms of classification performances of the models when using CW layers. Using normalization types other than BatchNorm in the black-box can in some cases benefit the performance of the corresponding interpretable model. Finally, we propose two new activation modes, *topk_pool* and *weighted_topk_pool*, that guarantee comparable or, in some cases, better results than those obtained with *max*, *mean* and *pos_mean*.

Future works might try to use structural properties of the molecules, such as pharmacophoric points, as concepts. Additionally, it would be interesting to analyze whether changing small parts of the network leads to specific changes in the concepts' representation and how this influences the classification performance of the model. Finally, future research might further investigate the use of CW in multi-label classification tasks.

Appendix A: Concept selection

In this section, we aim at providing a brief explanation of the meaning of each concept that has been tested and of how the threshold values have been selected. The molecular properties we have considered are the following:

- Quantitative estimate of drug-likeness (QED), proposed by Bickerton et al. (2012), which expresses the similarity of a compound's properties to those of oral drugs based on eight commonly used molecular properties, namely molecular weight, lipophilicity (logP), number of hydrogen bond donors, number of hydrogen bond acceptors, (total) polar surface area (PSA or TPSA), number of rotatable bonds, number of aromatic rings, and count of alerts for undesirable substructures. The QED is obtained by taking the geometric mean of the desirability, expressed by a number between 0 and 1, of such individual properties.
- TPSA, proposed by Prasanna and Doerksen (2009), which shows the correlation with passive molecular transport through membranes, and is therefore significantly useful for the BBBP prediction.
- Logarithm of the octanol-water partition coefficient (logP), one of the principal parameters for the estimation of the molecules' hydro-lipophilicity balance and it is known to affect drugs absorption, bioavailability, hydrophobic drug-receptor interactions, metabolism of molecules, as well as their toxicity (Kujawski et al., 2012).
- The number of nitrogen and oxygen atoms (NOCOUNT).
- The number of heteroatoms (# Heteroatoms), i.e., the number of atoms that are not carbon nor hydrogen.
- Molecular weight (mol_weight), which is a measure of the sum of the atomic weight values of the atoms in a molecule.
- Number of H-bond acceptors (HBA). A hydrogen bond (H-bond) is a primarily electrostatic force of attraction between a hydrogen (H) atom which is covalently bound to a more electronegative donor atom or group, and another electronegative atom bearing a lone pair of electrons, which is the hydrogen bond acceptor.
- Number of H-bond donors (HBD).

All these molecular descriptors have been computed using RDKit,² which is an open-source tool for cheminformatics.

For molecular weight, logP, HBA, and HBD we use the threshold values based on the *Lipinski's rule of five* (Lipinski et al., 2001) for evaluating drug likeliness, according to which an orally active drug has no more than one violation of the following criteria:

- Molecular weight smaller than 500 daltons;
- LogP smaller than 5;
- Less than 5 H-bond donors;
- Less than 10 H-bond acceptors.

For all the other molecular properties, we have used as threshold values the average across each dataset. Table 7 contains the results of the experiments made to choose the concepts to be used with ClinTox.

² RDKit: Open-source cheminformatics. <https://www.rdkit.org>.

Table 7 Mean accuracies and ROC-AUC over 15 runs on ClinTox before and after the addition of the three concept whitening layers, using one concept at a time

Architecture	Accuracy	ROC-AUC
Baseline	0.94 ± 0.01	0.86 ± 0.08
CW-QED	0.95 ± 0.02	0.95 ± 0.04
CW-TPSA	0.94 ± 0.02	0.93 ± 0.04
CW-mol_weight	0.95 ± 0.02	0.93 ± 0.08
CW-HBA	0.95 ± 0.01	0.95 ± 0.02
CW-HBD	0.94 ± 0.01	0.93 ± 0.08
CW-logP	0.95 ± 0.01	0.93 ± 0.04
CW-NOCcount	0.95 ± 0.02	0.91 ± 0.09
CW-# Heteroatoms	0.95 ± 0.02	0.95 ± 0.03

We write in bold the results obtained with the concepts that guaranteed the greatest improvement in both accuracy and ROC-AUC

The baseline model used in these experiments consists of 3 GAT layers of 128 units followed by ReLU activation function. The readout layer is represented by a global max pooling. In order to perform these experiments, we have just considered *max* activation mode and BatchNorm, as we are only interested in understanding which concept determines the best improvement with respect to the same baseline model when used within the CW layers. All the models were trained with the batch sizes, learning rates, weight decay, number of epochs and early stopping patience specified in Table 2 for each dataset.

Table 7 shows that, considering the concepts used for BBBP and BACE, the improvement in accuracy values is at most 0.01. The increase in ROC-AUC, instead, is much more significant. Considering the overall improvement in both accuracy and ROC-AUC, we decided to use as concepts QED, mol_weight, HBA, logP, and # Heteroatoms.

Appendix B: Classification performances

In Tables 8 and 9, we report the accuracy values for the experiments on BBBP, BACE, ClinTox and HIV. The results are generally in accordance with ROC-AUC values presented in Sect. 4.5

Appendix C: Multi-label classification

In this section, we present the experiments performed using Tox21 dataset. It is a multi-label classification problem, since we need to predict more than one class label at a time. In fact, Tox21 contains chemical compounds associated with 12 binary labels representing the outcome (active/inactive) of 12 different toxicological experiments.

