# Heterogeneous graph convolutional neural network for protein-ligand scoring

Kevin Crampon, Alexis Giorkallos, Xavier Vigouroux, Stephanie Baud, Luiz Angelo Steffenel

Exploration of Drug Science

Open Exploration

Open Access Original Article

# Heterogeneous graph convolutional neural network for protein-ligand scoring

Kevin Crampon[1,2,3]*, Alexis Giorkallos[1], Xavier Vigouroux[1], Stephanie Baud[2], Luiz Angelo Steffenel[3]*

[1]Eviden, 38130 Echirolles, France
[2]UMR CNRS/URCA 7369 MEDyC, Université de Reims Champagne-Ardenne, 51687 Reims, France
[3]LICIIS, Université de Reims Champagne-Ardenne, 51687 Reims, France

***Correspondence:** Kevin Crampon, kevin.crampon@univ-reims.fr; Luiz Angelo Steffenel, angelo.steffenel@univ-reims.fr. LICIIS, Université de Reims Champagne-Ardenne, 51687 Reims, France
**Academic Editor:** Walter Filgueira de Azevedo Jr., Pontifical Catholic University of Rio Grande do Sul, Brazil
**Received:** February 9, 2023 **Accepted:** March 24, 2023 **Published:** April 30, 2023

**Cite this article:** Crampon K, Giorkallos A, Vigouroux X, Baud S, Steffenel LA. Heterogeneous graph convolutional neural network for protein-ligand scoring. Explor Drug Sci. 2023;1:126–39. https://doi.org/10.37349/eds.2023.00010

## Abstract

**Aim:** Drug discovery is a long process, often taking decades of research endeavors. It is still an active area of research in both academic and industrial sectors with efforts on reducing time and cost. Computational simulations like molecular docking enable fast exploration of large databases of compounds and extract the most promising molecule candidates for further *in vitro* and *in vivo* tests. Structure-based molecular docking is a complex process mixing both surface exploration and energy estimation to find the minimal free energy of binding corresponding to the best interaction location.

**Methods:** Hereafter, heterogeneous graph score (HGScore), a new scoring function is proposed and is developed in the context of a protein-small compound-complex. Each complex is represented by a heterogeneous graph allowing to separate edges according to their class (inter- or intra-molecular). Then a heterogeneous graph convolutional network (HGCN) is used allowing the discrimination of the information according to the edge crossed. In the end, the model produces the affinity score of the complex.

**Results:** HGScore has been tested on the comparative assessment of scoring functions (CASF) 2013 and 2016 benchmarks for scoring, ranking, and docking powers. It has achieved good performances by outperforming classical methods and being among the best artificial intelligence (AI) methods.

**Conclusions:** Thus, HGScore brings a new way to represent protein-ligand interactions. Using a representation that involves classical graph neural networks (GNNs) and splitting the learning process regarding the edge type makes the proposed model to be the best adapted for future transfer learning on other (protein-DNA, protein-sugar, protein-protein, etc.) biological complexes.

## Keywords

Molecular docking, ligand-protein, scoring function, deep learning, graph neural network

## Introduction

The study of molecular interaction is the basis of drug or toxicity research. *In vitro* and *in vivo* experiments can be preceded by *in silico* filtering. These numerical simulations need huge computing resources, making them a relevant high-performance computing (HPC) use case. Moreover, the rise of artificial intelligence (AI) and specifically machine learning (ML) over the last decade has permitted better accuracy and reduced computational time. In that context, molecular docking is one of the most prominent modern techniques and is used to predict the likelihood that a small molecule, namely the ligand, will bind to a macromolecule, the receptor. In the case of that study, the former is a small chemical compound while the latter is a protein. Unlike ligand-based molecular docking which uses known ligand-receptor interactions to determine the score of the complex, structure-based molecular docking uses three-dimensional (3D) structures of both molecules to predict the complex conformation and its strength (the score). In the remainder of this article, all molecular dockings are structure-based. Molecular docking is a two-step process [1]. The first step, also called sampling, explores the ligand conformational space and in some cases the protein conformational space as well. Numerous exploration methods exist such as shape matching or systematic (iterative or incremental search), and both are based on simulation processes. However, the most popular methods are stochastic ones and rely on genetic algorithms. The second step consists in scoring each protein-ligand complex pose produced by the aforementioned step. That second step lists a great number of classic AI-free methods, which can be categorized into three classes. The first category is physics-based methods which use a weighted sum of energy terms to produce a score, the second is empirical methods where functions use a weighted sum of computationally-easy physicochemical terms, and the last is knowledge-based and uses experimental knowledge about docked protein-ligand complexes and their associated score. Finally, a mix of scoring functions from those three classes can act as an election quorum to produce a consensus score. In the case of stochastic exploration methods, that score is also used as a support guide for sampling.

Numerical simulations can also be used to reduce the exploration surface. Blind docking is used when no experimental knowledge or intuition-driven search is available, then the whole protein surface must be explored, and this tends to result in poorer models and less accurate scoring. Instead, targeted docking focuses only on promising sites, which is why binding site detection methods now exist to predict the putative binding site of a protein using its 3D structure.

Most recently, AI and ML have been investigated to speed up molecular docking. Indeed, the use of AI and ML has grown considerably in all scientific fields, and, in the case of molecular docking, ML started in the 2010s with random forest (RF) based methods such as RF-score [2]. The first use of neural network methods also dates to that period, but their use soared with convolutional neural networks (CNNs), introduced in 2015 with AtomNet [3]. The simplest way to represent data in these methods is the atom-based discretization of both molecules on a 3D grid of predefined cell size. Physicochemical properties are then added as features on each atom.

Lately, new variants of deep learning (DL) methods have emerged, using graph convolutional networks (GCNs) that generalize the well-known grid-based convolution to unstructured or graph-structured data. The graph structure allows for representing a molecule more reliably and accurately, allowing the proposal of more powerful scoring functions.

Hence, this paper proposes the heterogeneous graph score (HGScore) method, a protein-ligand complex scoring function based on a heterogeneous GCN (HGCN). The method is available on GitHub: github.com/KevinCrp/HGScore. After introducing existing AI methods for scoring, a dataset often used in the literature and the data representation are presented. Then, the proposed scoring method is described and this article is concluded by presenting a comparative study against existing methods.

### Related works

In a previous work [1], a comprehensive survey was performed on the use of ML and especially DL for protein-ligand binding scoring. From this work, representative binding scoring methods are extracted, and they will be used as a comparison basis for the evaluations in this paper. For instance, Pafnucy, proposed by Stepniewska-Dziubinska et al. [4], discretizes all the atoms on a 3D grid by adding a set of physicochemical

features on each point. Their proposed neural network comprises three convolutional layers followed by three fully connected layers, the last layer has only one neuron producing the binding score.

