



Article

SolPredictor: Predicting Solubility with Residual Gated Graph Neural Network

Waqar Ahmad ^{1,†} , Hilal Tayara ^{2,†} , HyunJoo Shim ^{3,*} and Kil To Chong ^{1,4,*}

¹ Department of Electronics and Information Engineering, Jeonbuk National University, Jeonju 54896, Republic of Korea

² School of International Engineering and Science, Jeonbuk National University, Jeonju 54896, Republic of Korea

³ School of Pharmacy, Jeonbuk National University, Jeonju 54896, Republic of Korea

⁴ Advanced Electronics and Information Research Center, Jeonbuk National University, Jeonju 54896, Republic of Korea

* Correspondence: shimhj@jbnu.ac.kr (H.S.); kitchong@jbnu.ac.kr (K.T.C.)

† These authors contributed equally to this work.

Abstract: Computational methods play a pivotal role in the pursuit of efficient drug discovery, enabling the rapid assessment of compound properties before costly and time-consuming laboratory experiments. With the advent of technology and large data availability, machine and deep learning methods have proven efficient in predicting molecular solubility. High-precision *in silico* solubility prediction has revolutionized drug development by enhancing formulation design, guiding lead optimization, and predicting pharmacokinetic parameters. These benefits result in considerable cost and time savings, resulting in a more efficient and shortened drug development process. The proposed SolPredictor is designed with the aim of developing a computational model for solubility prediction. The model is based on residual graph neural network convolution (RGNN). The RGNNs were designed to capture long-range dependencies in graph-structured data. Residual connections enable information to be utilized over various layers, allowing the model to capture and preserve essential features and patterns scattered throughout the network. The two largest datasets available to date are compiled, and the model uses a simplified molecular-input line-entry system (SMILES) representation. SolPredictor uses the ten-fold split cross-validation Pearson correlation coefficient R^2 0.79 ± 0.02 and root mean square error (RMSE) 1.03 ± 0.04 . The proposed model was evaluated using five independent datasets. Error analysis, hyperparameter optimization analysis, and model explainability were used to determine the molecular features that were most valuable for prediction.

Keywords: molecular solubility; drug discovery; ADMET; artificial intelligence; regression; graph neural network; residual gated graph neural network; simplified molecular-input line-entry system



Citation: Ahmad, W.; Tayara, H.; Shim, H.; Chong, K.T. SolPredictor: Predicting Solubility with Residual Gated Graph Neural Network. *Int. J. Mol. Sci.* **2024**, *25*, 715. <https://doi.org/10.3390/ijms25020715>

Academic Editor: Bruno Rizzuti

Received: 6 December 2023

Revised: 26 December 2023

Accepted: 4 January 2024

Published: 5 January 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Drug discovery is the process of identifying and developing new drugs for the treatment and prevention of diseases. This process has a significant impact on human health and society. Drug discovery drives innovation, meets medical needs, positively affects the economy, and improves public health.

Solubility is an important parameter in drug discovery for several reasons. Solubility affects the bioavailability, synthesis, and manufacture of drugs as well as the different drug design stages [1]. Chemists seek to improve the solubility of molecules through molecular structure optimization during drug design [2]. When a drug-like compound has adequate ADMET (absorption, distribution, metabolism, excretion, and toxicity) properties, the compound can be developed as a new medication [3]. Solubility is a major factor influencing drug absorption [4]. The solubility of a drug in intestinal fluid [5] is a prerequisite for achieving both concentration in the blood and therapeutic effects. Understanding the solubility of a compound enables chemical teams to make appropriate decisions during the

early stages of drug development. Poorly soluble compounds may be discarded in the early stages of drug development in order to save time and cost. Solubility is a very important physicochemical parameter and driver in drug bioavailability, and is considered during the early-stage drug screening process [6]. Although traditional methods of determining solubility are time consuming and costly [7], with recent technological advancements it is now possible to develop computational methods to minimize the time and cost of traditional methods, and in certain cases replace experimental work. Computational methods are data-driven, and are trained using physicochemical properties and molecular representations. These computational methods are trained for physicochemical properties and molecule representation. Over the past two decades, owing to technological advancements in every field of life, a large amount of data has been produced. Traditional analytical techniques are inadequate for processing this large amount of data [8]. Several machine learning (ML) approaches have been used to predict the molecular properties of materials, including their toxicity, solubility, lipophilicity, bandgaps, and conductivity, as well as protein structures [9–13]. ML models use different molecular representations and designs, including fingerprint representation with artificial neural network (ANN), a combination of multiple linear regression (MLP), ANN, and complex deep networks [14–17], and molecular descriptors of physicochemical properties, fingerprints, and topological indices. SMILES representations use recurrent neural networks (RNN), long short-memory models (LSTM), and the gated current unit (GRU) [18–20].

Graph-based representations explicitly represent a molecule's structural characteristics such as complex ring structures, branching, and other spatial connections between atoms. A neural network that is designed specifically to handle graph-structured data, with atoms representing nodes and edges expressing connections among them, is called a graph neural network (GNN). GNNs can learn spatial patterns and relationships among atoms. Different GNN architectures have been studied and utilized for solubility prediction. Graphically represented models use graph convolutional networks (GCNs), graph attention networks (GATs), and graph transformers [21–23]. Considerable research effort has been expended in developing computational models for solubility prediction [10,24–26]. Early solubility computational models utilized molecular, electronic, and structural properties as inputs [27,28], represented molecules as 2D/3D images [29], or used the simplified molecular-input line-entry system (SMILES), SELF-referencing embedded strings (SELFIES), or SMILES arbitrary target specifications (SMARTS) [30–32]. For solubility determination, Gao et al. [33] proposed a deep neural network for molecular property prediction using m -ordered descriptors and a graph convolution network using graphs. The effectiveness of both deep neural networks (DNNs) and GCNs for solubility prediction was then compared. Cui et al. [34] applied a deeper-net model to the 9943 training set. Bae et al. [35] developed support vector machine (SVM), gradient boosting tree (GBT), k -nearest neighbors (kNN), multilayer perceptron (MLP), and random forest (RF) models based on five fingerprint variants (RDKit, Morgan, Layered, Pattern, and MACCS). Maziarka et al. [36] developed a molecular attention transformer (MAT) architecture based on the present state-of-the-art transformer architecture for natural language processing services. MAT applies self-attention to the molecular graph representation of a molecule, while a feature vector identifies each node. This feature vector is then paired with a matrix called the adjacency matrix, which represents the molecule's relationship, and a distance matrix that represents the distance between each atom in the resulting 3D conformer of the molecule. SolTranNet [37] is a machine learning network based on the MAT architecture that estimates the solubility of a material in aqueous systems and is trained using AqSolDB. The RMSE and R^2 of SolTranNet were 1.459 and 0.6764, respectively, using cross-validation on the AqSolDB dataset.

