A Study On The Stability of Graph Edit Distance Heuristics

Linlin Jia, Vincent Tognetti, Laurent Joubert, Benoit Gaüzère, Paul Honeine

▶ **To cite this version:**

Linlin Jia, Vincent Tognetti, Laurent Joubert, Benoit Gaüzère, Paul Honeine. A Study On The Stability of Graph Edit Distance Heuristics. Electronics, 2022, 11 (20), pp.3312. 10.3390/electronics11203312 . hal-03816056

HAL Id: hal-03816056

https://normandie-univ.hal.science/hal-03816056

Submitted on 21 May 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

electronics

MDPI

Communication

A Study on the Stability of Graph Edit Distance Heuristics

Linlin Jia [1,*], Vincent Tognetti [2], Laurent Joubert [2], Benoit Gaüzère [3] and Paul Honeine [4]

[1] The COBRA Lab, INSA Rouen Normandie, 76800 Rouen, France
[2] The COBRA Lab, Université de Rouen Normandie, 76000 Rouen, France
[3] The LITIS Lab, INSA Rouen Normandie, 76800 Rouen, France
[4] The LITIS Lab, Université de Rouen Normandie, 76000 Rouen, France
* Correspondence: linlin.jia@insa-rouen.fr

Abstract: Graph edit distance (GED) is a powerful tool to model the dissimilarity between graphs. However, evaluating the exact GED is NP-hard. To tackle this problem, estimation methods of GED were introduced, e.g., `bipartite` and `IPFP`, during which heuristics were employed. The stochastic nature of these methods induces the stability issue. In this paper, we propose the first formal study of stability of GED heuristics, starting with defining a measure of these (in)stabilities, namely the relative error. Then, the effects of two critical factors on stability are examined, namely, the number of solutions and the ratio between edit costs. The ratios are computed on five datasets of various properties. General suggestions are provided to properly choose these factors, which can reduce the relative error by more than an order of magnitude. Finally, we verify the relevance of stability to predict performance of GED heuristics, by taking advantage of an edit cost learning algorithm to optimize the performance and the k-nearest neighbor regression for prediction. Experiments show that the optimized costs correspond to much higher ratios and an order of magnitude lower relative errors than the expert cost.

Keywords: graph edit distances; stability analyses; heuristic methods; edit cost learning

check for updates

Citation: Jia, L.; Tognetti, V.; Joubert, L.; Gaüzère, B.; Honeine, P. A Study on the Stability of Graph Edit Distance Heuristics. *Electronics* **2022**, *11*, 3312. https://doi.org/10.3390/electronics11203312

Academic Editor: Gemma Piella

Received: 12 September 2022
Accepted: 8 October 2022
Published: 14 October 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Graphs provide a flexible representation framework to encode elements as well as the relationships between them, enabling to capture the underlying structural information of the data. Despite providing rich expressiveness, the complexity lying in graph structures becomes its Achilles' heel when applying machine learning methods for graph data, which are mainly designed to operate on vector representations [1,2]. To leverage this flaw, several approaches have been designed to learn models on graphs, representatives among which include graph embedding strategy [3], graph kernels [4,5], and more recently graph neural networks [6], some of which are closely connected with signal processing on graphs [7–12]. Despite their state-of-the-art performances, they seldom operate directly in a graph space, hence reducing the interpretability of the underlying operations. To overcome these issues and preserve the properties of a graph space, some (dis)similarity measure or metric is usually assigned to that space, since most machine learning algorithms rely on (dis)similarity measures between data. One of the most used dissimilarity measures between graphs is the graph edit distance (GED) [13,14]. The GED of two graphs G_1 and G_2 is the minimal amount of distortion required to transform G_1 into G_2. This transformation includes a series of six elementary edit operations, namely vertex and edge substitutions (v_s and e_s), removals (v_r and e_r), and insertions (v_i and e_i), as shown in Figure 1. This sequence of edit operations constitutes an edit path π. A non-negative cost $c(e)$ can be assigned to each edit operation e. The sum of all edit operation costs included within π is defined as the cost associated with π. The minimal cost among all edit paths defines the GED between G_1 and G_2.

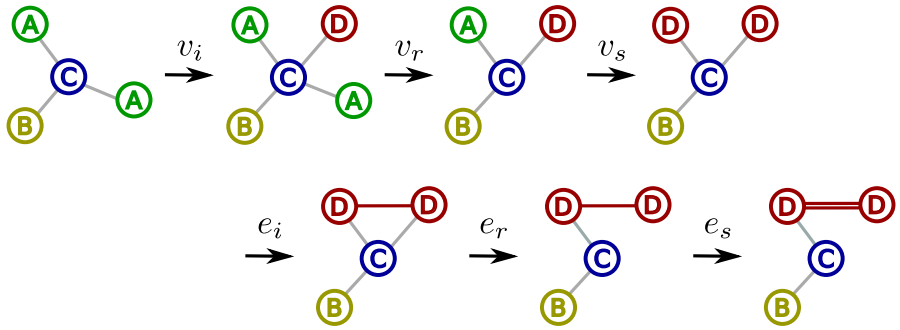

Figure 1. An illustration of graph edit operations. v_i, v_r, v_s, e_i, e_r, and e_s denote, respectively, the insertions, removals, and substitutions of vertices and edges. Different letters along with colors illustrate different vertex labels.