Table 8 Mean accuracies over 15 runs before and after the addition of the three concept whitening layers for all datasets

Model	BBBP	BACE	ClinTox	HIV
RF	0.89 ± 0.01	0.79 ± 0.01	0.90 ± 0.01	0.97 ± 0.00
MLP	0.80 ± 0.02	0.73 ± 0.01	0.86 ± 0.02	0.97 ± 0.00
MPNN	0.87 ± 0.02	0.64 ± 0.03	0.51 ± 0.10	0.97 ± 0.00
GCN	0.87 ± 0.02	0.78 ± 0.02	0.94 ± 0.01	0.97 ± 0.00
– <i>weighted_topk_pool</i>	0.84 ± 0.03	0.85 ± 0.02	0.95 ± 0.01	0.96 ± 0.00
– <i>topk_pool</i>	0.86 ± 0.03	0.84 ± 0.02	0.94 ± 0.02	0.96 ± 0.01
– <i>pos_mean</i>	0.87 ± 0.03	0.84 ± 0.02	0.95 ± 0.02	0.96 ± 0.01
– <i>max</i>	0.88 ± 0.02	0.84 ± 0.03	0.95 ± 0.01	0.96 ± 0.01
– <i>mean</i>	0.86 ± 0.01	0.84 ± 0.04	0.95 ± 0.02	0.96 ± 0.00
GAT	0.88 ± 0.03	0.79 ± 0.02	0.94 ± 0.02	0.97 ± 0.00
– <i>weighted_topk_pool</i>	0.87 ± 0.02	0.85 ± 0.03	0.93 ± 0.02	0.97 ± 0.04
– <i>topk_pool</i>	0.89 ± 0.02	0.86 ± 0.03	0.94 ± 0.02	0.97 ± 0.03
– <i>pos_mean</i>	0.89 ± 0.02	0.84 ± 0.02	0.94 ± 0.02	0.97 ± 0.02
– <i>max</i>	0.93 ± 0.01	0.85 ± 0.03	0.94 ± 0.02	0.97 ± 0.02
– <i>mean</i>	0.88 ± 0.02	0.85 ± 0.03	0.94 ± 0.02	0.97 ± 0.02
GIN	0.87 ± 0.02	0.78 ± 0.03	0.93 ± 0.02	0.97 ± 0.00
– <i>weighted_topk_pool</i>	0.85 ± 0.03	0.84 ± 0.03	0.93 ± 0.02	0.96 ± 0.01
– <i>topk_pool</i>	0.84 ± 0.03	0.83 ± 0.03	0.94 ± 0.02	0.97 ± 0.00
– <i>pos_mean</i>	0.87 ± 0.02	0.84 ± 0.03	0.94 ± 0.02	0.97 ± 0.00
– <i>max</i>	0.90 ± 0.02	0.83 ± 0.03	0.94 ± 0.01	0.97 ± 0.00
– <i>mean</i>	0.86 ± 0.02	0.85 ± 0.02	0.93 ± 0.02	0.96 ± 0.00

The results obtained using different activation modes have been reported. We write in bold the results obtained with the activation mode that leads to greatest overall improvement in both ROC-AUC and accuracy

Appendix C.1: Hyperparameters setup

To address this type of problem, we have configured our network by setting the number of hidden units in the output layer equal to the number of labels that need to be predicted. Then, we have used the sigmoid function for each node in the output layer to obtain the probability that a sample belongs to each class. Finally, we have used binary cross-entropy loss to perform the training.

In Table 10, we report the hyperparameters we use. As described in Sect. 4.4, all models are made up of 3 convolutional layers, but for Tox21 we have increased the number of hidden units to 256 in the case of GCN architecture. Moreover, with GAT and GIN we have not added any dense layer. Another difference is that we use ReLU function both after convolutional layers in the case of GAT and GIN, while with GCN we use LeakyReLU and ELU after convolutional and dense layers, respectively. The readout layer is a global max pooling, and as for the other datasets, it followed by a dense layer that performs the classification. Finally, by following the same procedure we have described in Sect. 4.3, we implemented early stopping with patience 20 for the black-box models and 5 for the interpretable ones. Each experiments was run 15 times to compare different seeds.

Table 9 Mean accuracies over 15 runs on all datasets before and after the addition of the three concept whitening layers using different normalization layers in the baseline models corresponding to the best interpretable models