We combined the Cui et al. [34] and AqSolDB [38] datasets and propose a residual-gated graph neural network (RGNN). The RGNN obtains both temporal and structural dependencies. Five independent datasets (Cui 2020 [34], Boobier 2017 [39], Lovric 2020 [40], Llinas 2020 set1, and set2 [41]) were evaluated, with R^2 values of 0.547, 0.814, 0.805, 0.373,

and 0.677 and RMSE values of 0.597, 0.743, 0.783, 0.991, and 1.142, respectively. Overall, molecular solubility is a critical parameter in drug discovery that influences bioavailability, formulation development, synthesis, manufacture, structure–activity relationship (SAR) studies, ADMET profiling, and lead optimization. By addressing solubility challenges and optimizing the solubility characteristics of drug candidates, researchers can increase the likelihood of successful drug development and improve the therapeutic potential of new medications.

2. Results and Discussion

SolPredictor was evaluated based on two metrics: R^2 and RMSE. The correlational measures allow us to determine how effectively the models rank the compounds in terms of solubility, whereas the error metric allows us to assess the average degree of error in the model predictions. Table 1 shows the ten-fold RMSE and R^2 . The RMSE values ranged from 0.93 to 1.09, with an average of 1.03 ± 0.04 . Fold 2 had the minimum RMSE, whereas fold 10 had the maximum RMSE value. The R^2 values ranged from 0.73 to 0.81, with an average R^2 0.79 ± 0.02 . Fold 5 had the maximum R^2 value, whereas fold 1 had the minimum value of 0.73.

Table 1. SolPredictor ten-fold cross-validation performance; fold 2 has the lowest RMSE.

Fold	1	2	3	4	5	6	7	8	9	10
RMSE	1.00	0.93	1.02	1.03	1.04	1.08	1.02	1.01	1.06	1.09
R^2	0.73	0.79	0.80	0.74	0.81	0.78	0.77	0.76	0.80	0.79

The scatter plot in Figure 1 shows the relationship between the actual labels and model prediction values. The R^2 values are shown for each fold. There is a strong pattern in which the predicted values increase as the actual data values increase, depicting a strong relationship. Furthermore, the line with a positive slope from the bottom left to the top right of the plot's points shows a positive correlation. The datapoints are tightly concentrated around the line of best fit, demonstrating the strength of the relationship. When two variables on a scatter plot have a significant positive correlation, it means that the relationship between them is consistent and tends to increase as one variable rises.

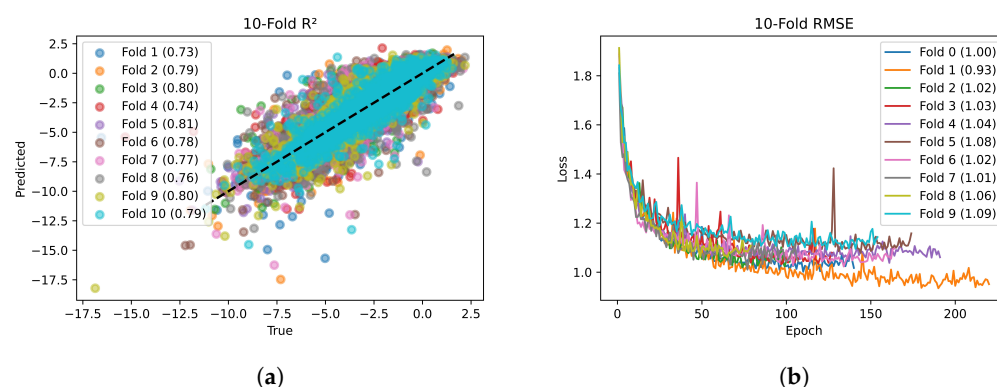


Figure 1. (a) R^2 scatter plot and (b) validation loss from ten-fold cross-validation.

2.1. Evaluation of Independent Datasets

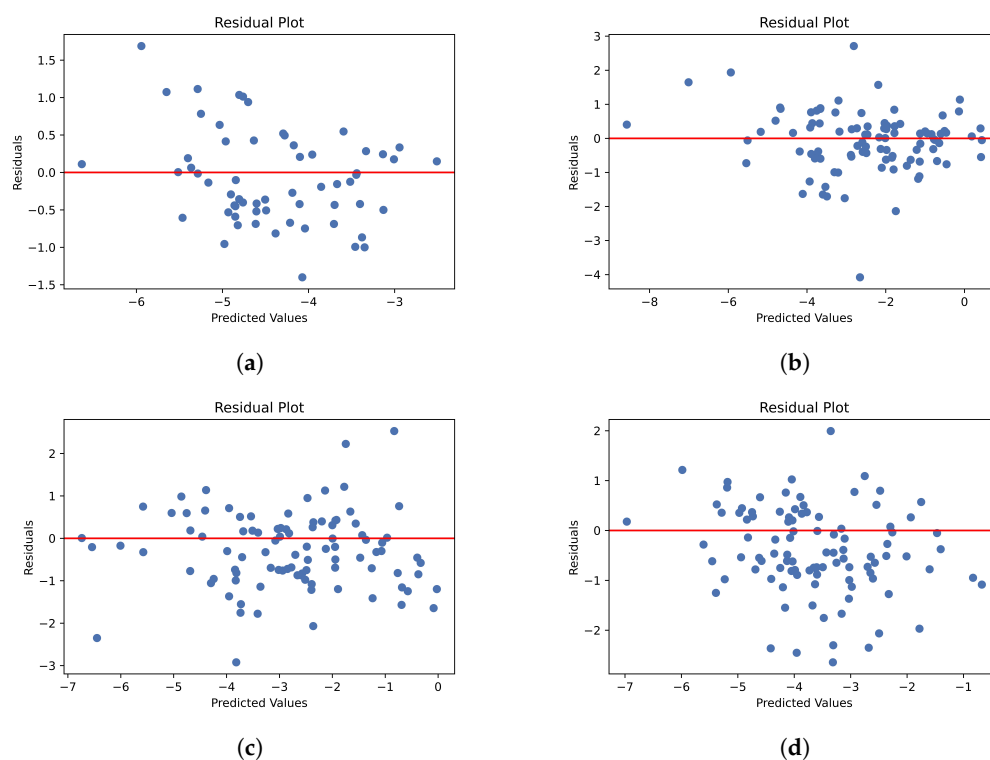
The results for the independent dataset are summarized in Table 2. The R^2 and RMSE results for SolPredictor were compared with SolTranNet. The number of molecules used in the present model prediction may differ from those used by other authors owing to cleaning and duplication removal. Each row in Table 2 shows the best R^2 and RMSE values in bold.

