Pattern Analysis and Applications (2023) 26:1161–1170
https://doi.org/10.1007/s10044-023-01141-3

THEORETICAL ADVANCES

Weighted edit distance optimized using genetic algorithm for SMILES-based compound similarity

In-Hyuk Choi[1] · Il-Seok Oh[1,2]

Received: 13 April 2022 / Accepted: 24 January 2023 / Published online: 18 February 2023

Abstract
A method for developing new drugs is the ligand-based approach, which requires intermolecular similarity computation. The simplified molecular input line entry system (SMILES) is primarily used to represent the molecular structure in one dimension. It is a representation of molecular structure; the properties can be completely different even if only one character is changed. Applying the conventional edit distance method makes it difficult to obtain optimal results, because the insertion, deletion, and substitution of molecules are considered the same in calculating the distance. This study proposes a novel edit distance using an optimal weight set for three operations. To determine the optimal weight set, we present a genetic algorithm with suitable hyperparameters. To emphasize the impact of the proposed genetic algorithm, we compare it with the exhaustive search algorithm. The experiments performed with four well-known datasets showed that the weighted edit distance optimized with the genetic algorithm resulted in an average performance improvement in approximately 20%.

Keywords Drug-target interaction prediction · Similarity-based DTI · SMILES · SMILES similarity · Edit distance · Weighted edit distance · Genetic algorithm

1 Introduction

Developing drugs that interact with arbitrary targets is difficult. Although great progress has been made using artificial intelligence, this is still a challenging problem. The ligand-based approach is a method used to develop new drugs [1]. It analyzes the ligands of a protein that are already known. The underlying assumption is that chemically similar compounds bind to the same or similar proteins and that targets with similar binding sites bind to similar ligands [2]. In the ligand-based approach, information about the similarity between the targets and drugs is required. To measure the similarity between the targets and drugs, we must represent them. Normally, molecular structures are represented in 1D, 2D, or 3D. The 2D and 3D representations use graphs, as shown in the left figure of Fig. 1 [3–8]. The 1D representation commonly uses a simplified molecular input line entry system (SMILES), which is a character string [9, 10]. Figure 1 shows the structure of *cyclohexane* using a 2D graph and 1D SMILES.

SMILES is represented by a string, many string-based distance algorithms are adaptable to SMILES. Some of these have were introduced in [11], including an edit distance [12], normalized longest common subsequence (LCS) [13], combination of LCS models [13], SMILES representation-based string kernel [14], SMILES fingerprint [15] with city block distance [16] or Tanimoto coefficient [4], LINGOsim [17], LINGO-based TF [18], TF-IDF [19], and combination of SIMCOMP with TF-IDF or LINGOsim [11].

Among these algorithms, the edit distance has demonstrated a mediocre performance [11]. The primary limitation in edit distance is its use of the same weights for insertion, deletion, and substitution. This is inappropriate for SMILES and leads to low performance. In the molecular structure, the deletion of an element may differ in impact from the insertion of the same element. We assume that endowing the optimal weights to each operation significantly improves performance. For this purpose, an exhaustive search and genetic algorithm were used in our experiments. An

✉ Il-Seok Oh
isoh@jbnu.ac.kr

In-Hyuk Choi
inhyuk05@jbnu.ac.kr

[1] Division of Computer Science and Engineering, Jeonbuk National University, Jeonju 54896, South Korea

[2] Center for Advanced Image Information Technology, Jeonju 54896, South Korea

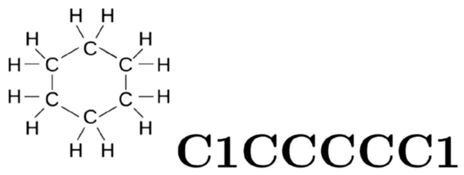

Fig. 1 Graph (left) and SMILES (right) representations of cyclohexane

average performance improvement of approximately 20% was obtained. Our code is available at https://github.com/Sabro98/GA-WeightedEditSimilarity.

The paper presents the following key contributions:

- To measure more accurately the similarity of SMILES representation, we reformulate the edit distance by introducing the optimal weights to three edit operations, insertion, deletion, and substitution.
- We devise a genetic algorithm suitable for finding the optimal weight set. We prove that the genetic algorithm is superior to the exhaustive search both in computation time and accuracy.
- Using four well-known datasets, we present experimental results showing that the weighted edit distance is significantly superior to the conventional edit distance.

In Sect. 2, the concept of drug-target interactions (DTI) and related works are reviewed. In Sect. 3, we introduce the edit distance and its limitations in terms of DTI. The weighted edit distance is proposed as a way of mitigating the limitation. Section 4 covers the datasets and evaluation methods. Section 5 describes an exhaustive search and discusses its limitation. In Sect. 6, we propose a genetic algorithm that is much faster and performs better than the exhaustive search. The experimental results and improvement in the genetic algorithm are presented. Section 7 concludes this paper.

2 Related works

In this section, we will review the literatures related to DTI in two different approaches, feature-based and similarity-based methods. In the similarity-based approach, the similarity measurement methods that is main theme of this paper are reviewed.

Fast and accurate DTI prediction is of great help in quickly screening new drug development. The stream of recent DTI research is divided into the feature-based and similarity-based approaches [20].

The feature-based approach first extracts a feature vector from a pair of drug and target and then classifies the feature vector into pre-specified classes, usually two classes of the positive and negative. A good survey paper is available for this approach [21]. This approach used various machine learning models such as support vector machine (SVM), random forest, and neural networks. Recently, adoption of deep learning models significantly improves the performance. For example, Karimi et al. [22] used a recurrent neural network (RNN). Lee et al. [23] used a convolution neural network (CNN). Lim et al. [24] used a three-dimensional graph and a graph neural network (GNN). Huang et al. [25] proposed a transformer-based DTI prediction methods called as MolTrans. The deep learning methods need sufficiently large datasets for a high accuracy. This requirement is satisfied in some segment of DTI applications.

