Graph edit distance: restrictions to be a metric

Francesc Serratosa[1]

Universitat Rovira i Virgili, Tarragona, Catalonia, Spain

Abstract

In the presentation of the graph edit distance in 1983 and other newer bibliography, authors state that it is necessary to apply the distance restrictions (non-negativity, identity of indiscernible elements, symmetry and triangle inequality) to each of the edit functions (insertion, deletion and substitution of nodes and edges) involved in the process of computing the graph edit distance to make the graph edit distance a true distance. Moreover, graph edit distance algorithms presented in the last three decades have been based on mapping the edit path that transforms a graph into the other one into a bijection of the graphs in which some null nodes have been added. In this paper, we show that the triangle inequality does not need to be imposed in each edit function if the graph edit distance is defined through an edit path; however, it is necessary if it is defined as a graph bijection. This is an important finding since the triangle inequality is the only restriction that relates different edit functions and concerns the process of tuning the edit functions given a specific application. Hence, on one hand, it would encourage research to define new algorithms based on the edit path instead of the graph bijection and, on the other hand, use edit functions without the restriction,

[1] Email: francesc.serratosa@urv.cat

for instance, that the sum of the costs of insertion and deletion of a pair of nodes has to be larger or equal than the cost of substituting them, which could increase the recognition ratio of a concrete application.

Keywords: Graph edit distance, distance definition, sub-optimality.

---

1. Introduction

Attributed graphs have been of crucial importance in pattern recognition for more than four decades [1], [2], [3] since they have been used to model several kinds of problems. Interesting reviews of techniques and applications are [4], [5], [6], [7]. If elements in pattern recognition are modelled through attributed graphs, error-tolerant graph-matching algorithms are needed that aim to compute a matching between nodes of two attributed graphs that minimises some kind of objective function. To that aim, one of the most widely used methods to evaluate an error-correcting graph isomorphism is the graph edit distance [8], [9], [10].

Input parameters of the graph edit distance are the pair of attributed graphs to be compared and also other calibration parameters. These parameters have to be tuned in order to maximise the recognition ratio in a classification scenario. In the first papers where the graph edit distance was defined [1], [2], the graph edit distance was considered a distance subject to certain restrictions. Since then, no researcher has analysed the validity of these restrictions again. Moreover, the book [11], which analyses in depth the graph edit distance and certain algorithms that have appeared to solve it, also suggests the same restrictions. In this paper, we show that these restrictions can be relaxed while the graph edit distance continues to be a true distance. This relaxation is important since it turns out that, in several analysed classification

scenarios with different databases, the recognition ratio is maximised in the combination of parameters where these restrictions do not hold.

Moreover, when elements of a database are represented by graphs, metric-trees are applied on these databases [12]. In this case, if the graph-matching algorithm is based on a sub-optimal method, the number of false rejections may increase. For this reason, the whole restrictions on the graph matching parameters are usually considered. If we can partially relax some of these restrictions while keeping the graph edit distance as a true distance, we can widen the parameters domain and therefore set them at a point where the query precision becomes higher.

Attributed graphs have the main advantage of having the versatile ability to properly define a prototype through their semantic and structural information. Nevertheless, they have the drawback that the computational cost of comparing graphs is exponential with regard to the order of the graphs. Moreover, their dissimilarities are not Euclidean, in other words, they do not represent the distances between points in a Euclidean space. For this reason, some methods have been presented to generate a prototype for a set of graphs, with the aim of reducing the runtime and increasing the representational power of the set. For instance, in [13], [14], [15] certain structures are presented, which take into consideration the first and second order relations between nodes and edges. More recently, in [16], authors have presented a method for constructing a prototype by adopting a minimum description length approach. Furthermore, several embedding methods have been presented and analysed in depth [17]. Other methods, [18], [19], use permutation invariants computed from the trace of the heat kernel to characterise graphs for the purposes of measuring similarity and clustering.

The first papers that defined the graph edit distance and its computation, [1], [2], presented an algorithm that deduces the graph edit distance between two graphs as the minimum cost to transform one graph into another. Nevertheless, and with the aim of reducing the computational effort, the current graph matching algorithms deduce the graph edit distance as the minimum cost of all possible correspondences between nodes of the involved graphs (or almost all correspondences, in the case of sub-optimal computation), [11].