Table 2. Comparison of R^2 and RMSE across independent datasets.

Datasets	SolTranNet		SolPredictor	
	R^2	RMSE	R^2	RMSE
Cui et al. [34]	0.611	0.624	0.547	0.597
Boobier et al. [39]	0.724	1.010	0.814	0.743
Lovric et al. [40]	0.783	0.720	0.805	0.783
Llinas et al. [41] set1	0.527	0.952	0.373	0.991
Llinas et al. [41] set2	0.824	1.243	0.677	1.142

2.2. Error Analysis

Scatter charts were constructed for the test set residuals and predicted values. Residuals are the differences between the actual values (labels) and model predictions. Residual plots are used to visually observe the regression model errors. The plots show the model predictions as independent variables and the residuals as the dependent variable. The residual values are randomly distributed around the horizontal zero line. If the residuals have a U-shaped pattern, then the model does not capture nonlinear relationships between dependent and independent variables, whereas a funnel-shaped pattern shows that either the use of another model or data transformation is required. Figure 2 shows that the values are distributed randomly around the horizontal line, meaning that the model displayed a random pattern with no evident systematic structure.

**Figure 2.** Residual plots for test sets: (a) Cui et al.; (b) Lovric; (c) Boobier; (d) Llinas.

2.3. Feature Importance

GNNs have become popular for processing graph-structured data. Because of the complexity of graph-based data and the nonlinear interactions between graph nodes, it might be difficult to explain why a GNN generates a specific prediction. Hence, there are explanatory problems for GNNs, and it is difficult to explain how these models make predictions. GraphFramX [42] provides explainability techniques. Traditionally there are two ways to explain GNNs, local explanation and second global explanation. Local explanation emphasizes a single node or edge, whereas the global approach uses the overall GNN behavior.

Herein, local phenomena were used for SolPredictor explanation. We used the Pyg captum explainability module to explain the feature importance. The model configurations were regression, graph, and raw for the mode, task level, and return type, respectively. SolPredictor uses nine types of node features. As shown in the Figure 3, the explainability mechanism shows that the atomic number is the most important node property.

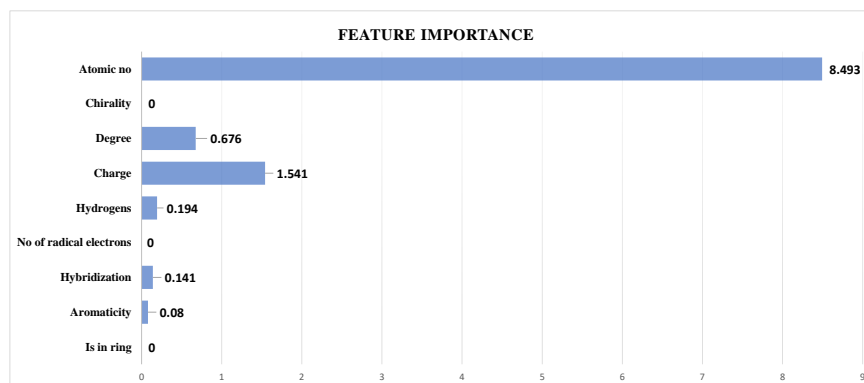


Figure 3. Feature importance chart, showing that atomic number is the most important node feature.

2.4. Hyperparameter Tuning

The best model architecture for a specific model is not always clear. Therefore, we investigated several choices. In a typical machine learning method, the machine is tasked to help with this exploration and to dynamically select the optimal model architecture. The hyperparameters determine the model architecture, and the process of obtaining the best model architecture is known as hyperparameter tuning. The hyperparameters were tuned for the SolPredictor using Optuna [43]. Optuna is a software (v.3.5.0) framework that facilitates automatic hyperparameter adaptability. The hyperparameters for SolPredictor are the learning rate, weight decay, optimizer, hidden features, number of layers, number of time steps, and dropout. The hyperparameter ranges are presented in the Supplementary Materials. As shown in Figure 4, the learning rate is the most important hyperparameter, whereas the number of layers has the least effect on the model.

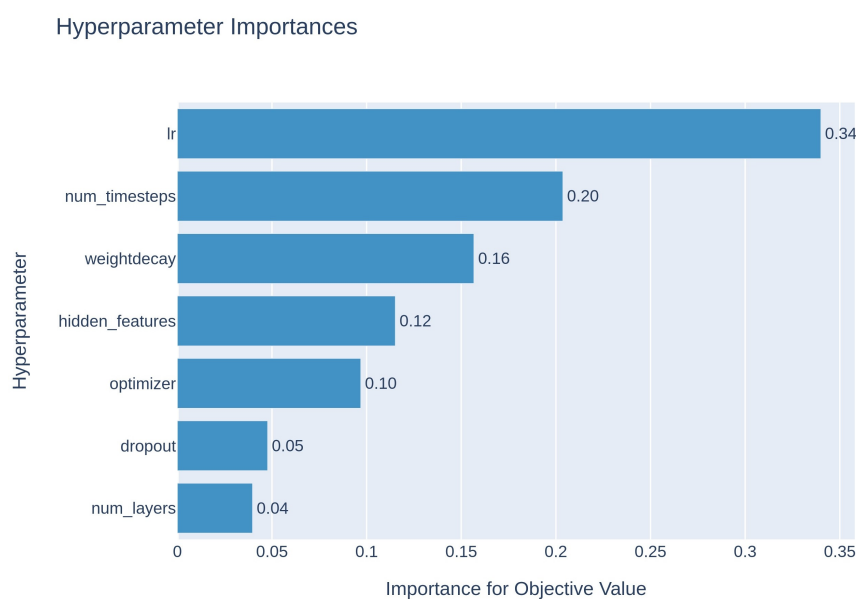


Figure 4. Importance of hyperparameters, showing that the learning rate is most important.

2.5. Web Server for Solubility

Finally, a web server was developed for researchers and pharmaceutical industry experts. The server was based on a residual graph convolution model. The page takes

SMILES list input, then calculates and shows predictions. The web server is demonstrated in Figure 5. The web server tool is available at <https://nscbio.jbnu.ac.kr/tools/SolPredictor/> (accessed on: 2 January 2024).

