International Journal of
*Molecular Sciences*

MDPI

*Article*

# Ligand-Based Virtual Screening Based on the Graph Edit Distance

**Elena Rica *, Susana Álvarez and Francesc Serratosa**

Departament d'Enginyeria Informàtica i Matemàtiques, Universitat Rovira i Virgili, 43007 Tarragona, Spain; susana.alvarez@urv.cat (S.Á.); francesc.serratosa@urv.cat (F.S.)
* Correspondence: mariaelena.rica@urv.cat

**Abstract:** Chemical compounds can be represented as attributed graphs. An attributed graph is a mathematical model of an object composed of two types of representations: nodes and edges. Nodes are individual components, and edges are relations between these components. In this case, pharmacophore-type node descriptions are represented by nodes and chemical bounds by edges. If we want to obtain the bioactivity dissimilarity between two chemical compounds, a distance between attributed graphs can be used. The Graph Edit Distance allows computing this distance, and it is defined as the cost of transforming one graph into another. Nevertheless, to define this dissimilarity, the transformation cost must be properly tuned. The aim of this paper is to analyse the structural-based screening methods to verify the quality of the Harper transformation costs proposal and to present an algorithm to learn these transformation costs such that the bioactivity dissimilarity is properly defined in a ligand-based virtual screening application. The goodness of the dissimilarity is represented by the classification accuracy. Six publicly available datasets—CAPST, DUD-E, GLL&GDD, NRLiSt-BDB, MUV and ULS-UDS—have been used to validate our methodology and show that with our learned costs, we obtain the highest ratios in identifying the bioactivity similarity in a structurally diverse group of molecules.

**Keywords:** virtual screening; molecular similarity; extended reduced graph; structure activity relationships; machine learning; graph edit distance

check for updates

**Citation:** Rica, E.; Álvarez, S.; Serratosa, F. Ligand-Based Virtual Screening Based on the Graph Edit Distance . *Int. J. Mol. Sci.* **2021**, *22*, 12751. https://doi.org/10.3390/ijms222312751

Academic Editor: Gerard Pujadas

Received: 5 October 2021
Accepted: 13 November 2021
Published: 25 November 2021

## 1. Introduction

The high increase in chemical compounds data has created the need to develop computational tools to reduce the drug synthesis and drug test cycle runtimes. When activity data are analysed, these tools are required to generate new models for virtual screening techniques [1–3]. In the drug discovery process, virtual screening is a common step in which computational techniques are used to search and filter chemical compounds in databases. Basically, there are two main types of methods in the virtual screening: ligand-based virtual screening (LBVS) [4] and structure-based virtual screening (SBVS) [5]. In this work, we focus only in LBVS applications. The idea of the LBVS method is to predict the unknown activity of new molecules [6] using the information about the known activity of some molecules—specifically, their behaviour as ligands that bind to a receptor.

Some LBVS approaches are shape-based similarity [7], pharmacophore mapping [6], fingerprint similarity and machine learning methods [8]. According to [9], structurally similar molecules are presumed to have similar activity properties, then, in the context of LBVS methods, the chosen molecular similarity metric is important because it can determine the success of a virtual screening method to discover proper drug candidates. Various similarity methods are used in several applications [10–14].

To compute molecular similarity, it is necessary to define a distance and define a descriptor representing the molecule. Hundreds of molecular descriptors have been reported in the literature [15]. For instance, one-dimensional descriptors include general molecular properties, such as size, molecular weight, logP or dipole moment, or BCUT parameters [16–19]. Two-dimensional descriptors generate an array of representations of the molecules by simplifying the atomic information within them, such as 2D fingerprints [20–22]. Finally, three-dimensional descriptors use 3D information, such as molecular volume [23,24]. Other existing methods, instead of representing molecules by an N-dimensional vector, use relational structures, such as trees [25] or graphs [26,27]. Regarding the molecule representation by graphs, some methods represent compounds using reduced graphs [28–31] and other ones, such as extended reduced graphs (ErGs) [28]. Reduced graphs group atomic sub-structures that have related features, e.g., pharmacophoric features, ring systems, hydrogen-bonding or others. Moreover, ErGs are an extension of reduced graphs that introduce some changes to better represent shape, size and pharmacophoric properties of the molecules. The method presented in [28] has demonstrated its use as a powerful tool for virtual screening.

To perform reduced graph comparisons, three different similarity measures have been used: In [28,29,32], they map the reduced graphs into a 2D fingerprint. In [33], they map reduced graphs into sets of shortest paths. Finally, in [34,35], they perform the comparison on the graphs using the Graph Edit Distance (GED). GED considers the distance between two graphs as the minimum cost of modifications required to transform one graph into another. Each modification can be one of the following six operations: insertion, deletion and substitution of both nodes and edges in the graph [36–38]. The main goal of this paper is to present an algorithm that learns the edit costs in the GED to improve the classification ratio returned by the system when the Harper costs were used.

In an initial paper [34], the edit costs were imposed and extracted from [33], given the chemical expertise of the authors and considering the different node and edge types. Later, in [35], authors presented an algorithm for optimising those edit costs based on minimising the distance between correctly classified molecules and maximising the distance between incorrectly classified molecules. That work was inspired in a similar one carried out by Birchall et al. [39], in which the authors optimise the transformation costs of a String Edit Distance method to compare molecules using reduced graphs.

The main problem of the algorithm in [35] was the huge computational cost, which depends on the number of edit costs to be optimised. Thus, for practical reasons, in the experimental section in [35], they presented four experiments, in which only one edit cost was optimised in each experiment. They imposed the other costs (126 in total) to be the ones defined in [33]. In contrast, starting from the costs defined by [33], our method learns the whole edit costs of the GED to compare molecules with a lower computational cost obtaining higher classification ratios in the ligand-based screening application, as shown in the experimental section.

The outline of this paper is as follows. First, materials and methods are explained in detail, including the datasets, the GED and the learning algorithm. Second, the results of the practical experiments are described and discussed. Third and finally, a general discussion about the method and the results is presented.

## 2. Materials and Methods

### *2.1. Datasets*

In this study, we have used six available public datasets: ULS-UDS [40], GLL&GDD [41], CAPST [42], DUD-E [43], NRLiSt-BDB [44] and MUV [45]. All these datasets had been formatted and standardized by the LBVS benchmarking platform developed by Skoda and Hoksza [46]. The datasets are composed of various groups of active and inactive molecules arranged according to the purpose of a target. Each group is split in two halves, the test and train sets, which are required when using machine learning methods. The train set is used to optimize the transformation costs, and the test set is used to evaluate the classification ratio. The targets of the datasets are shown in Table 1. In our experimentation, we have taken a subset of the first 100 active molecules and 100 of the first inactive molecules per target. Some datasets have less than 100 active molecules; in this case, all active molecules are taken and also the same number of inactive molecules.

