Vol.2019-MPS-123 No.51
Vol.2019-BIO-58 No.51
2019/6/19

A Method for the Inverse QSAR/QSPR Based on Artificial Neural Networks and Mixed Integer Linear Programming

Rachaya Chiewvanichakorn[1,a)] Chenxi Wang[1,b)] Zhe Zhang[1,c)] Aleksandar Shurbevski[1,d)]
Hiroshi Nagamochi[1,e)] Tatsuya Akutsu[2,f)]

概要：In this study, we propose a novel method for the inverse QSAR/QSPR. Given a set of chemical compounds G and their values $a(G)$ of a chemical property, we define a feature vector $f(G)$ of each chemical compound G. By using a set of feature vectors as training data, the first phase of our method constructs a prediction function ψ with an artificial neural network (ANN) so that $\psi(f(G))$ takes a value nearly equal to $a(G)$ for many chemical compounds G in the set. Given a target value a^* of the chemical property, the second phase infers a chemical structure G^* such that $a(G^*) = a^*$ in the following way. We compute a vector f^* such that $\psi(f^*) = a^*$, where finding such a vector f^* is formulated as a mixed integer linear programming problem (MILP). Finally we generate a chemical structure G^* such that $f(G^*) = f^*$. For acyclic chemical compounds and some chemical properties such as heat of formation, boiling point, and retention time, we conducted some computational experiments with our method.

1. Introduction

One of the important challenges in bioinformatics and chemo-informatics is computational design of a novel chemical compound that has desirable properties. This problem has been extensively studied under the name of inverse QSAR/QSPR (quantitative structure-activity and structure-property relationships) [13], [19]. It can be formulated as computation of a graph structure representing a chemical compound that maximizes (or minimizes) an objective function under various constraints, where objective functions are often derived from a set of training data consisting of known molecules and their activities/properties using statistical and/or machine learning methods. Various heuristic and statistical methods have been developed for finding optimal or near optimal graph structures under given objective functions [7], [13], [17]. In QSAR/QSPR, chemical compounds are often represented as a vector of real or integer numbers, which is called a feature vector or (a set of) descriptors. Therefore, it is an important subtask in inverse QSAR/QSPR to infer or enumerate graph structures from a given feature vector and extensive studies have also been done for solving this subtask [9], [15]. We also analyzed the computational complexity of this inference problem [3], [14] and developed efficient enumeration algorithms [4], [11].

Based on the recent progress of Artificial Neural Network (ANN) and deep learning technologies, novel approaches have been proposed for design of chemical compounds. For example, methods using variational autoencoder [5], grammar variational autoencoder [10], and recurrent neural networks [18], [20] have been developed. In these approaches, ANNs are trained using existing chemical compound data and then novel chemical graphs are obtained by solving a kind of inverse problem on ANN, in which an input vector of real numbers is computed from given ANN and output vector. In order to solve this in-

[1] Graduate School of Informatics, Kyoto University, Sakyo, Kyoto 606–8501, Japan
[2] Bioinformatics Center, Institute for Chemical Research, Kyoto University Uji-city, Kyoto 611–0011, Japan
[a)] ch.rachaya@amp.i.kyoto-u.ac.jp
[b)] chenxi@amp.i.kyoto-u.ac.jp
[c)] zzx@amp.i.kyoto-u.ac.jp
[d)] shurbevski@amp.i.kyoto-u.ac.jp
[e)] nag@amp.i.kyoto-u.ac.jp
[f)] takutsu@kuicr.kyoto-u.ac.jp

verse problem or its variants, various statistical methods have been employed. Since the optimality of the solution is not necessarily guaranteed by statistical methods, an integer linear programming (ILP)-based method has also been proposed for solving a kind of inverse problem on ANNs with linear threshold functions [12]. However, linear threshold functions are not widely used in recent ANNs. Therefore, we have recently developed novel methods for solving the inverse problem on ANNs with ReLU functions and sigmoid functions [1], [2]. Since it is known that the inverse problem is NP-hard even for ANNs with linear threshold functions [12], we employed Mixed Integer Linear Programming Problem (MILP) formulations, where MILP is one of widely used approaches to solving NP-hard problems. In this method, activation functions on neurons are efficiently represented as piece-wise linear functions, which can exactly represent ReLU functions and well approximate sigmoid functions.