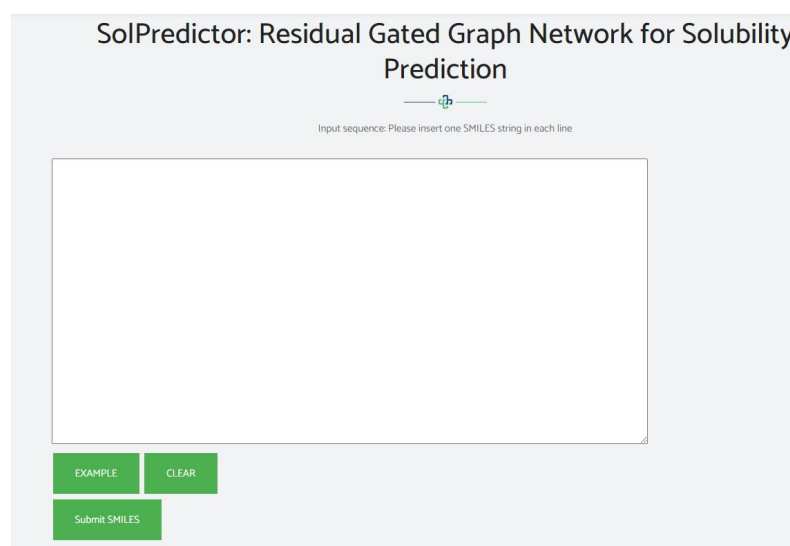


Figure 5. Web server for SolPredictor.

3. Materials and Methods

3.1. Dataset

We learned that neural network models perform well with more datapoints; thus, we utilized the datasets of Cui et al. [34], AqSolDB [38], and Lovric [40] for SolPredictor training. The datasets contained 9943, 9982, and 583 datapoints, respectively. All datasets were concatenated and duplicate SMILES were removed. The final combined dataset contained 17,826 datapoints. The dataset was divided into 90% for training and 10% for validation. The validation dataset was used in the SolPredictor hyperparameter optimization process. Each record had a SMILES string and logS value. logS refers to the logarithm of the water solubility of a compound and is measured in mol/L; a higher logS indicates higher solubility and vice versa. The logS values ranged from -17.46 to 2.13 . The length max, min, mean, std, and var for SMILES were 783, 2, 34, 25, and 662, respectively. The model was tested on five independent datasets from: Cui et al. [34], Boobier et al. [39], Lovric et al. [40], and Llinas et al. [41] set1 and set2, which contained 62, 95, 95, 99, and 32 datapoints, respectively. Each test dataset was checked against a training dataset for SMILES duplicates, and datapoints were deleted from the test dataset if any duplication was found. As shown in Figure 6, kernel density estimation plots represent the training, Cui2020 [34], Boobier2017 [39], Lovric2020 [40], and Llinas2020 set1 and set2 [41] test datasets. The x -axis represents the logS range, whereas the y -axis represents the density. The higher the y -axis value at a particular point on the x -axis, the higher the probability of that point. Figure 7 shows boxplots for the training and test datasets. Several of the logS values from the datasets of Cui et al. [34] and Llinas et al. [41] set2 datasets were non-overlapping with the training dataset. The Llinas et al. [41] set2 boxplot median line was lower in the interquartile range, which shows that the logS values are positively skewed.

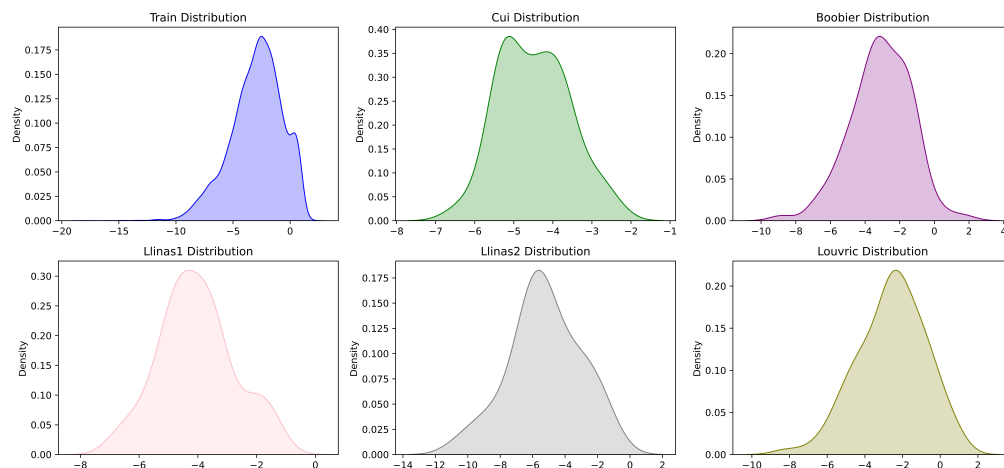


Figure 6. Solubility histograms: training dataset values ranged from -18.21 to 1.7 and test dataset values ranged from -6.52 to -2.36 .

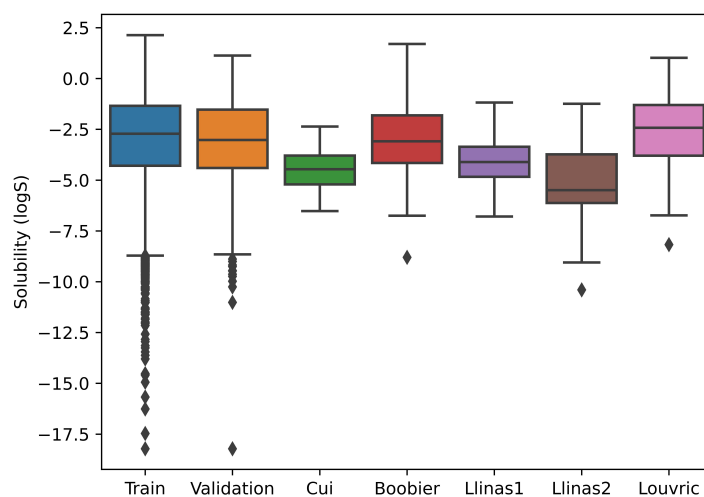


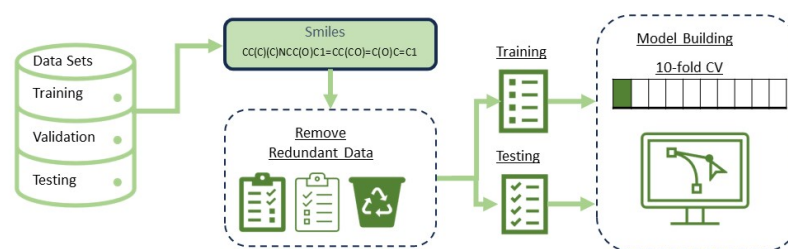
Figure 7. Boxplots for training, validation, and five test datasets. The training dataset had a median at -2.8 , maximum at 2.3 , and minimum at -8 , with outliers at -9 to -17.5 .

