Learning Relevant Molecular Representations via Self-Attentive Graph Neural Networks

Shoma Kikuchi
Hokkaido University
Sapporo, Japan
kicchi-s@ist.hokudai.ac.jp

Ichigaku Takigawa
RIKEN
Kyoto, Japan
ichigaku.takigawa@riken.jp

Satoshi Oyama
Hokkaido University
Sapporo, Japan
oyama@ist.hokudai.ac.jp

Masahito Kurihara
Hokkaido University
Sapporo, Japan
kurihara@ist.hokudai.ac.jp

***Abstract*—Molecular graphs are one of the established representations for small molecules, and even steric or electronic information can be encoded as node and edge features. Naturally, graph neural networks have been intensively investigated to solve various chemical problems at molecular levels. However, it remains unclear how to encode relevant chemical information into graphs. We investigate this problem by proposing three models of graph neural networks with self-attention mechanisms at different levels to adaptively select relevant chemical information for each input. Using neural graph fingerprint (NFP) as a baseline, we introduce three types of attention mechanisms on the top of NFPs. Our experimental evaluations suggest that introducing these self-attention mechanisms contributes to not only improving the prediction accuracy but also providing quantitative interpretation using obtained attention coefficients.**

***Index Terms*—Graph Neural Network, Self-Attention, Deep Learning, Machine Learning**

I. Introduction

Graphs are data structures used in various knowledge representations such as chemical structure data and XML clause data. They can also be used for supervised learning problems such as regression and classification for graph-structured data if associated target values or class labels are available. In chemical science, supervised machine learning is attracting increasing attention as a data-driven method for predicting physical properties or biological activities of small molecules. Since supervised learning is usually defined for fixed-dimensional multivariate vectors, its extension to graph data is far from trivial.

The use of neural networks for graph structures (graph neural networks, GNNs) has recently been attracting interest as a technique for learning feature representations of graphs. Various types of GNNs can be described in the Message Passing Neural Network (MPNN) framework [4]. MPNNs operate on graphs with nodes and edges, each of which can have a feature vector or state vector. They have two phases, a message passing phase and a readout phase. In the message passing phase, the node state vectors are updated by aggregating the state vectors of adjacent nodes and edges. In the readout phase, the feature vector of the graph itself is computed using all the node state vectors in the graph, and it is used as the fixed-dimensional representation of that graph that can be fed into standard machine learning pipelines.

Duvenaud et al. [2] proposed using neural graph fingerprints (NFPs) to extend convolutional neural networks to graphs and showed the effectiveness of this approach experimentally using several chemical datasets. They represented a molecule as a graph in which each node corresponds to an atom, and each edge to a bond. Chemical information of each atom or bond is represented as a multi-dimensional feature vector. These node and edge features are used as the initial state vectors of the corresponding node or edge in the graph. There are, however, various other options for how to encode chemical domain knowledge into input graphs for better prediction. The suitability of the node and edge features depends on individual tasks, datasets, and molecules. It is thus difficult in general to choose any relevant representation or chemical information beforehand.

Veličković et al. [7] proposed using graph attention networks (GATs) that can more flexibly learn feature representations of graphs using attention mechanisms. GATs have neural network architectures and operate on graph-structured data. Each node in the graph has a network with self-attention layers. Different weights can be specified for nodes with different neighbors when graph convolution is performed, and a suitable weight can be computed for each input. Therefore, weights can be dynamically computed not only for nodes but also for other elements.

In this paper, we present GNN models based on NFPs that are equipped with self-attention mechanisms at different levels. They assume multiple input representations that correspond to different sets of node and edge features, i.e., different ways of encoding domain-specific information onto input graphs. Focusing on this point is crucial because most chemical problems currently targeted by machine learning cannot be characterized only by the topology of graphs. They always require other accompanying domain-specific information on what each node and edge represents, and these features, in some way, need to be encoded into input graphs appropriately. In our models, the self-attention layer adaptively selects a relevant representation at different levels for each graph from available multiple features. We show that the selected representation is suitable by comparing prediction accuracy with that of the plain NFP. For self-attention mechanisms introduced, we present three models to attend at different levels: model Ⅲ-B (Fig. 2), model Ⅲ-C (Fig. 2) and model Ⅲ-

978-1-7281-0858-2/19/$31.00

D (Fig. 4) that make it easy to understand what information the self-attention layer emphasizes. This method helps to assess the importance of each domain information quantitatively as the obtained attention coefficients. Using this method thus improves the interpretability of the prediction results. We demonstrate providing the attention coefficients would be informative by exploring the distributions of obtained attention coefficients for chemical datasets.