Concerning this change of paradigm, this paper shows that:

a) If the graph edit distance is defined through graph transformations (old method) then fulfilling the triangle inequality in the edit functions is not necessary.

b) If the graph edit distance is defined through node graph correspondences (current method) then fulfilling the triangle inequality between edit functions is necessary.

c) The edit functions that maximise the recognition ratio do not fulfil the triangle inequality in the experiments we have carried out.

And thus, we conclude that:

a) If we want to use the current paradigm and define the graph edit distance as a true distance, then the edit functions cannot be tuned at the point where the recognition ratio is maximised.

b) If we want to maximise the recognition ratio, then we have two options:

    1) Use a current algorithm but analyse how it is influenced by the fact that the graph edit distance is not a true distance.

2) Use a graph matching algorithm based on computing the edit paths, although it could be more computationally expensive.

The outline of the paper is as follows; in Section 2, we define the attributed graphs and the graph edit distance. In Section 3, we explain the restrictions that a function needs to fulfil to be considered a metric and we reprint the restrictions proposed in the presentation of the graph edit distance [1]. Moreover, we show that some restrictions are not necessary. Finally, in Section 4, we show a practical experiment in which the recognition ratio is maximised in the parameter values where the initial restrictions were not fulfilled. Section 5 concludes the paper. Table 1 summarises the main notation of this paper.

2. Attributed graphs and graph edit distance

G, G' and G'': Attributed graphs. v_i and v'_a: Graph nodes in G and G'. e_{ij} and e'_{ab}: Graph edges in G and G'. $(e_1, \dots, e_t, \dots e_k)$: An edit path between G and G'. $E(G, G')$: Set of edit paths $(e_1, \dots, e_t, \dots e_k)$ that transform G into G'. vs, vd, vi: Set of node substitutions, deletions and insertions in $E(G, G')$. es, ed, ei: Set of edge substitutions, deletions and insertions in $E(G, G')$. C_{vs} and C_{es}: Cost of a node and an edge substitution operation. C_{vd} and C_{ed}: Cost of a node and an edge deletion operation. C_{vi} and C_{ei}: Cost of a node and an edge insertion operation. f: A bijection between nodes of G and G'. $B(G, G')$: Set of bijections f between nodes of G and G'. GED: Graph edit distance. $GEDpath$ and $GEDbij$: GED computed through edit paths and bijections.

Table 1. Summary of notations.

Let $G = (\Sigma_v, \Sigma_e, \gamma_v, \gamma_e)$ be an attributed graph representing a first object, and $G' = (\Sigma'_v, \Sigma'_e, \gamma'_v, \gamma'_e)$ be an attributed graph representing a second object. $\Sigma_v = \{v_i\}$ represents the set of nodes and $\Sigma_e = \{e_{ij}\}$ represents the set of edges. Functions γ_v, γ'_v, γ_e and γ'_e assign attribute values in any domain to nodes and edges respectively.

Graph edit distance [8], [10], [9], [11] is the most well-known and used distance function between attributed graphs. It is defined as the minimum amount of required distortion to transform one graph into another. To this end, a number of distortion or edit functions consisting of deletion, insertion, and substitution of nodes and edges are defined. A cost on these edit functions is introduced to quantitatively evaluate the edit operations. The basic idea is to assign a penalty cost to each edit operation according to the amount of distortion that it introduces in the transformation.

More formally, given two attributed graphs G and G', let $E(G, G')$ denote the set of all edit paths $(e_1, e_2, \dots e_k)$ that transform G into G', then the graph edit distance, called $GEDpath$, is defined as,

(1)

$$GEDpath(G, G') = \min_{\forall (e_1, e_2, \dots e_k) \in E(G, G')} \left\{ \underset{(e_1, e_2, \dots e_k)}{CED(G, G')} \right\}$$

Where CED is the cost of the edit path,

(2)

$$\underset{(e_1,e_2,\dots e_k)}{CED}(G,G') = \sum_{\forall e_t \in vs} C_{vs}(e_t) + \sum_{\forall e_t \in es} C_{es}(e_t) + \sum_{\forall e_t \in vd} C_{vd}(e_t)$$

$$+ \sum_{\forall e_t \in ed} C_{ed}(e_t) + \sum_{\forall e_t \in vi} C_{vi}(e_t) + \sum_{\forall e_t \in ei} C_{ei}(e_t)$$

vs, vd and vi represent the set of node substitutions, deletions and insertions. C_{vs}, C_{vd} and C_{vi} represent the cost of the substitution, deletion and insertion operations on nodes. Similar definitions apply for the edges and expressions es, ed, ei, C_{es} C_{ed} and C_{ei}.