3.1.1. Data Preprocessing

In this step, the datasets of Cui et al., AqsolDB, and Lovric were combined and SMILES duplication was removed using Pandas [44]. SMILES entries with a length of 1 were removed as well, as a single-node graph was not valid. In implementing the model, the SMILES were input features and the logS values were labels. The model used a graph as the input. A graph is a data structure created using two-component nodes and edges. In a molecular context, nodes are atoms, whereas edges are bonds. The SMILES data were converted into molecules using the rdkit library [45] and checked for validity, then a graph object was generated using the node features in Table 3 and saved on disk. The SMILES data were converted to a graph using Pytorch Geometric (PyG) [46]. The SMILES data were processed one by one, as shown in Figure 8. An unambiguous graphical representation was obtained by assigning a number to each atom in the molecule and then traversing the molecular graph in that order.

Table 3. Nine features of atoms.

Atom Features	Description
Atom number	1 to 119
Chirality	Atom chirality
Degree	0 to 11
Charge	−5 to 7
Hydrogens	Connected hydrogens
No of radical electrons	0 to 5
Hybridization	s, sp ² , sp ^{3d2} , sp, sp ³ , sp ^{3d} , other
Aromaticity	False or True
Is in ring	False or True

**Figure 8.** Data preprocessing flow chart showing molecule and graph generation from SMILES data.

3.1.2. Ten-Fold Data Split

Cross-validation is a resampling method, commonly known as out-of-sample estimation or rotation estimation, which is used to evaluate how statistically-determined results generalize to independent datasets by testing machine learning models on a limited set of data. The algorithm has a single parameter k that determines the number of groups into which a given data sample must be divided. Therefore, this technique is commonly referred to as k -fold cross-validation. The entire training dataset was randomly divided into ten folds of equal size and independence, with no rows repeated in any other fold. The model was trained with nine folds and then tested with the remaining fold to obtain a performance score. This process was repeated ten times to create ten models and scores for each model.

3.2. Methods

Deep learning approaches have been shown to be effective in predicting the molecular properties of compounds [47], and are becoming a growing instrument in modern computer-aided drug design toolboxes. Because molecules can be represented as graphs, one apparent strategy for deep learning is to utilize a graph-based architecture, which enables the use of graph-based neural networks (GNNs).

3.2.1. Molecular Feature Extraction

The extraction of molecular features is a vital step in the preparation of molecular data for graph neural network (GNN) models. Molecular atoms are the basic building blocks of SolPredictor. Nine types of atomic features were extracted; these are listed in Table 3. The feature vector encapsulates the features of each atom in a molecule. The atomic number, degree, formal charge, number of radical electrons, hybridization, aromaticity, hydrogen atoms, chirality, and 'is in ring' property are all common atomic-level characteristics. The ranges for the atomic number, degree, and charge are 1–119, 0–11, and −5 to 7, respectively. The aromaticity and 'is in ring' property have true or false values. The SMILES string was converted to the rdkit [45] molecule and the validity of the molecule was checked. The graph was created using the Pytorch geometric library. The atomic features were treated as node features in the graph; for example, if a molecule has 32 atoms, then the graph size is 32×9 .

3.2.2. Graph Neural Network (GNN)

A graph neural network (GNN) is a type of neural network specifically designed to operate on graph-structured data. GNNs obtain local and global interactions between atoms and their neighbors. The graphs consist of vertices (or nodes) and links (or edges) that represent the relationships between the vertices. GNNs are capable of learning and extracting meaningful representations from the nodes and edges of a graph, enabling them to perform various tasks such as graph regression/classification, node classification, link prediction, and graph generation.

$$G = (V, E) \quad (1)$$

The graph consists of a set of nodes V and some edges E that connect them. Each node V has a predefined vector associated with it that encodes its state, that is, the node's hidden state. Hidden state vectors are initialized with node-related features from the input graph at the start of GNN execution. Edges may include a set of features if desired. Following initialization of the hidden states, a message-passing algorithm is executed based on the connections of the input graph. The message-passing process comprises three major phases: (i) message, (ii) aggregation, and (iii) update. During the GNN message-passing phase, each vertex is updated according to the aggregation of the neighboring vertices. The embedding $h_u^{(k)}$ associated with vertex $u \in V$.

$$h_u^{(l+1)} = \text{update}^{(l)}(h_u^l, \text{AGGREGATE}^{(l)}(\{h_v^l, \forall v \in N_u\})) \quad (2)$$

$$= \text{update}^{(l)}(h_u^l, m_{N(u)}^{(l)}) \quad (3)$$

AGGREGATE and update are learnable functions and message m node u neighbors aggregation.

3.2.3. Residual Gated Graph Convolution

An RNN is a neural network specifically designed for sequential or time-series data. Each datapoint input to the RNN depends on the previous data input. RNNs maintain hidden feature vectors or contexts from previous inputs. At each time step, the RNN takes the input vector and hidden feature vector and produces the output along with the updated hidden vector:

$$h_i = f_{RNN}(w_i, \{h_j : j = i - 1\}) \quad (4)$$

where the vector h_i for word i is computed from the previous feature vector h_j and word w_i in the sequence. Graph RNN, introduced in 2005 [48], proposed a multilayer perceptron with an RNN on a graph. In this case, the feature vector h_i of vertex i is

$$h_i = f_{GRNN}(w_i, \{h_j : j \rightarrow i\}) = \sum_{j \rightarrow i} C_{GRNN}(w_i, h_j), \quad (5)$$

where w_i is a feature vector for the current vertex and h_j is the feature vector set of the current vector neighbor's feature vector vertices. Here, $C_{GRNN}(w_i, h_j)$ is defined as

$$C_{GRNN}(w_i, h_j) = P\sigma(Q\sigma(Rx_i + Sh_j)), \quad (6)$$

where P , Q , R , and S are weighted parameters and σ is a sigmoid function. In 2016, Sukhbaatar et al. [49] proposed the ConvNet graph, represented as follows:

$$h_i^{l+1} = f_{GCNN}(w_i^l, \{h_j^l : j = i \rightarrow 1\}) = \text{ReLU}(Q^l h_i^l + R^l \sum_{j \rightarrow i} h_j^l), \quad (7)$$

where ReLU is the rectified linear unit and l represents the layer level. In 2017, Marcheggiani et al. [50] proposed a function for edge gate importance:

$$h_i^{l+1} = f_{GCNN}(w_i^l, \{h_j^l : j = i \rightarrow 1\}) = \text{ReLU}(\sum_{j \rightarrow i} \eta_{ij} h_j^l), \quad (8)$$

where η_{ij} works as an edge gate and is represented as

$$\eta_{ij} = \sigma(Ch_i^l + Dh_j^l). \quad (9)$$

The gated graph convolution is

$$h_i^{l+1} = f_{GCGNN}(w_i^l, \{h_j^l : j = i \rightarrow 1\}) = \text{ReLU}(R^l h_i^l + \sum_{j \rightarrow i} \eta_{ij} h_j^l). \quad (10)$$