Evaluating exact GED is *NP*-hard even for uniform edit costs [15]. In practice, it cannot be done for graphs having more than 12 vertices in general [16]. To avoid this computational burden, strategies to approximate GED in a limited computational time have been proposed [17,18] with acceptable classification or regression performances. Of particular interest are the two famous methods, `bipartite` [19] and `IPFP` [20], where upper and lower bounds are estimated as an approximation of GED. The computation of the bounds relies highly on the design of the algorithm, as well as the randomness during the procedure, which leads to a reduction of stability.

As this paper will illustrate, the stability of the GED heuristics is highly relevant to the choice of heuristic method and their prediction performance. As GED is a widely used similarity between graphs, the study of stability can be useful to help promote the performance of GED in various tasks.

In this paper, it is the first time that the stability of GED heuristics has been formally studied. We define the instability of a GED heuristic in terms of the variability of the GED approximations over repeated trials. Methods that can potentially alleviate this problem are proposed. For instance, by carrying out several local searches in parallel, the multi-start counterparts of `bipartite` and `IPFP`, named m`bipartite` and m`IPFP`, respectively, may acquire better approximation with higher stability [21]. Description and analyses of these approximations and the root of randomness are presented in Sections 3 and 4.

Another essential ingredient of GED is the underlying edit cost function $c(e)$, which quantifies the distortion carried by any edit operation e. The values of the edit costs for each edit operation have a major impact on the computation of GED and its performance, including its stability [17,22]. Besides fixing costs provided a priori by domain experts for a given dataset, methods are proposed to optimize these costs, e.g., by aligning metrics in the graph to target spaces [23]. Analyzing the optimized edit costs helps further explore its relevance to the stability of GED heuristics.

The remaining part of the paper is organized as follows: Section 2 introduces widely used GED heuristics paradigms. Section 3 defines a measure of the (in)stability of these heuristics, as well as two factors of critical influence. Then, Section 4 gives experiments and analyses. Finally, Section 5 concludes the work and open perspectives.

2. Graph Edit Distances Heuristics

Over the years, many heuristics have been proposed to approximate GED. The authors of [18] categorize these heuristics according to their underlying paradigms. As milestones and baselines to many other heuristics, both `bipartite` and `IPFP` achieve high performance [17]. Thereby, in the following sections, we focus on these two heuristics and the related paradigms. First, we provide preliminary definitions of graphs and graph edit distances.

2.1. Graphs and Graph Edit Distances

A graph $G = (V, E)$ is an ordered pair of disjoint sets, where V is the vertex set and $E \in V \times V$ is the edge set. A graph can have a label set L from a label space and a labeling function ℓ that assigns a label $l \in L$ to each vertex and/or edge.

Given a set $\mathcal{G}$ of N graphs, the Graph Edit Distance (GED) between two graphs G_i and $G_j \in \mathcal{G}$ is defined as the cost of minimal transformation [19]:

$$\texttt{ged}(G_i, G_j) = \min_{\pi \in \Pi(G_i, G_j)} C(\pi, G_i, G_j), \tag{1}$$

where $\pi(G_i, G_j)$ is a mapping between $V_i \cup \varepsilon$ and $V_j \cup \varepsilon$ encoding the transformation from G_i to G_j, and ε represents a dummy element [20]. As described in Section 1, this transformation consists of a series of six elementary operations: removing or inserting a vertex or an edge, and substituting a label of a vertex or an edge by another. $C(\pi, G_i, G_j)$ measures the cost associated with π:

$$\begin{aligned} C(\pi, G_i, G_j) = & \sum_{\substack{v \in V_j \\ \pi^{-1}(v) = \varepsilon}} c_{vfi} + \sum_{\substack{v \in V_i \\ \pi(v) = \varepsilon}} c_{vfr} + \sum_{\substack{v \in V_i \\ \pi(v) \neq \varepsilon}} c_{vfs} \\ & + \sum_{\substack{e=(v_i,v_j) \in E_j | \\ \pi^{-1}(v_i) = \varepsilon \vee \\ \pi^{-1}(v_j) = \varepsilon \vee \\ (\pi^{-1}(v_i), \pi^{-1}(v_j)) \notin E_i}} c_{efi} + \sum_{\substack{e=(v_i,v_j) \in E_i | \\ \pi(v_i) = \varepsilon \vee \\ \pi(v_j) = \varepsilon \vee \\ (\pi(v_i), \pi(v_j)) \notin E_j}} c_{efr} + \sum_{\substack{e=(v_i,v_j) \in E_i | \\ \pi(v_i) \neq \varepsilon \wedge \\ \pi(v_j) \neq \varepsilon \wedge \\ (\pi(v_i), \pi(v_j)) \in E_j}} c_{efs}, \end{aligned} \tag{2}$$

where $c_{vfr}, c_{vfi}, c_{vfs}, c_{efr}, c_{efi}, c_{efs}$ are the edit costs associated with the six edit operations: respectively, vertex removal, insertion, substitution and edge removal, insertion, and substitution. Without loss of generality, these costs are set to be constant for each edit operation in the following part, denoted, respectively, as $c_{vr}, c_{vi}, c_{vs}, c_{er}, c_{ei}, c_{es}$.