The similarity-based approach constructs the similarity matrices for drug-drug, drug-target, and target-target between pairs of drugs and targets and decides their interactions based on the similarity matrices. A good survey paper is available for this approach [26]. Yamanashi et al. [27] used the kernel regression method to estimate the similarity. This method has a problem with the bipartite model requiring a massive computational power. Bleakley and Yamanishi [28] overcome this problem by using the bipartite local models. An and Yu [29] constructed multiple heterogeneous networks for drugs, proteins, diseases, and side effects and built a similarity network between drugs and proteins. There are also studies that performed DTI using matrix factorization that factorizes the similarity matrices using singular value decomposition (SVD) [30, 31].

As such, a similarity-based approach has many DTI models. The performance of the models is heavily dependent on the accuracy of the similarity measure. Unfortunately, there is no universal metric that perfectly measures the similarity. For this reason, there are very many algorithms for measuring similarity. Protein sequence embedding represents sequences as points in a high-dimensional space. This makes it possible to measure the similarity of the protein sequence. Väth et al. [32] studied the performance comparison of general-purpose protein sequence embedding methods. Similarity is also measured through metric learning like k-nearest-neighbor classifier [33] and k-means classification [34]. In addition, a study of learning vector quantization (LVQ) that learns vector data through competitive learning has been conducted [35–38]. A study applying relational LVQ to sequential data was conducted [39].

SMILES is a special type of string representation for chemical compounds. Due to this fact, various string similarity algorithms such as those mentioned in Sect. 1 can be used. Recently, a similarity measure combining edit distance, LCS and longest common substring (LCSS) has been studied [40]. Moreover, methods available in the field of NLP can be applied. The DTi2Vec is a good instance of applying the word2vec [41] technique to DTI [42]. A comparative analysis was done for many distance metrics with SMILES representation [11].

To improve the performance of the similarity-based approach, effective similarity metrics should be prioritized. In this study, we improve the conventional edit distance metric by introducing the optimal weights for the edit operations.

3 Edit distance

This section introduces edit distance. Then, a method of measuring similarity based on edit distance is discussed. At the end of this section, we introduce our approach.

3.1 Edit distance in general

The edit distance is the minimum number of operations required to transform the source string $S1$ into the target string $S2$ [12]. The operations comprise insertion, deletion, and substitution. Figure 2 illustrates an example of computing the edit distance, where $S1$ = 'ebiot' and $S2$ = 'edit'. Infinite transformation cases exist, of which three cases are shown in Fig. 2. The topmost case uses two operations, which is the minimum. Therefore, the edit distance ($S1$, $S2$) is 2.

Dynamic programming was used to efficiently compute the edit distance [43]. Many algorithms based on dynamic programming have been developed, and we use the Levenshtein algorithm [12]. Below is the pseudocode for this algorithm.

Algorithm 1: Levenshetein Edit Distance

```
Input: source string $S1$, target string $S2$
Output: edit distance d

1. r ← length of $S1$, c ← length of $S2$
2. Generate an array D of size (r + 1) * (c + 1)
3. D[0, 0..c] ← [0..c] // initialization of row 0.
4. D[0..r, 0] ← [0..r] // initialization of column 0.
5. for j ← 1 to r:
6.   for i ← 1 to c:
7.     if $x_i = y_i$ then rcost ← 0 else rcost ← 1
         // Assign substitute operation cost.
8.     D[j, i] ← min(D[j-1, i] + 1, D[j, i-1]+1, D[j-1, i-1] + rcost)
         // Select minimum value of insert, delete, or substitute operations.
9. d ← D[r, c] // The lower-right element is the edit distance.
10. output d
```

3.2 Similarity of SMILES representations

Equation (1) computes the similarity between the two SMILES representations S1 and S2, using the edit distance [11], where $edit(S1, S2)$ is the Levenshtein edit distance described in Sect. 3.1, n_i, n_d, and n_s are the minimum number of insertions, deletions, and substitutions required to transform S1 to S2, respectively.

$$EditSimilarity(S1, S2) = 1 - \frac{edit(S1, S2)}{MAX(len(S1), len(S2))} \tag{1}$$

where $edit(S1, S2) = n_i + n_d + n_s$.

Suppose $S1$ = "OC(O) = O" and $S2$ = "CCCC(O) = C4". To create $S1$ to $S2$, at least three insertions of 'C' s, two substitutions of 'O' with 'C', and one insertion of '4' are required. Therefore, $edit(S1, S2) = 6$. Because $MAX(len(S1), len(S2)) = 11$, $EditSimilarity(S1, S2) = 1 - \frac{6}{11} = \frac{5}{11}$. This SMILES-based edit distance algorithm assigns the same weight of 1 to all three operations (insertion, deletion, and substitution). However, we must note that the influence on the molecular structure is different when the same element is inserted, deleted, or substituted with other elements.

3.3 Weighted edit similarity

Algorithm 1 treats SMILES as simple strings. However, SMILES is a representation of a molecular structure as a string such that each character has a special meaning. Therefore, an advanced approach is required to compute the similarity between SMILES. Each insertion, deletion, and substitution operation of the same element has a different influence on the structure. Therefore, different weights must be assigned to each operation. Below is the pseudocode of an algorithm that considers element weights.