Normalization type	BBBP	BACE	ClinTox	HIV
BatchNorm	0.88 ± 0.03	0.78 ± 0.02	0.94 ± 0.01	0.97 ± 0.00
– <i>weighted_topk_pool</i>	0.87 ± 0.02	0.85 ± 0.02	0.95 ± 0.01	0.97 ± 0.04
– <i>topk_pool</i>	0.89 ± 0.02	0.84 ± 0.02	0.94 ± 0.02	0.97 ± 0.03
– <i>pos_mean</i>	0.89 ± 0.02	0.84 ± 0.03	0.95 ± 0.02	0.97 ± 0.02
– <i>max</i>	0.93 ± 0.01	0.84 ± 0.03	0.95 ± 0.01	0.97 ± 0.02
– <i>mean</i>	0.88 ± 0.02	0.84 ± 0.04	0.95 ± 0.02	0.97 ± 0.02
LayerNorm	0.86 ± 0.02	0.76 ± 0.03	0.94 ± 0.02	0.97 ± 0.00
– <i>weighted_topk_pool</i>	0.86 ± 0.02	0.77 ± 0.02	0.93 ± 0.03	0.97 ± 0.00
– <i>topk_pool</i>	0.87 ± 0.02	0.78 ± 0.04	0.92 ± 0.02	0.97 ± 0.00
– <i>pos_mean</i>	0.88 ± 0.02	0.78 ± 0.02	0.92 ± 0.02	0.97 ± 0.00
– <i>max</i>	0.86 ± 0.02	0.79 ± 0.04	0.93 ± 0.02	0.97 ± 0.01
– <i>mean</i>	0.87 ± 0.02	0.78 ± 0.03	0.93 ± 0.02	0.97 ± 0.00
InstanceNorm	0.65 ± 0.19	0.65 ± 0.04	0.86 ± 0.02	0.97 ± 0.00
– <i>weighted_topk_pool</i>	0.86 ± 0.02	0.88 ± 0.03	0.95 ± 0.02	0.96 ± 0.01
– <i>topk_pool</i>	0.85 ± 0.03	0.88 ± 0.02	0.95 ± 0.02	0.97 ± 0.00
– <i>pos_mean</i>	0.88 ± 0.03	0.87 ± 0.03	0.95 ± 0.02	0.97 ± 0.00
– <i>max</i>	0.88 ± 0.02	0.88 ± 0.03	0.95 ± 0.01	0.97 ± 0.00
– <i>mean</i>	0.86 ± 0.02	0.88 ± 0.02	0.95 ± 0.02	0.97 ± 0.00
GraphNorm	0.75 ± 0.08	0.67 ± 0.04	0.94 ± 0.02	0.96 ± 0.00
– <i>weighted_topk_pool</i>	0.85 ± 0.02	0.85 ± 0.03	0.94 ± 0.02	0.97 ± 0.00
– <i>topk_pool</i>	0.84 ± 0.03	0.85 ± 0.03	0.92 ± 0.02	0.97 ± 0.00
– <i>pos_mean</i>	0.86 ± 0.03	0.83 ± 0.02	0.95 ± 0.02	0.97 ± 0.00
– <i>max</i>	0.86 ± 0.02	0.84 ± 0.02	0.93 ± 0.03	0.97 ± 0.00
– <i>mean</i>	0.85 ± 0.03	0.83 ± 0.03	0.94 ± 0.01	0.97 ± 0.00

The results obtained using different activation modes have been reported. For each normalization type, we write in bold the results obtained with the activation mode that leads to greatest overall improvement in both ROC-AUC and accuracy

Appendix C.2: Concepts selection

We have chosen the concepts to be used for Tox21 by following the same procedure we described in Appendix A for ClinTox dataset. In particular, we selected the concepts that used alone within the CW layers lead to the greatest improvement in performance. However, together with the concepts we had used for BBBP, BACE and HIV, we also tested some additional molecular properties used by both Jaganathan et al. (2022) and Badri and Jaims (2021). The additional concepts we considered are:

- Atom-based calculation of partition coefficient based on Crippen’s approach (Wildman & Crippen, 1999) (CrippenLogP);
- Atom-based calculation of molar refractivity based on Crippen’s approach (Wildman & Crippen, 1999) (CrippenMR);
- The number of chlorine atoms (nCl);
- The number of double bonds (# DoubleBonds);

Table 10 Hyperparameters used to train the models on the three considered datasets

Hyperparameter	Tox21
Batch size	64
Learning rate	0.001
Weight decay	5×10^{-4}
# Epochs	200
Early stopping patience ^c	20/5
Conv. hidden units	128/256 ^a
Conv. activation	ReLU/LeakyReLU ^a
Dense hidden units	–/256 ^a
Dense activation	–/ELU ^a
Readout ^b	max

^aThe second value refers just to GCN architecture

^bAll the layers are global pooling layers

^cThe first value refers to the training of the black-box model, while the second one is used while tuning the CW layers in the pre-trained models

Table 11 Mean accuracies and ROC-AUC over 15 runs on Tox21 before and after the substitution of concept whitening layers to the normalization layers within the GAT model for Tox21, using one concept at a time

Architecture	Accuracy	ROC-AUC
Baseline	0.84 ± 0.10	0.90 ± 0.10
CW–mol_weight	0.91 ± 0.01	0.96 ± 0.01
CW–QED	0.90 ± 0.01	0.97 ± 0.01
CW–TPSA	0.90 ± 0.01	0.97 ± 0.01
CW–HBA	0.91 ± 0.01	0.97 ± 0.01
CW–HBD	0.91 ± 0.01	0.98 ± 0.01
CW–logP	0.90 ± 0.02	0.97 ± 0.01
CW–NOCount	0.90 ± 0.01	0.97 ± 0.01
CW–# Heteroatoms	0.90 ± 0.01	0.97 ± 0.01
CW–CrippenLogP	0.91 ± 0.01	0.97 ± 0.01
CW–CrippenMR	0.91 ± 0.01	0.98 ± 0.00
CW–nCl	0.91 ± 0.00	0.98 ± 0.00
CW–# DoubleBonds	0.90 ± 0.01	0.97 ± 0.00
CW–complexity	0.90 ± 0.01	0.97 ± 0.01
CW–# AromaticBonds	0.90 ± 0.01	0.98 ± 0.00

We write in bold the results obtained with the concepts that guaranteed the greatest improvement in both accuracy and ROC-AUC

- A topological index meant to quantify “complexity” of molecules proposed by Bertz (1981). In particular, it consists of the sum of two terms, representing the complexity of the bonding and the complexity of the distribution of heteroatoms, respectively;
- The number of aromatic bonds.