II. Related Work

We propose introducing the self-attention mechanism to neural network models for graph-structured data. Our purpose is to construct graph neural networks with attention mechanisms on the top of several existing methods.

A. Neural Graph Fingerprints (NFP)

Our proposed models are based on the method by Duvenaud et al. [2] for learning molecular fingerprints from graph data. They introduced a convolutional neural network that directly manipulates graphs and can take a graph of any size and shape as input. Their proposed architecture generalizes a standard molecular feature extraction method based on circular fingerprints [6] and showed excellent prediction performance on various tasks. A molecule can be regarded as a graph structure if atoms are regarded as nodes and bonds are regarded as edges. Each node or edge is associated with a fixed-dimensional feature vector representing the characteristics of the corresponding atom or bond. These features are used as the initial state vectors of the nodes and edges for computing NFPs. A node is convolved with a vector of adjacent nodes and edges by a convolutional operation. In [2], the node features are atom features such as atomic number and total degree. However, there are other characteristics of atoms and bonds that might be relevant depending on the data and tasks. Prediction accuracy can be expected to improve by selecting appropriate information.

B. Graph Attention Networks (GATs)

GATs [7] are neural network architectures that take graph-structured data as input. In order to overcome the drawbacks of conventional methods based on graph convolution, masked self-attention layers have been introduced. Stacked layers that allow nodes to attend to neighboring features implicitly allow different weighting of neighboring nodes without the need for costly matrix operations or prior knowledge of the graph structure. This mechanism for specifying different weights can be applied to entities other than nodes and can be expected to improve accuracy and interpretability.

C. Message Passing Neural Networks (MPNNs)

Gilmer et al. [4] showed that many existing models can be formulated as a special case of the MPNN framework. MPNNs take an undirected graph G with node features x_v and edge features e_{vw} as input. The forward propagation path consists of a message passing phase and a readout phase. The message passing phase consists of T steps, in which the message function M_t and the update function U_t are called. The message function computes the hidden state of node v using information residing on adjacent nodes w_j and edge e_{vw_j} between node v and the adjacent nodes. The update function updates the hidden state on the basis of the output of the message function and the current hidden state. In the readout phase, after step T of the message passing phase, a graph feature vector is computed using the hidden states of all nodes. Since functions M_t, U_t, and R are all differentiable, they can be trained by back propagation.

III. Methodology

We introduce a *soft self-attention layer* (Fig. 1) to compute the attention coefficients that represent the relevance of each input element. We then present three models that combine a soft self-attention layer with NFPs at different levels. We incorporate the attention mechanism in three different models (model Ⅲ-B, model Ⅲ-C, model Ⅲ-D), each of which focus on different levels of relevance optimized for accuracy and interpretability.

A. Soft Self-Attention Layer

The soft attention layer computes the attention coefficients that are used for multiplicative weighting of the input elements. In GATs [7], a node adjacent to a certain node v is used as input, and the attention coefficient of each node is computed. In our proposed method, graph feature vectors and initial state vectors are used as inputs. The soft self-attention layers (Fig. 1) can be written as

$$\begin{aligned} \boldsymbol{e} &= NN(\boldsymbol{f}; D, D) \\ \alpha_i &= \mathrm{softmax}(\boldsymbol{e})_i = \exp(e_i)/\textstyle\sum_{k=1}^{D} \exp(e_k) \end{aligned} \tag{1}$$

$$f'_i = \alpha_i f_i \tag{2}$$

where $\boldsymbol{f} \in \mathbb{R}^D$ is an input vector for weighting, and $NN(\cdot; D, D)$ is neural network with one hidden layer. This neural network has two parameters and performs a linear transformation from the first parameter dimension to the second parameter dimension.Since the total sum of the outputs of the softmax function is 1, the output value represents the normalized ratio of relevance of each input. In the soft attention layer, the attention coefficients are learned from the input itself (Eqs. (1)). With this method, the attention coefficients for each input is computed and adaptively weighted (Eq. (2)).