Algorithm 2: Weighted Edit Distance

```
Input: source string S1, target string S2, insertion weight $w_i$,
       deletion weight $w_d$, substitution weight $w_s$
Output: weighted edit distance d

1. r ← length of S1, c ← length of S2
2. Generate an array D of size (r + 1) * (c + 1)
3. D[0, 0..c] ← [0..c] // initialization of row 0.
4. D[0..r, 0] ← [0..r] // initialization of column 0.
5. for j ← 1 to r:
6.   for i ← 1 to c:
7.     if $x_i = y_i$ then rcost ← 0 else rcost ← $w_r$
         // Assign substitute operation cost.
8.     D[j, i] ← min(D[j-1, i] + $w_i$, D[j, i-1]+$w_d$, D[j-1, i-1] + rcost)
         // Select minimum value of insert, delete, or substitute
         // operations applied weight.
9. d ← D[r, c] // The lower-right element is the weighted edit distance.
10. output d
```

Using Algorithm 2, the similarity can be calculated as in Eq. (2). It applies the weights w_i, w_d, and w_s to calculate the similarity.

$$WeightedEditSimilarity(S1, S2) = 1 - \frac{WEdit(S1, S2)}{MAX(len(S1), len(S2))}$$

where $WEdit(S1, S2) = w_i * n_i + w_d * n_d + w_s * n_s$. (2)

The optimal weights varied from data to data. Therefore, an optimal weight selection should be performed for each dataset.

4 Datasets and evaluation

This section introduces the datasets and our evaluation method.

4.1 Datasets

We used the drugs SMILES data and dataset provided by Yamanishi et al. [27, 43]. It is composed of four datasets: Enzyme, Ion Channels, G Protein-Coupled Receptor (GPCR), and Nuclear Receptor.

4.2 Evaluation

The Weighted Closest Neighbor-Gaussian Interaction Profile (WNN-GIP), which is a state-of-the-art drug-target interaction prediction model [44], was used. WNN-GIP requires three tables: target similarity, drug similarity, and drug-target interaction.

4.2.1 Target similarity table

This indicates the degree of similarity between targets. It has a value in the range of [0, 1] in proportion to the similarity. The table has a size of $n_t * n_t$, where n_t is the number of targets. Table 1 lists the number of targets in each dataset.

4.2.2 Drug similarity table

This table shows the similarity between drugs. It expresses a value in the range of [0, 1] according to the similarity and has a size of $n_d * n_d$, where n_d is the number of targets.

$$\begin{aligned} s &= \text{ebiot} \xrightarrow{d} \text{ebit} \xrightarrow{s} \text{edit} \quad (2) \\ s &= \text{ebiot} \xrightarrow{d} \text{ebit} \xrightarrow{d} \text{eit} \xrightarrow{i} \text{edit} \quad (3) \\ s &= \text{ebiot} \xrightarrow{d} \text{eiot} \xrightarrow{s} \text{edot} \xrightarrow{s} \text{edit} \quad (3) \\ &\dots \end{aligned}$$

Fig. 2 Computation of edit distance

4.2.3 Drug-target interaction table

This contains bipartite information about drug-target interactions. For the pairs of drugs and targets interacting, the table stores 1, and 0 otherwise. The table has a size of $n_d * n_t$, where n_d is the number of drugs, and n_t the number of targets.

When evaluating, for a weight set (w_i, w_d, w_s), the drug-drug similarity table is created using the weighted edit distance algorithm. We evaluate the quality of the weight set through the following process. This is the same as the way how WNN-GIP works. Two kernels are created from drug-drug and target-target similarity tables, respectively. The kernels are modified to be symmetric and positive definite. Another kernel is created from drug-target interaction table. Then first two kernels are combined into one using Kronecker product. Finally, using Regularized Least Squares (RLS) algorithm, the model predicts drug-target interactions [44]. The goodness of the weight set is measured by area under curve of receiver operating characteristics (AUC-ROC). It is close to 1 if a good prediction is made, and close to 0 otherwise.

The WNN-GIP algorithm has a time complexity of $O\left(n_d^3 + n_t^3\right)$ when the number of drugs is n_d and the number of targets is n_t [44]. This is because RLS and Kronecker products dominate the entire time.

5 Exhaustive search

Exhaustive search (ES) is a method that examines all cases. In this section, we first introduce the process, and results of using ES in our experiments. Finally, the limitations of this method are described.

Table 1 Number of components included in the drug-target interaction datasets of Yamanishi et al.

Dataset	Drugs	Targets	Interactions	Avg. length (Drugs)
Enzyme	445	664	2926	55
Ion Channels	210	204	1476	23
GPCR	223	95	635	26
Nuclear Receptor	54	26	90	8

5.1 Exhaustive search for weighted edit distance

To apply the weighted edit similarity of Eq. (2) determining the optimal weight combination is necessary. Because the weights are real numbers, the possible combinations are infinite. In this study, the optimal solution is determined by limiting and discretizing the ranges of w_i, w_d, and w_r. The weight range was set to [0, 1], and each weight was discretized into 40 intervals (one interval is 0.025). Therefore, 64,000 cases were investigated.

Table 2 lists the AUC-ROCs obtained using the optimal weight combinations. The Nuclear Receptor dataset demonstrates a high performance when w_s is high and w_d is low. The other datasets show high performances when w_s is low. When w_d and w_i are similar, Ion Channels and GPCR show high performances. Enzyme achieved a high performance when w_i is lower than w_d. This implies that the optimal weight combinations are highly dependent on the datasets.