2.2. Paradigm *LSAPE-GED* and Heuristic *Bipartite*

GED can be approximated by solving a linear sum assignment problem with edition or error correction (LSAPE). For any two sets V_1 and V_2, consider a transformation from V_1 to V_2, with elementary operations on each element $i \in V_1$: substitution ($i \to j$), insertion ($\varepsilon \to j$), and removal ($i \to \varepsilon$), where $j \in V_2$ and ε represents a dummy element. An assignment with edition, also known as the ε-assignment [24], is a bijection between set $V_1^\varepsilon = V_1 \cup \{\varepsilon\}$ and set $V_2^\varepsilon = V_2 \cup \{\varepsilon\}$ relaxed on ε, namely $\pi : V_1^\varepsilon \to V_2^\varepsilon$, where $|\pi(i)| = 1$ for any $i \in V_1$, $|\pi^{-1}(j)| = 1$ for any $j \in V_2$, and $\pi(\varepsilon) = \varepsilon$. We denote the set of all possible ε-assignments from V_1^ε to V_2^ε as $\Pi(V_1, V_2)$.

Each elementary operation in an ε-assignment π can be associated with a non-negative cost c. Consequently, a cost C is associated with π, namely,

$$C(\pi) = \sum_{\substack{i \in V_1 \\ \pi(i) = j}} c(i, j) + \sum_{\substack{j \in V_2 \\ \pi^{-1}(j) = \varepsilon}} c(\varepsilon, j) + \sum_{\substack{i \in V_1 \\ \pi(i) = \varepsilon}} c(i, \varepsilon), \tag{3}$$

where each term on the right side successively represents substitutions, insertions, and removals. The costs of all operations induced by π can be represented by a matrix $\mathbf{C} \in \mathbb{R}^{(|V_1|+1) \times (|V_2|+1)}$. LSAPE aims at minimizing this cost over all $\pi \in \Pi$, namely finding $C^\star(\pi^\star) = \min_{\pi \in \Pi(V_1, V_2)} C(\pi)$. We denote the set of all optimal solutions as $\Pi^\star(V_1, V_2)$. Variants of the Hungarian algorithm have been used to acquire an optimal solution [25,26].

The GED between two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ can be approximated by solving an LSAPE between vertex sets V_1 and V_2. Each row and column in the cost matrix $\mathbf{C}$ respectively correspond to a vertex in V_1 and V_2; each entry $\pi^\star(G_1, G_2) = \pi^\star(V_1, V_2) \in \Pi^\star(G_1, G_2)$ represents a optimal feasible transformation from G_1 to G_2 as in (1), and the optimal cost $C^\star$ corresponds to the approximation of GED. This paradigm is named `LSAPE-GED`.

`bipartite` is a representative heuristic under the `LSAPE-GED` paradigm. It constructs the cost matrix $\mathbf{C}$ by adding the cost of vertices and the cost of the edges adjacent to them. After that, an optimal LSAPE solution for $\mathbf{C}$ and the corresponding cost are computed.

2.3. Paradigm `LS-GED` and Heuristic `IPFP`

The local search (`LS-GED`) paradigm is composed of two steps. First, the transformation π and the cost $C(\pi)$ are initialized randomly or by a heuristic, such as `bipartite`. Then, starting at these initial results, a refinement procedure is carried out by a local search method to search for improved transformation with a lower cost. With different strategies applied in the second step, various heuristics have been designed. `IPFP` is a well-known representative one.

GED can be modeled as a quadratic problem. The `LSAPE-GED` paradigm simplifies this problem by only considering the linear part of GED, namely the costs of vertex transformations, where costs of edge transformations can only be implied as patches, as done by `bipartite`. In contrast, the `IPFP` heuristic under the `LS-GED` paradigm provides a method to extend `LSAPE-GED`, by including the edge transformations as a quadratic part of GED.

We define a binary matrix $\mathbf{X} \in \{0,1\}^{(|V_1|+1)\times(|V_2|+1)}$ equivalent to an ε-assignment π. As a result, the cost of the transformation π can be formalized as

$$C(\mathbf{x}) = g\mathbf{x}^\top \mathbf{Q}\mathbf{x} + \mathbf{c}^\top \mathbf{x}, \tag{4}$$

where $\mathbf{X}$ is vectorized by the binary vector $\mathbf{x} = \mathtt{vec}(\mathbf{X}) \in \{0,1\}^{(|V_1|+1)(|V_2|+1)}$ by concatenating its rows, $\mathbf{c} = \mathtt{vec}(\mathbf{C})$ is the edit cost vector, and the coefficient g is set to 0.5 if the graphs are undirected and 1 otherwise [1].

The `IPFP` heuristic approximates the GED by adapting the integer projected fixed point (IPFP) algorithm [27] designed for the quadratic assignment problem (QAP) [20,28]. The algorithm is first initialized randomly or by a heuristic such as `bipartite`, and then updated by iterations. In each iteration, a linear approximation is computed by a `LSAPE` solver. Then, the local minimum of the cost and the corresponding binary solution is estimated by a line search [28].

3. Stability of GED Heuristics

The nature of the GED heuristics leads to a drop in computational stability, namely different trials may lead to different results. In the following, we analyze this instability by measuring the variability of the GED approximations over repeated trials. For instance, in the `LSAPE-GED` paradigm, the cost matrix $\mathbf{C}$ may vary given vertex set with different orders, which affects the solution of the LSAPE problem, furthermore causing the instability. Likewise, in the `LS-GED` paradigm, the instability can be traced back to the initial procedure where a random transformation or a GED heuristic such as `bipartite` implying stochasticity may be assigned.