**Table 1.** Datasets used for the experiments. Each dataset on the left contains the targets on the right.

| Dataset | Used Targets |
|---|---|
| CAPST | CDK2, CHK1, PTP1B, UROKINASE |
| DUD-E | COX2, DHFR, EGFR, FGFR1, FXA, P38, PDGFRB, SRC, AA2AR |
| GLL&GDD | 5HT1A_Agonist, 5HT1A_Antagonist, 5HT1D_Agonist, 5HT1D_Antagonist, 5HT1F_Agonist, 5HT2A_Antagonist, 5HT2B_Antagonist, 5HT2C_Agonist, 5HT2C_Antagonist, 5HT4R_Agonist, 5HT4R_Antagonist, AA1R_Agonist, AA1R_Antagonist, AA2AR_Antagonist, AA2BR_Antagonist, ACM1_Agonist, ACM2_Antagonist, ACM3_Antagonist, ADA1A_Antagonist, ADA1B_Antagonist, ADA1D_Antagonist, ADA2A_Agonist, ADA2A_Antagonist, ADA2B_Agonist, ADA2B_Antagonist, ADA2C_Agonist, ADA2C_Antagonist, ADRB1_Agonist, ADRB1_Antagonist, ADRB2_Agonist, ADRB2_Antagonist, ADRB3_Agonist, ADRB3_Antagonist, AG2R_Antagonist, BKRB1_Antagonist, BKRB2_Antagonist, CCKAR_Antagonist, CLTR1_Antagonist, DRD1_Antagonist, DRD2_Agonist, DRD2_Antagonist, DRD3_Antagonist, DRD4_Antagonist, EDNRA_Antagonist, EDNRB_Antagonist, GASR_Antagonist, HRH2_Antagonist, HRH3_Antagonist, LSHR_Antagonist, LT4R1_Antagonist, LT4R2_Antagonist, MTR1A_Agonist, MTR1B_Agonist, MTR1L_Agonist, NK1R_Antagonist, NK2R_Antagonist, NK3R_Antagonist, OPRD_Agonist, OPRK_Agonist, OPRM_Agonist, OXYR_Antagonist, PE2R1_Antagonist, PE2R2_Antagonist, PE2R3_Antagonist, PE2R4_Antagonist, TA2R_Antagonist, V1AR_Antagonist, V1BR_Antagonist, V2R_Antagonist |
| MUV | 466, 548, 600, 644, 652, 689, 692, 712, 713, 733, 737, 810, 832, 846, 852, 858, 859 |
| NRLiSt_BDB | AR_Agonist, AR_Antagonist, ER_Alpha_Agonist, ER_Alpha_Antagonist, ER_Beta_Agonist, FXR_Alpha_Agonist, GR_Agonist, GR_Antagonist, LXR_Alpha_Agonist, LXR_Beta_Agonist, MR_Antagonist, PPAR_Alpha_Agonist, PPAR_Beta_Agonist, PPAR_Gamma_Agonist, PR_Agonist, PR_Antagonist, PXR_Agonist, RAR_Alpha_Agonist, RAR_Beta_Agonist, RAR_Gamma_Agonist, RXR_Alpha_Agonist, RXR_Alpha_Antagonist, RXR_Gamma_Agonist, VDR_Agonist |
| ULS-UDS | 5HT1F_Agonist, MTR1B_Agonist, OPRM_Agonist, PE2R3_Antagonist |

### 2.2. Molecular Representation

Reduced graphs are compact representations of chemical compounds, in which the main information is condensed in feature nodes to give abstractions of the chemical structures. Different versions of reduced graphs have been presented [26,28,30,32,33] that depend on the features they summarise or the use that is given to them. In the virtual screening context, the structures are reduced to track down features or sub-structures that have the potential to interact with a specific receptor and, at the same time, try to keep the topology and spatial distribution of those features. Figure 1 presents an example of molecule reduction.

### 2.3. The Proposed Method

We explain our proposed method in the next three subsections. The first one explains the classification of compounds based on structural information; in the second one, we explain the learning algorithm; and in the third one, we detail the code of the algorithm.

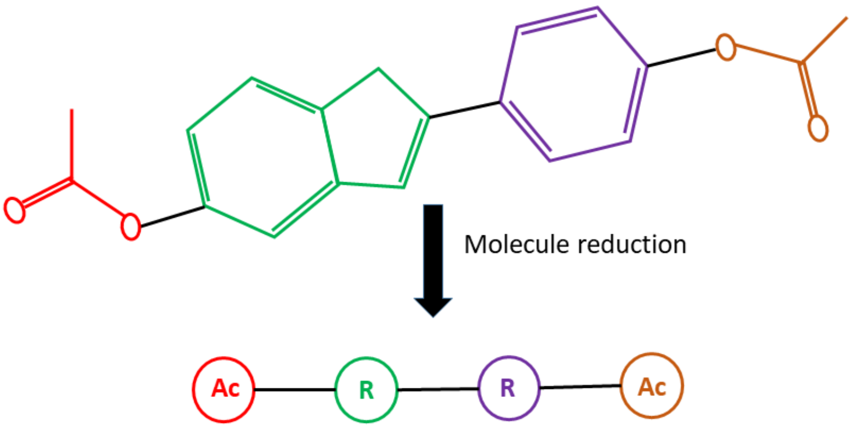

**Figure 1.** Example of molecule reduction using ErG. The original molecule is on the top and its ErG representation is below. Elements of the same colour on the top are reduced to nodes on the ErG. R: Ring system, Ac: Acyclic components.

### 2.3.1. Classification of Molecules

Once the molecules are represented as ErGs, we can compare them by means of the Graph Edit Distance (GED) [47,48]. The GED is defined as the minimum cost of transformations required to convert one graph into the other. Thus, in our application, it is the cost to transform an ErG into the other one. These modifications are called edit operations, and six of them have been defined: insertion, deletion and substitution of both nodes and edges. To classify a molecule, we apply the Nearest Neighbour (NN) strategy that consists of calculating the GED between this molecule and the other ones, which we know their class, and predicting its class (active or inactive) to be the class of the nearest molecule. In the case the molecule is equidistant from more than one classified molecule, the method arbitrarily selects one of the closest molecules.

Edit costs have been introduced to quantitatively evaluate each edit operation. The aim of the edit costs is to designate a coherent transformation penalty in proportion to the extent to which it modifies the transformation sequence. For instance, when ErGs are compared, it makes sense that the cost of substituting a “hydrogen-bond donor" feature with a joint “hydrogen-bond donor-acceptor" feature be less heavily penalized than the cost of substituting a “hydrogen-bond donor" feature with an “aromatic ring" system. Similarly, inserting a single bond should have a lower penalization cost than inserting a double bond, and so on. In a previous work [34], the edit costs proposed by Harper et al. [33] were used. The node and edge descriptions are shown in Table 2, and the specific costs proposed by Harper et al. are exposed in Tables 3 and 4.

The final edit cost for a given transformation sequence is obtained by adding up all of the individual edition costs. Figure 2 shows a schematic example of a transformation of a molecule $G_1$ into another one, $G_2$. As we can see, the executed operations in this transformation are: a deletion of node type [1], a deletion of a simple edge, an insertion of node type [5], an insertion of a simple edge a substitution of node type [7] by node of type [2], and a substitution of a simple edge with a double edge. If we sum the values of Harper costs associated with these operations in Tables 3 and 4, we obtain that the cost of this transformation equals: $2 + 0 + 2 + 0 + 3 + 3 = 10$.

**Table 2.** Node and edge attributes description in an ErG.

| Node Attributes | |
|---|---|
| **Attribute** | **Description** |
| [0] | hydrogen-bond donor |
| [1] | hydrogen-bond acceptor |
| [2] | positive charge |
| [3] | negative charge |
| [4] | hydrophobic group |
| [5] | aromatic ring system |
| [6] | carbon link node |
| [7] | non-carbon link node |
| [0, 1] | hydrogen-bond donor + hydrogen-bond acceptor |
| [0, 2] | hydrogen-bond donor + positive charge |
| [0, 3] | hydrogen-bond donor + negative charge |
| [1, 2] | hydrogen-bond acceptor + positive charge |
| [1, 3] | hydrogen-bond acceptor + negative charge |
| [2, 3] | positive charge + negative charge |
| [0, 1, 2] | hydrogen-bond donor + hydrogen-bond acceptor + positive charge |

| Edge Attributes | |
|---|---|
| **Attribute** | **Description** |
| - | single bond |
| = | double bond |
| ≡ | triple bond |

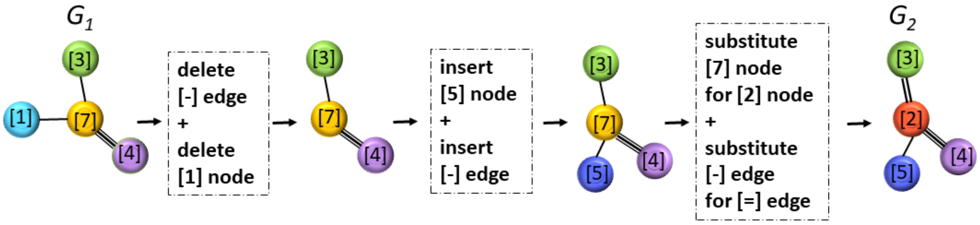

**Figure 2.** Transformation sequence from graph $G_1$ to graph $G_2$.

Since several transformation sequences can be applied to transform a graph into another one, the GED resulting for any pair of graphs is defined as the minimum cost under all those possible transformation sequences. Usually, the final distance is normalized according to the number of nodes in both graphs being compared. This is performed in order to make the measure independent of the size of the graphs.