Table 11 shows the results of these experiments, which led to the selection of HBA, HBD, CrippenLogP, CrippenMR, and nCl. For HBA and HBD we use as threshold the values

Table 12 Mean accuracies over 15 runs before and after the addition of the three concept whitening layers for the Tox21 dataset

Architecture	Accuracy	ROC-AUC
RF	0.63 ± 0.004	0.62 ± 0.003
MLP	0.65 ± 0.02	0.70 ± 0.01
MPNN	0.72 ± 0.06	0.77 ± 0.01
GCN	0.63 ± 0.02	0.71 ± 0.03
– <i>weighted_topk_pool</i>	0.63 ± 0.02	0.69 ± 0.03
– <i>topk_pool</i>	0.64 ± 0.01	0.70 ± 0.03
– <i>pos_mean</i>	0.67 ± 0.01	0.85 ± 0.02
– <i>max</i>	0.73 ± 0.01	0.92 ± 0.01
– <i>mean</i>	0.63 ± 0.01	0.70 ± 0.03
GAT	0.84 ± 0.10	0.90 ± 0.10
– <i>weighted_topk_pool</i>	0.63 ± 0.01	0.66 ± 0.03
– <i>topk_pool</i>	0.62 ± 0.01	0.64 ± 0.01
– <i>pos_mean</i>	0.75 ± 0.02	0.91 ± 0.01
– <i>max</i>	0.83 ± 0.01	0.94 ± 0.01
– <i>mean</i>	0.63 ± 0.01	0.68 ± 0.02
GIN	0.82 ± 0.02	0.94 ± 0.02
– <i>weighted_topk_pool</i>	0.62 ± 0.01	0.63 ± 0.05
– <i>topk_pool</i>	0.61 ± 0.01	0.62 ± 0.03
– <i>pos_mean</i>	0.66 ± 0.01	0.84 ± 0.02
– <i>max</i>	0.68 ± 0.01	0.85 ± 0.01
– <i>mean</i>	0.61 ± 0.01	0.64 ± 0.02

The results obtained using different activation modes have been reported. We write in bold the best results obtained with each architecture

suggested by Lipinski’s rule *Lipinski’s rule of five* (Lipinski et al., 2001), while for CrippenLogP, CrippenMR, and nCl we use the mean value across the whole dataset, namely 2.3657, 72.1641, and 0.2822.

Appendix C.3: Multi-label classification performances

In Table 12, we report the mean values for the accuracy and ROC-AUC using GCN, GAT and GIN on Tox21 dataset. Again, we compare the results obtained with and without CW, using different activation modes.

Differently from what we have seen for the other datasets, CW does not always allow improving the performance of the network. Specifically, the substitution of CW modules to the normalization layers within the GIN black-box model makes both accuracy and ROC-AUC values worse, independently from the adopted activation mode. However, this is not true for GCN and GAT, for which we obtain the best performance using *max* activation mode. In particular, in the case of GCN the accuracy increases by 0.10 and the ROC-AUC by 0.21, while in the case of GAT the accuracy slightly decreases by 0.01 and the ROC-AUC improves by 0.04.

Another important dissimilarity with respect to our previous analyses is that there is a significant divergence among the results obtained using different activation modes. For

Table 13 Mean accuracies over 15 runs before and after the addition of the three concept whitening layers using different normalization layers in the baseline models with GAT layers for the Tox21 dataset

Normalization type	Accuracy	ROC-AUC
BatchNorm	0.84 ± 0.10	0.90 ± 0.10
– <i>weighted_topk_pool</i>	0.63 ± 0.01	0.66 ± 0.03
– <i>topk_pool</i>	0.62 ± 0.01	0.64 ± 0.01
– <i>pos_mean</i>	0.75 ± 0.02	0.91 ± 0.01
– <i>max</i>	0.83 ± 0.01	0.94 ± 0.01
– <i>mean</i>	0.63 ± 0.01	0.68 ± 0.02
LayerNorm	0.86 ± 0.01	0.95 ± 0.00
– <i>weighted_topk_pool</i>	0.64 ± 0.01	0.75 ± 0.02
– <i>topk_pool</i>	0.65 ± 0.02	0.78 ± 0.03
– <i>pos_mean</i>	0.82 ± 0.01	0.93 ± 0.01
– <i>max</i>	0.81 ± 0.01	0.95 ± 0.01
– <i>mean</i>	0.65 ± 0.01	0.79 ± 0.01
InstanceNorm	0.61 ± 0.01	0.65 ± 0.03
– <i>weighted_topk_pool</i>	0.65 ± 0.01	0.80 ± 0.01
– <i>topk_pool</i>	0.64 ± 0.01	0.78 ± 0.03
– <i>pos_mean</i>	0.86 ± 0.01	0.94 ± 0.01
– <i>max</i>	0.89 ± 0.00	0.95 ± 0.00
– <i>mean</i>	0.67 ± 0.02	0.83 ± 0.01
GraphNorm	0.62 ± 0.01	0.75 ± 0.01
– <i>weighted_topk_pool</i>	0.64 ± 0.01	0.68 ± 0.04
– <i>topk_pool</i>	0.64 ± 0.01	0.74 ± 0.02
– <i>pos_mean</i>	0.84 ± 0.01	0.94 ± 0.01
– <i>max</i>	0.89 ± 0.01	0.95 ± 0.01
– <i>mean</i>	0.65 ± 0.01	0.77 ± 0.02