3.1. Measure of (In)Stability

To measure the (in)stability of GED heuristics, we define a criterion named **relative error**. Given a set of graphs $G_1, G_2 \ldots, G_N$, we compute the GED with a heuristic between each pair of graphs N_t times (trials). The relative error E_r is then defined as

$$E_r = \frac{1}{N^2} \sum_{k=1}^{N_t} \frac{\sum_{i,j=1}^{N} \|(\mathtt{ged}^{(k)}(G_i, G_j) - \mathtt{ged}_0(G_i, G_j)\|}{\frac{1}{2}\left(\sum_{i,j=1}^{N} \mathtt{ged}^{(k)}(G_i, G_j) + \mathtt{ged}_0(G_i, G_j)\right)}, \tag{5}$$

where $\mathtt{ged}^{(k)}(G_i, G_j)$ is the approximation of the GED in the k-th trial using a GED heuristic such as Algorithm 1, and $\mathtt{ged}_0(G_i, G_j)$ is the exact GED between G_i and G_j. As evaluating

$\texttt{ged}_0$ is normally impractical, we replace it with the minimum approximation over all trials, namely

$$\underset{0}{\texttt{ged}}(G_i, G_j) = \min_{1 \leqslant k \leqslant N_t} \overset{(k)}{\texttt{ged}}(G_i, G_j).$$

The relative error E_r measures the average ratio between the offsets and the exact GEDs over trials and pairs of graphs. A smaller value indicates higher stability.

3.2. Influential Factors of Stability

Low stability can degrade the performance of GED heuristics, which implies a broader range of the confidence interval in a prediction task such as regression and classification, or instability of the produced graphs in a pre-image task [29]. A straightforward method to mitigate this problem is repeating the GED computation. The minimum cost over repetitions is then chosen as the GED approximation. Strategies have been proposed to refine this method. Well-known ones are the m`bipartite` and m`IPFP`, which are the multi-start counterparts of `bipartite` and `IPFP` [21]. These two heuristics start several initial candidates simultaneously to acquire tighter upper bounds. The stability is concurrently ameliorated. As examined in Section 4, the relative error can be reduced by up to four orders of magnitude. Algorithm 1 presents the procedure of m`IPFP` as an example.

Algorithm 1 Approximation of GED using m`IPFP`

Input: Graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$.
Vertex edit cost c_v, edge edit cost c_e. The number of solutions m.
Output: An approximation $C^\star$ of GED between G_1 and G_2.

Initialize cost $C^\star = C_0 = \infty$.
Let $j = 0$.
while $j < m$ **do**
  Approximate a new cost C_{j+1} with the `IPFP` heuristic.
  if $C^\star > C_{j+1}$ **then**
    $C^\star = C_{j+1}$.
  end if
  $j = j + 1$.
end while
$\texttt{ged}(G_i, G_j) = C^\star$.

Another factor that significantly influences the stability of GED heuristics turns out to be the relative values of vertex and edge edit costs. When vertex costs are markedly larger than edge costs, the GED stability often shows an observable improvement. This phenomenon is detailed in Section 4. When optimized edit cost values are applied, the relative error can be reduced by up to around 30 times compared to using the worst cost values.

Based on the aforementioned information, we analyze the stability with respect to two factors. The first one is the number of random initial candidates of the GED heuristics, namely "# of solutions". For m`bipartite` and m`IPFP`, it is equal to the parameter m, as in Algorithm 1. The second factor is the ratio between vertex and edge edit costs. Let c_{vfs}, c_{vfi}, c_{vfr}, c_{efs}, c_{efi}, $c_{efr} \in \mathbb{R}_+$ be the cost functions associated with, respectively, vertex substitutions, insertions, removals and edge substitutions, insertions, and removals. Then, the ratio is defined as

$$R_{ec} = \frac{\texttt{average}(c_{vfi}, c_{vfr}, c_{vfs})}{\texttt{average}(c_{efi}, c_{efr}, c_{efs})}, \tag{6}$$

where $\texttt{average}(\cdot)$ computes the average value of its inputs.

4. Experiments

In this section, we conduct experimental analyses on the GED stability. First, the influence of the two factors introduced in Section 3.2 is examined, and then the relevance of stability and prediction performance of GED heuristics is verified, taking advantage of a state-of-the-art edit costs learning strategy.

Five well-known public datasets are applied in the experiments: *Alkane* and *Acyclic* are composed of acyclic molecules modeled respectively as unlabeled and vertex-labeled graphs, while *MAO*, *Monoterpens*, and *MUTAG* consist of cycles-included molecules represented as graphs labeled on both vertices and edges. The sizes of datasets vary from 68 to 286 (https://brunl01.users.greyc.fr/CHEMISTRY/, accessed on 9 September 2022).

We exploit two multi-start heuristics, m`bipartite` and m`IPFP`, to evaluate the stability as described in Section 2. The former induces randomness by permuting vertices and consequently the cost matrix for each graph. The two heuristics employ respectively the implementation in the `graphkit-learn` [30] and `GEDLIB` [31] libraries.

4.1. Effects of the Two Factors

Figure 2 shows the effect of the two factors, namely "# of solutions" and the ratio between vertex and edge edit costs R_{ec}, on the relative error E_r defined in (5), considering the m`bipartite` and m`IPFP` heuristics on five datasets. The Figure 2a,c on the left side exhibit how E_r drops with the increase of the "# of solutions" m. In most cases, E_r drops rapidly when the solution number increases from 1 to around a certain number $\mathbb{N}$, and reaches at a relatively small value; the tendency mitigates afterwards. $\mathbb{N}$ is around 20 for m`bipartite` and 10 for m`IPFP`. Take datasets *Alkane* and *Acyclic* for examples. When using m`bipartite`, E_r's on these two datasets drop respectively from 0.23 to 0.08, and from 0.12 to 0.03, as m increases from 1 to 100; when using m`IPFP`, E_r's drop respectively from 0.08 to 5×10^{-6}, and from 0.03 to 7.2×10^{-7}. This result indicates that an adequately large number of solutions is necessary, thus a trade-off decision between stability and time complexity needs to be made for different applications.