The six cost functions have the edit operation e_t as a parameter. Nevertheless, these functions depend on the attributes on the nodes or arcs that are the input of the edit operations. For the rest of the paper, we suppose that:

If $e_t \in vs$ then $e_t = [v_a\ , v_i']$.

If $e_t \in es$ then $e_t = [e_{ab}, e_{ij}']$.

If $e_t \in vd$ then $e_t = [v_a\ ]$.

If $e_t \in ed$ then $e_t = [e_{ab}]$.

If $e_t \in vi$ then $e_t = [v_i']$.

If $e_t \in ei$ then $e_t = [e_{ij}']$.

In the last three decades, a number of methods addressing the high computational complexity of graph edit distance computation have been proposed. Probabilistic relaxation labelling [20], [21] adopts a Bayesian perspective on graph edit distance and iteratively applies edit operations to improve a maximum a posteriori criterion. As an alternative to this hill climbing approach, genetic algorithms have been proposed for optimisation in [22]. In [23], a linear programming method for

computing the edit distance of graphs with unlabelled edges is reported. And also, dominant sets have been applied to sub-optimally compute the edit distance [24]. Finally, in [25] the graph edit distance is approximated by the Hausdorff distance. Recently, a new algorithm called bipartite (BP) [26] and two other versions of this algorithm have appeared that are called fast bipartite [27] (FBP) and square fast bipartite [28], [29] (SFBP). Some methods [30], [31] improve this distance and obtain a new correspondence starting from the correspondence computed by BP, FBP or SFBP at the expense of increasing the runtime. Recently, an algorithm to deduce a sub-optimal graph edit distance in linear cost has been presented [32] and other methods also return sub-optimal graph edit distances that tend to be more precise than [26] at the expense of increasing the computational cost [33], [34], [35].

These algorithms do not work directly on the edit path. Instead, they redefine the graph edit distance through a bijection between nodes of both graphs. To do so, the involved graphs need to have the same order (sometimes, only in the theory) and for this reason, algorithms assume some nodes are added with a concrete attribute (they are usually called null nodes).

Formally, given two attributed graphs G and G', in which some null nodes have been added to have the same order and given $B(G, G')$ that denotes the set of all bijections that map G into G', then, the graph edit distance, called $GEDbij$, is defined as,

(3)

$$GEDbij(G, G') = \min_{\forall f \in B(G,G')} \left\{ CED(\underset{f}{G}, G') \right\}$$

Where CED is the cost of the node-to-node bijection,

$$CED_{f}(G, G') = \sum_{\forall v_a \in \Sigma_v} C_v\left(v_a, f(v_a)\right) + \sum_{\forall e_{ab} \in \Sigma_e} C_e\left(e_{ab}, f(e_{ab})\right) \quad (4)$$

In this new definition, there is only one cost function on nodes, C_v, and on edges, C_e. While comparing to the definition in Equation (1), mapping a non-null node to a non-null node can be seen as a substitution operation, mapping a non-null node to a null node can be seen as a deletion operation and mapping a null node to a non-null node can be seen as a deletion operation.

3. Metric definition and graph edit distance restrictions

A metric is a function that defines a dissimilarity between elements of a set, such as x, y or z. The domain is $[0, \infty)$ and it holds the following restrictions for all elements in the set [36]:

(5)

1) Non-negativity: $\mathrm{dist}(x, y) \geq 0$.

2) Identity of indiscernible elements: $\mathrm{dist}(x, y) = 0 \Leftrightarrow x = y$.