The results obtained using different activation modes have been reported. We write in bold the best results obtained with each normalization type

instance, using GAT architecture, *topk_pool* leads to just 0.62 of accuracy and a ROC-AUC of 0.64, which are far lower than the values obtained using *max* activation mode. In fact, analogously to BBBP and BACE, *max* is the activation mode that performs best with all three architectures and GAT is the model that leads to the best results.

Although we could expect the models to perform worse than in the previously considered tasks, since Tox21 addresses a multi-label classification problem, the best results that we obtained are good, and encourage a further investigation of CW in this type of classification tasks.

Successively, we have trained the GAT baseline model using different normalization types. The results we obtained are reported in Table 13. LayerNorm is the type of normalization that allows having better results for the baseline model. However, none of the available activation modes allows improving such performance, even if *pos_mean* and *max* let us reach good accuracy and ROC-AUC values.

On the other hand, InstanceNorm and GraphNorm lead to very poor results of the baseline models, but the addition of CW modules allows overcoming the performance obtained by the baseline model with LayerNorm. In particular, the highest gap is reported between the ROC-AUC values of the baseline model with InstanceNorm and

the corresponding interpretable model with *max* activation mode, which passes from a value of 0.65 to 0.95. Consequently, as for BACE, the best results are obtained using GAT pre-trained with InstanceNorm layers. However, while for BACE the activation mode that leads to the best results is *weighted_topk_pool*, in the case of Tox21 it is *max*.

Author contributions Alessio Ragno (AR) and Biagio La Rosa (BLR) conceived of the presented idea. Michela Proietti (MP) developed the theory and performed the computations. Rino Ragno (RR) encouraged MP to investigate the structural explanations of the concept activations and analyzed the findings concerning chemical aspects. Roberto Capobianco (RC) and Rino Ragno (RC) supervised the findings of this work. All authors discussed the results and contributed to the final manuscript.

Funding Open access funding provided by Università degli Studi di Roma La Sapienza within the CRUI-CARE Agreement. Not applicable.

Data availability The datasets that were used in all the experiments are publicly available (Wu et al., 2018).

Code availability All the code of this project will be publicly shared at the following link <https://github.com/KRLGroup/Molecular-CW> and it will be soon implemented in the www.3d-qsar.com portal (Ragno, 2019).

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval Not applicable.

Consent to participate All authors ensure their consent to participate.

Consent for publication All authors release their consent for publication.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Ba, J., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. arXiv [arXiv:abs/1607.06450](https://arxiv.org/abs/1607.06450)
- Badri, T., & Jaims, K. (2021). Determining the best set of molecular descriptors for a toxicity classification problem. *RAIRO - Operations Research*, 55. <https://doi.org/10.1051/ro/2021134>
- Bertz, S. H. (1981). The first general index of molecular complexity. *Journal of the American Chemical Society*, 103(12), 3599–3601. <https://doi.org/10.1021/ja00402a071>
- Bickerton, R., Paolini, G., Besnard, J., Muresan, S., & Hopkins, A. L. (2012). Quantifying the chemical beauty of drugs. *Nature Chemistry*, 4, 90–8. <https://doi.org/10.1038/nchem.1243>
- Bongini, P., Bianchini, M., & Scarselli, F. (2021). Molecular generative graph neural networks for drug discovery. *Neurocomputing*, 450, 242–252.
- Bongini, P., Pancino, N., Dimitri, G. M., Pancino, N., & Lio, P. (2022). Modular multi-source prediction of drug side-effects with drug. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 20, 1211–1220. <https://doi.org/10.1109/TCBB.2022.3175362>