With DeepBindRG, Zhang et al. [5] used a 2D map to represent the protein-ligand interactions as images, allowing to use of residual network (ResNet) [6], a well-known CNN architecture. In the end, the 1D vector produced is used as input by a fully connected layer producing the score.

In 2019, Zheng et al. [7] proposed the OnionNet method, where the main idea is to represent the protein-ligand contacts by a series of shells. A series of shells is built around each ligand atom, then for each shell, all ligand-protein atom pairs are counted, according to their atom type (8 types for the protein and 8 types for the ligand) resulting in 64 features for each shell. The authors use 60 shells for each ligand atom, thus each of them is represented by 3,840 features. Finally, their model is composed of three convolutional layers followed by three fully connected layers.

The algebraic graph learning score (AGL-Score) [8] is not a DL method but it outperforms the previous method. The protein-ligand complex is transformed into a graph where node features are atom type and position, whereas the graph edges are the non-covalent bonds that connected the atoms. Finally, several descriptive statistics are produced from the graph adjacency (or Laplacian) matrix eigenvalues, and that vector is then used as an input to train a gradient-boosting tree [9].

The InteractionGraphNet (IGN) method proposed by Jiang et al. [10] represents the molecular complex using three independent graphs, one representing the protein, one for the ligand and the former is the protein-ligand graph (a bipartite graph whose nodes represent atoms from the ligand or the protein, and edges represent inter-molecular bonds). The method starts by working on both molecular graphs using convolutions, then the nodes embedding from both molecules are used as input features for the protein-ligand graph.

In 2021, the Extended Connectivity Interaction Features (ECIF) has been proposed by Sánchez-Cruz et al. [11]. The featurization produces an interaction fingerprint which represents the number of atomic pairs according to the atom type in the interaction. Their method called ECIF6::LD-GBT takes as input the fingerprint and uses a gradient-boosting tree to produce a binding score.

## Materials and methods

### Dataset

To compare the performance of the proposed method and other works, the Protein Data Bank-bind (PDBbind) 2020 dataset [12] is used. It contains 19,443 ligand-protein complexes, each input of which is composed of a Protein Data Bank (PDB) file representing the protein, a PDB file for the protein binding pocket, and a MOL2 file for the ligand. In PDBbind, a binding pocket contains all residues whose at least one heavy atom is closer than 10 Å to one of the ligand atoms. Each ligand-protein complex has its associated binding affinity, express by $-\log(K_i)$ or $-\log(K_d)$ as a function of the complexes. Hereafter, only the protein binding pocket is used to describe the protein part. PDBbind authors have proposed splitting their database into several sets, as shown in Figure 1. The first is the general set containing all database inputs, while the second is the refined set created by selecting only the best-quality complexes from that first set. Both sets are updated each year. Finally, the third set consists in a core set that contains very few inputs but with high-quality [13].

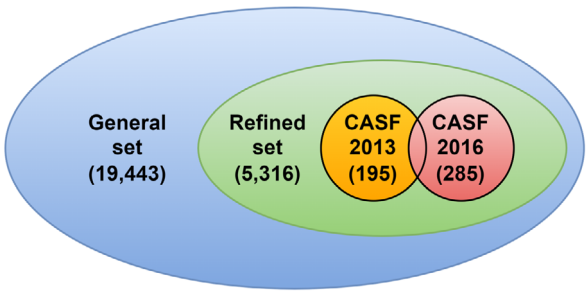


**Figure 1.** The PDBbind dataset and its components. The general set with 19,443 protein-ligand complexes, the refined set (included in the general) with 5,316 complexes, and both comparative assessment of scoring functions (CASF) benchmarks with 195 and 285 complexes respectively for the 2013 and the 2016 versions

In this work, the versions 2013 and 2016 of the core set are used as test sets (195 and 285 items, respectively), and 1,000 inputs from the refined set are randomly extracted to build the validation set and the remaining input as the training set (18,070 items). It is ensured that there are no duplicates between sets, except for the core sets which naturally have overlapping entries (elements from 2016 already present in the 2013 dataset). Also, in the pre-processing step, the protein files are cleaned by keeping only protein atoms, so that water and other hetero atoms are removed.

## Data representation

A graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is a set of $V$ nodes (also called vertices) $\mathcal{V}$ linked by a set of $E$ edges $\mathcal{E}$. Each node $v_i \in \mathcal{V}$ can have a set of features named $\overrightarrow{v_i} \in \mathbb{R}^n$. Idem for the edges with their respective $\overrightarrow{e_i} \in \mathbb{R}^m$.

A heterogeneous graph, as presented in Figure 2, is composed of several graphs but in that case, only a pair of graphs, $[\mathcal{G}_1(\mathcal{V}_1, \mathcal{E}_1)$ and $\mathcal{G}_2(\mathcal{V}_2, \mathcal{E}_2)]$ is used, each can have a different set of features: $\mathcal{V}_1 \in \mathbb{R}^{N_1 \times n_1}$, $\mathcal{V}_2 \in \mathbb{R}^{N_2 \times n_2}$, $\mathcal{E}_1 \in \mathbb{R}^{E_1 \times m_1}$ and $\mathcal{E}_2 \in \mathbb{R}^{E_2 \times m_2}$. In addition, two new sets of edges ($\mathcal{E}_{1\to2}$ and $\mathcal{E}_{2\to1}$) allow connecting nodes from one graph to the second: $\mathcal{E}_{1\to2} \in \mathbb{R}^{E_{12} \times l_{12}}$ and $\mathcal{E}_{2\to1} \in \mathbb{R}^{E_{21} \times l_{21}}$.