We drew a surface mesh chart to visually examine the performance changes. Figure 3 shows the performance surface for the Enzyme dataset by fixing the optimal w_i. Figure 4 illustrates the performance surfaces of Nuclear Receptor with the optimal w_d, and Fig. 5 for GPCR with the optimal w_s. These graphs show that the performances change smoothly according to the change in weight combinations. The results demonstrate a need for optimizing the weights for each dataset.

5.2 Comparisons with original algorithm

Table 3 compares the algorithm of Eq. (2) with the conventional algorithm of Eq. (1) using the optimal weights found by ES.

The Enzyme and GPCR datasets showed a high improvement of over 30%. Ion Channels improved by more than 10%, and Nuclear Receptor a slight improvement. These improvements indicate that the influences of the three operations are different in the SMILES-based edit distance. However, in terms of computation time, the weighted SMILES similarity method requires an optimization process that is not required in the conventional algorithm. In particular, the chemical properties of each dataset were different, and performing optimization for each dataset is a burden.

Table 2 Optimal weight combinations and corresponding AUC-ROC for each dataset

Dataset	w_i	w_d	w_s	AUC-ROC
Enzyme	0.675	1.000	0.075	0.9179
Ion Channels	1.000	0.925	0.700	0.9505
GPCR	1.000	1.000	0.825	0.9096
Nuclear Receptor	0.725	0.450	0.950	0.9092

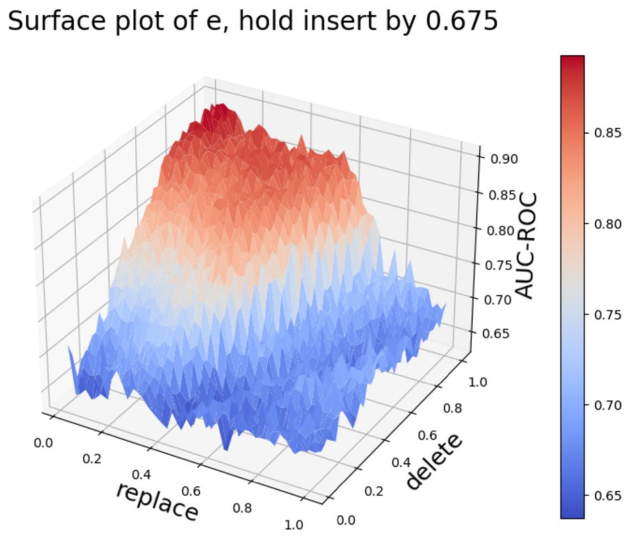

Fig. 3 Enzyme, keeping w_i at 0.675

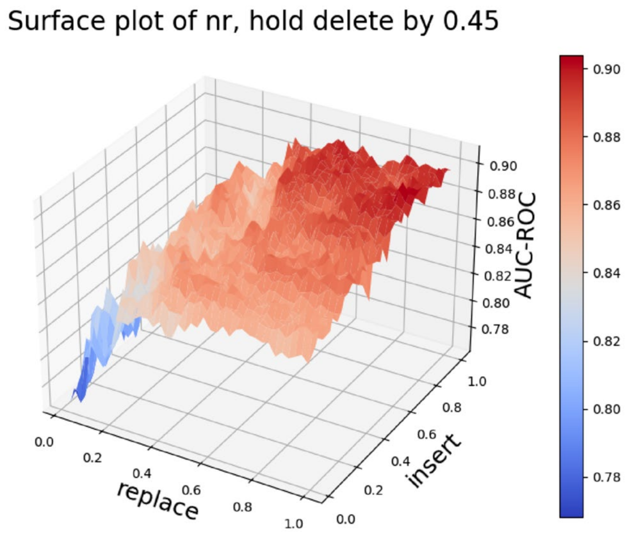

Fig. 4 Nuclear Receptor, keeping w_d at 0.45

5.3 Limitations of exhaustive search

The first limitation is that the number of cases to be examined is unknown. Equation (3) represents the time complexity of the ES. In this case, L indicates $MAX(len(S_1), len(S_2))$, N the number of intervals, and $O(n_d^3 + n_t^3)$ the evaluation time for each case using WNN-GIP where n_d is the number of drugs and n_t is the number of targets for each datasets [44]. L is different for each dataset and are proportional to the average length of the SMILES string and number of drugs, respectively. Table 1 lists the drugs' average length of the SMILES string and number of samples for each dataset.

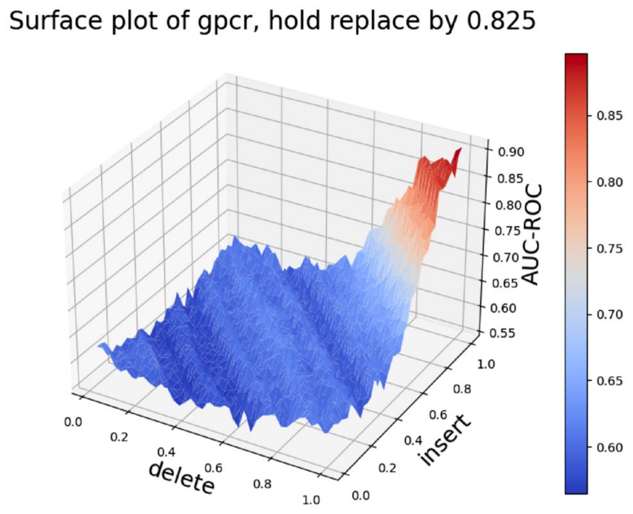

Fig. 5 GPCR, keeping w_s at 0.825

$$O\left(N^3\left(L^2 + n_d^3 + d_t^3\right)\right) \quad (3)$$

In our experiment, we set $N=40$. Therefore, we examined 64,000 cases. The optimization process required 18.7 h for Enzyme, 3.8 h for Ion channels, 3.4 h for GPCR and 0.4 h for Nuclear Receptor.