- Bruna, J., Zaremba, W., Szlam, A., & LeCun, Y. (2014). Spectral networks and locally connected networks on graphs. In: *International conference on learning representations (ICLR2014)*. CBLIS.
- Cai, T., Luo, S., Xu, K., He, D., Liu, T. Y. & Wang, L. (2021). Graphnorm: A principled approach to accelerating graph neural network training. In: M. Meila, & T. Zhang (Eds.) *Proceedings of the 38th international conference on machine learning, proceedings of machine learning research* (Vol. 139, pp. 1204–1215). PMLR.
- Chen, J., Si, Y. W., Un, C. W., & Siu, S. W. (2021). Chemical toxicity prediction based on semi-supervised learning and graph convolutional neural network. *Journal of Cheminformatics*. <https://doi.org/10.21203/rs.3.rs-733550/v1>
- Chen, Z., Bei, Y., & Rudin, C. (2020). Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12), 772–782. <https://doi.org/10.1038/s42256-020-00265-z>
- Dai, E., & Wang, S. (2021). Towards self-explainable graph neural network. In *Proceedings of the 30th ACM international conference on information & knowledge management*. ACM. <https://doi.org/10.1145/3459637.3482306>.
- Deac, A., Huang, Y.H., Velickovic, P., Liò, P. & Tang, J., (2019). Drug-drug adverse effect prediction with graph co-attention. ArXiv [arXiv:abs/1905.00534](https://arxiv.org/abs/1905.00534).
- Debnath, A. K., Compadre, R. L. L., Shusterman, A. J., & Hansch, C. (1992). Quantitative structure-activity relationship investigation of the role of hydrophobicity in regulating mutagenicity in the Ames test: 2. Mutagenicity of aromatic and heteroaromatic nitro compounds in salmonella typhimurium TA100. *Environmental and Molecular Mutagenesis*, 19.
- Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In D. Lee, M. Sugiyama, U. Luxburg, et al. (Eds.) *Advances in neural information processing systems* (Vol. 29). Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2016/file/04df4d434d481c5bb723be1b6df1ee65-Paper.pdf>.
- Fang, Z., Kuang, K., Lin, Y., Wu, F., & Yao, Y. F. (2020). Concept-based explanation for fine-grained images and its application in infectious keratitis classification. In *Proceedings of the 28th ACM international conference on multimedia*. ACM. <https://doi.org/10.1145/3394171.3413557>.
- Fey, M., & Lenssen, J. E. (2019). Fast graph representation learning with pytorch geometric. CoRR [arXiv:abs/1903.02428](https://arxiv.org/abs/1903.02428).
- Gao, H., & Ji, S. (2019). Graph u-nets. In K. Chaudhuri, & R. Salakhutdinov (Eds.) *Proceedings of the 36th international conference on machine learning, proceedings of machine learning research* (Vol. 97, pp. 2083–2092). PMLR. <https://proceedings.mlr.press/v97/gao19a.html>.
- Ghorbani, A., Wexler, J., Zou, J. Y., & Kim, B. (2019). Towards automatic concept-based explanations. In *Advances in neural information processing systems* (pp. 9273–9282).
- Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *Proceedings of the 34th international conference on machine learning, ICML'17* (Vol. 70, pp. 1263–1272). JMLR.org.
- Gui, S., Yuan, H., Wang, J., Lao, Q., Li, K., & Ji, S.(2022). Flowx: Towards explainable graph neural networks via message flows.
- Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. In *Proceedings of the 31st international conference on neural information processing systems, NIPS'17* (pp. 1025–1035). Curran Associates Inc.
- Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., & Leskovec, J. (2020). Strategies for pre-training graph neural networks. In *International conference on learning representations*.
- Huang, L., Liu, X., Lang, B., Yu, A., Wang, Y., & Li, B. (2018a). Orthogonal weight normalization: Solution to optimization over multiple dependent Stiefel manifolds in deep neural networks. In *AAAI* (pp. 3271–3278).
- Huang, L., Yang, D., Lang, B., & Deng, J. (2018b). Decorrelated batch normalization. In *2018 IEEE/CVF Conference on computer vision and pattern recognition* (pp. 791–800).
- Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In F. Bach, & D. Blei (Eds.) *Proceedings of the 32nd international conference on machine learning, proceedings of machine learning research* (Vol. 37, pp. 448–456). PMLR.
- Ishida, S., Terayama, K., Kojima, R., Takasu, K., & Okuno, Y. (2019). Prediction and interpretable visualization of retrosynthetic reactions using graph convolutional networks. *Journal of Chemical Information and Modeling*.
- Jaganathan, K., Tayara, H., & Chong, K. T. (2022). An explainable supervised machine learning model for predicting respiratory toxicity of chemicals using optimal molecular descriptors. *Pharmaceutics*, 14(4), 832.