This equation has a source vertex, neighboring vertex information, and edge gate capability. Subsequently Bresson and Laurent [51] formulated a residual gated graph convolution by adding an identity operator between the layers:

$$h_i^{l+1} = f^l(h_i^l, \{h_j^l : j \rightarrow i\}) + h_i^l \quad (11)$$

where h_i^{l+1} represents the node features for the $l + 1$ layer, h_i^l represents the node features for the current layer l , and h_j^l represents the neighboring nodes. Multiple layers of the RGNN can be stacked iteratively using Equation (11) to capture and transfer information across the graph structure.

3.2.4. Implementation Details

A model was developed for solubility prediction using a residual gated graph neural network. As shown in Figure 9, SMILES data were converted to graphs using the PyG library. The model uses nine node features as inputs. The features were normalized and passed to the linear layer, and the RGNN convolution was applied. After global pooling, embedding was employed as a sequence and the GRU was used to capture long-term dependencies. Finally, a dense layer was used to calculate the solubility.

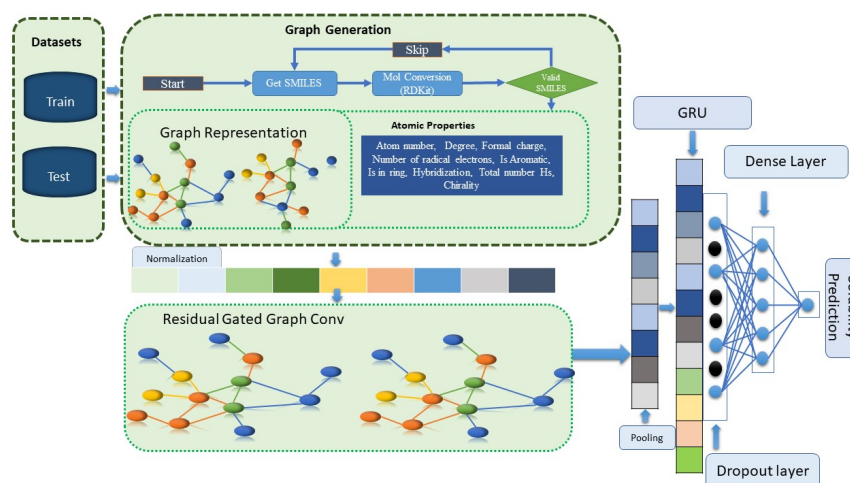


Figure 9. Flow diagram showing conversion of SMILES to graph and application of RGNN layers.

3.2.5. Evaluation Metrics

RMSE and R^2 were used as metrics to evaluate the performance of the SolPredictor regression model. RMSE is the square root of the average difference between the expected and actual values of a target variable. It is a statistical measure of the extent to which the predicted values of a model diverge from the actual values. The RMSE is always positive; it measures how close the predicted and true values are, with lower values indicating better model performance. The RMSE is calculated as follows:

$$RMSE = \sqrt{\frac{\sum_{i=0}^n (y_i - \hat{y}_i)^2}{n}} \quad (12)$$

The Pearson correlation coefficient is frequently used in statistics, data analysis, and machine learning to assess the linear relationships between variables. R^2 indicates the correlation between two variables, namely, the model predictions and actual label values. R^2 ranges from -1 to $+1$, and indicates the direction and strength of the variables. The direction of this relationship is represented by a coefficient; a positive sign implies a positive connection, whereas a negative sign indicates a negative correlation.

$$R^2 = 1 - \frac{\sum_{i=0}^n (y_i - \hat{y}_i)^2}{\sum_{i=0}^n (y_i - \bar{y})^2} \quad (13)$$

4. Conclusions

Aqueous solubility is crucial in physical chemistry. The in silico prediction of solubility with a high degree of reliability may significantly reduce the cost and time required for developing medications. In this paper, the SolPredictor model is proposed based on a RGNN. During the graph convolution process, the gating mechanism of the RGNN allows the model to emphasize local and global information selectively. This gating technique regulates the flow of data, allowing the model to focus on relevant network neighbors or to collect data from distant nodes. This feature enables SolPredictor to capture both local and global dependencies in a graph, which improves the ability of the model to grasp and describe complicated graph patterns and build deeper network robustness for noisy or incomplete data. Other tasks that could benefit from the improved molecular representation of AI models include yield prediction, synthesis planning, toxicity prediction, reaction outcome prediction, bioactivity prediction, and retrosynthesis analysis. SolPredictor demonstrated a ten-fold split cross-validation coefficient of R^2 0.79 ± 0.02 and root mean square error (RMSE) of 1.03 ± 0.04 . The model was tested on five independent datasets: Cui et al. [34], Boobier et al. [39], Lovric et al. [40], and set 1 and set2 from Llinas et al. [41]. Additional chemical representation features (e.g., bond features) or additional experimental solubility data may improve the solubility prediction ability of this model. The method shown here can be extended to the development of other high performance deep learning models for physical chemistry problems, such as toxicity and lipophilicity prediction.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/ijms25020715/s1>.

Author Contributions: The manuscript was written through contributions of all authors. W.A., H.T. and H.S. prepared the datasets and performed data analysis, algorithm design, and SolPredictor web server implementation, and wrote the manuscript with help from K.T.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. 2020R1A2C2005612), the Technology Innovation Program (20012892) funded by the Ministry of Trade, Industry and Energy (MOTIE, Korea) and the Korea Big Data Station (K-BDS) with computing resources including technical support.

Data Availability Statement: Code and datasets are available at SolPredictor (<https://github.com/waqarahmadm019/SolPredictor>, accessed on 2 January 2024).

Conflicts of Interest: The authors declare that they do not have any competing interests.