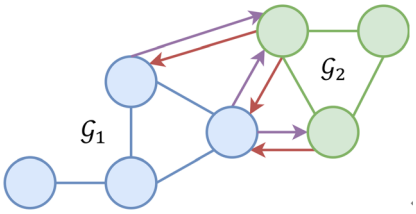


**Figure 2.** Example of a heterogeneous graph. Blue nodes belong to $\mathcal{G}_1$ and green nodes belong to $\mathcal{G}_2$. Blue edges belong to $\mathcal{E}_1$, green edges belong to $\mathcal{E}_2$, purple arrows correspond to $\mathcal{E}_{1\to2}$ and red arrows correspond to $\mathcal{E}_{2\to1}$

Each molecule (protein and ligand) is represented with a graph of heavy atoms, thus, each graph node represents an atom, and the graph edges represent molecular bonds. A total of 23 physicochemical features are computed using Open Drug Discovery Toolkit (ODDT) [14] and are presented in Table 1. The 23 features are used as the second dimension for both the protein and ligand nodes. Then each ligand-protein atom pair closer than 4 Å is connected, also completed by a set of features enumerated in Table 2. Then after featurization is complete, each complex is represented by a protein graph $\mathcal{G}_p(\mathcal{V}_p, \mathcal{E}_p)$, $\mathcal{V}_p \in \mathbb{R}^{N_P \times 23}$, $\mathcal{E}_p \in \mathbb{R}^{E_P \times 6}$, a ligand graph $\mathcal{G}_L(\mathcal{V}_L, \mathcal{E}_L)$, $V_L \in \mathbb{R}^{N_L \times 23}$, $\mathcal{E}_L \in \mathbb{R}^{E_L \times 6}$ and two inter-molecular edge sets $\mathcal{E}_{P\to L} \in \mathbb{R}^{E_{PL} \times 6}$ and $\mathcal{E}_{L\to P} \in \mathbb{R}^{E_{LP} \times 6}$. A protein-ligand complex and its corresponding heterogeneous graph are represented in Figure 3. Heterogeneous graphs are the way to get the most accurate representation, allowing to represent atoms and bonds while differentiating them, in that case, according to their belonging to the protein or ligand.

**Table 1.** Nodes features

| Feature | Size | Description |
|---|---|---|
| Atom type | 8 | One-hot encoded (B, C, N, O, F, P, S, and others) |
| Hybridization | 7 | One-hot encoded (other, sp, $sp^2$, $sp^3$, sq planar, trigonal bipyramidal, octahedral) |
| Partial charge | 1 | Float (Gasteiger charge model) |
| Is hydrophobic | 1 | Boolean |
| Is in an aromatic cycle | 1 | Boolean |
| Is H-bond acceptor | 1 | Boolean |
| Is H-bond donor | 1 | Boolean |
| Is H-bond donor hydrogen | 1 | Boolean |
| Is negatively charged/chargeable | 1 | Boolean |
| Is positively charged/chargeable | 1 | Boolean |

**Table 2.** Intra- and Inter-molecular edge features

| Feature | Size | Intra | Inter | Description |
|---|---|---|---|---|
| Length/distance | 1 | X | X | Float |
| Belongs to an aromatic cycle | 1 | X | - | Boolean |

**Table 2.** Intra- and Inter-molecular edge features (*continued*)

| Feature | Size | Intra | Inter | Description |
|---|---|---|---|---|
| Belongs to a ring | 1 | X | - | Boolean |
| Type | 3 | X | - | One-hot (simple, double, or triple) |
| Is a hydrogen bond | 1 | - | X | Boolean |
| Is a salt bridge | 1 | - | X | Boolean |
| Is a hydrophobic contact | 1 | - | X | Boolean |
| Has π-stacking | 1 | - | X | Boolean |
| Has π-cation | 1 | - | X | Boolean |

π-stacking, and π-cation are only available for inter-molecular edges whose terminal atom from the protein side belongs to a ring. X: applicable; -: not applicable

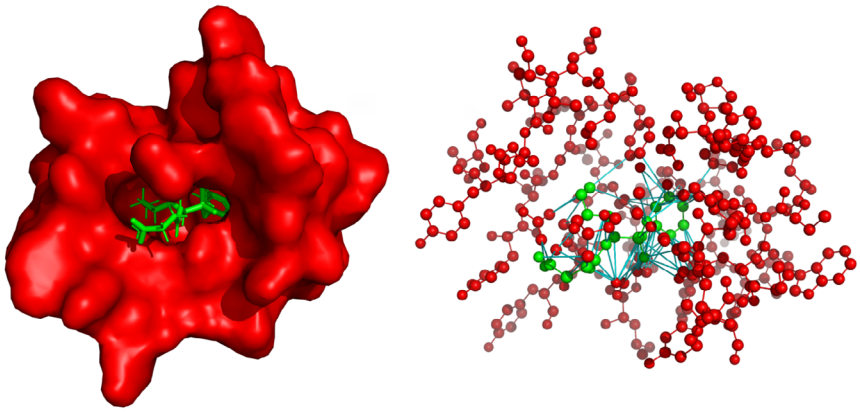

**Figure 3.** A protein-ligand complex and its associated heterogeneous graph. On the left, the 29G11 protein (in red) complexed with the ligand HEP in green (PDB: 1a0q). On the right, the heterogeneous graph represents the same complex, red nodes represent the protein binding site atoms, and the green nodes are the ligand atoms, both without hydrogen. Red and green edges represent intramolecular bonds, red edges belong to the protein while green edges belong to the ligand. Finally, the cyan edges are the inter-molecular links representing the protein-ligand interaction

## Method

### Graph convolution

Graph neural networks (GNNs) are a flexible type of neural objects that exist in various flavors, as classified by Wu et al. [15]. While recurrent GCNs (RGCNs) were the first attempt at modern graph neural modeling, nowadays GCNs are the most popular kind. GCNs are created by stacking multiple graph convolutional layers and updating the state of the graph at each step.

The implementation is based on PyTorch Geometric [16], a framework relying on the concept of message-passing neural network (MPNN, Equation 1) that includes libraries for sparse computation. Hence, the principle of spatial convolution as presented in Figure 4 is used.

$$h_i^{(l+1)} = U_l\left(h_i^{(l)}, \sum_{j \in \mathcal{N}(i)} M_l\left(h_i^{(l)}, h_j^{(l)}, e_{j \to i}\right)\right) \quad \text{Equation 1}$$

Where $h_i^{(l)}$ is the hidden features vector of node $i$ at layer $l$, $h_l^{(0)} = \vec{v_l}$, $U_l$ and $M_l$ are functions with learnable parameters and $N(i)$ is the neighborhood of node $i$.

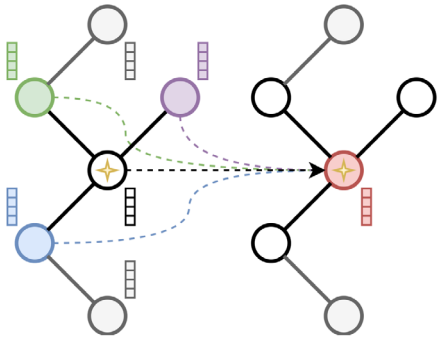

**Figure 4.** The spatial convolution process. In spatial convolution, the current node, here identified by a star, aggregates information coming from its direct neighbors here represented in green, purple, and blue. The current node feature vector is then updated as shown in the graph on the right

### Layers

The proposed method is based on Attentive FP [17], which was adapted to work with molecular complexes. While the neural model was originally designed to produce a molecular fingerprint allowing the prediction of chemical properties such as toxicity or solubility, it was adapted to be applied to a molecular complex (ligand-protein) to produce a binding affinity score. That model is composed of graph attentional convolution (GATConv) and gated recurrent unit cell (GRUCell) layers, explained below.