The second limitation is that the real values of the weights are discretized to apply the ES. For example, if the optimal w_i is 0.03, 0.025 is the output of w_i because the interval of the discretization is 0.025 in our experiment. The accuracy increased as the interval becomes finer with a larger N, but the search space grows rapidly as N^3. This implies that, if 40 intervals are increased to 80 intervals, the calculation time increases by eight times.

We need an efficient search algorithm that uses the real values of the weights.

6 Genetic algorithm

This section first introduces the genetic algorithm (GA) and describes why this method was chosen. Next, we introduce the operation designed for weights selection and the hyperparameters used. Finally, we conclude with an analysis of the results.

6.1 Introduction to genetic algorithm

In nature, genes are passed down from the parents to the offspring over several generations. In this process, unfavorable survival genes become extinct. Conversely, genes favorable survival genes are conserved and transmitted to future generations. A genetic algorithm mimics this process. That is, this process is a computational model based on a natural process and a technique used to solve the optimization problem. The GA maintains a set of chromosomes in a population. A chromosome consists of genes, and its fitness is evaluated before entering the population. The mechanism is as follows: select the parent, generate offspring by crossing over, mutate the offspring, and replace the parents with offspring. The GA repeats this and finally selects the optimal chromosome with the best fitness. Because our problem optimizes only three parameters (insertion, deletion, and substitution weights), we can quickly determine the optimal solution using the GA.

To apply the GA to our problem, we must design a chromosome representation, fitness evaluation method, and selection, crossover, mutation, and replacement operations suitable for the problem. In addition, to prevent premature convergence, the hyperparameters that influence the selection pressure should be properly set.

Various methods exist for multivariate optimization. Many methods are based on differentiation, and a representative example is stochastic gradient descent (SGD), which is mainly used in neural networks [45]. However, it is very difficult to define the gradient of the evaluation function used in this study. We can use other real number optimization methods that do not use gradient. However, the GA has been proven to near optimal values in many domains [46]. Our primary aim is to prove that the GA is also suitable in optimizing the weighted edit distance in DTI application area.

Table 3 AUC-ROC comparison of Eqs. (1) and (2)

Dataset	AUC-ROC of Eq. (1)	AUC-ROC of Eq. (2) with optimal weights	Improvement ratio
Enzyme	0.6785	0.9179	35.3%
Ion Channels	0.858	0.9505	10.7%
GPCR	0.6858	0.9096	32.6%
Nuclear Receptor	0.9024	0.9092	0.7%

6.2 Components of genetic algorithm

The process of GA can be found in Algorithm 3. We designed the necessary components in this algorithm and introduce each of them here.

Algorithm 3: Genetic algorithm for optimizing the weights of edit distance operations

Input: size of population P, maximum generation T

Output: Optimal weights combination

```
1. population ← empty **list**
2. **for** $P$ times:
3.    chromosome ← **generate** chromosome
4.    **append** chromosome to population
5. **for** $T$ times:
6.    parents ← **select** two parents $p_1$ and $p_2$ from population
7.    offspring ← **crossover** parents
8.    **mutate** offspring
9.    **evaluate** offspring
10.   **replace** one of parents to offspring
11. **output** the fittest chromosome so far
```

6.2.1 Chromosome

In our problem, the optimization targets are the weights w_i, w_d, and w_s; therefore, the chromosome is expressed as in Eq. (4). For example, w_i=0.12, w_d=0.43, and w_s=0.55 are represented by the chromosome of [0.12, 0.43, 0.55]. Genes are real numbers within the range (0, 1]. Owing to this property, discretization is not required, making more accurate measurements possible.

$$\text{chromosome} = \left[w_i, w_d, w_r\right] \tag{4}$$

To evaluate the fitness of a chromosome, the AUC-ROC was evaluated using the WNN-GIP by applying the genes to Eq. (2).

To secure population diversity, the genes are initialized by random values in the range of (0, 1].

6.2.2 Selection

Selection should ensure a high probability of parents with higher fitness. For this purpose, we adopted a tournament algorithm. This algorithm selects two chromosomes from the population, and, if the random number is lower than threshold P_t, the chromosome with the higher fitness is selected; the lower chromosome is selected otherwise. This process repeats until two different chromosomes are selected. The value of P_t is greater than 0.5. The larger the P_t is, the larger the selection pressure. A large selection pressure means that the algorithm is more favorable to the fitter chromosome.

6.2.3 Crossover and mutation

An arithmetic crossover is used to generate the offspring from the selected parents. The genes of the offspring are generated by averaging the parent genes. For example, if the two parents are [0.1, 0.4, 0.2] and [0.3, 0.1, 0.4], the offspring is [0.2, 0.25, 0.3].

Table 4 Hyperparameter values used in the GA

Hyperparameter	Value
P_T	0.75
P_M	0.025
R_M	0.15
Population size	27
Maximum generation	2000

The mutation is applied to the offspring. This is important for maintaining the population diversity. A gene is mutated if a random number within [0, 1] is smaller than the threshold P_M. The P_M is a small value; we set it to be 0.025. The smaller the P_M is, the larger the selection pressure. The gene is mutated by adding a random value in the range $[-R_M, R_M]$ to the gene. Finally, the offspring fitness is evaluated using the WNN-GIP.

6.2.4 Replacement

To maintain the population size, a chromosome in the population must be replaced with the offspring. We used a simple scheme that replaces the inferior parent with the offspring.