- Jiménez-Luna, J., Grisoni, F., & Schneider, G. (2020). Drug discovery with explainable artificial intelligence. *Nature Machine Intelligence*, 2(10), 573–584. <https://doi.org/10.1038/s42256-020-00236-4>
- Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., & Viegas, F. (2018). Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *ICML* (pp. 2673–2682).
- Kim, J., Park, S., Min, D., & Kim, W. (2021). Comprehensive survey of recent drug discovery using deep learning. *International Journal of Molecular Sciences*. <https://doi.org/10.3390/ijms22189983>
- Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *5th International conference on learning representations, ICLR 2017, Toulon, France, April 24–26, 2017, conference track proceedings*. OpenReview.net. <https://openreview.net/forum?id=SJU4ayYgl>.
- Kiralj, R., & Ferreira, M. M. (2003). A priori molecular descriptors in QSAR: A case of HIV-1 protease inhibitors: I. The chemometric approach. *Journal of Molecular Graphics and Modelling*, 21(5), 435–448. [https://doi.org/10.1016/S1093-3263\(02\)00201-2](https://doi.org/10.1016/S1093-3263(02)00201-2)
- Knyazev, B., Taylor, G. W., & Amer, M., et al. (2019). Understanding attention and generalization in graph neural networks. In H. Wallach, H. Larochelle, & A. Beygelzimer (Eds.), *Advances in neural information processing systems*. (Vol. 32). Curran Associates Inc.
- Kujawski, J., Popielarska, H., Myka, A., et al. (2012). The log p parameter as a molecular descriptor in the computer-aided drug design—an overview. *Computational Methods in Science and Technology*, 18, 81–88. <https://doi.org/10.12921/cmst.2012.18.02.81-88>
- Lenselink, E., Dijke, N., Bongers, B., et al. (2017). Beyond the hype: Deep neural networks outperform established methods using a ChEMBL bioactivity benchmark set. *Journal of Cheminformatics*, 9, 45. <https://doi.org/10.1186/s13321-017-0232-0>
- Li, Y., Vinyals, O., Dyer, C., Pascanu, R., & Battaglia, P. (2018). Learning deep generative models of graphs. *CoRR arXiv:abs/1803.03324*.
- Lim, J., Hwang, S. Y., Kim, S., et al. (2019). Scaffold-based molecular design with a graph generative model. *Chemical Science*, 11, 1153–1164.
- Lipinski, C. A., Lombardo, F., Dominy, B. W., et al. (2001). Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Advanced Drug Delivery Reviews*, 46(1–3), 3–26.
- Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems, NIPS'17* (pp. 4768–4777). Curran Associates Inc.
- Martins, I. F., Teixeira, A. L., Pinheiro, L., & Falcao, A. O. (2012). A Bayesian approach to in silico blood-brain barrier penetration modeling. *Journal of Chemical Information and Modeling*, 52(6), 1686–97.
- McCloskey, K., Taly, A., Monti, F., Brenner, M. P., & Colwell, L. J. (2019). Using attribution to decode binding mechanism in neural network models for chemistry. *Proceedings of the National Academy of Sciences of the United States of America*, 116(24), 11624–11629. <https://doi.org/10.1073/pnas.1820657116>
- Paszke, A., Gross, S., Massa, F., et al. (2019). *PyTorch: An imperative style, high-performance deep learning library*. Curran Associates Inc.
- Peng, Y., Lin, Y., Jing, X. Y., Zhang, H., Huang, Y., & Luo, G. S. (2020). Enhanced graph isomorphism network for molecular admet properties prediction. *IEEE Access*, 8, 168344–168360.
- Pope, P. E., Kolouri, S., Rostami, M., Martin, C. E., & Hoffmann, H. (2019). Explainability methods for graph convolutional neural networks. In *2019 IEEE/CVF Conference on computer vision and pattern recognition (CVPR)* (pp. 10764–10773). <https://doi.org/10.1109/CVPR.2019.01103>.
- Prasanna, S., & Doerksen, R. (2009). Topological polar surface area: A useful descriptor in 2D-QSAR. *Current Medicinal Chemistry*, 16, 21–41. <https://doi.org/10.2174/092986709787002817>
- Ragno, A., La Rosa, B., & Capobianco, R. (2022). Prototype-based interpretable graph neural networks. *IEEE Transactions on Artificial Intelligence*, PP, 1–11. <https://doi.org/10.1109/TAI.2022.3222618>
- Ragno, R. (2019). www.3d-qsar.com: a web portal that brings 3-d QSAR to all electronic devices—the py-ComFA web application as tool to build models from pre-aligned datasets. *Journal of Computer-Aided Molecular Design*, 33(9), 855–864. <https://doi.org/10.1007/s10822-019-00231-x>
- Rodríguez-Pérez, R., & Bajorath, J. (2020). Interpretation of machine learning models using shapley values: Application to compound potency and multi-target activity predictions. *Journal of Computer-Aided Molecular Design*, 34, 1013–1026.
- Rodríguez-Pérez, R., & Bajorath, J. (2019). Interpretation of compound activity predictions from complex machine learning models using local approximations and shapley values. *Journal of Medicinal Chemistry*. <https://doi.org/10.1021/acs.jmedchem.9b01101>
- Sakiyama, H., Fukuda, M., & Okuno, T. (2021). Prediction of blood-brain barrier penetration (BBBP) based on molecular descriptors of the free-form and in-blood-form datasets. *Molecules*.

- Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2009). Computational capabilities of graph neural networks. *IEEE Transactions on Neural Networks*, 20(1), 81–102. <https://doi.org/10.1109/TNN.2008.2005141>
- Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1), 61–80. <https://doi.org/10.1109/TNN.2008.2005605>
- Schnake, T., Eberle, O., Lederer, J., Nakajima, S., Schütt, K. T., Müller, K. R., & Montavon, G. (2022). Higher-order explanations of graph neural networks via relevant walks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11), 7581–7596. <https://doi.org/10.1109/TPAMI.2021.3115452>
- Schwarzenberg, R., Hübner, M., Harbecke, D., Alt, C., & Hennig, L. (2019). Layerwise relevance visualization in convolutional text graph classifiers. In *Proceedings of the thirteenth workshop on graph-based methods for natural language processing (TextGraphs-13)*, Hong Kong (pp. 58–62). <https://doi.org/10.18653/v1/D19-5308>.
- Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2019). Grad-CAM: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2), 336–359. <https://doi.org/10.1007/s11263-019-01228-7>
- Shang, C., Liu, Q., Tong, Q., Sun, J., Song, M., & Bi, J. (2021). Multi-view spectral graph convolution with consistent edge attention for molecular modeling. *Neurocomputing*, 445, 12–25. <https://doi.org/10.1016/j.neucom.2021.02.025>
- Sirois, S., Tsoukas, C., Chou, K. C., Wei, D., Boucher, C., & Hatzakis, G. E. (2005). Selection of molecular descriptors with artificial intelligence for the understanding of HIV-1 protease peptidomimetic inhibitors-activity. *Medicinal Chemistry (Shāriqah (United Arab Emirates))*, 1, 173–84. <https://doi.org/10.2174/1573406053175238>
- Subramanian, G., Ramsundar, B., Pande, V. S., & Denny, R. A. (2016). Computational modeling of β -secretase 1 (BACE-1) inhibitors using ligand based approaches. *Journal of Chemical Information and Modeling*, 56(10), 1936–1949.
- Sundararajan, M., Taly, A., & Yan, Q. (2017). Axiomatic attribution for deep networks. In *Proceedings of the 34th international conference on machine learning, ICML'17* (Vol. 70, pp. 3319–3328). JMLR.org
- Ulyanov, D., Vedaldi, A., & Lempitsky, V. S. (2016). Instance normalization: The missing ingredient for fast stylization. CoRR [arXiv:abs/1607.08022](https://arxiv.org/abs/1607.08022).
- Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2018). Graph attention networks. In *International conference on learning representations*.
- Wang, X., Li, Z., Jiang, M., Wang, S., Zhang, S., & Wei, Z. (2019). Molecule property prediction based on spatial graph embedding. *Journal of Chemical Information and Modeling*.
- Wen, Z., & Yin, W. (2013). A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142, 397–434.
- Wieder, O., Kohlbacher, S., Kuenemann, M., Garon, A., Ducrot, P., Seidel, T., & Langer, T. (2020). A compact review of molecular property prediction with graph neural networks. *Drug Discovery Today: Technologies*. <https://doi.org/10.1016/j.ddtec.2020.11.009>
- Wildman, S. A., & Crippen, G. M. (1999). Prediction of physicochemical parameters by atomic contributions. *Journal of Chemical Information and Computer Sciences*, 39, 868–873.
- Withnall, M., Lindelöf, E., Engkvist, O., & Chen, H. (2020). Building attention and edge message passing neural networks for bioactivity and physical-chemical property prediction. *Journal of Cheminformatics*. <https://doi.org/10.1186/s13321-019-0407-y>
- Wu, Z., Ramsundar, B., Feinberg, E. N., Gomes, J., Geniesse, C., Pappu, A. S., Leswing, K., & Pande, V. (2018). MoleculeNet: A benchmark for molecular machine learning. *Chemical Science*, 9(2), 513–530. <https://doi.org/10.1039/c7sc02664a>
- Xu, K., Hu, W., Leskovec, J., & Jegelka, S. (2019). How powerful are graph neural networks? In *International conference on learning representations*.
- Xu, Y., Pei, J., & Lai, L. (2017). Deep learning based regression and multiclass models for acute oral toxicity prediction with automatic chemical feature extraction. *Journal of Chemical Information and Modeling*, 57(11), 2672–2685. <https://doi.org/10.1021/acs.jcim.7b00244>
- Ying, Z., Bourgeois, D., You, J., et al. (2019). Gnnexplainer: Generating explanations for graph neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, et al. (Eds.), *Advances in neural information processing systems*. (Vol. 32). Curran Associates Inc.
- Yuan, H., Yu, H., Gui, S., & Ji, S. (2020). Explainability in graph neural networks: A taxonomic survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Zhang, J., Bargal, S. A., Lin, Z., Brandt, J., Shen, X., & Sclaroff, S. (2017). Top-down neural attention by excitation backprop. *International Journal of Computer Vision*, 126(10), 1084–1102. <https://doi.org/10.1007/s11263-017-1059-x>

- Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization. In *2016 IEEE Conference on computer vision and pattern recognition (CVPR)*. IEEE. <https://doi.org/10.1109/cvpr.2016.319>.
- Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., & Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI Open*, *1*, 57–81. <https://doi.org/10.1016/j.aiopen.2021.01.001>
- Zitnik, M., Agrawal, M., & Leskovec, J. (2018). Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics (Oxford, England)*, *34*(13), i457–i466. <https://doi.org/10.1093/bioinformatics/bty294>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Authors and Affiliations

Michela Proietti¹  · Alessio Ragno¹ · Biagio La Rosa¹ · Rino Ragno² · Roberto Capobianco^{1,3}

✉ Michela Proietti
mproietti@diag.uniroma1.it

Alessio Ragno
ragno@diag.uniroma1.it

Biagio La Rosa
larosa@diag.uniroma1.it

Rino Ragno
rino.ragno@uniroma1.it

Roberto Capobianco
capobianco@diag.uniroma1.it

¹ Department of Computer, Control and Management Engineering “Antonio Ruberti”, Sapienza University, Rome, Italy

² Rome Center for Molecular Design, Department of Drug Chemistry and Technologies, Sapienza University, Rome, Italy

³ Sony AI, Zurich, Switzerland