The GATConv layer proposed by Veličković et al. [18] allows using an attention mechanism on GCN. In this work, the second version of the GATConv (GATv2Conv) layer proposed by Brody et al. [19] that fixes the static attention problem of GATConv is used. Thus, the network can learn which neighbor nodes are more relevant thanks to implicit weights. Indeed, in addition to the matrix of weights $W$, the GATConv operators learn an attention coefficient $a_{ij}$ representing the magnitude of the edge $j \rightarrow i$ or in other words the importance of node features of $j$ for explaining node $i$. All neighbors $j$ of node $i$ have a coefficient representing their importance for $i$. Equations 2 and 3 are used to compute the attention coefficient, and Equation 4 updates the current node features according to this coefficient.

$$e(h_i, h_j) = a^{\top}\text{LeakyReLU}(Wh_i \,||\, Wh_j) \quad \text{Equation 2}$$

Where the function $e$ allows scoring the edge from $j$ to $i$ nodes, $a$ and $W$ are learned, the $||$ is the vector concatenation operator, $\top$ the transposition operator, and LeakyReLU the leaky rectified linear unit.

$$\alpha_{ij} = \text{softmax}_j\left(e\left(h_i, h_j\right)\right) = \frac{\exp\left(e\left(h_i, h_j\right)\right)}{\sum_{j' \in \mathcal{N}_i} \exp\left(e\left(h_i, h_{j'}\right)\right)} \quad \text{Equation 3}$$

As explained by Veličković et al. [18], all attention coefficients are normalized by a soft-max function over all node neighbors.

$$h_i' = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij} W h_j\right) \quad \text{Equation 4}$$

Where $\sigma$ is a nonlinearity function.

The GRUCell layer proposed by Cho et al. [20] is a recurrent neural network layer that keeps the memory of items over long sequences of data. A GRUCell layer is described by four equations to compute the layer update. The first two equations are the update gate (Equation 5) and the reset gate (Equation 6), respectively allowing relevant information to flow from the past ($h_{t-1}$) or to be forgotten. The current memory content follows Equation 7 and stores the relevant information from the past. Finally, the layer is updated with Equation 8.

$$z_t = \sigma\left(W^{(z)}x_t + U^{(z)}h_{t-1}\right) \quad \text{Equation 5}$$

$$r_t = \sigma\left(W^{(r)}x_t + U^{(r)}h_{t-1}\right) \quad \text{Equation 6}$$

$$h_t' = \tanh(Wx_t + r_t \odot Uh_{t-1}) \quad \text{Equation 7}$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h_t' \quad \text{Equation 8}$$

Where $\odot$ is the Hadamard product.

### HGScore

As mentioned above, the heterogeneous graph contains two sub-graphs (protein and ligand) in which nodes are connected. Thus, 4 types of edges: $\mathcal{G}_p \leftrightarrow \mathcal{G}_p$, $\mathcal{G}_L \leftrightarrow \mathcal{G}_L$, $\mathcal{G}_p \leftrightarrow \mathcal{G}_L$, and $\mathcal{G}_L \leftrightarrow \mathcal{G}_p$ are used, resulting in 6 independent GCNs (one for each edge type in atomic part and one for each molecule in molecular part), and ending with a multi-layer perceptron (MLP) to compute the final score as presented in Figures 5–7. Different ways to construct the network including one that does not use the molecular embedding part and feeds the atomic embedding pooling to the MLP have been tested. The proposed method, used in the remainder of this paper, is the strategy that showed the best results.

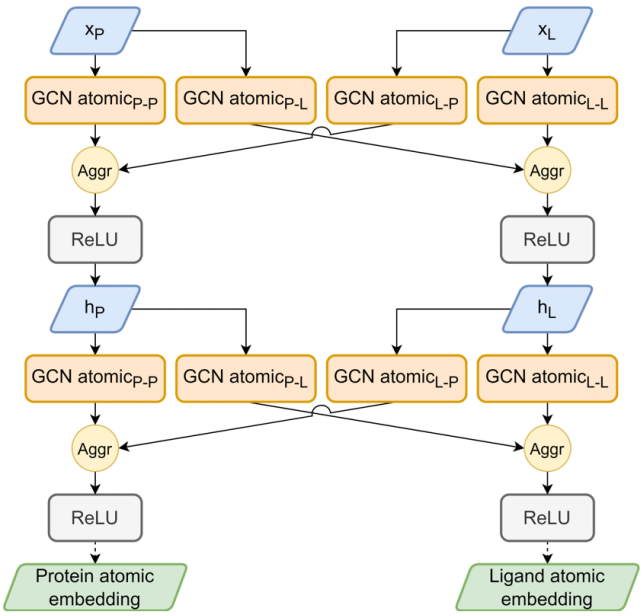


**Figure 5.** The HGScore atomic embedding part. The atomic embedding part extends Attentive FP atomic embedding to work on the heterogeneous graph to produce an atomic embedding of both molecules. The aggregation (Aggr) module allows a combination of previous layer outputs using mean, add, or max Aggr. ReLu: rectified linear unit

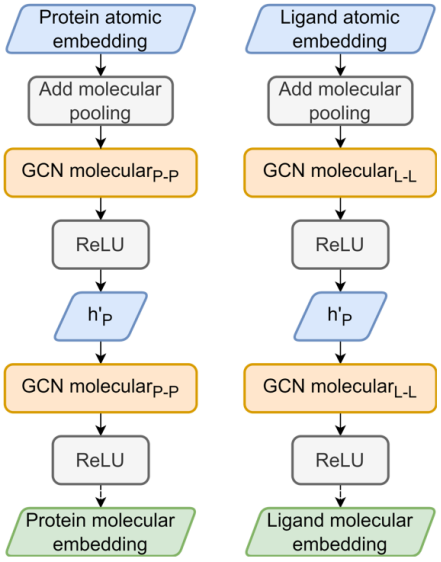


**Figure 6.** The HGScore molecular embedding part. The two atomic part embeddings are pooled by a sum to produce a molecular embedding which is used in the classical Attentive FP molecular embedding part to produce protein and ligand fingerprints