More formally, we define the GED as follows,

$$GED(G_a, G_b, C_1, \ldots, C_n) = \min_{\{N_i:i=1,\ldots,n\}} \frac{C_1N_1 + \ldots + C_nN_n}{L} \tag{1}$$

where $C_t$ is the imposed cost of the $t$th edit operation on nodes and edges, and $N_t$ is the number of times this edit operation has been applied. Moreover, the combination of $N_1$, $N_2, \ldots$ is restricted to be one that transforms $G_a$ into $G_b$. Finally, $L$ is the sum of the number of nodes of both graphs, and $n$ is the number of different edit operations on nodes and edges.

Several GED computational methods have been proposed during the last three decades, they can be classified into two groups: those returning the exact value of the GED in the exponential computational cost with respect to the number of nodes [49], and those returning an approximation of the GED in the polynomial cost [50–53]. These two groups of GED computational methods have been widely studied [54,55]. In our experiments, we used the fast bipartite graph matching method [50] (polynomial computational cost), although our learning method is independent of the matching algorithm.

**Table 3.** Substitution, insertion and deletion costs for nodes proposed by Harper et al. [33].

| | **Substitution Costs for Nodes** | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **[0]** | **[1]** | **[2]** | **[3]** | **[4]** | **[5]** | **[6]** | **[7]** | **[0, 1]** | **[0, 2]** | **[0, 3]** | **[1, 2]** | **[1, 3]** | **[2, 3]** | **[0, 1, 2]** |
| **[0]** | 0 | **2** | **2** | 2 | **2** | **2** | **2** | 3 | **1** | **1** | 1 | 2 | 2 | 2 | **1** |
| **[1]** | **2** | 0 | 2 | 2 | **2** | **2** | **2** | 3 | **1** | **2** | 2 | **1** | 1 | 2 | **1** |
| **[2]** | **2** | 2 | 0 | 2 | 2 | 2 | **2** | 3 | 2 | **1** | 2 | 1 | 2 | 1 | **1** |
| **[3]** | 2 | 2 | 2 | 0 | 2 | 2 | **2** | 3 | **2** | 2 | 1 | 2 | 1 | 1 | 2 |
| **[4]** | **2** | **2** | 2 | 2 | 0 | **2** | **2** | 3 | **2** | **2** | 2 | 2 | 2 | 2 | 2 |
| **[5]** | **2** | **2** | 2 | 2 | **2** | 0 | **2** | 3 | **2** | **2** | 2 | 2 | 2 | 2 | **2** |
| **[6]** | **2** | **2** | **2** | **2** | **2** | **2** | 0 | 3 | 2 | **2** | 2 | 2 | 2 | 2 | **2** |
| **[7]** | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 0 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| **[0, 1]** | **1** | **1** | 2 | **2** | **2** | **2** | 2 | 3 | 0 | **2** | 2 | 2 | 2 | 2 | **2** |
| **[0, 2]** | **1** | **2** | **1** | 2 | **2** | **2** | **2** | 3 | **2** | 0 | 2 | 2 | 2 | 2 | 2 |
| **[0, 3]** | 1 | 2 | 2 | 1 | 2 | 2 | 2 | 3 | 2 | 2 | 0 | 2 | 2 | 2 | 2 |
| **[1, 2]** | 2 | **1** | 1 | 2 | 2 | 2 | 2 | 3 | 2 | 2 | 2 | 0 | 2 | 2 | 2 |
| **[1, 3]** | 2 | 1 | 2 | 1 | 2 | 2 | 2 | 3 | 2 | 2 | 2 | 2 | 0 | 2 | 2 |
| **[2, 3]** | 2 | 2 | 1 | 1 | 2 | 2 | 2 | 3 | 2 | 2 | 2 | 2 | 2 | 0 | 2 |
| **[0, 1, 2]** | **1** | **1** | **1** | 2 | 2 | **2** | **2** | 3 | **2** | 2 | 2 | 2 | 2 | 2 | 0 |
| | **Insertion/Deletion Costs for Nodes** | | | | | | | | | | | | | | |
| | **[0]** | **[1]** | **[2]** | **[3]** | **[4]** | **[5]** | **[6]** | **[7]** | **[0, 1]** | **[0, 2]** | **[0, 3]** | **[1, 2]** | **[1, 3]** | **[2, 3]** | **[0, 1, 2]** |
| **insert** | **2** | **2** | 2 | 2 | **2** | **2** | **1** | 1 | **2** | **2** | 2 | **2** | 2 | 2 | **2** |
| **delete** | **2** | **2** | 2 | 2 | **2** | **2** | **1** | 1 | **2** | **2** | 2 | **2** | 2 | 2 | **2** |

**Table 4.** Substitution, insertion and deletion costs for edges proposed by Harper et al. [33].

| | **Substitution Costs For Edges** | | |
|---|---|---|---|
| | **-** | **=** | ≡ |
| **-** | 0 | 3 | 3 |
| **=** | 3 | 0 | 3 |
| ≡ | 3 | 3 | 0 |
| | **Insertion/Deletion Costs For Edges** | | |
| | **-** | **=** | ≡ |
| insert | 0 | **1** | 1 |
| delete | 0 | **1** | 1 |

Initially, the edit costs were manually set in a trial and error process considering the application at hand [33,34]. (As previously commented, Tables 3 and 4 show their edit cost proposal.) Nevertheless, there has been a tendency to automatically learn these costs since it has been seen that a proper tuning of them is crucial to achieve good classification ratios in virtual screening [35] and other applications [56–60]. In [35], authors presented a learning algorithm that is forced to learn only one edit cost at once due to runtime restrictions. Thus, they perform four different experiments on the same data as [34] in which they use all the costs of [34] except the one that is learned. These experiments are: C1: Learning the deletion/insertion cost of the carbon link ([6]). C2: Learning the cost of substituting a carbon link node ([6]) with an aromatic ring system ([5]). C3: Learning the insertion/deletion cost of the bond edge ([-]). C4: Learning the substitution cost between a single bond edge ([-]) and a double bond edge ([=]). Table 5 shows their learnt costs.

**Table 5.** Costs obtained in [35]. Each row corresponds to one of their experiments.

| | Type of Cost | CAPST | DUD-E | GLL&GDD | MUV | NRLiSt_BDB | ULS-UDS |
|---|---|---|---|---|---|---|---|
| C1 | Ins/Del [6] | 0.000002 | 0.005 | 0.014 | 0.490 | 0.012 | 0.115 |
| C2 | Subs [5] by [6] | 0.013 | 0.145 | 0.333 | 0.867 | 0.104 | 0.500 |
| C3 | Ins/Del [-] | 0.004 | 0.001 | 0.003 | 0.327 | 0.003 | 0.011 |
| C4 | Subs [-] by [=] | 0.017 | 0.186 | 0.206 | 1.005 | 0.024 | 0.607 |

The next section presents our method, which has the advantage of learning the whole set of edit costs at once.

### 2.3.2. The Learning Method

We present an iterative algorithm, in which, in each iteration, the NN strategy is applied and the initial edit costs are modified such that one molecule that has been incorrectly classified becomes correctly classified. Modifying the edit costs could cause other incorrectly classified molecules to also be properly classified, but, unfortunately, some other ones that were properly classified become incorrectly classified. This is the reason why we want to generate the minimum modification on the edit costs. To do so, the selected molecule is the one that it is easier to move from the incorrectly classified ones to the correctly classified ones. In the next paragraphs, our learning algorithm is explained in detail.

Let $G_j$ be a molecule in the learning set that has been incorrectly classified using the NN strategy and the current costs $C_1, \ldots, C_n$. We define $D_j$ as the minimal GED between $G_j$ and all the molecules but restricted to be the ones that have a different class:

$$D_j = \min_q GED(G_j, G_q, C_1, \ldots, C_n)\text{ , where class}(G_q) \neq \text{class}(G_j). \quad (2)$$

Moreover, we define $D'_j$ as the minimal GED between $G_j$ and all the molecules of the learning set but restricted to be the ones that have the same class:

$$D'_j = \min_p GED(G_j, G_p, C_1, \ldots, C_n),\text{ where class}(G_p) = \text{class}(G_j) \quad (3)$$

Since $G_j$ is incorrectly classified, we can confirm that $D'_j > D_j$. Figure 3 schematically shows this situation. It turns out that $G_j$ and $G_q$ belong to different classes even though the distance between them is smaller than the distance between $G_j$ and its closest molecule that has the same class, $G_p$.