In this work, we combine our previous approaches; efficient enumeration of tree-like graphs [4], and MILP-based formulation of the inverse problem on ANNs [1], [2]. This combined framework for QSAR/QSPR mainly consists of two phases; one for constructing a prediction function to a chemical property, and the other for constructing graphs based on the inverse of the prediction function. In the first phase, an ANN is trained from existing chemical compounds and their properties using a standard learning algorithm, where each chemical compound is transformed into a feature vector that is used as an input for the ANN. In the second phase, a feature vector is inferred from the trained ANN and a given chemical property and then a set of chemical structures is inferred whose feature vectors are the same as or close to the inferred feature vector. That is, the second phase solves two inverse problems; inference of a feature vector from a trained ANN and a given chemical property, and inference of chemical structures from a given feature vector. In order to effectively combine these two problems, we develop a new set of graph theoretical descriptors. In this report, we describe the outline of our proposed framework and present results of preliminary computational experiments using several data sets consisting of chemical compounds with acyclic graph structures and their chemical properties.

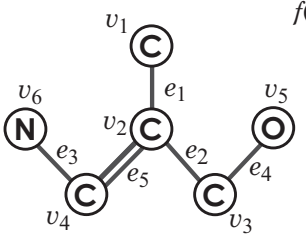
2. Preliminary

Let $\mathbb{R}$ and $\mathbb{Z}$ denote the sets of reals and non-negative integers, respectively.

Graphs A *graph* stands for a simple undirected graph, where an edge joining two vertices u and v is denoted by uv $(= vu)$. Let $G = (V, E)$ be a graph with a set V of vertices and a set E of edges. For a vertex $v \in V$, the set of neighbors of v in G is denoted by $N_G(v)$, and the *degree* $\deg_G(v)$ of v is defined to be $|N_G(v)|$. The length of a path is defined to be the number of edges in the path. The *distance* $\mathrm{dist}_G(u, v)$ between two vertices $u, v \in V$ is defined to be the minimum length of a path connecting u and v in G. The *diameter* $\mathrm{dia}(G)$ of G is defined to be the maximum distance between two vertices in G; i.e., $\mathrm{dia}(G) \triangleq \max_{u,v \in V} \mathrm{dist}_G(u, v)$. The *sum-distance* $\mathrm{smdt}(G)$ of G is defined to be the sum of distances over all vertex pairs; i.e., $\mathrm{smdt}(G) \triangleq \sum_{u,v \in V} \mathrm{dist}_G(u, v)$.

Chemical Graphs We represent the graph structure of a chemical compound as a graph with labels on vertices and multiplicity on edges in a hydrogen-suppressed model. Let Λ be a set of labels each of which represents a chemical element such as `C` (carbon), `O` (oxygen), `N` (nitrogen), `S` (sulfur) and so on, where we assume that Λ does not contain `H` (hydrogen). Let mass(`a`) and val(`a`) denote the mass and valance of a chemical element $\mathtt{a} \in \Lambda$, respectively. Two atoms `a` and `b` joined with a bond of multiplicity k is denoted by a tuple $\gamma = (\mathtt{a}, \mathtt{b}, k)$ $(= (\mathtt{b}, \mathtt{a}, k))$. Let Γ denote the set $\{(\mathtt{a}, \mathtt{b}, k) \mid \mathtt{a}, \mathtt{b} \in \Lambda, k \in \{1, 2, 3\}\}$ of tuples.

A *chemical graph* in a hydrogen-suppressed model is defined to be a tuple $G = (H, \alpha, \beta)$ of a graph $H = (V, E)$, a function $\alpha : V \to \Lambda$ (called an *atom-function*) and a function $\beta : E \to \{1, 2, 3\}$ (called a *bond-function*) such that (i) H is connected; and (ii) $\sum_{v \in N_H(u)} \beta(uv) \leq \mathrm{val}(\alpha(u))$ for each vertex $u \in V$. Figure 1 illustrates an example of a chemical graph $G = (H, \alpha, \beta)$.



$f(G)$=(n=6, $\overline{m}$=13,
$n_{\mathtt{C}}$=4, $n_{\mathtt{O}}$=1, $n_{\mathtt{N}}$=1,
$n_{(\mathtt{C},\mathtt{C},1)}$=2, $n_{(\mathtt{C},\mathtt{C},2)}$=1,
$n_{(\mathtt{C},\mathtt{O},1)}$=1, $n_{(\mathtt{C},\mathtt{N},1)}$=1,
n_1=3, n_2=3, n_3=1,
n_4=0, n_5=0, n_6=0,
b_2=1, b_3=0, $\overline{\mathrm{dia}}$=0.667,
$\overline{\mathrm{smdt}}$=0.1435)

図 1 An example of a chemical graph $G = (H, \alpha, \beta)$ and its feature vector $f(G)$, where $V = \{v_1, v_2, \ldots, v_6\}$ $E = \{e_1 = v_1v_2, e_2 = v_2v_3, e_3 = v_4v_6, e_4 = v_3v_5, e_5 = v_2v_4\}$, $\alpha(v_i) = \mathtt{C}$, $1 \leq i \leq 4$, $\alpha(v_5) = \mathtt{O}$, $\alpha(v_6) = \mathtt{N}$, $\beta(e_i) = 1$, $1 \leq i \leq 4$ and $\beta(e_5) = 2$.