3) Symmetry: $\mathrm{dist}(x, y) = \mathrm{dist}(y, x)$

4) Triangle inequality: $\mathrm{dist}(x, y) \leq \mathrm{dist}(x, z) + \mathrm{dist}(z, y)$

The paper that defined the graph edit distance [1] considered these restrictions and translated them in the case of the edit functions (Lemma 5 in [1]). These restrictions were also rewritten in the book [11] (section 2.1.1). Thus, the main idea is that if the edit functions, seen as elements in a specific domain, fulfil the metric restrictions

(Equation 5), it is certain that the graph edit distance is a true distance. This is because the graph edit distance is a linear function of the edit functions. Specifically, they state the distance restrictions in the following three points, Only the nomenclature has been changed with regard to [1], moreover it is written $C_{\cdot}(\cdot)$ instead of $C_{\cdot}([\cdot])$.

1) A. $C_{vs}(v_a, v_i') > 0$ if $\gamma_v(v_a) \neq \gamma_v'(v_i')$

   $C_{vs}(v_a, v_i') = 0$ otherwise

   B. $C_{es}(e_{ab}, e_{ij}') > 0$ if $\gamma_e(e_{ab}) \neq \gamma_v'(e_{ij}')$

   $C_{es}(e_{ab}, e_{ij}') = 0$ otherwise

   C. $C_{vd}(v_a, v_i') > 0$

   D. $C_{ed}(e_{ab}, e_{ij}') > 0$

   E. $C_{vi}(v_a, v_i') > 0$

   F. $C_{ei}(e_{ab}, e_{ij}') > 0$

2) A. $C_{vs}(v_a, v_i') = C_{vs}(v_i', v_a)$

   B. $C_{es}(e_{ab}, e_{ij}') = C_{es}(e_{ij}', e_{ab})$

   C. $C_{vd}(v_a) = C_{vi}(v_i')$

   D. $C_{ed}(e_{ab}) = C_{ei}(e_{ij}')$

3) A. $C_{vs}(v_a, v_i') \leq C_{vs}(v_a, v_j'') + C_{vs}(v_j'', v_i')$

   B. $C_{es}(e_{ab}, e_{ij}') \leq C_{es}(e_{ab}, e_{kl}'') + C_{es}(e_{kl}'', e_{ij}')$

   C. $C_{vs}(v_a, v_i') \leq C_{vd}(v_a) + C_{vi}(v_i')$

D. $C_{es}(e_{ab}, e'_{ij}) \leq C_{ed}(e_{ab}) + C_{ei}(e'_{ij})$

4. Redefining the restrictions for the graph edit distance

In the first part of this section, we demonstrate in Theorem 1 that Points 3.C and 3.D (the ones that refers to the triangle inequality between substitution, deletion and insertion costs) do not needed to be imposed for the $GEDpath$ to be a true distance. Moreover, we also show in Theorem 2 that they are needed to be imposed to be the $GEDbij$ a true distance. In the second part of the section, we concretise these finding in the string edit distance and tree edit distance cases.

Theorem 1.

The $GEDpath$ is a true distance if points 1., 2., 3.A and 3.B are fulfilled.

Proof:

First, if points 1. and 2. are fulfilled, then it is trivial to realise that the non-negativity: $GEDpath(G, G') >= 0$; the identity of indiscernible elements: $GEDpath(G, G') = 0 \Leftrightarrow G = G'$; and the symmetry properties: $GEDpath(G, G') = GEDpath(G', G)$ are fulfilled in the graph edit distance.

Second, we have to prove the triangle inequality, which means that $GEDpath(G, G') \leq GEDpath(G, G'') + GEDpath(G'', G')$ always holds. Suppose that $e_{G,G'} \in E(G, G')$, $e_{G,G''} \in E(G, G'')$ and $e_{G'',G'} \in E(G'', G')$ are the edit path that generate the minimum cost CED in $E(G, G')$, $E(G, G'')$ and $E(G'', G')$, respectively. In other words, the ones used to compute $GEDpath(G, G')$, $GEDpath(G, G'')$ and $GEDpath(G'', G')$. We define a new edit path $\hat{e}_{G,G'}$ that transforms G into G' as a

concatenation of $e_{G,G''}$ and $e_{G'',G'}$. By construction, it holds that $CED_{\hat{e}_{G,G'}}(G,G')$= $CED_{e_{G,G''}}(G,G'')$+ $CED_{e_{G'',G'}}(G'',G')$ and by definition of $CED_{e_{G,G'}}(G,G')$, it holds that $CED_{e_{G,G'}}(G,G') \leq CED_{\hat{e}_{G,G'}}(G,G')$. Therefore, $GEDpath(G,G') = CED_{e_{G,G'}}(G,G') \leq CED_{\hat{e}_{G,G'}}(G,G')= GEDpath(G,G'') + GEDpath(G'',G')$.