6.2.5 Hyperparameters

The GA has a few hyperparameters that influence the selection pressure. They should be set appropriately to balance exploitation and exploration. When GA is biased toward exploitation, premature convergence occurs, and the performance remains low. Convergence requires a lot of time when the GA is biased toward exploration. In this study, appropriate hyperparameter values were set through an experiment. Table 4 lists these values.

To obtain the optimal values of hyperparameters, we conducted repetitive experiments by varying four hyperparameters ($P_T, P_M, R_M, \textit{population size}$) except for the maximum generation.

6.3 Result

6.3.1 Experiment result

We performed the experiment five times for each dataset. Table 5 lists the optimal weights and AUC-ROC of the best solution for each of the four datasets.

Table 6 shows each optimal AUC-ROC using Eq. (1), Eq. (2) with ES, and Eq. (2) with the GA. It also shows the performance improvement by comparing Eq. (1) to Eq. (2) with GA.

Table 5 Optimal weights and AUC-ROC for each dataset using Eq. (2) with the GA

Dataset	Insert	Delete	Substitute	AUC-ROC
Enzyme	0.7973	0.826	0.0739	0.9200
Ion Channels	0.1938	0.9112	0.6745	0.9537
GPCR	0.9565	0.9935	0.7671	0.9125
Nuclear Receptor	0.7156	0.4692	0.9630	0.9116

Table 6 Optimal AUC-ROC each Eq. (1), Eq. (2) with ES and GA, and final improvement ratio

Dataset	Equation (1)	Equation (2) with ES	Equation (2) with GA	Final improvement (%)
Enzyme	0.6785	0.9179	0.9200	35.6
Ion Channels	0.8580	0.9505	0.9537	11.1
GPCR	0.6858	0.9096	0.9125	33.0
Nuclear Receptor	0.9024	0.9092	0.9116	1.0

The GA shows similar or slightly superior to ES. The reason for the superiority is that ES determines the optimal value among 40 discretized intervals, whereas the GA determines the optimal value more precisely among real values within (0, 1].

As a result, the weighted edit distance optimized with the genetic algorithm resulted in an average performance improvement of approximately 20%.

6.3.2 Time complexity of GA

Let $MAX\left(len\left(S_1\right), len\left(S_2\right)\right) = L$, the number of generations be G, and the fitness evaluation time using WNN-GIP be $O\left(n_d^3 + n_t^3\right)$ where n_d is the number of drugs and n_t is the number of targets for each datasets [44]. Equation (5) represents the time complexity of GA.

$$O\left(G\left(L^2 + n_d^3 + n_t^3\right)\right) \tag{5}$$

The difference from the time complexity of the ES is that the number of cases is a fixed constant. In ES, the number of cases increases exponentially with the degree of discretization. In the GA, the optimal value can be obtained at a fixed constant. The GA is faster than the ES by $\frac{N^3}{G}$ times. In this experiment, $N = 64{,}000$ and $G = 2000$ were set, and thus, the GA determines the optimal value approximately 32 times faster. The difference increases as the discretization in the ES becomes finer.

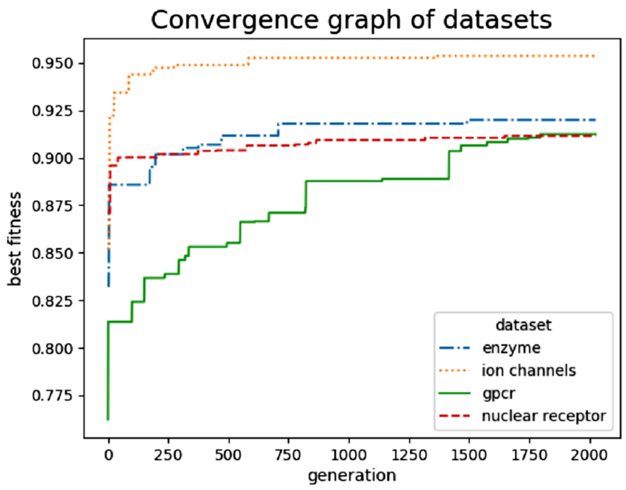

Fig. 6 Convergence graphs of the GA for four datasets

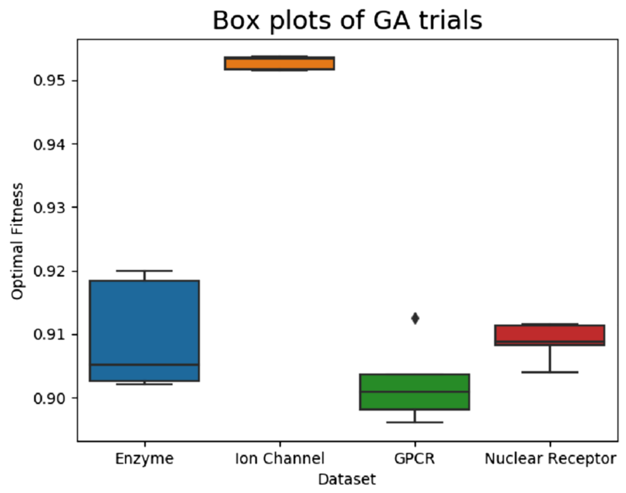

Fig. 7 Box plots of the five GA trials and their corresponding fitness

6.3.3 Convergences of GA

Figure 6 shows the convergence graphs of the four datasets.

The graphs show that convergence was achieved around the 1500th generation for all four datasets. The GCPR dataset was the slowest to converge.

Figure 7 shows the boxplots of the five AUC-ROCs for each of the four datasets. The Ion channel dataset was the most stable, whereas the Enzyme dataset demonstrates the highest variations.