The Figure 2b,d on the right side reveal the relation between E_r and the ratio R_{ec}. The edge costs are set to 1 and the vertex costs to be the ratio value (for insertions, removals, and substitutions). The removal costs of vertices (resp. edges) are set to 0 if vertices (resp. edges) are not labeled. For both m`bipartite` and m`IPFP`, E_r is relatively large when the ratio is smaller than 1, namely when edge costs are bigger than vertex costs, and drops with the increase of the ratio. We can observe that a larger ratio leads to higher stability. Take datasets *Alkane* and *Acyclic* for examples. When using m`bipartite`, E_r's on these two datasets drop respectively from 0.5 to 0.16, and from 0.34 to 0.07, as R_{ec} increases from 0.1 to 10; when using m`IPFP`, E_r's drop respectively from 0.03 to 5.6×10^{-3}, and from 0.02 to 6.9×10^{-4}. A possible cause of this phenomenon is that large edge costs amplify the arbitrariness of the edge edit operations. For graphs with n vertices, there are n^2 possible edges that can be inserted, removed, and substituted, which causes more uncertainty when constructing edit paths and computing their costs. Taking `IPFP` for instance, large edge costs lead to a big cost matrix $\mathbf{Q}$ in (4), implying the possibility of more variance on the value of the term $g\mathbf{x}^\top \mathbf{Q}\mathbf{x}$. Many edit costs given by domain experts are in accordance with this empirical rule, such as the ones in [17]. In the next section, we further validate the relevance of stability and prediction performance benefitting from an edit cost learning method.

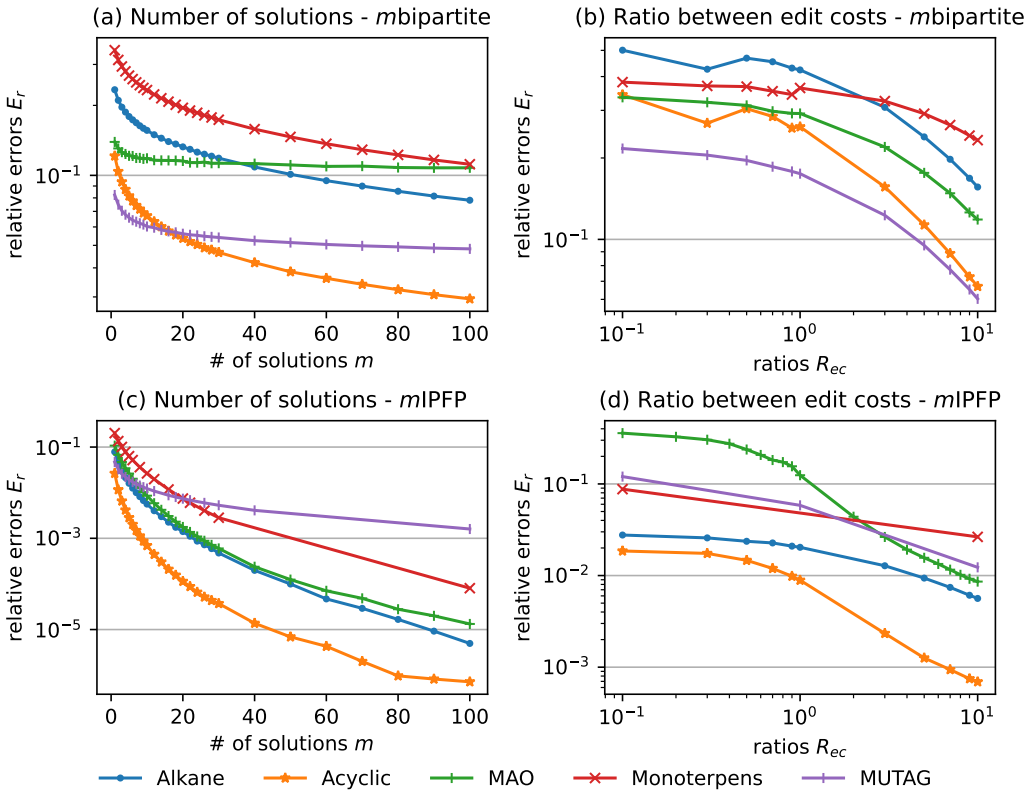

Figure 2. The relative errors of m`bipartite` and m`IPFP` on five datasets with respect to the numbers of solutions m and ratios between vertex and edge edit costs R_{ec}. Colors along with the markers indicate different datasets.

4.2. Stability vs. Prediction Performance

As stated in the Introduction, the choice of edit costs has a major impact on the computation of graph edit distance, and thus on the performance associated with a prediction task. To challenge these predefined expert costs with how they can improve the prediction performance, methods to tune the edit costs and thus the GED were proposed in the literature [23,32,33]. These methods provide an opportunity to observe the connection between the prediction performance of GEDs and the choices of edit costs, which further relate to the GED stability, as examined in Section 4.1.