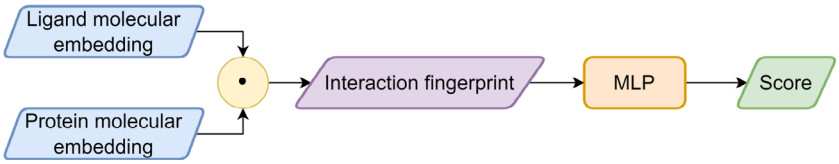


**Figure 7.** The HGScore prediction part. The two molecular part fingerprints are concatenated into an interaction fingerprint allowing an MLP to predict an affinity score

The atomic-GCN embedding part of the model, presented in Figure 5, encodes atomic features in a latent space. For both molecules, nodes are updated by two GCNs, one for each edge type. This resulted in four GCNs, all Attentive FP-inspired and using GATv2Conv and GRUCell. The main idea is to dissociate the embedding for each edge type so that each model will mix an intertwined pair of input signals enriched by their specific context (protein-protein, protein-ligand, ligand-protein, ligand-ligand). The resulting signal propagates information from context to context several times over by repeating the same operation.

Once the atomic-GCN embedding is completed, the same algorithm (Figure 6) proposed by Attentive FP (the molecular embedding) is applied to each molecule fingerprint, which consists of a sum global pooling,

i.e., all node vectors are summed to produce a unique molecular representation. This vector is then processed by a molecular-GCN composed of GATv2Conv and GRUCell, also built from Attentive FP.

Finally, the concatenation of the two out-coming molecular fingerprints is used to produce the interaction fingerprint, which is fed to a classic feed-forward MLP (Figure 7), resulting in an affinity score of the complex.

## Metrics

To test and compare HGScore, the metrics and the two benchmarks (CASF 2013 and CASF 2016, also known as PDBbind core set 2013 and 2016) provided by Su et al. [13] are used. The Pearson correlation coefficient $R_p$ (Equation 9) is used as the main metric, and the standard deviation (SD) in regression is the second (Equation 10). These metrics were proposed by Su et al. [13] as a good way to measure the scoring power of a scoring function. $R_p$ allows determining if two sets of values are linearly correlated. $R_p$ closer to 1.0 is considered best, with minimal SD. Other metrics of interest: the mean absolute error (MAE, Equation 11) and the root mean square error (RMSE) are also used (Equation 12).

$$R_p = \frac{\sum_{i=1}^{n}(x_i - \overline{x})\,(y_i - \overline{y})}{\sum_{i=1}^{n}(x_i - \overline{x})^2 \sum_{i=1}^{n}(y_i - \overline{y})^2} \quad \text{Equation 9}$$

$$\text{SD} = \sqrt{\frac{\sum_{i=1}^{n}\left(y_i - (a + bx_i)\right)^2}{n-1}} \quad \text{Equation 10}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - x_i| \quad \text{Equation 11}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2} \quad \text{Equation 12}$$

Where $x_i$ is the predicted score for item $i \in [\![1, n]\!]$, $y_i$ the target score, $\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $\overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$. $a$ and $b$ are the linear regression coefficients on predicted values.

Su et al. [13] also provide metrics to measure the ranking power of a scoring function; this power reflects the capacity of the function to correctly order a set of ligands docked on the same protein. Thus, Spearman's rank correlation coefficient ($\rho$, Equation 13) [21], Kendall's rank correlation coefficient ($\tau$, Equation 14) [22], and the predictive index (PI) are used (Equation 15), all being within the interval [−1, 1] and 1 indicating a perfect ordering.

$$\rho = \frac{\sum_{i=1}^{n}(rx_i - \overline{rx})\,(ry_i - \overline{ry})}{\sqrt{\sum_{i=1}^{n}(rx_i - \overline{rx})^2}\sqrt{\sum_{i=1}^{n}(ry_i - \overline{ry})^2}} \quad \text{Equation 13}$$

$$\tau = \frac{P_{Concord} - P_{Discord}}{\frac{1}{2}n(n-1)} \quad \text{Equation 14}$$

Where $n$ is the number of ligands ranked for each target, it is 3 for CASF 2013 and 5 for the 2016 version, $rx_i$ is the rank of the predicted score $x_i$, and $ry_i$ is the rank of the ground truth (obtained experimentally). $P_{Concord}$ and $P_{Discord}$ correspond to the number of concording and discording pairs. Let a sample be a couple ($x_i$, $y_i$) with $x_i$ the predicted value and $y_i$ the ground truth, then a pair of sample [($x_i$, $y_i$), ($x_j$, $y_j$)] is concording if $x_i < x_j$ and $y_i < y_j$ or $x_i > x_j$ and $y_i > y_j$, is discording if $x_i < x_j$ and $y_i > y_j$ or $x_i > x_j$ and $y_i < y_j$, else the pair is not counted.

$$\text{PI} = \frac{\sum_{j>i}^{n}\sum_{i}^{n} W_{ij}\,C_{ij}}{\sum_{j>i}^{n}\sum_{i}^{n} W_{ij}} \quad \text{Equation 15}$$

$$\text{Where } W_{ij} = |y_j - y_i| \text{ and } C_{ij} = \begin{cases} 1 \text{ if } \frac{y_j - y_i}{x_j - x_i} > 0 \\ -1 \text{ if } \frac{y_j - y_i}{x_j - x_i} < 0 \\ 0 \text{ else} \end{cases}.$$

The model is also assessed on the docking power [13] which allows to measure the capacity of a scoring function to detect near-native ligand binding poses among decoys. Two metrics are used, the first is the success rate in tops 1, 2, and 3 (the lowest predicted score is associated with the top 1 ligand, the second lowest to the top 2 ligand, and so on); then a given prediction is a success if the root mean square deviation (RMSD) (Equation 16) between the ligand in question and the native ligand pose is less than a cutoff value (2 Å). The second metric is the Spearman correlation coefficient (Equation 13) between the RMSD and the predicted score on different subsets of all ligands (decoys and native). The subset $i$ contains all ligands whose RMSD (with the native one) is less than $i$, for $i \in [\![2, 10]\!]$.

$$\text{RMSD} = \sqrt{\frac{\sum_{i=0}^{n}\left(X_P^i - X_N^i\right)^2}{n}} \quad \text{Equation 16}$$

Where $P$ is the predicted atom coordinates, $N$ is the native ligand atom coordinates, and $n$ is the number of ligand atoms.