Vol.2019-MPS-123 No.51
Vol.2019-BIO-58 No.51
2019/6/19

3. A Method for Inferring Chemical Graphs

Our method for the inverse QSAR/QSPR mainly consists of two phases; one for constructing a prediction function to a chemical property, and the other for constructing graphs based on the inverse of the prediction function. For a specified chemical property such as boiling point, we denote by $a(G)$ the value of the specified chemical property for a given chemical compound G, which is represented by a chemical graph $G = (H, \alpha, \beta)$.

Phase 1.

1. Prepare a data set $D = \{(G_i, a(G_i)) \mid i = 1, 2, \ldots, p\}$ for a specified chemical property, where G_i is a chemical graph. Set reals $\underline{a}, \overline{a} \in \mathbb{R}$ such that $\underline{a} \leq \min\{a(G_i) \mid i = 1, 2, \ldots, p\}$ and $\overline{a} \geq \max\{a(G_i) \mid i = 1, 2, \ldots, p\}$.
2. Set a graph class $\mathcal{G}$ to be a set of chemical graphs such that $\mathcal{G} \supseteq \{G_i \mid i = 1, 2, \ldots, p\}$. Introduce a function $f : \mathcal{G} \to \mathbb{R}^K$ for a positive integer K. We call $f(G)$ the *feature vector* of $G \in \mathcal{G}$, and call each entry of a vector $f(G)$ a *descriptor* of G. (Our choice of descriptors will be discussed in Section 4.)
3. Select an architecture and an activation function of an artificial neural network (ANN) $\mathcal{N}$ that, given a vector in $\mathbb{R}^K$, returns a real in the range $[\underline{a}, \overline{a}]$. Using the data set D as training data, choose weights and biases of $\mathcal{N}$ to construct a prediction function ψ with $\mathcal{N}$ so that $\psi(f(G))$ takes a value nearly equal to $a(G)$ for many chemical graphs in D.

The novelty of our method is to directly treat the following inverse problems.

Inverse Prediction

Input: A real $a^* \in [\underline{a}, \overline{a}]$.
Output: A vector $f^* \in \mathbb{R}^K$ such that $\psi(f^*) = a^*$.

Graph Enumeration

Input: A vector $f^* \in \mathbb{R}^K$.
Output: All graphs $G^* \in \mathcal{G}$ such that $f(G^*) = f^*$.

It is known [1], [2] that given an ANN $\mathcal{N}$ with a piecewise-linear activation function (such as a rectified linear unit (Relu) function), a mixed integer linear programming problem (MILP) $P(\mathcal{N})$ with variables x_i, $i = 1, 2, \ldots, K$ and y (and some other others) can be formulated so that the set of feasible solutions $(x_1, x_2, \ldots, x_K)$ to $P(\mathcal{N})$ with $y = a^*$ is equal to the set of vectors $f^* \in \mathbb{R}^K$ such that $\psi(f^*) = a^*$. When the activation function of $\mathcal{N}$ is not piecewise-linear function, we approximate the function with a piecewise-linear function to introduce an MILP $P(\mathcal{N})$ whose feasible solution $f^* = (x_1, x_2, \ldots, x_K)$ with $y = a^*$ nearly satisfies $\psi(f^*) = a^*$.

An algorithm to Graph Enumeration is designed based on the branch-and-bound method (see [4] for enumerating acyclic chemical compounds).

Phase 2.

4. Formulate Inverse Prediction for the resulting prediction function ψ as an MILP $P(\mathcal{N})$ based on $\mathcal{N}$. Find a set F^* of vectors $f^* \in \mathbb{R}^K$ such that $(1 - \varepsilon)a^* \leq \psi(f^*) \leq (1 + \varepsilon)a^*$ for a small real $\varepsilon > 0$.
5. Enumerate all graphs $G^* \in \mathcal{G}$ such that $f(G) = f^*$ for some $f^* \in F^*$.

In Step 4, we introduce a tolerance $\varepsilon > 0$ because there may not exist a vector f^* such that $\psi(f^*)$ is numerically equal to a real a^*.