To inspect this relevance, we utilize a state-of-the-art cost learning algorithm [23], where the edit costs are optimized according to a particular prediction task. An alternate iterative procedure is proposed to preserve the distances in both the graph space (GEDs) and target space (Euclidean or Manhattan distances between targets), where the update on edit costs obtained by solving a constrained linear problem and a re-computation of the optimal edit paths according to the newly computed costs are performed alternately. The GEDs with optimized edit costs are then used to train a k-nearest-neighbors (KNN) regression [34] model. KNN predicts the property value or class of an object based on the value or class of its neighbors. It has been widely used for various types of data, such as traffic [35], infrared vision [36], sensors [37], and graphs [38]. Thus, it is suitable for the current experiments. Experiments on *Alkane* and *Acyclic* show that optimized costs gain a significant improvement in accuracy compared to random or expert costs [1]. Table 1 summarizes the optimized values of edit costs.

Table 1. Average and standard deviation of fitted edit costs' values.

Dataset	Edit Cost	Distance	c_{ni}	c_{nr}	c_{ns}	c_{ei}	c_{er}	c_{es}
Alkane	bipartite	Euclidean	26.45 ± 0.48	26.24 ± 0.60	-	0.13 ± 0.06	0.14 ± 0.09	-
		Manhattan	26.67 ± 0.37	26.63 ± 0.58	-	0.11 ± 0.04	0.11 ± 0.06	-
	IPFP	Euclidean	26.12 ± 0.24	25.88 ± 0.25	-	0.74 ± 0.23	0.78 ± 0.23	-
		Manhattan	25.94 ± 0.38	25.71 ± 0.44	-	0.89 ± 0.30	0.77 ± 0.29	-
Acyclic	bipartite	Euclidean	13.81 ± 0.48	13.83 ± 0.80	10.46 ± 0.40	1.37 ± 0.46	1.45 ± 0.46	1.41 ± 0.09
		Manhattan	13.76 ± 0.39	14.14 ± 0.57	10.28 ± 0.44	1.44 ± 0.20	1.45 ± 0.19	1.45 ± 0.07
	IPFP	Euclidean	11.61 ± 0.45	11.68 ± 0.43	11.07 ± 0.53	4.49 ± 0.30	4.46 ± 0.24	4.48 ± 0.18
		Manhattan	11.52 ± 0.40	11.40 ± 0.40	10.61 ± 0.52	4.50 ± 0.31	4.50 ± 0.31	4.50 ± 0.10

We then examine the stability of the two heuristics when using these edit costs. Figure 3 demonstrates the relations between the relative errors E_r and the ratios R_{ec} between vertex and edge costs (See Section 3.2 and Figure 2 for more details). Therein, the colors represent datasets (i.e., blue for *Alkane* and orange for *Acyclic*), and shapes represent different edit costs, with ✖, ●, and ◆ respectively for the expert costs, the optimized costs using the Euclidean and Manhattan distances. When applying *m*IPFP, for the expert costs, R_{ec}'s for the two datasets are both 1, and E_r's are 0.02 for *Alkane* and 0.013 for *Acyclic*; when using the Euclidean distance, $R_{ec} = 34.21$, $E_r = 0.002$ for *Alkane*, and $R_{ec} = 2.6$, $E_r = 0.002$ for *Acyclic*; when using the Manhattan distance, $R_{ec} = 3.11$, $E_r = 2.33 \times 10^{-3}$ for *Alkane*, and $R_{ec} = 2.55$, $E_r = 2.68 \times 10^{-3}$ for *Acyclic*. It can be observed that the optimized costs correspond to much higher ratios and an order of magnitude lower relative errors than the expert costs. Similar conclusions can be observed for *m*bipartite as well. Thus, an empirical conclusion can be derived: the obtained optimized edit costs correspond to higher stability of GEDs, while obtaining a higher performance.

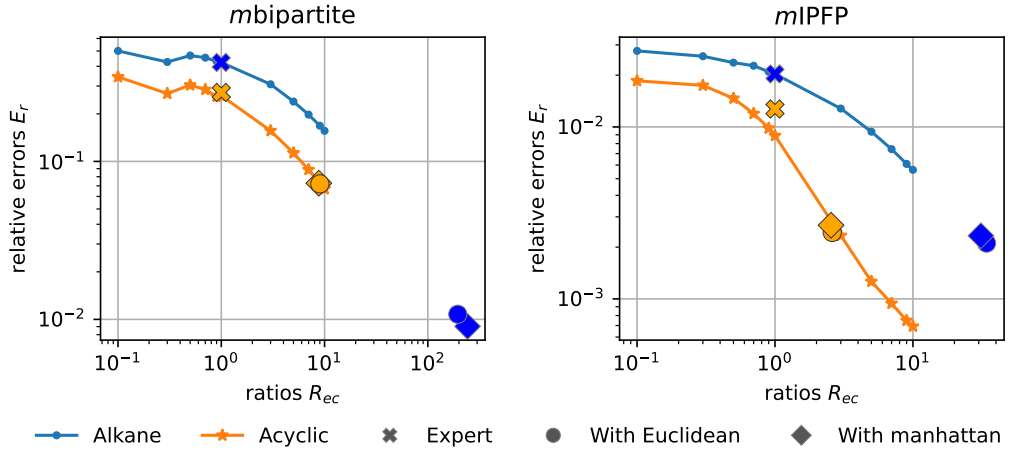

Figure 3. The relative errors of *m*bipartite and *m*IPFP on datasets *Alkane* and *Acyclic* with respect to the ratios R_{ec} between vertex and edge edit costs using different edit costs optimization methods. The colors represent datasets and the shapes of markers represent different edit costs.