References

1. Lipinski, C.A.; Lombardo, F.; Dominy, B.W.; Feeney, P.J. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Adv. Drug Deliv. Rev.* **2012**, *64*, 4–17. [CrossRef]
2. Lipinski, C.A.L.F. Poor aqueous solubility—An industry-wide problem in drug discovery. *Am. Pharm. Rev.* **2002**, *5*, 82–85.
3. Di, L.; Kerns, E.H. *Drug-like Properties: Concepts, Structure Design and Methods from ADME to Toxicity Optimization*; Academic Press: Cambridge, MA, USA, 2015.
4. Kostewicz, E.S.; Brauns, U.; Becker, R.; Dressman, J.B. Forecasting the oral absorption behavior of poorly soluble weak bases using solubility and dissolution studies in biorelevant media. *Pharm. Res.* **2002**, *19*, 345. [CrossRef] [PubMed]
5. McPherson, S.; Perrier, J.; Dunn, C.; Khadra, I.; Davidson, S.; Ainousah, B.; Wilson, C.G.; Halbert, G. Small scale design of experiment investigation of equilibrium solubility in simulated fasted and fed intestinal fluid. *Eur. J. Pharm. Biopharm.* **2020**, *150*, 14–23. [CrossRef] [PubMed]
6. Chaudhary, A.; Nagaich, U.; Gulati, N.; Sharma, V.; Khosa, R.; Partapur, M. Enhancement of solubilization and bioavailability of poorly soluble drugs by physical and chemical modifications: A recent review. *J. Adv. Pharm. Educ. Res.* **2012**, *2*, 32–67.
7. Tu, M.; Cheng, S.; Lu, W.; Du, M. Advancement and prospects of bioinformatics analysis for studying bioactive peptides from food-derived protein: Sequence, structure, and functions. *TrAC Trends Anal. Chem.* **2018**, *105*, 7–17. [CrossRef]

8. Jan, B.; Farman, H.; Khan, M.; Imran, M.; Islam, I.U.; Ahmad, A.; Ali, S.; Jeon, G. Deep learning in big data analytics: A comparative study. *Comput. Electr. Eng.* **2019**, *75*, 275–287. [[CrossRef](#)]
9. Tang, W.; Chen, J.; Wang, Z.; Xie, H.; Hong, H. Deep learning for predicting toxicity of chemicals: A mini review. *J. Environ. Sci. Health Part C* **2018**, *36*, 252–271. [[CrossRef](#)]
10. Wang, X.; Liu, M.; Zhang, L.; Wang, Y.; Li, Y.; Lu, T. Optimizing pharmacokinetic property prediction based on integrated datasets and a deep learning approach. *J. Chem. Inf. Model.* **2020**, *60*, 4603–4613. [[CrossRef](#)]
11. Khan, A.; Tayara, H.; Chong, K.T. Prediction of organic material band gaps using graph attention network. *Comput. Mater. Sci.* **2023**, *220*, 112063. [[CrossRef](#)]
12. Qin, G.; Wei, Y.; Yu, L.; Xu, J.; Ojih, J.; Rodriguez, A.D.; Wang, H.; Qin, Z.; Hu, M. Predicting lattice thermal conductivity from fundamental material properties using machine learning techniques. *J. Mater. Chem. A* **2023**, *11*, 5801–5810. [[CrossRef](#)]
13. Stahl, K.; Graziadei, A.; Dau, T.; Brock, O.; Rappsilber, J. Protein structure prediction with in-cell photo-crosslinking mass spectrometry and deep learning. *Nat. Biotechnol.* **2023**, *41*, 1810–1819. [[CrossRef](#)] [[PubMed](#)]
14. Boothroyd, S.; Kerridge, A.; Broo, A.; Buttar, D.; Anwar, J. Solubility prediction from first principles: A density of states approach. *Phys. Chem. Chem. Phys.* **2018**, *20*, 20981–20987. [[CrossRef](#)] [[PubMed](#)]
15. Livingstone, D.J.; Ford, M.G.; Huuskonen, J.J.; Salt, D.W. Simultaneous prediction of aqueous solubility and octanol/water partition coefficient based on descriptors derived from molecular structure. *J. Comput. Aided Mol. Des.* **2001**, *15*, 741–752. [[CrossRef](#)] [[PubMed](#)]
16. Ma, X.; Li, Z.; Achenie, L.E.; Xin, H. Machine-learning-augmented chemisorption model for CO₂ electroreduction catalyst screening. *J. Phys. Chem. Lett.* **2015**, *6*, 3528–3533. [[CrossRef](#)] [[PubMed](#)]
17. Korotcov, A.; Tkachenko, V.; Russo, D.P.; Ekins, S. Comparison of deep learning with multiple machine learning methods and metrics using diverse drug discovery data sets. *Mol. Pharm.* **2017**, *14*, 4462–4475. [[CrossRef](#)] [[PubMed](#)]
18. Olivecrona, M.; Blaschke, T.; Engkvist, O.; Chen, H. Molecular de-novo design through deep reinforcement learning. *J. Cheminform.* **2017**, *9*, 48. [[CrossRef](#)]
19. Hirohara, M.; Saito, Y.; Koda, Y.; Sato, K.; Sakakibara, Y. Convolutional neural network based on SMILES representation of compounds for detecting chemical motif. *BMC Bioinform.* **2018**, *19*, 83–94. [[CrossRef](#)]
20. Arús-Pous, J.; Johansson, S.V.; Prykhodko, O.; Bjerrum, E.J.; Tyrchan, C.; Raymond, J.L.; Chen, H.; Engkvist, O. Randomized SMILES strings improve the quality of molecular generative models. *J. Cheminform.* **2019**, *11*, 71. [[CrossRef](#)]
21. Chen, S.; Wulamu, A.; Zou, Q.; Zheng, H.; Wen, L.; Guo, X.; Chen, H.; Zhang, T.; Zhang, Y. MD-GNN: A mechanism-data-driven graph neural network for molecular properties prediction and new material discovery. *J. Mol. Graph. Model.* **2023**, *123*, 108506. [[CrossRef](#)]
22. Cremer, J.; Sandonas, L.M.; Tkatchenko, A.; Clevert, D.A.; De Fabritiis, G. *Equivariant Graph Neural Networks for Toxicity Prediction*; ACS Publications: Washington, DC, USA, 2023.
23. Yang, L.; Jin, C.; Yang, G.; Bing, Z.; Huang, L.; Niu, Y.; Yang, L. Transformer-based deep learning method for optimizing ADMET properties of lead compounds. *Phys. Chem. Chem. Phys.* **2023**, *25*, 2377–2385. [[CrossRef](#)] [[PubMed](#)]
24. Atz, K.; Grisoni, F.; Schneider, G. Geometric deep learning on molecular representations. *Nat. Mach. Intell.* **2021**, *3*, 1023–1032. [[CrossRef](#)]
25. Chuang, K.V.; Gunsalus, L.M.; Keiser, M.J. Learning molecular representations for medicinal chemistry: Miniperspective. *J. Med. Chem.* **2020**, *63*, 8705–8722. [[CrossRef](#)] [[PubMed](#)]
26. Ghasemi, F.; Mehridehnavi, A.; Pérez-Garrido, A.; Pérez-Sánchez, H. Neural network and deep-learning algorithms used in QSAR studies: Merits and drawbacks. *Drug Discov. Today* **2018**, *23*, 1784–1790. [[CrossRef](#)] [[PubMed](#)]
27. Padula, D.; Simpson, J.D.; Troisi, A. Combining electronic and structural features in machine learning models to predict organic solar cells properties. *Mater. Horizons* **2019**, *6*, 343–349. [[CrossRef](#)]
28. Kang, B.; Seok, C.; Lee, J. Prediction of molecular electronic transitions using random forests. *J. Chem. Inf. Model.* **2020**, *60*, 5984–5994. [[CrossRef](#)] [[PubMed](#)]
29. Fan, Z.; Ma, E. Predicting orientation-dependent plastic susceptibility from static structure in amorphous solids via deep learning. *Nat. Commun.* **2021**, *12*, 1506. [[CrossRef](#)]
30. Wu, C.K.; Zhang, X.C.; Yang, Z.J.; Lu, A.P.; Hou, T.J.; Cao, D.S. Learning to SMILES: BAN-based strategies to improve latent representation learning from molecules. *Briefings Bioinform.* **2021**, *22*, bbab327. [[CrossRef](#)]
31. Shen, C.; Krenn, M.; Eppel, S.; Aspuru-Guzik, A. Deep molecular dreaming: Inverse machine learning for de-novo molecular design and interpretability with surjective representations. *Mach. Learn. Sci. Technol.* **2021**, *2*, 03LT02. [[CrossRef](#)]
32. Capecchi, A.; Raymond, J.L. Classifying natural products from plants, fungi or bacteria using the COCONUT database and machine learning. *J. Cheminform.* **2021**, *13*, 82. [[CrossRef](#)]
33. Gao, P.; Liu, Z.; Tan, Y.; Zhang, J.; Xu, L.; Wang, Y.; Jeong, S.Y. Accurate predictions of drugs aqueous solubility via deep learning tools. *J. Mol. Struct.* **2022**, *1249*, 131562. [[CrossRef](#)]
34. Cui, Q.; Lu, S.; Ni, B.; Zeng, X.; Tan, Y.; Chen, Y.D.; Zhao, H. Improved prediction of aqueous solubility of novel compounds by going deeper with deep learning. *Front. Oncol.* **2020**, *10*, 121. [[CrossRef](#)] [[PubMed](#)]
35. Bae, S.Y.; Lee, J.; Jeong, J.; Lim, C.; Choi, J. Effective data-balancing methods for class-imbalanced genotoxicity datasets using machine learning algorithms and molecular fingerprints. *Comput. Toxicol.* **2021**, *20*, 100178. [[CrossRef](#)]
36. Maziarka, Ł.; Danel, T.; Mucha, S.; Rataj, K.; Tabor, J.; Jastrzębski, S. Molecule attention transformer. *arXiv* **2020**, arXiv:2002.08264.