In Step 5, there may not exist a graph $G^* \in \mathcal{G}$ such that $f(G) = f^*$ for some vector $f^* \in F^*$. We try to avoid generating such vectors f^* in Step 4 by including some additional constraints into the MILP $P(\mathcal{N})$, as will be discussed in Section 5.

4. Descriptors in Feature Vectors

In our method, we use only graph theoretical descriptors for defining a feature vector, which facilitates our designing an algorithm for constructing graphs in the second phase. Given a chemical graph $G = (H = (V, E), \alpha, \beta)$, we define a *feature vector* $f(G)$ that consists of the following graph theoretical descriptors.

- n : the number of vertices; i.e., $n = |V|$.
- $\overline{m}$: the average mass of atoms in G; i.e., $\overline{m} = \sum_{v \in V} \mathrm{mass}(\alpha(v))/n$.
- $n_{\mathtt{a}}$, $\mathtt{a} \in \Lambda$: the number of vertices with label $\mathtt{a} \in \Lambda$; i.e., $n_{\mathtt{a}} = |\{v \in V \mid \alpha(v) = \mathtt{a}\}|$.
- n_γ, $\gamma = (\mathtt{a}, \mathtt{b}, k) \in \Gamma$: the number of label pairs $\{\mathtt{a}, \mathtt{b}\}$ with multiplicity k; i.e., $n_{(\mathtt{a},\mathtt{b},k)} = |\{uv \in E \mid \alpha(u) = \mathtt{a}, \alpha(v) = \mathtt{b}, \beta(uv) = k\}|$, $\mathtt{a}, \mathtt{b} \in \Lambda$, $k \in \{1, 2, 3\}$.
- n_d, $d \in \{1, 2, \ldots, 6\}$: the number of vertices of degree d in H; i.e., $n_d = |\{v \in V \mid \deg_H(v) = d\}|$, where the multiplicity of edges incident to a vertex v is ignored in the degree of v.
- b_i, $i = 2, 3$: the number of double and triple bonds; i.e., $b_i = \{e \in E \mid \beta(e) = i\}$, $i = 2, 3$.
- $\overline{\mathrm{dia}}$: the diameter of H divided by n; i.e., $\overline{\mathrm{dia}} = \mathrm{dia}(H)/n$.
- $\overline{\mathrm{smdt}}$: the sum of diameters of H divided by n^3, i.e., $\overline{\mathrm{smdt}} = \mathrm{smdt}(H)/n^3$.

Figure 1 illustrates an example of a feature vector $f(G)$.

We may not include a descriptor s as an entry of our feature vector if $s = 0$ for all chemical graphs in a data set D.

5. Additional Constraints in MILPs

An MILP $P(\mathcal{N})$ can be formulated so that a feasible solution $f^* = x \in \mathbb{R}^K$ to $P(\mathcal{N})$ satisfies $\psi(f^*) = a^*$ [1], [2]. This, however, does not guarantee that there always exists a graph $G^* \in \mathcal{G}$ such that $f(G) = f^*$. This is because the original MILP $P(\mathcal{N})$ does not contain any constraint that some of the descriptors of a chemical graph must obey. To reduce the chance of generating a vector f^* that does not admit any graph G^* with $f(G) = f^*$, we include additional constraints into the original MILP. For space limitation, we here list up only basic constraints.

The number of vertices in a graph H is equal to the total number of labels; i.e.,

$$n = \sum_{\mathtt{a}\in\Lambda} n_{\mathtt{a}}.$$

The molecular mass in G is the sum of mass over all labels; i.e.,

$$n \cdot \overline{m} = \sum_{\mathtt{a}\in\Lambda} \mathrm{mass}(\mathtt{a}) n_{\mathtt{a}}.$$

The sum of degree over all vertices in graph H is twice the number of edges; i.e.,

$$\sum_{d=1,2,\ldots,6} d n_d = 2 \sum_{\gamma\in\Gamma} n_\gamma.$$

The number of k-bonds in G is the total number of label-pairs with multiplicity k; i.e.,

$$\sum_{\gamma=(\mathtt{a},\mathtt{b},k)\in\Gamma} n_\gamma = b_k, \quad \forall k = 2, 3.$$

The number of vertices of degree at least d in G is bounded from above by the total number of vertices with label $\mathtt{a}$ and $\mathrm{val}(\mathtt{a}) \geq d$; i.e.,

$$\sum_{d\leq i\leq 6} n_i \leq \sum_{\mathtt{a}\in\Lambda:\ \mathrm{val}(\mathtt{a})\geq d} n_{\mathtt{a}}, \quad \forall d = 1, 2, \ldots, 6.$$

The total number of multiplicities incident to a label $\mathtt{a}$ in G is at most the total valence of vertices of $\mathtt{a}$; i.e.,

$$\sum_{k=1,2,3} \Big(\sum_{\gamma=(\mathtt{a},\mathtt{b},k):\mathtt{a}\neq\mathtt{b}} k n_\gamma + \sum_{\gamma=(\mathtt{a},\mathtt{a},k)} 2k n_\gamma \Big) \leq \mathrm{val}(\mathtt{a}) n_{\mathtt{a}}$$

for each label $\mathtt{a} \in \Lambda$.