## Results

The method's implementation is based on PyTorch, Lightning, and PyTorch Geometric [16] to build the graph data, the datasets, and the model, and perform training and evaluation. Optuna was used to perform hyper-optimization to find the best architecture and perform the training with the same hyperparameter set several times over because the model is sensitive to weight initialization [23]. As a result, the model is composed of 3 atomic embedding layers with mean Aggr whose sizes are {48, 57, 68} for the protein part and {48, 76, 121} for the ligand part, followed by 4 molecular embedding layers. All atomic parts have 2 heads for their respective GATv2Convs. The interaction fingerprint is a vector of size 189 fed to an MLP with channels {189, 94, 1}.

During training, a learning rate of $10^{-3}$ associated with the Adam optimizer and a weight decay equal to $10^{-4}$ are used. The neural model is trained on 300 epochs with a dropout set at 0.13 (applied to all layers), but unpromising, stagnant candidates are stopped (with early stopping). The model was trained on an Nvidia DGX [a server with two sockets of 20 cores each, 512 gigabytes (GB) of random access memory (RAM)] using one graphics processing unit [GPU, an Nvidia V100 with 16 GB of video RAM (VRAM)], 300 epochs taking about 3 h00. The mean inference time is 0.10 s using one central processing unit (CPU), however, the necessary time to construct the graph from the complex file must be considered. That processing takes nearly 0.05 s (on the CASF 2016 complexes), but that time does not include the time for extracting the pocket if necessary, which is approximately 0.30 s per complex. The different times were not deeply studied because the other articles do not provide a basis for comparison. The results for the scoring power are presented in Figures 8 and 9 and Table 3. The scoring power results show a good $R_p$, especially for the CASF 2016, and both error metrics (MAE and RMSE) are pretty low, thus, the model is validated with respect to the scoring power.

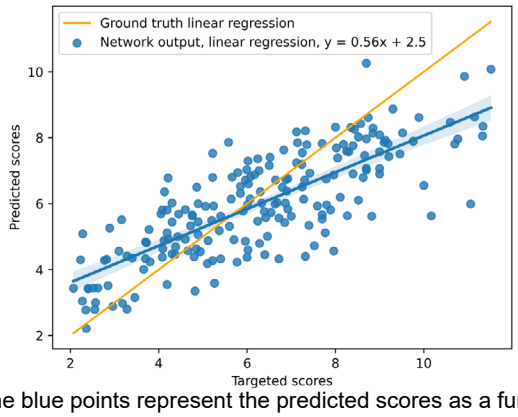


**Figure 8.** Results on CASF 2013. The blue points represent the predicted scores as a function of the targeted ones, the blue line is the blue points linear regression, and the orange line is the ground truth linear regression

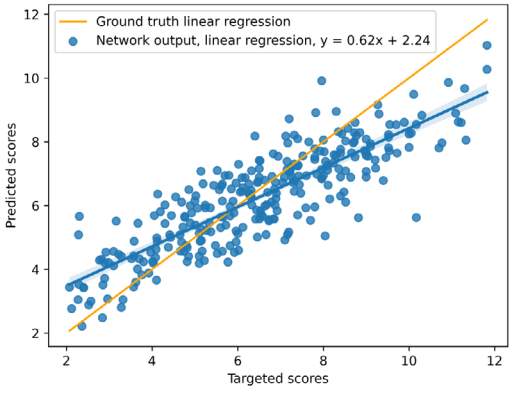


**Figure 9.** Results on CASF 2016. The blue points represent the predicted scores as a function of the targeted ones, the blue line is the blue points linear regression, and the orange line is the ground truth linear regression

**Table 3.** Scoring power metrics for HGScore method on CASF 2013 and 2016

| Dataset | $R_p$↑ | SD↓ | MAE↓ | RMSE↓ |
|---|---|---|---|---|
| CASF 2013 | 0.79 | 1.39 | 1.11 | 1.39 |
| CASF 2016 | 0.85 | 1.14 | 0.90 | 1.14 |

↑ indicates that the larger the value, the better; ↓ indicates that the smaller the value, the better

Ranking power results are presented in Table 4, which shows that the model has good performance for the ranking power, especially on the CASF 2016 dataset.

**Table 4.** Ranking power metrics for HGScore on CASF 2013 and 2016

| Dataset | $\rho$↑ | $\tau$↑ | PI↑ |
|---|---|---|---|
| CASF 2013 | 0.59 | 0.55 | 0.60 |
| CASF 2016 | 0.74 | 0.67 | 0.76 |

↑ indicates that the larger the value, the better

Docking power results are presented in Tables 5 and 6. As shown in Table 5, the method ranks perfectly 1/3 of the native ligands (or a closer one) among the decoys. Moreover, the method ranks more than half of the native ligands (or a closer one) in the top 2 and 3. However, Table 6 does not show a good ranking performance; that lack of performance may be caused by the large number of used decoys and the small range of all scores (the average score range is 3.0) so the score range being small the model has more chance of not respecting the RMSD ranking. It should be noted that the fact that the produced scores are in a small range is because all the experimental scores of the training set are themselves in a small range.

**Table 5.** The docking power success rate (%) on the CASF 2016 dataset

| TOP 1 | TOP 2 | TOP 3 |
|---|---|---|
| 32.63 | 51.93 | 60.35 |

**Table 6.** The docking power Spearman correlation coefficient metric on CASF 2016

| RMSD range (Å) | Coefficient↑ |
|---|---|
| [0, 2] | 0.17 |
| [0, 3] | 0.20 |
| [0, 4] | 0.23 |
| [0, 5] | 0.25 |
| [0, 6] | 0.26 |
| [0, 7] | 0.27 |
| [0, 8] | 0.28 |
| [0, 9] | 0.30 |
| [0, 10] | 0.30 |

↑ indicates that the larger the value, the better

## Discussion

In addition to the intrinsic characterization of the new scoring method, the HGScore results are compared to some other DL scoring methods in Tables 7 and 8 respectively for CASF 2013 and 2016. The other methods compared are Pafnucy [4], DeepBindRG [5], OnionNet [7], IGN [10], and ECIF6::LD-GBT [11]. Some classical scoring function performances on CASF 2013 [24] and CASF 2016 [13] are presented to provide a no AI-centric comparison.