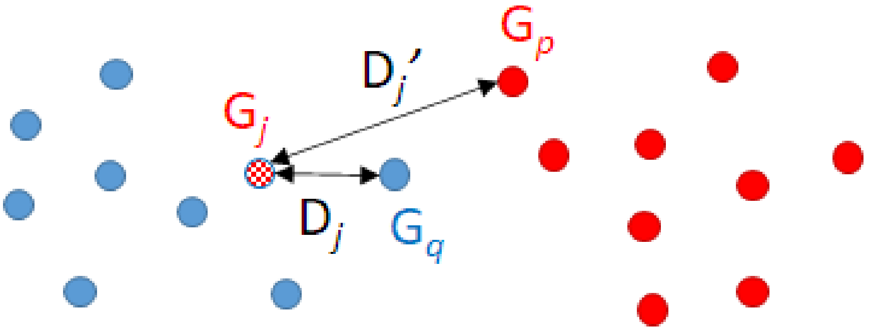

**Figure 3.** Classification of molecule $G_j$. The true classes are in solid colours. $G_j$ is classified in the wrong class (blue), but the correct class is the red one. The distance between $G_j$ and $G_q$ is lower than the distance between $G_j$ and $G_p$.

The main idea of our method is to permute $D'_j$ and $D_j$, modifying the edit costs. With this exchange, we achieve a lower distance between $G_j$ and the molecule of its same class ($G_p$) than the distance between $G_j$ and the molecule with different classes ($G_q$). Thus, $G_j$ will be correctly classified. However, considering that adapting these distances affects all the molecules' classifications, we select a molecule $G_i$ among the incorrectly-classified ones, $\{G_j | D'_j > D_j, \forall G_j\}$, which satisfies that the difference of the distances $D'_j - D_j$ is the minimum one, as shown in Equation (4). Note that in Equation (4), all the values of $D'_j - D_j$ are always positive because $D'_j > D_j$ by definition of $G_j$.

$$G_i = arg \min_{\{G_j | D'_j > D_j\}} (D'_j - D_j) \tag{4}$$

Figure 4 shows this idea. However, what is crucial to understand is that this modification is performed in the distances since the molecule representations are not modified. Furthermore, this is carried out by modifying the edit costs. Thus, the strategy is to define the new edit costs such that $D'_i$ becomes $D_i$ and vice versa.

The rest of this section is devoted to explaining how to modify the edit costs.

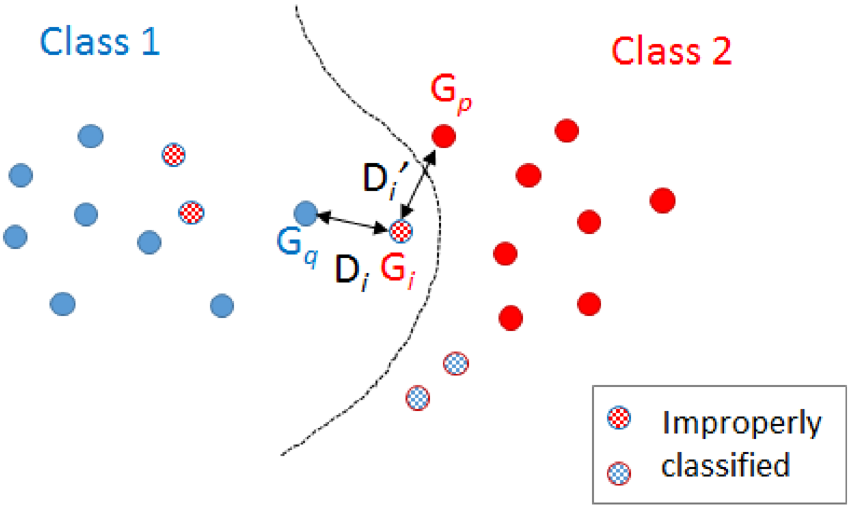

**Figure 4.** Stripped molecules have been improperly classified using NN strategy. $G_i$ is the one that minimises $D'_j - D_j$ being $D'_j > D_j$.

Considering Equation (1), the distance is composed of edit costs $C_1, \ldots, C_n$ and the number of times the specific edit operations have been taken $N_1, \ldots, N_n$. Our method modifies the edit costs without altering the number of operations $N_1, \ldots, N_n$.

Thus, we define $D_i$ and $D'_i$ as follows:

$$\begin{aligned} D_i &= \frac{C_1 N_1 + \ldots + C_n N_n}{L} \\ D'_i &= \frac{C_1 N'_1 + \ldots + C_n N'_n}{L'} \end{aligned} \tag{5}$$

Then, we exchange the distances $D_i$ and $D'_i$ and modify the edits costs by adding new terms:

$$\begin{aligned} D_i &= \frac{(C_1 + \alpha'_1) N'_1 + \ldots + (C_n + \alpha'_n) N'_n}{L'} \\ D'_i &= \frac{(C_1 + \alpha_1) N_1 + \ldots + (C_n + \alpha_n) N_n}{L} \end{aligned} \tag{6}$$

Note that these new terms $\alpha_1, \ldots, \alpha_n$ and also $\alpha'_1, \ldots, \alpha'_n$ are defined such that the new value of $D_i$ is $D'_i$ instead of $D_i$ and vice versa. Moreover, the edit costs $C_1, \ldots, C_n$ are the same in both expressions. We proceed to explain below how to deduce the terms $\alpha_1, \ldots, \alpha_n$ and also $\alpha'_1, \ldots, \alpha'_n$.

From Equation (6), we obtain:

$$\begin{aligned} D_i &= \frac{C_1N'_1 + \ldots + C_nN'_n}{L'} + \frac{\alpha'_1N'_1 + \ldots + \alpha'_nN'_n}{L'} \\ D'_i &= \frac{C_1N_1 + \ldots + C_nN_n}{L} + \frac{\alpha_1N_1 + \ldots + \alpha_nN_n}{L} \end{aligned} \tag{7}$$

We observe that the first terms in both expressions are $D'_j$ and $D_j$, respectively:

$$\begin{aligned} D_i &= D'_i + \frac{\alpha'_1N'_1 + \ldots + \alpha'_{n'}N'_n}{L'} \\ D'_i &= D_i + \frac{\alpha_1N_1 + \ldots + \alpha_nN_n}{L} \end{aligned} \tag{8}$$

By regrouping the terms again, we have:

$$\begin{aligned} D_i - D'_i &= \frac{\alpha'_1N'_1 + \ldots + \alpha'_{n'}N'_n}{L'} \\ D'_i - D_i &= \frac{\alpha_1N_1 + \ldots + \alpha_nN_n}{L} \end{aligned} \tag{9}$$

Furthermore, finally, we divide by $D_i - D'_i$ and $D'_i - D_i$ in each expression to arrive at the following normalised expressions:

$$\begin{aligned} 1 &= \frac{\alpha'_1N'_1}{(D_i - D'_i)L'} + \ldots + \frac{\alpha'_nN'_n}{(D_i - D'_i)L'} \\ 1 &= \frac{\alpha_1N_1}{(D'_i - D_i)L} + \ldots + \frac{\alpha_nN_n}{(D'_i - D_i)L} \end{aligned} \tag{10}$$

Note that, as commented in the definition of the GED, not all of the edit operations are used to transform a molecule into another. These edit operations are the ones that $N_t = 0$ or $N'_t = 0$. Because of this, in Equation (10), there are some addends that are null. We use $m$ and $m'$ to denote the number of edit operations that have been used, that is, the ones that $N_t \neq 0$ or $N'_t \neq 0$, respectively.