Figure 8 illustrates the boxplots of the optimal weights of the three operations for each of the four datasets. For the Enzyme dataset, the substitution weight was low, whereas the insertion and deletion weights were high. In contrast, for the Nuclear Receptor dataset, the substitution weight was higher than the insertion and deletion weights.

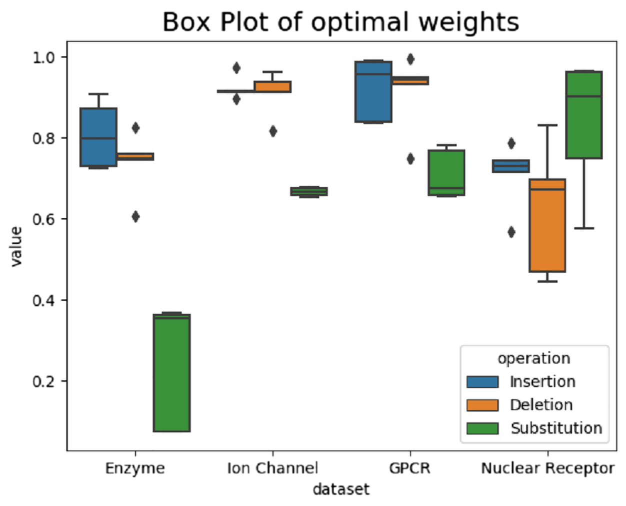

Fig. 8 Box plots of the five GA trials and their optimal weights

7 Conclusion

In this study, we improved the SMILES-based edit distance algorithm by adopting optimized weights for insertion, deletion, and substitution operations. Because the optimal weights vary from data to data, we determined the data-dependent optimal weights using a genetic algorithm. Performance improvements were observed for the Enzyme, Ion Channel, GPCR, and Nuclear Receptor datasets. We determined that the GA is much faster than the ES algorithm and superior in accuracy owing to its real value encoding of operation weights. We expect that the SMILES-based compound similarities optimized by the proposed GA will improve the performance and time for prescreening in the development of new drugs.

Data availability statement The datasets generated during and/or analyzed during the current study are available in this repository, https://github.com/Sabro98/GA-WeightedEditSimilarity/tree/master/data.

Declarations

Conflict of interest The authors did not receive support from any organization for the submitted work. And all authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

References

1. Gohlke H, Hendlich M, Klebe G (2000) Knowledge-based scoring function to predict protein-ligand interactions. J Mol Biol 295(2):337–356. https://doi.org/10.1006/jmbi.1999.3371
2. Tabei Y, Pauwels E, Stoven V, Takemoto K, Yamanishi Y (2012) Identification of chemogenomic features from drug–target interaction networks using interpretable classifiers. Bioinformatics 28(18):487–494
3. Sawada R, Kotera M, Yamanishi Y (2014) Benchmarking a wide range of chemical descriptors for drug-target interaction prediction using a chemogenomic approach. Mol Inf 33(11–12):719–731
4. Willett P, Barnard JM, Downs GM (1998) Chemical similarity searching. J Chem Inf Comput Sci 38(6):983–996
5. Schuffenhauer A, Gillet VJ, Willett P (2000) Similarity searching in files of three-dimensional chemical structures: analysis of the bioster database using two-dimensional fingerprints and molecular field descriptors. J Chem Inf Comput Sci 40(2):295–307
6. Helguera AM, Combes RD, González MP, Cordeiro M (2008) Applications of 2d descriptors in drug design: a dragon tale. Curr Top Med Chem 8(18):1628–1655
7. Hong H, Xie Q, Ge W, Qian F, Fang H, Shi L, Su Z, Perkins R, Tong W (2008) Mold2, molecular descriptors from 2d structures for chemoinformatics and toxicoinformatics. J Chem Inf Model 48(7):1337–1344
8. Kombo DC, Tallapragada K, Jain R, Chewning J, Mazurov AA, Speake JD, Hauser TA, Toler S (2013) 3d molecular descriptors important for clinical success. J Chem Inf Model 53(2):327–342
9. Weininger D (1998) Smiles, a chemical language and information system. 1. Introduction to methodology and encoding rules. J Chem Inf Comput Sci 28(1):31–36. https://doi.org/10.1021/ci00057a005
10. Weininger D, Weininger A, Weininger JL (1989) Smiles. 2. Algorithm for generation of unique smiles notation. J Chem Inf Comput Sci 29(2):97–101. https://doi.org/10.1021/ci00062a008
11. Öztürk H, Ozkirimli E, Özgür A (2016) A comparative study of SMILES-based compound similarity functions for drug-target interaction prediction. BMC Bioinform. https://doi.org/10.1186/s12859-016-0977-x
12. Levenshtein VI (1996) Binary codes capable of correcting deletions, insertions, and reversals. Sov Phys Doklady 10(8):707–710
13. Islam A, Inkpen D (2008) Semantic text similarity using corpus-based word similarity and string similarity. ACM Trans Knowl Discov Data 2(2):1–25
14. Cao DS, Zhao JC, Yang YN, Zhao CX, Yan J, Liu S, Hu QN, Xu QS, Liang YZ (2012) In silico toxicity prediction by support vector machine and smiles representation-based string kernel. SAR QSAR Environ Res 23(1–2):141–153
15. Schwartz J, Awale M, Reymond JL (2013) Smifp (smiles fingerprint) chemical space for virtual screening and visualization of large databases of organic molecules. J Chem Inf Model 53(8):1979–1989. https://doi.org/10.1021/ci400206h
16. Krause EF (1986) An adventure in non-euclidean geometry. Dover Publication, New York
17. Vidal D, Thormann M, Pons M (2005) LINGO, an efficient holographic text based method to calculate biophysical properties and intermolecular similarities. J Chem Inf Model. https://doi.org/10.1021/ci0496797
18. Luhn HP (1957) A statistical approach to mechanized encoding and searching of literary information. IBM J Res Dev 1(4):309–317. https://doi.org/10.1147/rd.14.0309
19. Jones KS (1972) A statistical interpretation of term specificity and its application in retrieval. J Doc 28:11–21
20. Bagherian M, Sabeti E, Wang K et al (2020) Machine learning approaches and databases for prediction of drug–target interaction: a survey paper. Brief Bioinform. https://doi.org/10.1093/bib/bbz157
21. Sachdev K, Gupta MK (2019) A comprehensive review of feature based methods for drug target interaction prediction. J Biomed Inform. https://doi.org/10.1016/j.jbi.2019.103159