**Table 7.** Metrics computed on the CASF 2013

| Method | Trained on PDBbind | *N* | $R_p$↑ | SD↓ | MAE↓ | RMSE↓ |
|---|---|---|---|---|---|---|
| GoldScore[a] [24] | - | 195 | 0.48 | 1.97 | - | - |
| ChemScore[b] [24] | - | 195 | 0.59 | 1.82 | - | - |

**Table 7.** Metrics computed on the CASF 2013 (*continued*)

| Method | Trained on PDBbind | *N* | $R_p$↑ | SD↓ | MAE↓ | RMSE↓ |
|---|---|---|---|---|---|---|
| X-Score$^{HM}$ [24] | - | 195 | 0.61 | 1.78 | - | - |
| DeepBindRG [5] | 2018 | 195 | 0.64 | - | 1.48 | 1.82 |
| Pafnucy [4] | 2016 | 195 | 0.70 | 1.61 | - | 1.62 |
| OnionNet [7] | 2016 | 108 | 0.78 | 1.45 | 1.21 | 1.50 |
| AGL-Score$^{c}$ [8] | 2007 (refined) | 195 | 0.79 | - | - | 1.97 |
| IGN$^{c}$ [10] | 2016 | 85 | 0.83 | - | - | - |
| HGScore | 2016 | 195 | 0.79 | 1.39 | 1.11 | 1.39 |

The scoring function was used with the software [a] Gold and [b] Sybyl. [c] the number of tested complexes is too low to be compared with the other methods. ↑ indicates that the larger the value, the better; ↓ indicates that the smaller the value, the better. *N* is the number of tested complexes, some methods have been tested only on a subset of the benchmark. -: not applicable

**Table 8.** Metrics computed on the CASF 2016

| Method | Trained on PDBbind | *N* | $R_p$↑ | SD↓ | MAE↓ | RMSE↓ |
|---|---|---|---|---|---|---|
| GoldScore$^{a}$ [13] | - | 285 | 0.42 | 1.99 | - | - |
| ChemScore$^{b}$ [13] | - | 285 | 0.59 | 1.76 | - | - |
| X-Score$^{HM}$ [13] | - | 285 | 0.61 | 1.73 | - | - |
| Pafnucy [4] | 2016 | 290 | 0.78 | 1.37 | 1.13 | 1.42 |
| Pafnucy* [4] | 2020 | 285 | 0.76 | - | - | 1.46 |
| OnionNet [7] | 2016 | 290 | 0.82 | 1.26 | 0.98 | 1.28 |
| AGL-Score$^{c}$ [8] | 2007 (refined) | 285 | 0.83 | - | - | 1.73 |
| IGN [10] | 2016 | 262 | 0.84 | - | 0.94 | 1.22 |
| ECIF6::LD-GBT$^{c}$ [11] | 2016 (refined) | 285 | 0.87 | - | - | - |
| HGScore | - | 285 | 0.85 | 1.14 | 0.90 | 1.14 |

The scoring function was used with the software [a] Gold and [b] Sybyl. [c] a problem for comparison, either the dataset is not fully tested, or the test set is also used for training or validation. ↑ indicates that the larger the value, the better; ↓ indicates that the smaller the value, the better. *N* is the number of tested complexes, some methods have been tested only on a subset of the benchmark, methods with *N* of 290 have been tested on a former CASF 2016 version. -: not applicable; *: retrained on the same split as HGScore

HGScore is among the best results on the CASF datasets for the scoring power metrics (Table 8), and the ability of the method to split the learning following the bond type may be an important part of the improvement. However, compared to existing scoring methods, the proposed new score does not bring significant improvement in the case of the CASF 2013 dataset (Table 7); it may be because the model of the present study has been trained on PDBbind version 2020. Indeed, the CASF datasets are the most representative complexes of PDBbind respectively in versions 2013 and 2016. Although the IGN method has a $R_p$ of 0.83, the CASF 2013 test set is reduced to only 85 items which is far from the whole set, thus it is difficult to conclude about this $R_p$ value. Concerning the ECIF6::LD-GBT method, the CASF 2016 is used for both the validation and test sets, whereas for HGScore the test set is unknown when the model is tested on it, which is the canonical way of proceeding. On the other hand, some authors have trained their method on the PDBbind refined set, which is composed of the highest quality complexes, but that choice may imply a lower prediction quality on the poorest quality complexes, which are the most frequent ones. It also should be noted that some methods do not provide any other metric than $R_p$, involving a greater difficulty in comparison.

The IGN method is the one that most resembles HGScore, indeed, the HGScore heterogeneous graphs can be considered as the assembly of the three independent graphs of the IGN method.

The present model has been trained on the 2020 version of PDBbind. The 2016 version of CASF is closer in terms of composition, which explains why HGScore improves the performance on the latter but not on the 2013 version. Nevertheless, it should be noted that on the CASF 2013, HGScore outperforms the others since it leads to lower error metrics values: the MAE and the RMSE.

Since the training set impacts the results of the scoring, it is of utmost importance to compare the performance of the HGScore with the performance of the score of other methods trained, validated, and tested identically. That is why the Pafnucy method [4] was also trained from scratch on the same HGScore

dataset split. Tested on CASF 2016 (Table 8), the HGScore method still outperforms Pafnucy trained in a completely identical way (starred line in Table 8). Because of the difficulty to get the source codes and then to execute them easily, in a strictly identical way to HGScore, only the Pafnucy method has been retrained.

In conclusion, the new proposed DL method allows scoring protein-ligand complexes. Based on the Attentive FP method, the presented model brings the originality to split the learning process regarding the interaction class, ligand to ligand, protein to protein, ligand to protein, and protein to ligand. That general idea allows the model to learn more precisely the effect of the molecules themselves and the interaction between them. The method does not propose a new type of layer but rather a new way to represent ligand-protein complexes, using a heterogeneous graph. Trained on the PDBbind dataset, like almost all counterpart methods, HGScore reaches the top of the methods assessed on the CASF 2016 dataset. Moreover, the method largely outperforms the classical scoring methods presented in a previous article [1]. Although HGScore does not bring a huge improvement on the main test metric ($R_p$ on CASF 2016), it outperforms all methods blindly tested on CASF. Moreover, the quality already achieved by the metrics implies that their improvement cannot be very large anymore.

As mentioned previously the network is split according to the interaction category. This method will allow to easily use transfer learning and thus reuse the feature extractor and high-level abstract representation based on the type of interaction between the two molecular groups. Of course, it will be necessary to ensure that the roles of the receptor (here the protein) and the ligand are preserved. Therefore, the use of transfer learning to adapt this new model to cases of protein-glycosaminoglycan (GAG) or protein-proteoglycan docking is very promising.

The next objective will be to integrate that scoring function into a complete docking process and analyze a) its impact on docking results and understand the root cause for better performance over other docking programs such as AutoDock and b) its impact on computation time. For now, HGScore can be used as an external scoring function allowing to reassess the score of a set of docked complexes to refine existing ranking.