We want to deduce $\alpha_1, \alpha_2, \ldots$ and also $\alpha'_1, \alpha'_2, \ldots$ such that Equation (10) is fulfilled. The easiest way is to impose that each non-null term in these expressions equal $1/m'$ or $1/m$, respectively. Then, we achieve the following expressions,

$$\begin{aligned} 1/m' &= \frac{\alpha'_tN'_t}{(D_i - D'_i)L'} \text{ being } N'_t > 0 \\ 1/m &= \frac{\alpha_tN_t}{(D'_i - D_i)L} \text{ being } N_t > 0 \end{aligned} \tag{11}$$

From the previous expressions, we arrive at the definitions of $\alpha_t$ that allow the modification from $D_i$ to $D'_i$. Moreover, we also arrive at the definitions of $\alpha'_{t'}$ that allow the modification from $D'_i$ to $D_i$.

$$\begin{aligned} \alpha'_t &= \frac{(D_i - D'_i)L'}{m'N'_{t'}}, N'_t > 0 \\ \alpha_t &= \frac{(D'_i - D_i)L}{mN_t}, N_t > 0 \end{aligned} \tag{12}$$

Note that considering Equations (5), (6) and (12), we have, on one hand, that the new costs $\overline{C}_t = C_t + \alpha_t$ and, on the other hand, that $\overline{C}_t = C_t + \alpha'_t$. Since it may happen that $\alpha_t \neq \alpha'_t$, we assume the average option is the best choice when both weights are computed,

$$\overline{C}_t = \begin{cases} C_t + \frac{\alpha_t + \alpha'_t}{2}, \text{ if } N_t > 0 \text{ and } N'_t > 0 \\ C_t + \alpha_t, \text{ if } N_t > 0 \text{ and } N'_t = 0 \\ C_t + \alpha'_t, \text{ if } N_t = 0 \text{ and } N'_t > 0 \\ C_t, \text{ if } N_t = 0 \text{ and } N'_t = 0 \end{cases} \tag{13}$$

In the next subsection, we present our algorithm.

#### 2.3.3. Algorithm

Algorithm 1 consists of an iterative process in which, in each iteration, the edit costs are updated to correct the classification of one selected molecule. The updated costs are used in the next iteration to classify all the molecules again, select a molecule and modify the costs again.

**Algorithm 1** Costs learning.

```
Input (Learning Set, Initial edit costs, $Max\_Iter$)
Output (Learnt edit costs)
1.  Initialise:
    $iter = 1$.
    $C_1, \ldots, C_n$ = Initial edit costs.
    While $iter \leq Max\_Iter$:
2.      Classify all molecules with nearest neighbour and $GED$ (Equation (1)) using $C_1, \ldots, C_n$.
3.      Compute $D_j$ and $D'_j$: (Equations (2) and (3)) for all $G_j$ incorrectly classified.
4.      Deduce $G_i$ (Equation (4)).
5.      Compute $\alpha_t, t = 1, \ldots, m$ and $\alpha'_t, t = 1, \ldots, m'$: (Equation (12)).
6.      Compute $\overline{C}_1, \ldots, \overline{C}_n$ (Equation (13)).
7.      Update costs: $C_t = \overline{C}_t, t = 1, \ldots, n$.
8.      $iter = iter + 1$.
    End While
End Algorithm
```

This algorithm has been coded in Matlab, and it is available in https://deim.urv.cat/~francesc.serratosa/SW/, accessed on 12 November 2021.

## 3. Results

Table 6 shows the classification ratios obtained in each dataset using different edit cost configurations, algorithms and initialisations. The first row corresponds to the accuracies obtained with the costs proposed by Harper [33], the second row corresponds to the accuracies deduced by setting all the costs to 1 (no learning algorithm). The next four rows correspond to the accuracies obtained using the costs deduced in García et al. [35] in their four experiments (C1, C2, C3 and C4). Finally, the last two rows present the accuracies obtained by our method: the first row by initialising the algorithm by the Harper costs and the second one by initialising all the costs to 1. We note the used costs are the mean of the learned costs in all the databases, and our algorithm performed 50 iterations.

**Table 6.** Accuracy (%) obtained in each dataset. In bold, the highest ones. The last column shows the mean accuracy.

| | CAPST | DUD-E | GLL&GDD | MUV | NRLiSt_BDB | ULS-UDS | Mean |
|---|---|---|---|---|---|---|---|
| **Harper** | 93.75 | 95.88 | 85.68 | **92.76** | 93.17 | **96.10** | 92.89 |
| **1s** | 92.93 | 91.25 | 93.03 | 56.01 | 94.75 | 92.94 | 86.82 |
| **C1** | 89.25 | 92.63 | 82.47 | 86.06 | 88.58 | 89.65 | 88.11 |
| **C2** | 89.75 | 91.13 | 82.51 | 87.35 | 88.21 | 91.69 | 88.44 |
| **C3** | 91.25 | 91.25 | 83.25 | 86.65 | 87.75 | 92.34 | 88.75 |
| **C4** | 89.50 | 90.88 | 82.43 | 86.00 | 89.92 | 92.59 | 88.55 |
| **Our method (Harper init.)** | **95.85** | **96.38** | **93.67** | 88.63 | **95.90** | 94.00 | **94.07** |
| **Our method (1s init.)** | 88.15 | 93.50 | 93.30 | 61.76 | 94.98 | 95.25 | 87.82 |

We realise that in all the datasets, except for MUV and ULS-UDS, our costs with Harper initialisation obtained the highest classification ratios. In these two datasets, the best accuracy is obtained by Harper costs. Note that our method initialised by all-ones costs returns lower accuracies than our method initialised by Harper costs, except for the ULS-UDS dataset. This behaviour makes us think that the initialisation point is very important in this type of algorithm. Another highlight is that we have achieved better accuracies than the four experiments presented by García et al. [35] in all the tests. In the ULS-UDS dataset, our method returns close accuracy to the Harper costs. Nevertheless, that is not the case for MUV dataset. To deeply analyse this behaviour, we have computed the accuracy using the costs learned by only the MUV targets. In this case, the accuracy is 64.9%, which is significantly lower than using mean costs. This is not the normal behaviour in learning algorithms since while conducting specific learning, the classification ratio tends to increase. We think there are other reasons for this abnormal behaviour: one could be the small size of this dataset and the other the separability between ligands and decoys in MUV is low, which makes our algorithm not to converge to a proper solution.

In Figure 5, we present the classification ratio obtained in the 127 targets in the six datasets. At a glance, we realise that our method achieves most of the highest accuracies in all the targets in CAPST, DUD-E, GLL&GDD and NRLiSt-BDB databases. Specifically, we point out targets from 19 to 31 in the GLL&GDD dataset where the other cost combinations have very low accuracies while our method achieves much higher results. We observe that targets in the datasets MUV and ULS-UDS, in which our method does not return the highest accuracies, have a high variability because the same costs produce very different results.

Note that in [35], authors computed a cost per each of the six datasets and each target. Conversely, we learn the edit costs given the six datasets at once. In general, using several datasets at once makes the learnt parameters less specific for the application at hand, and thus, the classification ratios tend to decrease. In spite of this possible disadvantage, our method returns better classification ratios than the one in [35] in all the datasets.

Figure 6 shows the percentage of times that each cost configuration obtains the highest classification ratio taking into account all the 127 targets, given the four configurations proposed in [35], one configuration proposed in [33] and our deduced configuration. Our method obtains the best classification ratio the highest number of times.