B. Soft Attention for Feature Vectors (model Ⅲ-B)

An *input representation* is a class of feature vectors associated to nodes (atoms) and edges (bonds or atom pairs). An example is given in TABLE I. Input representations $\boldsymbol{R}_i, i \in 1, 2, \ldots, K$ and NFPs are used to compute the final embedding vector of graph $\boldsymbol{g}_i \in \mathbb{R}^D$ fed into supervised learning. K is the number of input representations under consideration, and each representation characterizes different aspects of nodes and edges. In this model, the self-attention is introduced to

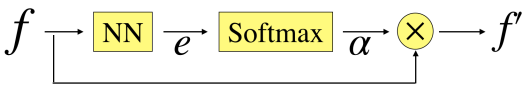

Fig. 1: General structures of self-attention layers. Since the attention coefficients are learned using the input itself, different weights can be applied to each input.

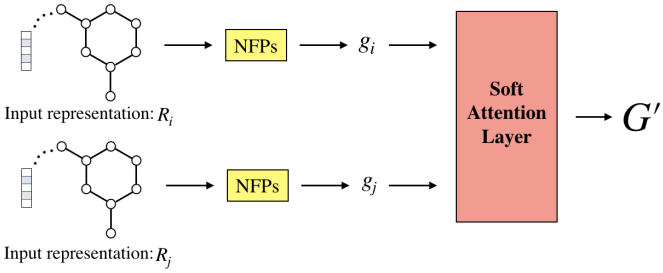

Fig. 2: Structures of models Ⅲ-B and Ⅲ-C when $K = 2$. An embedding vector of the graph is generated by NFP for each input representation. The vectors are then input to the soft attention layer, and one feature vector is computed. The difference between Ⅲ-B and Ⅲ-C is in Fig. 3.

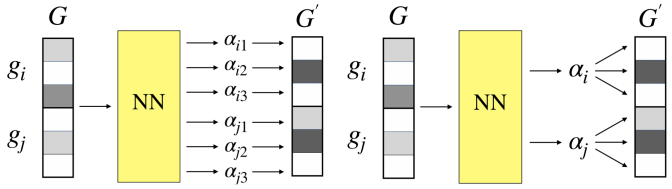

Fig. 3: **Left:** Structure of soft attention layer of Model Ⅲ-B. The number of attention coefficients is equal to the length of feature vectors. **Right:** Structure of soft attention layer of Model Ⅲ-C. The number of attention coefficients is equal to the number of input representations.

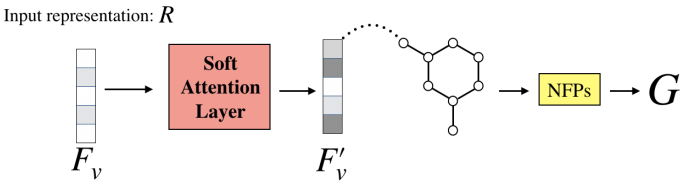

Fig. 4: Structure of model Ⅲ-D. A soft attention layer is applied to the inputs on nodes with the structure in Fig. 2 Left.

attend the concatenated vector from NPF outputs with different features (Fig. 2, Fig. 3).

$$
\begin{aligned}
\boldsymbol{g}_i &= NFP(\boldsymbol{R}_i) \\
\boldsymbol{G} &= concat(\boldsymbol{g}_1, \boldsymbol{g}_2, ..., \boldsymbol{g}_K) \\
\boldsymbol{e} &= NN(\boldsymbol{G}; K \times D, D) \\
\alpha_i &= \mathrm{softmax}(\boldsymbol{e})_i = \exp(e_i)/\textstyle\sum_{k=1}^{K \times D} \exp(e_k) \\
G'_i &= \alpha_i G_i,
\end{aligned}
$$

where $concat(\cdot)$ is the operation to concatenate multiple vectors into one. The model adaptively weights each feature vector to learn the relevant representations for better prediction. However, it is not obviously interpretable which input representation was important.