## Abbreviations

3D: three-dimensional

Aggr: aggregation

AGL-Score: algebraic graph learning score

AI: artificial intelligence

CASF: comparative assessment of scoring functions

DL: deep learning

GATConv: graph attentional convolution

GATv2Conv: graph attentional convolution version 2

GCNs: graph convolutional networks

GNNs: graph neural networks

GRUCell: gated recurrent unit cell

HGCN: heterogeneous graph convolutional network

HGScore: heterogeneous graph score

IGN: InteractionGraphNet

MAE: mean absolute error

ML: machine learning

MLP: multi-layer perceptron

PDB: Protein Data Bank

PDBbind: Protein Data Bank-bind

RMSD: root mean square deviation

RMSE: root mean square error

SD: standard deviation

## Declarations


### Acknowledgments

The authors thank the supercomputer center ROMEO for providing CPU-GPU times.


### Author contributions

KC: Conceptualization, Software, Writing—original draft, Writing—review & editing. AG: Software, Writing—review & editing, Supervision. XV, SB, and LAS: Writing—review & editing, Supervision. All authors read and approved the submitted version.

### Conflicts of interest

The authors declare that they have no conflicts of interest.

### Ethical approval

Not applicable.

### Consent to participate

Not applicable.

### Consent to publication

Not applicable.

### Availability of data and materials

The datasets analyzed for this study can be found in the PDBbind website http://pdbbind.org.cn/.


### Funding

This work was supported by the Association Nationale de la Recherche et de la Technologie (ANRT) through a CIFRE grant [2020/0691]. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.




## References


1. Crampon K, Giorkallos A, Deldossi M, Baud S, Steffenel LA. Machine-learning methods for ligand-protein molecular docking. Drug Discov Today. 2022;27:151–64.
2. Ballester PJ, Mitchell JBO. A machine learning approach to predicting protein-ligand binding affinity with applications to molecular docking. Bioinformatics. 2010;26:1169–75.
3. Wallach I, Dzamba M, Heifets A. AtomNet: a deep convolutional neural network for bioactivity prediction in structure-based drug discovery. arXiv:1510.02855 [Preprint]. 2015 [cited 2023 Jan 27]. Available from: http://arxiv.org/abs/1510.02855
4. Stepniewska-Dziubinska MM, Zielenkiewicz P, Siedlecki P. Development and evaluation of a deep learning model for protein-ligand binding affinity prediction. Bioinformatics. 2018;34:3666–74.
5. Zhang H, Liao L, Saravanan KM, Yin P, Wei Y. DeepBindRG: a deep learning based method for estimating effective protein-ligand affinity. PeerJ. 2019;7:e7362.

6. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Boult TE, Wong A, Ferryman J, Siva P, Christensen GE, Rudd EM, et al., editors. IEEE 2016: Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR); 2016 Jun 27–30; Las Vegas, NV, USA. IEEE; 2016. pp. 770–8.
7. Zheng L, Fan J, Mu Y. OnionNet: a multiple-layer intermolecular-contact-based convolutional neural network for protein-ligand binding affinity prediction. ACS Omega. 2019;4:15956–65.
8. Nguyen DD, Wei GW. AGL-Score: algebraic graph learning score for protein-ligand binding scoring, ranking, docking, and screening. J Chem Inf Model. 2019;59:3291–304.
9. Hastie T, Tibshirani R, Friedman J. The elements of statistical learning. 2nd ed. New York: Springer New York; 2009.
10. Jiang D, Hsieh CY, Wu Z, Kang Y, Wang J, Wang E, et al. InteractionGraphNet: a novel and efficient deep graph representation learning framework for accurate protein-ligand interaction predictions. J Med Chem. 2021;64:18209–32.
11. Sánchez-Cruz N, Medina-Franco JL, Mestres J, Barril X. Extended connectivity interaction features: improving binding affinity prediction through chemical description. Bioinformatics. 2021;37:1376–82.
12. Liu Z, Su M, Han L, Liu J, Yang Q, Li Y, et al. Forging the basis for developing protein-ligand interaction scoring functions. Acc Chem Res. 2017;50:302–9.
13. Su M, Yang Q, Du Y, Feng G, Liu Z, Li Y, et al. Comparative assessment of scoring functions: the CASF-2016 update. J Chem Inf Model. 2019;59:895–913.
14. Wójcikowski M, Zielenkiewicz P, Siedlecki P. Open Drug Discovery Toolkit (ODDT): a new open-source player in the drug discovery field. J Cheminform. 2015;7:26.
15. Wu Z, Pan S, Chen F, Long G, Zhang C, Yu PS. A comprehensive survey on graph neural networks. IEEE Trans Neural Netw Learning Syst. 2021;32:4–24.
16. Fey M, Lenssen JE. Fast graph representation learning with PyTorch Geometric. arXiv:1903.02428 [Preprint]. 2019 [cited 2023 Feb 8]. Available from: https://arxiv.org/abs/1903.02428
17. Xiong Z, Wang D, Liu X, Zhong F, Wan X, Li X, et al. Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism. J Med Chem. 2020;63:8749–60.
18. Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y. Graph attention networks. arXiv:1710.10903 [Preprint]. 2018 [cited 2023 Feb 8]. Available from: https://arxiv.org/abs/1710.10903
19. Brody S, Alon U, Yahav E. How attentive are graph attention networks? [Internet]. Credit to ICLR; [cited 2023 Feb 8]. Available from: https://iclr.cc/virtual/2022/poster/6366
20. Cho K, van Merrienboer B, Bahdanau D, Bengio Y. On the properties of neural machine translation: encoder-decoder approaches. arXiv:1409.1259 [Preprint]. 2014 [cited 2023 Jan 27]. Available from: http://arxiv.org/abs/1409.1259
21. Spearman C. The proof and measurement of association between two things. Int J Epidemiol. 2010;39:1137–50.
22. Knight WR. A computer method for calculating Kendall's tau with ungrouped data. J Am Stat Assoc. 1966;61:436–9.
23. Akiba T, Sano S, Yanase T, Ohta T, Koyama M. Optuna: a next-generation hyperparameter optimization framework. In: KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining; 2019 Aug 4–8; Anchorage, USA. New York: Association for Computing Machinery; 2019.
24. Li Y, Su M, Liu Z, Li J, Liu J, Han L, et al. Assessing protein-ligand interaction scoring functions with the CASF-2013 benchmark. Nat Protoc. 2018;13:666–80.