22. Karimi M, Wu D, Wang Z et al (2019) DeepAffinity: interpretable deep learning of compound–protein affinity through unified recurrent and convolutional neural networks. Bioinformatics. https://doi.org/10.1093/bioinformatics/btz111
23. Lee I, Keum J, Nam H (2019) DeepConv-DTI: prediction of drug-target interactions via deep learning with convolution on protein sequences. PLoS Comput Biol 15(6):e100719. https://doi.org/10.1371/journal.pcbi.1007129
24. Lim J, Ryu S, Park K et al (2019) Predicting drug–target interaction using a novel graph neural network with 3D structure-embedded graph representation. J Chem Inf Model. https://doi.org/10.1021/acs.jcim.9b00387
25. Huang K, Xiao C, Glass LM et al (2020) MolTrans: molecular Interaction Transformer for drug–target interaction prediction. Bioinformatics. https://doi.org/10.1093/bioinformatics/btaa880
26. Wang C, Kurgan L (2020) Survey of similarity-based prediction of drug-protein interactions. Curr Med Chem. https://doi.org/10.2174/0929867326666190808154841
27. Yamanishi Y, Araki M, Gutteridge A, Honda W, Kanehisa M (2008) Prediction of drug-target interaction networks from the integration of chemical and genomic spaces. Bioinformatics 24(13):232–240. https://doi.org/10.1093/bioinformatics/btn162
28. Bleakley K, Yamanishi Y (2009) Supervised prediction of drug–target interactions using bipartite local models. Bioinformatics. https://doi.org/10.1093/bioinformatics/btp433
29. An Q, Yu L (2021) A heterogeneous network embedding framework for predicting similarity-based drug-target interactions. Brief Bioinform. https://doi.org/10.1093/bib/bbab275
30. Zheng X, Ding H, Mamitsuka H et al (2013) Collaborative matrix factorization with multiple similarities for predicting drug-target. https://doi.org/10.1145/2487575.2487670
31. Ezzat A, Zhao P, Wu M et al (2017) Drug–target interaction prediction with graph regularized matrix factorization. IEEE/ACM Trans Comput Biol Bioinform. https://doi.org/10.1109/TCBB.2016.2530062
32. Väth P, Münch M, Raab C et al (2022) PROVAL: a framework for comparison of protein sequence embeddings. J Comput Math Data Sci. https://doi.org/10.1016/j.jcmds.2022.100044
33. Cover T, Hart P (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory. https://doi.org/10.1109/TIT.1967.1053964
34. Lloyd S (1982) Least squares quantization in PCM. IEEE Trans Inf Theory. https://doi.org/10.1109/TIT.1982.1056489
35. Biehl M, Bunte K, Schneider P (2013) Analysis of flow cytometry data by matrix relevance learning vector quantization. PLoS ONE. https://doi.org/10.1371/journal.pone.0059401
36. Kirstein S, Wersing H, Gross H-M et al (2012) A life-long learning vector quantization approach for interactive learning of multiple categories. Neural Netw. https://doi.org/10.1016/j.neunet.2011.12.003
37. Backhaus A, Seiffert U (2014) Classification in high-dimensional spectral data: accuracy vs. interpretability vs. model size. Neurocomputing. https://doi.org/10.1016/j.neucom.2013.09.048
38. Hammer B, Hofmann D, Schleif F-M et al (2014) Learning vector quantization for (dis-)similarities. Neurocomputing. https://doi.org/10.1016/j.neucom.2013.05.054
39. Mokbel B, Paassen B, Schleif F-M et al (2015) Metric learning for sequences in relational LVQ. Neurocomputing. https://doi.org/10.1016/j.neucom.2014.11.082
40. Zhang S, Hu Y, Bian G (2017) Research on string similarity algorithm based on Levenshtein Distance. https://doi.org/10.1109/IAEAC.2017.8054419
41. Mikolov T, Chen K, Corrado G et al (2013) Efficient estimation of word representations in vector space. https://arxiv.org/abs/1301.3781
42. Thafar MA, Olayan RS, Albaradei S et al (2021) DTi2Vec: drug–target interaction prediction using network embedding and ensemble learning. J Cheminform. https://doi.org/10.1186/s13321-021-00552-w
43. Thomas H (2009) Cormen, introduction algorithms, 3rd edn. MIT Press, Cambridge
44. van Laarhoven T, Marchiori E (2013) Predicting drug-target interactions for new drug compounds using a weighted nearest neighbor profile. PLoS ONE 8(6):66952. https://doi.org/10.1371/journal.pone.0066952
45. Ruder S (2016) An overview of gradient descent optimization algorithms. https://arxiv.org/abs/1609.04747
46. Katoch S, Chauhan SS, Kumar V (2020) A review on genetic algorithm: past, present, and future. Multimed Tools Appl. https://doi.org/10.1007/s11042-020-10139-6