C. Importance Ratio of Input Representation (model Ⅲ-C)

This model makes it easier to understand which input representation was important for better prediction. If there are K input representations, the soft attention layer has K outputs. The importance of each input representation can be found by checking the value of the attention coefficient. The larger the coefficient, the greater the importance. The feature vector of the graph is the concatenation of the products of the attention coefficient of each input representation and the feature vector (Fig. 2, Fig. 3).

$$
\begin{aligned}
\boldsymbol{g}_i &= NFP(\boldsymbol{R}_i) \\
\boldsymbol{G} &= concat(\boldsymbol{g}_1, \boldsymbol{g}_2, ..., \boldsymbol{g}_K) \\
\boldsymbol{e} &= NN(\boldsymbol{G}; K \times D, K) \\
\alpha_i &= \mathrm{softmax}(\boldsymbol{e})_i = \exp(e_i)/\textstyle\sum_{k=1}^{K} \exp(e_k) \\
\boldsymbol{g}'_i &= \alpha_i \boldsymbol{g}_i \\
\boldsymbol{G}' &= concat(\boldsymbol{g}'_1, \boldsymbol{g}'_2, ..., \boldsymbol{g}'_K),
\end{aligned} \tag{3}
$$

where the calculation of $\boldsymbol{e}$ for Ⅲ-B and Ⅲ-C is the same. Since the attention factor is a scalar value, Eq. (3) is the product of a scalar and a vector. In other words, all elements of the vector are multiplied by the same attention coefficient. Since the number of learning parameters is less than with model Ⅲ -B, the degree of freedom of the model is smaller.

D. Soft Attention for Input Representation (model Ⅲ-D)

In models Ⅲ-B and Ⅲ-C, the embedding vectors of graphs by NFPs are input to the soft attention layer. In this model, the initial state vector $\boldsymbol{F}_v$ for node v is input to the soft attention layer. The output of the soft attention layer is $\boldsymbol{e} \in \mathbb{R}^P$, where P is the number of node features in the input representation (Fig. 4).

$$
\begin{aligned}
\boldsymbol{F}_v &= concat(\boldsymbol{r}_1, \boldsymbol{r}_2, ..., \boldsymbol{r}_P) \\
\boldsymbol{e}_v &= NN(\boldsymbol{F}_v; P, P) \\
\alpha_{vi} &= \mathrm{softmax}(\boldsymbol{e}_v)_i = \exp(e_{vi})/\textstyle\sum_{k=1}^{P} \exp(e_{vk}) \\
\boldsymbol{r}'_i &= \alpha_{vi} \boldsymbol{r}_i \\
\boldsymbol{F}'_v &= concat(\boldsymbol{r}'_1, \boldsymbol{r}'_2, ..., \boldsymbol{r}'_P),
\end{aligned}
$$

where $\boldsymbol{r}_i$ is a vector representing the i th feature of the input representation. This model represents which feature is important more finely by the attention coefficient.

IV. Evaluation

A. Datasets

We used the same three datasets used in the original research of NFPs [2] to evaluate the performance of the proposed attentive models on the top of plain NFPs.

- **Solubility** [1]: The aqueous solubility of 1,144 molecules.

***Node feature 1**: Physical properties of atoms*		
Feature	Description	Lengths
Atomic number	43 kinds of elements and other elements (one-hot)	44
Degree	Number of adjacent atoms (one-hot)	6
Total hydrogen	Number of adjacent hydrogens (one-hot)	5
Valance electrons	Number of valence electrons (one-hot)	6
Aromatic	Presence or absence (binary)	1
Total length		62

***Node feature 2**: Chemical properties of atoms*		
Feature	Description	Length
Donor	Donates electrons (binary)	1
Acceptor	Accepts electrons (binary)	1
Aromatic	In an aromatic system (binary)	1
Halogen	Halogen family (binary)	1
Acidic	Acidic (binary)	1
Basic	Basic (binary)	1
Total length		6