The number of edges is at least the number of vertices minus 1 in a connected graph H; i.e.,

$$\sum_{\gamma\in\Gamma} n_\gamma \geq n - 1.$$

There are several other constraints, some of which are specialized to acyclic chemical graphs, but we omit describing them here due to space limitation.

6. Experimental Results

We implemented our method for inferring acyclic chemical graphs and executed on a PC with Intel Core i5 1.6 GHz CPU and 8GB RAM running under the Mac OS operating system version 10.14.4. We select five chemical properties: heat of atomization (Ha), heat of formation (Hf), boiling point (Bp), octanol/water partition coefficient (Kow) and retention time (Rt).

For Ha, Hf, Bp and Kow (resp., Rt), we used the acyclic chemical graphs in [16] (resp., [8]). Table 1 shows the size and range of data sets that we prepared for each chemical property, where we denote the following: $a()$: chemical property; $|D|$: the size $|D|$ of data set D; Λ: chemical elements in D; $[\underline{n}, \overline{n}]$: the minimum and maximum number of vertices in H over data set D; $[\underline{a}, \overline{a}]$: the minimum and maximum values of $a(G)$ over data set D; and K: the number of descriptors in $f(G)$.

表 1 The size and range of data sets in Steps 1 and 2

$a()$	$\lvert D\rvert$	Λ	$[\underline{n}, \overline{n}]$	$[\underline{a}, \overline{a}]$	K
Ha	128	C,O,S	[2,11]	[450.3, 3009.6]	19
Hf	88	C,O,S	[2,16]	[20.2, 94.8]	19
Bp	131	C,O,S	[2,16]	[-103.7, 286.8]	19
Kow	62	C,O,S	[2,16]	[-0.77, 8.20]	19
Rt	39	C,O	[11,16]	[1422, 1919]	18

We set a graph class $\mathcal{G}$ to be the set of all acyclic chemical graphs on the label set Λ in Table 1

Results on Phase 1. We use `scikit-learn` to construct ANNs where the tool and activation function are set to be MLPRegressor and Relu, respectively. We tested several different architectures of ANNs for each chemical property. To evaluate the performance of the resulting prediction function ψ with cross-validation, we partition a given data set D into five subsets D_i, $i = 1, 2, 3, 4, 5$ randomly, where $D \setminus D_i$ is used for a training set and D_i is used for a test set in five trials $i = 1, 2, 3, 4, 5$.

Table 2 shows the results on Phase 1, where we denote the following: arch.: architecture of ANN $\mathcal{N}$, time: the average time (sec) to construct ANNs for each trial; test R^2 (ave): the average of coefficient of determination over the five test sets; and test R^2 (best): the largest value of coefficient of determination over the five test sets.

Results on Phase 2. We implemented Steps 4 and 5 in Phase 2 as follows.

Step 4. In this step, we also specify a size $n \in [\underline{n}, \overline{n}]$ of graph. We choose two target values $a^* \in [\underline{a}, \overline{a}]$ for each

表 2 Results on constructing ANNs in Step 3

$a()$	arch.	time	test R^2 (ave)	(best)
HA	(19,10,1)	6.251	0.999	0.999
HF	(19,10,1)	0.818	0.985	0.993
BP	(19,15,1)	3.974	0.965	0.985
KOW	(19,10,10,1)	0.219	0.968	0.980
RT	(18,10,10,1)	4.869	0.892	0.931

chemical property, and two values $n \in [\underline{n}, \overline{n}]$ for each target value a^*. We set $\varepsilon = 0.02$. For each pair (a^*, n), we fix one of some three values for n_1 (the number of vertices of degree 1), and divide a range A for possible values of diameter into two ranges A_1 and A_2 and a range B for possible values of sum of distances into three ranges B_i, $i = 1, 2, 3$. This scheme results in $3 \times 2 \times 3 = 18$ MILPs with different restrictions for each pair (a^*, n), where each MILP is either feasible or infeasible and we find one feasible vector f^* to each feasible MILP. Let F^* denote the set of vectors f^* generated from these 18 MILPs, where $|F^*| \leq 18$. To solve an MILP $P(\mathcal{N})$ in Step 4, we use CPLEX (ILOG CPLEX version 12.8) [6].