37. Francoeur, P.G.; Koes, D.R. SolTranNet—A Machine Learning Tool for Fast Aqueous Solubility Prediction. *J. Chem. Inf. Model.* **2021**, *61*, 2530–2536. [[CrossRef](#)]
38. Sorkun, M.C.; Khetan, A.; Er, S. AqSolDB, a curated reference set of aqueous solubility and 2D descriptors for a diverse set of compounds. *Sci. Data* **2019**, *6*, 143. [[CrossRef](#)]
39. Boobier, S.; Osbourn, A.; Mitchell, J.B. Can human experts predict solubility better than computers? *J. Cheminform.* **2017**, *9*, 63. [[CrossRef](#)]
40. Lovrić, M.; Pavlović, K.; Žuvela, P.; Spataru, A.; Lučić, B.; Kern, R.; Wong, M.W. Machine learning in prediction of intrinsic aqueous solubility of drug-like compounds: Generalization, complexity, or predictive ability? *J. Chemom.* **2021**, *35*, e3349. [[CrossRef](#)]
41. Llinas, A.; Oprisiu, I.; Avdeef, A. Findings of the second challenge to predict aqueous solubility. *J. Chem. Inf. Model.* **2020**, *60*, 4791–4803. [[CrossRef](#)]
42. Amara, K.; Ying, R.; Zhang, Z.; Han, Z.; Shan, Y.; Brandes, U.; Schemm, S.; Zhang, C. Graphframex: Towards systematic evaluation of explainability methods for graph neural networks. *arXiv* **2022**, arXiv:2206.09677.
43. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A next-generation hyperparameter optimization framework. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 2623–2631.
44. McKinney, W. Data Structures for Statistical Computing in Python. In Proceedings of the 9th Python in Science Conference, Austin, TX, USA, 28 June–3 July 2010; pp. 56–61. [[CrossRef](#)]
45. Landrum, G. Rdkit documentation. *Release* **2013**, *1*, 4.
46. Fey, M.; Lenssen, J.E. Fast graph representation learning with PyTorch Geometric. *arXiv* **2019**, arXiv:1903.02428.
47. Walters, W.P.; Barzilay, R. Applications of deep learning in molecule generation and molecular property prediction. *Acc. Chem. Res.* **2020**, *54*, 263–270. [[CrossRef](#)] [[PubMed](#)]
48. Gori, M.; Monfardini, G.; Scarselli, F. A new model for learning in graph domains. In Proceedings of the 2005 IEEE International Joint Conference on Neural Networks, Montreal, QC, Canada, 31 July–4 August 2005; Volume 2, pp. 729–734.
49. Sukhbaatar, S.; Fergus, R. Learning multiagent communication with backpropagation. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; Volume 29.
50. Marcheggiani, D.; Titov, I. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv* **2017**, arXiv:1703.04826.
51. Bresson, X.; Laurent, T. Residual gated graph convnets. *arXiv* **2017**, arXiv:1711.07553.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.