***Edge feature**: Properties of bonds*		
Feature	Description	Length
Bond type	Single, double, triple, or aromatic (one-hot)	4
Conjugated system	Whether it is conjugate (binary)	1
Same ring	Whether the atoms in the pair are In the same ring (binary)	1
Total length		6

TABLE I: Input representations for node and edge features

- **Drug efficacy** [3]: The EC_{50} value (in vitro) of $10,000$ molecules against a malaria parasite *P. falciparum*.
- **Photovoltaic efficiency** [5]: The photovoltaic efficiency of $20,000$ organic molecules.

B. Evaluation Metric

We ran experiments to compare the predictive performance of our models and NFP. As with experiments by Duvenaud [2], we use the root mean squared error (RMSE) to compare prediction accuracy:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}, \quad (4)$$

where y_i is the target value of data instance x_i, $\hat{y}_i$ is the predicted value, and N is the number of data instances.

C. Input Representations

We prepared two types of input representations. These two choices have been most widely used in chemoinformatics, and originated from circular topological fingerprints [6] respectively called Extended-Connectivity Fingerprints (ECFPs) and Functional-Class Fingerprints (FCFPs) with different kinds of abstraction. Note that NFPs are proposed as an extension of ECFPs. One represents the physical properties of atoms (node features 1 in TABLE I), as was also used by Duvenaud et al. [2]. The other represents the chemical properties of atoms (node feature 2 TABLE I). Each feature is represented in a one-hot encoding. Some features described in node feature 1 and 2 are binary, and it is usually enough to represent them by single-bit expressions. However, in model Ⅲ-D, two-bits expression by a one-hot encoding is used to incorporate the attention coefficients naturally. The input representation for edges in a molecule is binary expression. We compared the prediction accuracy of each proposed model with the RMSE of the prediction accuracy of NFP methods.

Model	Size of feature vector	Size of NN
Ⅲ-B	50 (25+25)	(50,100,50)
Ⅲ-C	50 (25+25)	(50,100,2)
Ⅲ-D (Node feature 1)	50	(63,50,5)
Ⅲ-D (Node feature 2)	50	(12,50,6)

TABLE II: hyperparameters of models

	Solubility	Drug efficacy	Photovoltaic efficiency
#data(original)	1,144	10,000	29,978
Training	600	6,000	18,000
Validation	200	1,900	5,900
Test partitioning	200	2,000	6,000
Total	1,000	9,900	29,000

TABLE III: The numbers of training, validation, and test data.

D. Setup

Our own implementation of NFP [2] were used as a baseline model. We trained NFPs and the three proposed models on the top of plain NFPs. NFPs are trained with three settings of only node feature 1, only node feature 2, and concatenation of both 1 and 2 as input expressions. For the third option, all nodes are labeled with concatenated vectors of both feature 1 and node feature 2. This was used to compensate for the difference in the amount of information used for the proposed method. The initial state vector assigned to the edge for all models is shown in edge feature in TABLE I. The size of feature vector and the neural network in soft attention layer of the three models are shown in TABLE II. The number of data instances used for training, validation, and test partitioning for each dataset is shown in TABLE III.Due to missing values in the original data, some data was excluded from the experiment. We used total size data for random subsets.

For model Ⅲ-B, whether to improve the predictive performance by flexible attention mechanisms is our interest. For model Ⅲ-C, we also check the learned attention coefficients for each input representation and evaluate which input representations are important. For model Ⅲ-D, we investigate what node features on the nodes are important for better prediction from each given input representation.

E. Results and Discussion

The results of the prediction performance of our three attention models for three datasets are shown in TABLE IV. Our baseline models are NFPs with three different node features (only feature 1, only feature 2, and both feature 1 and 2). The numbers in the table are RMSEs (Eq.(4)), and the best results are shown in bold.