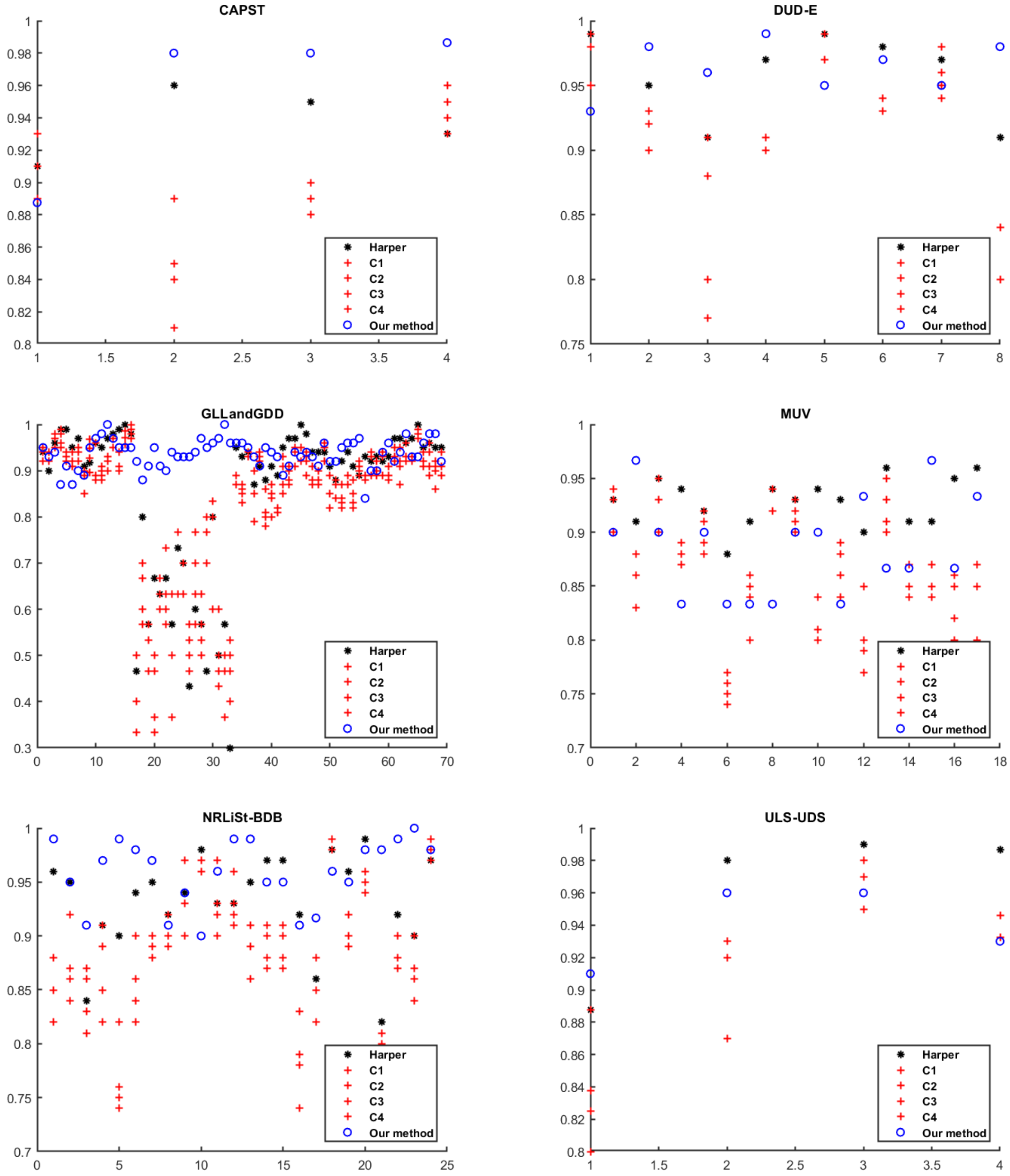

**Figure 5.** Classification ratio in the test set over the 127 targets available in the six datasets. The horizontal axis represents the index of the targets presented in Table 1.

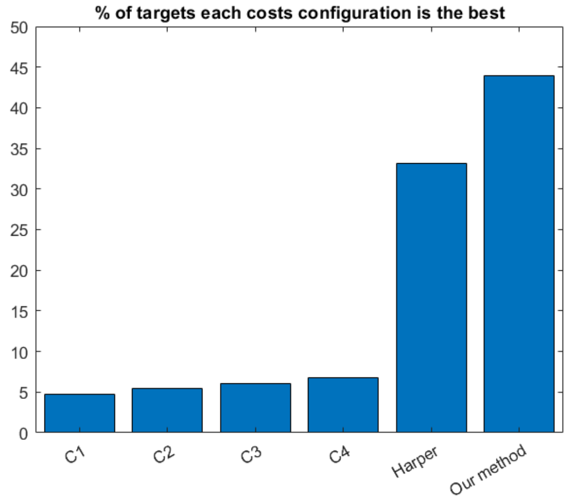

**Figure 6.** Percentage of times that each set of costs returns the best classification ratio.

Tables 7 and 8 show our learned edition costs for nodes and edges, respectively. In red and bold are the ones that are different to the ones proposed by Harper et al. [33]. As we can see, the results are very similar to Harper costs because we introduce a very small modification in each step. In addition, there are many costs that have not been modified. This is because these costs were not involved in the modifications of molecules that are improperly classified, minimising $D'_i - D_i$.

**Table 7.** Substitution, insertion and deletion costs of nodes obtained with our method. In bold, the ones that are different from Tables 3 and 4.

| | | | | | | | | **Substitution Costs for Nodes** | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **[0]** | **[1]** | **[2]** | **[3]** | **[4]** | **[5]** | **[6]** | **[7]** | **[0, 1]** | **[0, 2]** | **[0, 3]** | **[1, 2]** | **[1, 3]** | **[2, 3]** | **[0, 1, 2]** |
| **[0]** | 0 | **1.99** | **2.02** | 2.00 | **1.99** | **2.04** | **2.05** | 3.00 | **1.06** | **0.99** | 1.00 | 2.00 | 2.00 | 2.00 | **0.97** |
| **[1]** | **1.99** | 0 | 2.00 | 2.00 | **1.98** | **1.99** | **1.96** | 3.00 | **1.02** | **1.99** | 2.00 | **1.02** | 1.00 | 2.00 | **1.04** |
| **[2]** | **2.02** | 2.00 | 0 | 2.00 | 2.00 | 2.00 | **1.99** | 3.00 | 2.00 | **0.99** | 2.00 | 1.00 | 2.00 | 1.00 | **0.98** |
| **[3]** | 2.00 | 2.00 | 2.00 | 0 | 2.00 | 2.00 | **2.05** | 3.00 | **1.99** | 2.00 | 1.00 | 2.00 | 1.00 | 1.00 | 2.00 |
| **[4]** | **1.99** | **1.98** | 2.00 | 2.00 | 0 | **1.99** | **2.01** | 3.00 | **2.01** | **2.01** | 2.00 | 2.00 | 2.00 | 2.00 | 2.00 |
| **[5]** | **2.04** | **1.99** | 2.00 | 2.00 | **1.99** | 0 | **1.99** | 3.00 | **1.96** | **1.96** | 2.00 | 2.00 | 2.00 | 2.00 | **2.02** |
| **[6]** | **2.05** | **1.96** | **1.99** | **2.05** | **2.01** | **1.99** | 0 | 3.00 | 2.00 | **2.01** | 2.00 | 2.00 | 2.00 | 2.00 | **1.98** |
| **[7]** | 3.00 | 3.00 | 3.00 | 3.00 | 3.00 | 3.00 | 3.00 | 0 | 3.00 | 3.00 | 3.00 | 3.00 | 3.00 | 3.00 | 3.00 |
| **[0, 1]** | **1.06** | **1.02** | 2.00 | **1.99** | **2.01** | **1.96** | 2.00 | 3.00 | 0 | **2.02** | 2.00 | 2.00 | 2.00 | 2.00 | **2.01** |
| **[0, 2]** | **0.99** | **1.99** | **0.99** | 2.00 | **2.01** | **1.96** | **2.01** | 3.00 | **2.02** | 0 | 2.00 | 2.00 | 2.00 | 2.00 | 2.00 |
| **[0, 3]** | 1.00 | 2.00 | 2.00 | 1.00 | 2.00 | 2.00 | 2.00 | 3.00 | 2.00 | 2.00 | 0 | 2.00 | 2.00 | 2.00 | 2.00 |
| **[1, 2]** | 2.00 | **1.02** | 1.00 | 2.00 | 2.00 | 2.00 | 2.00 | 3.00 | 2.00 | 2.00 | 2.00 | 0 | 2.00 | 2.00 | 2.00 |
| **[1, 3]** | 2.00 | 1.00 | 2.00 | 1.00 | 2.00 | 2.00 | 2.00 | 3.00 | 2.00 | 2.00 | 2.00 | 2.00 | 0 | 2.00 | 2.00 |
| **[2, 3]** | 2.00 | 2.00 | 1.00 | 1.00 | 2.00 | 2.00 | 2.00 | 3.00 | 2.00 | 2.00 | 2.00 | 2.00 | 2.00 | 0 | 2.00 |
| **[0, 1, 2]** | **0.97** | **1.04** | **0.98** | 2.00 | 2.00 | **2.02** | **1.98** | 3.00 | **2.01** | 2.00 | 2.00 | 2.00 | 2.00 | 2.00 | 0 |
| | | | | | | | | **Insertion/Deletion Costs for Nodes** | | | | | | | |
| | **[0]** | **[1]** | **[2]** | **[3]** | **[4]** | **[5]** | **[6]** | **[7]** | **[0, 1]** | **[0, 2]** | **[0, 3]** | **[1, 2]** | **[1, 3]** | **[2, 3]** | **[0, 1, 2]** |
| **insert** | **1.95** | **1.98** | 2.00 | 2.00 | **1.99** | **1.89** | **0.97** | 1.00 | **2.03** | **2.02** | 2.00 | **1.99** | 2.00 | 2.00 | **1.96** |
| **delete** | **1.95** | **1.98** | 2.00 | 2.00 | **1.99** | **1.89** | **0.97** | 1.00 | **2.03** | **2.02** | 2.00 | **1.99** | 2.00 | 2.00 | **1.96** |