5. Conclusions and Future Work

In this paper, we conducted analyses of the GED heuristics' stability, which is the first time it is formally investigated in the literature. After defining an (in)stability measure, namely, the relative error, we show the strong connection of the stability with the number of random initial candidates of multi-start GED heuristics and the relation between vertex and edge edit costs. Experiments on five datasets and two GED heuristics indicate that the proper choice of these factors can reduce the relative error by more than an order of magnitude. A further investigation indicates higher stability of GED computation

corresponds to the optimized edit costs and thus better prediction performance, where an edit cost learning algorithm is applied to optimize the performance and the k-nearest neighbor regression for prediction.

There are still several challenges to address in future work. First, examining other influential factors and conducting theoretical analyses can help deepen understanding of the stability of the heuristics. Second, it will be helpful to perform more thorough experiments, including other state-of-the-art GED heuristics on datasets from a wider range of fields and statistical properties. Third, higher stability comes with the cost of higher time complexity. Methods that can better balance the stability, time complexity, and prediction performance in practice need to be developed.

Author Contributions: Conceptualization, L.J. (Linlin Jia), B.G. and P.H.; methodology, L.J. (Linlin Jia) and B.G.; software, L.J. (Linlin Jia); validation, L.J. (Linlin Jia), B.G. and P.H.; formal analysis, L.J. (Linlin Jia); investigation, L.J. (Linlin Jia); resources, V.T., L.J. (Laurent Joubert), B.G. and P.H.; data curation, L.J. (Linlin Jia); writing—original draft preparation, L.J. (Linlin Jia); writing—review and editing, B.G. and P.H.; visualization, L.J. (Linlin Jia); supervision, V.T., L.J. (Laurent Joubert), B.G. and P.H.; project administration, V.T., L.J. (Laurent Joubert) and P.H.; funding acquisition, V.T., L.J. (Laurent Joubert) and P.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been partially supported by the University of Rouen Normandy, INSA Rouen Normandy, the "Centre National de la Recherche Scientifique" (CNRS), the European Regional Development Fund (ERDF), Labex SynOrg (ANR-11-LABX-0029), Carnot Institut I2C, the graduate school for research XL-Chem (ANR-18-EURE-0020 XL CHEM), the "Région Normandie", China Scholarship Council (CSC), and the French national research agency (ANR) under the grant APi (ANR-18-CE23-0014).

Data Availability Statement: Publicly available datasets were analyzed in this study. These data can be found here: https://brunl01.users.greyc.fr/CHEMISTRY/, accessed on 9 September 2022.

Acknowledgments: The authors would like also to gratefully acknowledge the "Centre Régional Informatique et d'Applications Numériques de Normandie" (CRIANN) for computing resources.

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

References

1. Jia, L. Bridging Graph and Kernel Spaces: A Pre-Image Perspective. Ph.D. Thesis, INSA Rouen Normandie, Saint-Étienne-du-Rouvray, France, 2021.
2. Borgwardt, K.; Ghisu, E.; Llinares-López, F.; O'Bray, L.; Rieck, B. Graph Kernels: State-of-the-Art and Future Challenges. *arXiv* **2020**, arXiv:2011.03854.
3. Goyal, P.; Ferrara, E. Graph embedding techniques, applications, and performance: A survey. *Knowl.-Based Syst.* **2018**, *151*, 78–94. [CrossRef]
4. Kriege, N.M.; Johansson, F.D.; Morris, C. A survey on graph kernels. *Appl. Netw. Sci.* **2020**, *5*, 1–42. [CrossRef]
5. Gaüzère, B.; Brun, L.; Villemin, D. Graph kernels in chemoinformatics. In *Quantitative Graph Theory Mathematical Foundations and Applications*; Dehmer, M., Emmert-Streib, F., Eds.; CRC Press: Boca Raton, FL, USA, 2015; pp. 425–470.
6. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Philip, S.Y. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4-24. [CrossRef] [PubMed]
7. Balcilar, M.; Renton, G.; Héroux, P.; Gaüzère, B.; Adam, S.; Honeine, P. Analyzing the Expressive Power of Graph Neural Networks in a Spectral Perspective. In Proceedings of the International Conference on Learning Representations, Virtual Event, Austria, 3–7 May 2021.
8. Ortega, A.; Frossard, P.; Kovačević, J.; Moura, J.M.; Vandergheynst, P. Graph signal processing: Overview, challenges, and applications. *Proc. IEEE* **2018**, *106*, 808–828. [CrossRef]
9. Dong, X.; Thanou, D.; Toni, L.; Bronstein, M.; Frossard, P. Graph signal processing for machine learning: A review and new perspectives. *IEEE Signal Process. Mag.* **2020**, *37*, 117–127. [CrossRef]
10. Zhang, C.; Florêncio, D.; Chou, P.A. *Graph Signal Processing—A Probabilistic Framework*; Tech. Rep. MSR-TR-2015-31; Microsoft Research Lab: Redmond, WA, USA, 2015.
11. Richiardi, J.; Achard, S.; Bunke, H.; Van De Ville, D. Machine learning with brain graphs: Predictive modeling approaches for functional imaging in systems neuroscience. *IEEE Signal Process. Mag.* **2013**, *30*, 58–70. [CrossRef]