Dataset	Solubility [1]	Drug efficacy [3]	Photovoltaic efficiency [5]
unit	$[\log \mathrm{Mol}/L]$	$[EC_{50}]$ in nM	%
Average	4.29 ± 0.40	1.47 ± 0.07	6.40 ± 0.09
NFPs + Node feature 1	1.09 ± 0.04	1.10 ± 0.03	1.89 ± 0.00
NFPs + Node feature 2	1.26 ± 0.05	1.12 ± 0.02	2.89 ± 0.02
NFPs + Node feature 1,2	1.14 ± 0.04	1.09 ± 0.02	1.87 ± 0.04
Ⅲ-B	**0.79 ± 0.09**	**1.06 ± 0.01**	**1.59 ± 0.02**
Ⅲ-C	1.00 ± 0.02	1.09 ± 0.03	1.68 ± 0.00
Ⅲ-D + Node feature 1	1.10 ± 0.03	1.63 ± 0.03	1.79 ± 0.01
Ⅲ-D + Node feature 2	1.16 ± 0.03	1.79 ± 0.05	1.99 ± 0.13

TABLE IV: Prediction performance of diffrent models

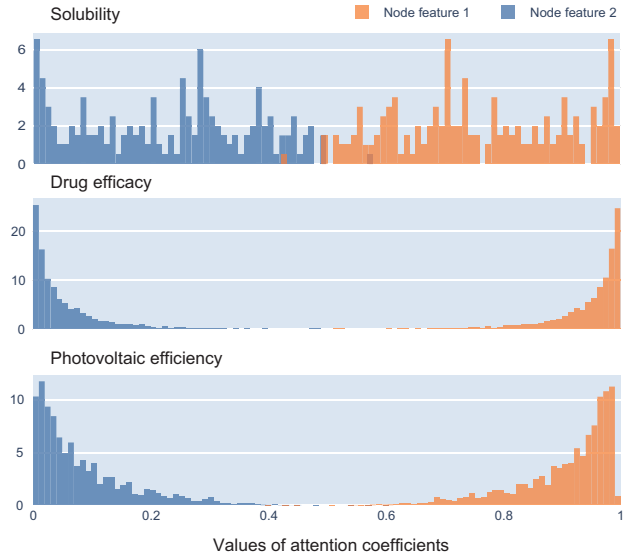

Fig. 5: The distributions of two attention weights for node feature 1 and 2 for three datasets.

1) Soft Attention for Feature Vectors (Ⅲ-B model): For all of three datasets, the Ⅲ-B model showed the highest accuracy. Since the prediction performance was higher than that of NFP with both feature 1 and 2, this result implies that introducing the attention mechanism can improve the prediction performance by adaptively selecting relevant information depending on each input. In the Ⅲ-B model, we used the attention layer in the left of Fig. 3, thus we can flexibly combine two types of chemical information for each input.

2) Importance Ratio of Input Representation (Ⅲ-C model): The Ⅲ-C model also improved the prediction performance compared to NFP with both feature 1 and 2 and the best NFPs using one of them, but also resulted in a lower prediction accuracy than the best model Ⅲ-B above. In this Ⅲ-C model, the attention coefficients are defined only block-wise corresponding to each node feature 1 or 2. Thus only two values of attention coefficients are obtained for each input (each molecule). The distribution of the attention coefficient values is shown in the Fig. 5. In the figures, the histogram in orange is the attention coefficients for the input representation 1 (Node feature 1 in TABLE I) and the histogram in blue is those for the input representation 2 (Node feature 2 in TABLE I). In this case, the distributions are symmetric since the sum of two coefficients for each graph is 1, but in general cases for $K > 3$, visualizing individual distributions is informative. For all three datasets, coefficients for feature 1 are larger, i.e., close to 1, where those for feature 2 close to 0. This would mean that weighing more on the input representation 1 improves the prediction performance. In fact, with the NFP, the accuracy when using feature 1 was higher than that when using feature 2. However we also observed the differences in distributions. For the first dataset on solubility, the distribution is relatively broad, but for other two datasets on drug efficacy and photovoltaic efficiency, distribution for feature 1 is peaky around 1 (thus those for feature 2 around 0), which means that the attention only to feature 1 is dominating. This implies that for predicting the solubility, integrating feature 1 and 2 can contribute to improvement in prediction accuracy, whereas feature 1 would be more important for predicting the drug efficacy and photovoltaic efficiency. In model Ⅲ-B and Ⅲ-C, the predict performance increased more than other datasets.