Tables 3 to 7 show the results on Step 4, where we denote the following: a^*: a target value in $[\underline{a}, \overline{a}]$; n: a specified number of vertices in $[\underline{n}, \overline{n}]$; $|F^*|$: the number $|F^*|$ of vectors f^* generated from 18 MILPs; and f-time: the time (sec.) to compute a set F^* of vectors f^*.

表 3 Results on generating vectors in Step 4 for HA

a^*	n	$\lvert F^*\rvert$	f-time
2700	10	18	0.1208
2700	11	18	0.1288
2900	10	18	0.0717
2900	11	18	0.1284

表 4 Results on generating vectors in Step 4 for HF

a^*	n	$\lvert F^*\rvert$	f-time
70	11	15	0.0708
70	12	3	0.0611
90	12	18	0.1495
90	13	18	0.1693

表 5 Results on generating vectors in Step 4 for BP

a^*	n	$\lvert F^*\rvert$	f-time
190	11	18	0.269
190	12	18	0.193
220	12	18	0.188
220	13	18	0.111

We observe that MILPs with some restrictions were infeasible. For example, $|F^*| = 3$ for $(a^* = 70, n = 12)$ in

表 6 Results on generating vectors in Step 4 for KOW

a^*	n	$\lvert F^*\rvert$	f-time
5	11	18	0.205
5	12	18	0.214
7	14	12	0.444
7	15	15	0.296

表 7 Results on generating vectors in Step 4 for RT

a^*	n	$\lvert F^*\rvert$	f-time
1600	14	18	0.263
1600	15	18	0.243
1900	15	18	0.233
1900	16	18	0.237

Table 4 means that three out of 18 MILPs were feasible.

Step 5. In this step, we modified the algorithm proposed in [4] to enumerate all acyclic graphs. We conducted the following two options in our experiment:
(a) for each $f^* \in F^*$, enumerate all graphs $G^* \in \mathcal{G}$ such that $f(G) = f^*$; and
(b) for each $f^* = (c_1, c_2, \ldots, c_K) \in F^*$, enumerate all graphs $G^* \in \mathcal{G}$ such that $f(G) = (c_1, c_2, \ldots, c_{K-1}, \tilde{c}_K)$ and $(1 - \delta)c_K \leq \tilde{c}_K \leq (1 + \delta)c_K$ for a small real $\delta > 0$ (where c_K is a value to descriptor $\overline{\text{smdt}}$).
The aim of Step 5(b) is to find a more number of graphs than (a) by allowing some difference at the last entry of descriptor $\overline{\text{smdt}}$, and we later examine if the predicted value $\psi(f(G^*))$ remains nearly equal to a^*.

Tables 8 to 17 show the results on Step 5, where we denote the following: a^*: a target value in $[\underline{a}, \overline{a}]$; n: a specified number of vertices in $[\underline{n}, \overline{n}]$; $\#G^*$: the number of acyclic chemical graphs G^* such that $f(G^*) = f^*$, where the number of chemical graphs already registered in chemical database PubChem among the generated chemical graphs is indicated in the parenthesis; $[\underline{\psi}, \overline{\psi}]$: the minimum and maximum values among $\psi(f(G^*))$ predicted with the ANN for all generated chemical graphs G^*; and G-time: the time (sec.) to compute all chemical graphs G^* (or to detect that there is no graph G^*).

表 8 Results on enumerating graphs in Step 5(a) for HA

a^*	n	$\#G^*$	$[\underline{\psi}, \overline{\psi}]$	G-time
2700	10	2 (2)	[2736.9, 2737.7]	0.0331
2700	11	1 (1)	[2649.0, 2649.0]	0.0579
2900	10	2 (2)	[2922.2, 2923.1]	0.0297
2900	11	152 (7)	[2882.3, 2894.6]	0.0352

We observe that some vector $f^* \in F^*$ admits no graph G^*. For example, no graph G^* was found in the case of $(a^* = 220, n = 12)$ in Table 12. This is possible because

表 9 Results on Step 5(b) with $\delta = 0.03$ for HA

a^*	n	$\#G^*$	$[\underline{\psi}, \overline{\psi}]$	G-time
2700	10	15 (14)	[2672.3, 2737.7]	0.0315
2700	11	89 (13)	[2646.4, 2686.7]	0.0610
2900	10	12 (12)	[2915.7, 2928.0]	0.0308
2900	11	666 (48)	[2847.4, 2898.1]	0.0414