12. Richiardi, J.; Van De Ville, D.; Riesen, K.; Bunke, H. Vector space embedding of undirected graphs with fixed-cardinality vertex sequences for classification. In Proceedings of the 2010 20th International Conference on Pattern Recognition, Istanbul, Turkey, 23–26 August 2010; pp. 902–905.
13. Bunke, H.; Allermann, G. Inexact graph matching for structural pattern recognition. *Pattern Recognit. Lett.* **1983**, *1*, 245–253. [CrossRef]
14. Sanfeliu, A.; Fu, K.S. A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Syst. Man, Cybern.* **1983**, *SMC-13*, 353–362.
15. Zeng, Z.; Tung, A.K.; Wang, J.; Feng, J.; Zhou, L. Comparing stars: On approximating graph edit distance. *Proc. Vldb Endow.* **2009**, *2*, 25–36. [CrossRef]
16. Neuhaus, M.; Riesen, K.; Bunke, H. Fast suboptimal algorithms for the computation of graph edit distance. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*; Springer: Berlin, Germany, 2006; pp. 163–172.
17. Abu-Aisheh, Z.; Gaüzère, B.; Bougleux, S.; Ramel, J.Y.; Brun, L.; Raveaux, R.; Héroux, P.; Adam, S. Graph edit distance contest: Results and future challenges. *Pattern Recognit. Lett.* **2017**, *100*, 96–103. [CrossRef]
18. Blumenthal, D.B.; Boria, N.; Gamper, J.; Bougleux, S.; Brun, L. Comparing heuristics for graph edit distance computation. *VLDB J.* **2020**, *29*, 419–458. [CrossRef]
19. Riesen, K. Structural pattern recognition with graph edit distance. In *Advances in Computer Vision and Pattern Recognition*; Springer: Berlin, Germany, 2015.
20. Bougleux, S.; Gaüzère, B.; Brun, L. Graph edit distance as a quadratic program. In Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 4–8 December 2016; pp. 1701–1706.
21. Daller, É.; Bougleux, S.; Gaüzère, B.; Brun, L. Approximate graph edit distance by several local searches in parallel. In Proceedings of the 7th International Conference on Pattern Recognition Applications and Methods, Funchal, Portugal, 16–18 January 2018.
22. Bunke, H. Error correcting graph matching: On the influence of the underlying cost function. *IEEE Trans. Pattern Anal. Mach. Intell.* **1999**, *21*, 917–922. [CrossRef]
23. Jia, L.; Gaüzère, B.; Yger, F.; Honeine, P. A Metric Learning Approach to Graph Edit Costs for Regression. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*; Springer: Berlin, Germany, 2021; pp. 238–247.
24. Bougleux, S.; Brun, L. Linear sum assignment with edition. *arXiv* **2016**, arXiv:1603.04380.
25. Kuhn, H.W. The Hungarian method for the assignment problem. *Nav. Res. Logist. Q.* **1955**, *2*, 83–97. [CrossRef]
26. Munkres, J. Algorithms for the assignment and transportation problems. *J. Soc. Ind. Appl. Math.* **1957**, *5*, 32–38. [CrossRef]
27. Leordeanu, M.; Hebert, M.; Sukthankar, R. An integer projected fixed point method for graph matching and map inference. *Adv. Neural Inf. Process. Syst.* **2009**, *22*, 1114–1122.
28. Bougleux, S.; Brun, L.; Carletti, V.; Foggia, P.; Gaüzere, B.; Vento, M. A quadratic assignment formulation of the graph edit distance. *arXiv* **2015**, arXiv:1512.07494.
29. Jia, L.; Gaüzère, B.; Honeine, P. A graph pre-image method based on graph edit distances. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*; Springer: Berlin, Germany, 2021; pp. 216–226.
30. Jia, L.; Gaüzère, B.; Honeine, P. graphkit-learn: A Python library for graph kernels based on linear patterns. *Pattern Recognit. Lett.* **2021**, *143*, 113–121. [CrossRef]
31. Blumenthal, D.B.; Bougleux, S.; Gamper, J.; Brun, L. GEDLIB: A C++ Library for graph edit distance computation. In Proceedings of the International Workshop on Graph-Based Representations in Pattern Recognition, Tours, France, 19–21 June 2019; Springer: Berlin, Germany, 2019; pp. 14–24.
32. Neuhaus, M.; Bunke, H. Automatic learning of cost functions for graph edit distance. *Inf. Sci.* **2007**, *177*, 239–247. [CrossRef]
33. Cortés, X.; Conte, D.; Cardot, H. Learning edit cost estimation models for graph edit distance. *Pattern Recognit. Lett.* **2019**, *125*, 256–263. [CrossRef]
34. Altman, N.S. An introduction to kernel and nearest-neighbor nonparametric regression. *Am. Stat.* **1992**, *46*, 175–185.
35. Cai, L.; Yu, Y.; Zhang, S.; Song, Y.; Xiong, Z.; Zhou, T. A sample-rebalanced outlier-rejected *k*-nearest neighbor regression model for short-term traffic flow forecasting. *IEEE Access* **2020**, *8*, 22686–22696. [CrossRef]
36. Siriborvornratanakul, T. Color and Active Infrared Vision: Estimate Infrared Vision of Printed Color Using Bayesian Classifier and K-Nearest Neighbor Regression. In *Pacific Rim Conference on Multimedia*; Springer: Berlin, Germany, 2015; pp. 518–527.
37. Naimi, A.; Deng, J.; Shimjith, S.; Arul, A.J. Fault Detection and Isolation of a Pressurized Water Reactor Based on Neural Network and K-Nearest Neighbor. *IEEE Access* **2022**, *10*, 17113–17121. [CrossRef]
38. Kang, S. K-nearest neighbor learning with graph neural networks. *Mathematics* **2021**, *9*, 830. [CrossRef]