3) Soft Attention for Input Representation (Ⅲ-D model): In this Ⅲ-D model, different attention coefficients are obtained at each node in a graph, and they are used to attend to individual elements of node feature 1 or 2. The distributions of attention coefficients of node features for three datasets are shown in the Fig. 6. The larger the value of the attention coefficients, the more relevant the corresponding feature is. In TABLE IV, using node feature 2 for solubility, and node feature 1 and 2 for photovoltaic efficiency, respectively improved the prediction accuracy compared to the corresponding NFPs. However, it did not improve as much as Ⅲ-B and Ⅲ-C. We also observed that learning for the dataset on drug efficacy was failed because the resultant accuracy was worse even than a constant prediction ('Average' in TABLE IV). In this setting, we investigated the effect of only one of node feature 1 or 2, and the attention of model Ⅲ-D model was ineffective and also unstable in some cases. In future work, we also investigate results when using the concatenated feature of node feature 1 and 2 as well as results when using much more features for nodes and edges.

V. Conclusion

We investigated three proposed models of graph neural networks with self-attention mechanisms at different levels to adaptively select relevant chemical information for each input. Our experimental evaluations suggested that introducing these self-attention mechanisms contributed to not only improving the prediction accuracy but also providing quantitative interpretation using obtained attention coefficients. In particular, model Ⅲ-B showed better accuracy than any other methods including the original NFPs. Model Ⅲ-C and Ⅲ-D also slightly improved the performance of NFPs, providing informative insights via the distribution of attention coefficients to understand what kinds of chemical features are important for better prediction. Since the mechanism of the proposed method is essentially not limited to molecules, it can be a useful strategy in other general situations where domain-specific information needs to be encoded in input graphs for GNNs.

Fig. 6: **Left:** The distributions of node feature's attention weights for node feature 1 for three datasets. **Right:** The distributions of node feature's attention weights for node feature 2 for three datasets.

ACKNOWLEDGMENTS

This work was partially supported by JSPS KAKENHI Grants 17H01783, 15H05711, and 17K19953, by JST PRESTO and CREST, and by the Global Station for Big Data and CyberSecurity, a project of the Global Institution for Collaborative Research and Education at Hokkaido University.

REFERENCES

[1] Delaney, J. S.: ESOL: Estimating Aqueous Solubility Directly from Molecular Structure, *J. Chem. Inform. Comput. Sci.*, 44(3), 1000–1005 (2004)

[2] Duvenaud, D. K., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P.: Convolutional Networks on Graphs for Learning Molecular Fingerprints, in *Adv. Neural. Inf. Process. Syst (NIPS2015)*. 28, 22242232 (2015)

[3] Gamo, F.-J., Sanz, L. M., Vidal, J., de Cozar, C., Alvarez, E., Lavandera, J.-L., Vanderwall, D. E., Green, D. V., Kumar, V., Hasan, S., et al.: Thousands of chemical starting points for antimalarial lead identification, *Nature*, 465(7296), 305–310 (2010)

[4] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E.: Neural Message Passing for Quantum Chemistry, in *Proceedings of the 34th International Conference on Machine Learning (ICML2017)*, 1263–1272 (2017)

[5] Hachmann, J., Olivares-Amaya, R., Atahan-Evrenk, S., Amador-Bedolla, C., Sánchez-Carrera, R. S., Gold-Parker, A., Vogt, L., Brockway, A. M., and Aspuru-Guzik, A.: The Harvard clean energy project: large-scale computational screening and design of organic photovoltaics on the world community grid, *J. Phys. Chem. Lett.*, 2(17), 2241–2251 (2011)

[6] Rogers, D., Hahn, M.: Extended-Connectivity Fingerprints, *Journal of Chemical Information and Modeling*, 50 (5), 742–754 (2010)

[7] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lió, P., and Bengio, Y.: Graph Attention Networks, *International Conference on Learning Representations (ICLR2018)* (2018)