表 10 Results on enumerating graphs in Step 5(a) for HF

a^*	n	$\#G^*$	$[\underline{\psi}, \overline{\psi}]$	G-time
70	11	18 (16)	[70.89, 70.89]	0.0279
70	12	1 (1)	[71.05, 71.05]	0.0072
90	12	38 (0)	[88.76, 91.15]	0.0775
90	13	41 (0)	[91.37, 91.37]	0.0687

表 11 Results on Step 5(b) with $\delta = 0.03$ for HF

a^*	n	$\#G^*$	$[\underline{\psi}, \overline{\psi}]$	G-time
70	11	105 (85)	[68.62, 71.25]	0.0275
70	12	2 (2)	[71.05, 71.05]	0.0063
90	12	439 (2)	[88.30, 91.15]	0.8498
90	13	573 (3)	[88.82, 91.37]	0.0775

表 12 Results on enumerating graphs in Step 5(a) for BP

a^*	n	$\#G^*$	$[\underline{\psi}, \overline{\psi}]$	G-time
190	11	24 (0)	[187.7, 189.7]	0.0510
190	12	2 (0)	[216.8, 216.8]	0.0405
220	12	0 (0)	-	0.0477
220	13	1 (1)	[218.4, 218.4]	0.0346

表 13 Results on Step 5(b) with $\delta = 0.03$ for BP

a^*	n	$\#G^*$	$[\underline{\psi}, \overline{\psi}]$	G-time
190	11	132 (15)	[187.6, 189.8]	0.0539
190	12	55 (0)	[187.5, 216.8]	0.0413
220	12	23 (1)	[216.2, 224.5]	0.0479
220	13	32 (32)	[216.6, 223.8]	0.0361

表 14 Results on enumerating graphs in Step 5(a) for KOW

a^*	n	$\#G^*$	$[\underline{\psi}, \overline{\psi}]$	G-time
5	11	4 (1)	[5.05, 5.05]	0.0411
5	12	17 (0)	[5.05, 5.07]	0.0471
7	14	6(1)	[6.93, 6.95]	0.0425
7	15	2(2)	[6.88, 7.03]	0.1372

表 15 Results on in Step 5(b) with $\delta = 0.03$ for KOW

a^*	n	$\#G^*$	$[\underline{\psi}, \overline{\psi}]$	G-time
5	11	59 (14)	[4.51, 5.05]	0.0400
5	12	126 (3)	[4.40, 5.07]	0.0507
7	14	16(2)	[6.93, 6.95]	0.0400
7	15	3(3)	[6.88, 7.03]	0.1344

the current set of additional constraints in our MILP is a necessary condition for a feasible vector f^* to admit a graph G^* with $f(G^*) = f^*$. The graphs G^* in Tables 16 and 17 for RT have many double bonds between carbons and none of them has been registered in PubChem.

表 16 Results on enumerating graphs in Step 5(a) for RT

a^*	n	$\#G^*$	$[\underline{\psi}, \overline{\psi}]$	G-time
1600	14	0 (0)	-	0.533
1600	15	12 (0)	[1582.2, 1582.2]	2.367
1900	15	0 (0)	-	1.995
1900	16	5 (0)	[1903.4, 1903.4]	3.922

表 17 Results on Step 5(b) with $\delta = 0.03$ for RT

a^*	n	$\#G^*$	$[\underline{\psi}, \overline{\psi}]$	G-time
1600	14	1335 (0)	[1578.4, 1628.4]	0.540
1600	15	2002 (0)	[1575.2, 1631.8]	2.755
1900	15	399 (0)	[1872.5, 1923.5]	2.066
1900	16	1634 (0)	[1865.0, 1909.1]	4.103

7. Concluding Remarks

In this paper, we proposed a method for the inverse QSAR/QSPR and implemented it for inferring acyclic chemical graphs using a feature vector f with only graph theoretical descriptors. For a different chemical property, once we can successfully construct a good prediction function ψ with a new ANN $\mathcal{N}$, the second phase such as formulating an MILP $P(\mathcal{N})$ and generating vectors f^* and graphs G^* can be executed in a rather straightforward way. For the five chemical properties that we selected, it seems that our graph theoretical feature vector performs well for constructing a good prediction function. This is probably because the kinds of chemical elements in Λ is small. For a chemical property to which some electronic descriptors are useful to construct a good prediction function, it would be worth for predicting those electronic descriptors with a graph theoretical feature vector. Recently we found a set of linear constraints with integer variables that can be added to an MILP $P(\mathcal{N})$ in Step 4 so that any feasible solution f^* to $P(\mathcal{N})$ always admits a chemical graph G^* such that $f(G^*) = f^*$. It is left as a future work to implement the new MILP in our method.