**Table 8.** Substitution, insertion and deletion costs of edges obtained with our method.

| | **Substitution Costs<br>For Edges** | | |
|---|---|---|---|
| | **-** | **=** | **≡** |
| **-** | 0 | 3.00 | 3.00 |
| **=** | 3.00 | 0 | 3.00 |
| **≡** | 3.00 | 3.00 | 0 |
| | **Insertion/Deletion Costs<br>For Edges** | | |
| | **-** | **=** | **≡** |
| insert | 0 | **1.02** | 1.00 |
| delete | 0 | **1.02** | 1.00 |

## 4. Discussion

We present an iterative algorithm such that, in each iteration, the current costs are modified to properly classify an improperly classified molecule. While updating the costs, other improperly classified molecules could also be properly classified and vice versa. This is the reason why we cannot guarantee the algorithm's convergence. To reduce the no-convergence impact and the possible solution oscillation, the algorithm selects the molecule that requires the minimum modification of the costs with the aim of slightly moving to the best solution.

The algorithm requires some initial costs. We have initialised the algorithm by some aleatory costs and by the costs proposed by Harper [33]. In all the tests, the highest accuracies appeared while initialising the costs by the Harper proposal. We believe this behaviour appears because the optimisation function of the learning algorithm is highly non-convex. Generally, in these situations, the selected initialisation has a high impact on the solution. Finally, we have seen that the classification accuracy is highly dependent on the edit costs. That is, a slight modification of one of the costs could make the classification be completely different. Considering that the computational cost of this learning problem is extremely high, sub-optimal algorithms, as the one presented, are needed to achieve an acceptable classification accuracy. Thus, any proposal that achieves better classification ratios would have to be considered and analysed.

## 5. Conclusions and Future Research

In some ligand-based virtual screening (LBVS) methods, molecules are represented by extended reduced graphs. In this case, the Graph Edit Distance can be used to compute the dissimilarity between molecules.

In this article, we have presented a new method that automatically learns the edit costs in the Graph Edit Distance. In each iteration, our method introduces slight modifications in the current costs to correct the classification of a selected molecule that had been incorrectly classified in the previous step.

The obtained costs have been tested in six publicly available datasets and have been compared to previous works published in [34,35]. We achieve better classification ratios than [35] in the six datasets and better classification ratios than [34] in four of them.

In the experimental section, we realised that small modifications in the costs could produce considerable improvement in the classification ratio.

In future work, we will analyse which types of molecules cause the algorithm to converge and which are the best initial values to obtain higher classification accuracy.

**Author Contributions:** Conceptualization, methodology and investigation: E.R., S.Á. and F.S.; Supervision: S.Á. and F.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by Martí i Franquès grant number 2018PMF-PIPF-30.

**Acknowledgments:** This project has received funding from the Martí i Franquès fellowship of Universitat Rovira i Virgili (URV), Spain.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Kubinyi, H.; Mannhold, R.; Timmerman, H. *Virtual Screening for Bioactive Molecules*; John Wiley & Sons: Hoboken, NJ, USA, 2008; Volume 10.
2. Bajorath, J. Selected concepts and investigations in compound classification, molecular descriptor analysis, and virtual screening. *J. Chem. Inf. Comput. Sci.* **2001**, *41*, 233–245. [CrossRef]
3. Schneider, G.; Clément-Chomienne, O.; Hilfiger, L.; Schneider, P.; Kirsch, S.; Böhm, H.J.; Neidhart, W. Virtual screening for bioactive molecules by evolutionary de novo design. *Angew. Chem. Int. Ed.* **2000**, *39*, 4130–4133. [CrossRef]
4. Cereto-Massagué, A.; Ojeda, M.J.; Valls, C.; Mulero, M.; Garcia-Vallvé, S.; Pujadas, G. Molecular fingerprint similarity search in virtual screening. *Methods* **2015**, *71*, 58–63. [CrossRef]
5. Heikamp, K.; Bajorath, J. The future of virtual compound screening. *Chem. Biol. Drug Des.* **2013**, *81*, 33–40. [CrossRef]
6. Sun, H. Pharmacophore-based virtual screening. *Curr. Med. Chem.* **2008**, *15*, 1018–1024. [CrossRef]
7. Kirchmair, J.; Distinto, S.; Markt, P.; Schuster, D.; Spitzer, G.M.; Liedl, K.R.; Wolber, G. How to optimize shape-based virtual screening: Choosing the right query and including chemical information. *J. Chem. Inf. Model.* **2009**, *49*, 678–692. [CrossRef]
8. Melville, J.L.; Burke, E.K.; Hirst, J.D. Machine learning in virtual screening. *Comb. Chem. High Throughput Screen.* **2009**, *12*, 332–343. [CrossRef]
9. Johnson, M.A.; Maggiora, G.M. *Concepts and Applications of Molecular Similarity*; Wiley: Hoboken, NJ, USA, 1990.
10. Bender, A.; Glen, R.C. Molecular similarity: A key technique in molecular informatics. *Org. Biomol. Chem.* **2004**, *2*, 3204–3218. [CrossRef]
11. Nikolova, N.; Jaworska, J. Approaches to measure chemical similarity—A review. *Mol. Inf.* **2003**, *22*, 1006–1026. [CrossRef]
12. Willett, P. Evaluation of molecular similarity and Mol. Diversity methods using biological activity data. In *Chemoinformatics*; Springer: Berlin/Heidelberg, Germany, 2004; pp. 51–63.
13. Lajiness, M. Molecular similarity-based methods for selecting compounds for screening. In *Computational Chemical Graph Theory*; Nova Science Publishers, Inc.: Hauppauge, NY, USA, 1990; pp. 299–316.
14. Willett, J. *Similarity and Clustering in Chemical Information Systems*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 1987.
15. Xue, L.; Bajorath, J. Molecular descriptors in chemoinformatics, computational combinatorial chemistry, and virtual screening. *Comb. Chem. High Throughput Screen.* **2000**, *3*, 363–372. [CrossRef]
16. Menard, P.R.; Mason, J.S.; Morize, I.; Bauerschmidt, S. Chemistry space metrics in diversity analysis, library design, and compound selection. *J. Chem. Inf. Comput. Sci.* **1998**, *38*, 1204–1213. [CrossRef]
17. Pearlman, R.S.; Smith, K.M. Metric validation and the receptor-relevant subspace concept. *J. Chem. Inf. Comput. Sci.* **1999**, *39*, 28–35. [CrossRef]
18. Schnur, D. Design and diversity analysis of large combinatorial libraries using cell-based methods. *J. Chem. Inf. Comput. Sci.* **1999**, *39*, 36–45. [CrossRef]
19. Livingstone, D.J. The characterization of chemical structures using molecular properties. A survey. *J. Chem. Inf. Comput. Sci.* **2000**, *40*, 195–209. [CrossRef]
20. Barnard, J.M. Substructure searching methods: Old and new. *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 532–538. [CrossRef]
21. James, C.; Weininger, D. *Daylight, 4.41 Theory Manual*; Daylight Chemical Information Systems Inc.: Irvine, CA, USA, 1995.
22. McGregor, M.J.; Pallai, P.V. Clustering of large databases of compounds: Using the MDL "keys" as structural descriptors. *J. Chem. Inf. Comput. Sci.* **1997**, *37*, 443–448. [CrossRef]
23. Güner, O.F. *Pharmacophore Perception, Development, and Use in Drug Design*; Internat'l University Line: Geneva, Switzerland, 2000; Volume 2.
24. Beno, B.R.; Mason, J.S. The design of combinatorial libraries using properties and 3D pharmacophore fingerprints. *Drug Discov. Today* **2001**, *6*, 251–258. [CrossRef]
25. Rarey, M.; Dixon, J.S. Feature trees: A new molecular similarity measure based on tree matching. *J. Comput.-Aided Mol. Des.* **1998**, *12*, 471–490. [CrossRef]
26. Barker, E.J.; Buttar, D.; Cosgrove, D.A.; Gardiner, E.J.; Kitts, P.; Willett, P.; Gillet, V.J. Scaffold hopping using clique detection applied to reduced graphs. *J. Chem. Inf. Model.* **2006**, *46*, 503–511. [CrossRef]
27. Takahashi, Y.; Sukekawa, M.; Sasaki, S. Automatic identification of molecular similarity using reduced-graph representation of chemical structure. *J. Chem. Inf. Model.* **1992**, *32*, 639–643. [CrossRef]
28. Stiefl, N.; Watson, I.A.; Baumann, K.; Zaliani, A. ErG: 2D pharmacophore descriptions for scaffold hopping. *J. Chem. Inf. Model.* **2006**, *46*, 208–220. [CrossRef] [PubMed]
29. Gillet, V.J.; Willett, P.; Bradshaw, J. Similarity searching using reduced graphs. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 338–345. [CrossRef]