参考文献

[1] Akutsu, T., Nagamochi, H.: A mixed integer linear programming formulation to artificial neural networks, *2nd International Conference on Information Science and System*, Japan, March 16-19 (2019).

[2] Akutsu, T., Nagamochi, H.: A mixed integer linear programming formulation to artificial neural networks, *Technical Report 2019-001*, Department of Applied Mathematica and Physics, Kyoto University (2019). `http://www.amp.i.kyoto-u.ac.jp/tecrep/index.html`

[3] Akutsu, T., Fukagawa, D., Jansson, J., Sadakane, K.: Inferring a graph from path frequency, *Discrete Applied Mathematics*, vol. 160, 10-11, 1416-1428 (2012).

[4] Fujiwara, H., Wang, J., Zhao, L., Nagamochi, H.,

Akutsu, T.: Enumerating treelike chemical graphs with given path frequency, *Journal of Chemical Information and Modeling*, vol. 48, 7, 1345-1357 (2008).

[5] Gómez-Bombarelli, R., Wei, D. Duvenaud, J. N., Hermández-Lobato, J. M., Sanchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., Aspuru-Guzik, A.: Automatic chemical design using a data-driven continuous representation of molecules, *ACS Central Science*, vol. 4, 268-276 (2018).

[6] IBM ILOG CPLEX Optimization Studio 12.8, `https://www.ibm.com/support/knowledgecenter/SSSA5P_12.8.0/ilog.odms.studio.help/pdf/usrcplex.pdf`

[7] Ikebata, H., Hongo, K., Isomura, T., Maezono, R.: Bayesian molecular design with a chemical language model, *Journal of Computer Aided Molecular Design*, vol. 31, 379-391 (2017).

[8] Jalali-Heravi, M., Fatemi, M. H.: Artificial neural network modeling of Kováts retention indices for noncyclic and monocyclic terpenes, Journal of Chromatography A, 915, 177-183 (2001).

[9] Kerber, A., Laue, R., Gruner, T., Meringer, M.: MOLGEN 4.0, *Match Communications in Mathematical and in Computer Chemistry*, vol. 37, 205-208 (1998).

[10] Kusner, M. J., Paige, B., Hernández-Lobato, J. M.: Grammar variational autoencoder, In: *Proc. 34th International Conference on Machine Learning* (ICML 2017), 1945-1954 (2017).

[11] Li, J., Nagamochi, H., Akutsu, T.: Enumerating substituted benzene isomers of tree-like chemical graphs, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 15, 2, 633-646 (2018).

[12] Liu, P., Bao, Y., Hayashida, M., Akutsu, T.: Finding pre-images for neural networks: an integer linear programming approach, Poster abstract, *17th International Workshop on Bioinformatics and Systems Biology*, (2017).

[13] Miyao, T., Kaneko, H., Funatsu, K.: Inverse QSPR/QSAR analysis for chemical structure generation (from y to x), *Journal of Chemical Information and Modeling*, vol. 56, 2, 286-299 (2016).

[14] Nagamochi, H.: A detachment algorithm for inferring a graph from path frequency, *Algorithmica*, vol. 53, 2, 207-224 (2009).

[15] Reymond, J-L.: The chemical space project, *Accounts of Chemical Research*, vol, 48, 722-730 (2015).

[16] Roy K., Saha A.: Comparative QSPR studies with molecular connectivity, molecular negentropy and TAU indices. Part I: molecular thermochemical properties of diverse functional acyclic compounds, Journal of Molecular Modeling, 9(4), 259-270, (2003).

[17] Rupakheti, C., Virshup, A., Yang, W., Beratan, D. N.: Strategy to discover diverse optimal molecules in the small molecule universe, *Journal of Chemical Information and Modeling*, vol. 55, 2, 529-537 (2015).

[18] Segler, M. H. S., Kogej, T., Tyrchan, C., Waller, M. P.: Generating focused molecule libraries for drug discovery with recurrent neural networks, *ACS Central Science*, vol. 4, 120-131 (2018).

[19] Skvortsova, M. I., Baskin, I. I., Slovokhotova, O. L., Palyulin, V. A., Zefirov, N. S.: Inverse problem in QSAR/QSPR studies for the case of topological indices characterizing molecular shape (Kier indices), *Journal of Chemical Information and Computer Science*, vol. 33, 630-640 (1993).

[20] Yang, X., Zhang, J., Yoshizoe, K., Terayama, K., Tsuda, K.: ChemTS: an efficient Python library for de novo molecular generation, *Journal of Science and Technology of Advanced Materials*, vol. 18, 1, 972-976 (2017).