30. Gillet, V.J.; Downs, G.M.; Holliday, J.D.; Lynch, M.F.; Dethlefsen, W. Computer storage and retrieval of generic chemical structures in patents. 13. Reduced graph generation. *J. Chem. Inf. Comput. Sci.* **1991**, *31*, 260–270. [CrossRef]
31. Fisanick, W.; Lipkus, A.H.; Rusinko, A. Similarity Searching on CAS Registry Substances. 2. 2D Structural Similarity. *J. Chem. Inf. Comput. Sci.* **1994**, *34*, 130–140. [CrossRef]
32. Barker, E.J.; Gardiner, E.J.; Gillet, V.J.; Kitts, P.; Morris, J. Further development of reduced graphs for identifying bioactive compounds. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 346–356. [CrossRef]
33. Harper, G.; Bravi, G.S.; Pickett, S.D.; Hussain, J.; Green, D.V.S. The reduced graph descriptor in virtual screening and data-driven clustering of high-throughput screening data. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 2145–2156. [CrossRef]
34. Garcia-Hernandez, C.; Fernández, A.; Serratosa, F. Ligand-Based Virtual Screening Using Graph Edit Distance as Molecular Similarity Measure. *J. Chem. Inf. Model.* **2019**, *59*, 1410–1421. [CrossRef]
35. Garcia-Hernandez, C.; Fernández, A.; Serratosa, F. Learning the Edit Costs of Graph Edit Distance Applied to Ligand-Based Virtual Screening. *Curr. Top. Med. Chem.* **2020**, *20*, 1582–1592. [CrossRef]
36. Munkres, J. Algorithms for the Assignment and Transportation Problems. *J. Soc. Ind. Appl. Math.* **1957**, *5*, 32–38. [CrossRef]
37. Sanfeliu, A.; Fu, K.S. A distance measure between attributed relational graphs for Pattern Recognit. *IEEE Trans. Syst. Man Cybern.* **1983**, *SMC-13*, 353–362. [CrossRef]
38. Gao, X.; Xiao, B.; Tao, D.; Li, X. A survey of graph edit distance. *Pattern Anal. Appl.* **2010**, *13*, 113–129. [CrossRef]
39. Birchall, K.; Gillet, V.J.; Harper, G.; Pickett, S.D. Training similarity measures for specific activities: Application to reduced graphs. *J. Chem. Inf. Model.* **2006**, *46*, 577–586. [CrossRef]
40. Xia, J.; Tilahun, E.L.; Reid, T.E.; Zhang, L.; Wang, X.S. Benchmarking methods and data sets for ligand enrichment assessment in virtual screening. *Methods* **2015**, *71*, 146–157. [CrossRef]
41. Gatica, E.A.; Cavasotto, C.N. Ligand and decoy sets for docking to G protein-coupled receptors. *J. Chem. Inf. Model.* **2011**, *52*, 1–6. [CrossRef]
42. Sanders, M.P.; Barbosa, A.J.; Zarzycka, B.; Nicolaes, G.A.; Klomp, J.P.; de Vlieg, J.; Del Rio, A. Comparative analysis of pharmacophore screening tools. *J. Chem. Inf. Model.* **2012**, *52*, 1607–1620. [CrossRef] [PubMed]
43. Mysinger, M.M.; Carchia, M.; Irwin, J.J.; Shoichet, B.K. Directory of useful decoys, enhanced (DUD-E): Better ligands and decoys for better benchmarking. *J. Med. Chem.* **2012**, *55*, 6582–6594. [CrossRef]
44. Lagarde, N.; Ben Nasr, N.; Jeremie, A.; Guillemain, H.; Laville, V.; Labib, T.; Zagury, J.F.; Montes, M. NRLiSt BDB, the manually curated nuclear receptors ligands and structures benchmarking database. *J. Med. Chem.* **2014**, *57*, 3117–3125. [CrossRef] [PubMed]
45. Rohrer, S.G.; Baumann, K. Maximum unbiased validation (MUV) data sets for virtual screening based on PubChem bioactivity data. *J. Chem. Inf. Model.* **2009**, *49*, 169–184. [CrossRef] [PubMed]
46. Skoda, P.; Hoksza, D. Benchmarking platform for ligand-based virtual screening. In Proceedings of the 2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM 2016), Shenzhen, China, 15–18 December 2016; pp. 1220–1227. [CrossRef]
47. Solé, A.; Serratosa, F.; Sanfeliu, A. On the Graph Edit Distance Cost: Properties and Applications. *Intern. J. Pattern Recognit. Artif. Intell.* **2012**, *26*, 1260004. [CrossRef]
48. Serratosa, F. Redefining the Graph Edit Distance. *SN Comput. Sci.* **2021**, *2*, 438. [CrossRef]
49. Blumenthal, D.B.; Gamper, J. On the exact computation of the graph edit distance. *Pattern Recognit. Lett.* **2018**, 1–12. [CrossRef]
50. Serratosa, F. Fast computation of bipartite graph matching. *Pattern Recognit. Lett.* **2014**, *45*, 244–250. [CrossRef]
51. Santacruz, P.; Serratosa, F. Error-tolerant graph matching in linear computational cost using an initial small partial matching. *Pattern Recognit. Lett.* **2018**, 1–10. [CrossRef]
52. Serratosa, F. Speeding up Fast Bipartite Graph Matching Through a New Cost Matrix. *Int. J. Pattern Recognit. Artif. Intell.* **2014**, *29*, 1550010. [CrossRef]
53. Serratosa, F. Computation of graph edit distance: Reasoning about optimality and speed-up. *Image Vis. Comput.* **2015**, *40*, 38–48. [CrossRef]
54. Conte, D.; Foggia, P.; Sansone, C.; Vento, M. Thirty years of graph matching in Pattern Recognit. *Intern. J. Pattern Recognit. Artif. Intell.* **2004**, *18*, 265–298. [CrossRef]
55. Vento, M. A long trip in the charming world of graphs for Pattern Recognit. *Pattern Recognit.* **2015**, *48*, 291–301. [CrossRef]
56. Rica, E.; Álvarez, S.; Serratosa, F. On-line learning the graph edit distance costs. *Pattern Recognit. Lett.* **2021**, *146*, 55–62. [CrossRef]
57. Conte, D.; Serratosa, F. Interactive online learning for graph matching using active strategies. *Knowl. Based Syst.* **2020**, *205*, 106275. [CrossRef]
58. Santacruz, P.; Serratosa, F. Learning the Graph Edit Costs Based on a Learning Model Applied to Sub-optimal Graph Matching. *Neural Process. Lett.* **2020**, *51*, 881–904. [CrossRef]
59. Algabli, S.; Serratosa, F. Embedding the node-to-node mappings to learn the Graph edit distance parameters. *Pattern Recognit. Lett.* **2018**, *112*, 353–360. [CrossRef]
60. Cortés, X.; Serratosa, F. Learning Graph Matching Substitution Weights Based on the Ground Truth Node Correspondence. *Int. J. Pattern Recognit. Artif. Intell.* **2016**, *30*, 1650005:1–1650005:22. [CrossRef]