■

Theorem 2.

The $GEDbij$ is NOT a distance if points 3.C or 3.D are NOT fulfilled.

Proof:

We demonstrate this theorem through a simple example. Figure 1 shows three graphs composed of only one and two nodes. The attributes on the nodes are **a**, **b** and **c**. Moreover, we define $C_{vs} = 3$ if the attributes are different and $C_{vs} = 0$ otherwise. And also $C_{vd} = C_{vi} = 1$. Clearly, these edit function definitions do not hold point 3.C.

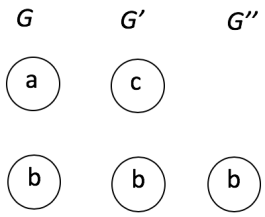


Figure 1: Attributed graphs G, G' and G''. Letters **a**, **b** and **c** are the node attributes.

Figure 2 shows the bijections that obtain the minimum value in Equation (4) between these graphs computed through Equation (3). It holds that $GEDbij(G,G') = 3$, $GEDbij(G,G'') = 1$ and $GEDbij(G'',G') = 1$ and we conclude that the triangle inequality is not fulfilled.

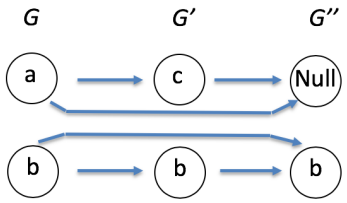


Figure 2: Optimal bijections between graphs G, G' and G''.

■

Finally, note that $GEDbij(G, G') = 3$ but $GEDpath(G, G') = 2$. This is because the path that generates the minimum value in Equation (2) is composed of two edit operations, delete the node in G that has attribute **a**, $C_{vd} = 1$, and insert a new node with attribute **c**, $C_{vi} = 1$. This edit path cannot be defined in a graph bijection because the bijection only maps each node once.

4.1. String and tree edit distance

Trees and strings are specific cases of graphs. In the first, loops are not accepted. In the second, each node has only one input and output edge except one node that has only one input edge and another node that has only one output edge. Thus, while demonstrating these properties on graphs we automatically demonstrate them on the tree edit distance and the string edit distance.

There are some papers that present new algorithms (optimal or suboptimal) to find the correspondence between strings or between trees. Others present new edit functions, for instance merging or splitting nodes in the trees ([37] and [38]). Nevertheless, in all of these papers, the restrictions on these edit operations to be a metric are not analysed. In the string case, there is an interesting paper [39] that proves

that all combinations of deletion, insertion and substitution costs which have the same ratio $(C_{vd} + C_{vi})/C_{vs}$ generate exactly the same correspondences. Although not mentioned in that paper, the cases in which this ratio is lower than one were considered as non-metric parameter settings. Thanks to Theorem 1, all combinations are now considered metric parameter settings if the string edit distance is computed through the edit path, C_{vs} is a distance and C_{vd} and C_{vi} are non-negative constants.

5. Practical experiment

The aim of this section is to show that the highest classification ratios appear in the settings of the edit functions, so that the triangle inequality between substitution, deletion and insertion is not fulfilled. Until now, it was thought that in these cases the graph edit distance would not have to be considered a metric and for this reason they were rarely applied. Conversely, we observe that the algorithms proposed to learn the edit functions, such as [40], do not intrinsically take into consideration these restrictions. Now, we know that the graph edit distance continues to be a metric (if it is computed through the edit path) and therefore it is worth defining the edit functions in the values, so that the recognition ratio is maximised even if the triangle inequality on the edit functions is not fulfilled.

We have used the following five public graph databases: Letter-high, Letter-med, Letter-low, Grec and Coil-Rag. The most common features and details of these databases are summarised in [41], [42] and shown in Table 2. For instance, the number of graphs in the test, reference and validation sets, the average and maximum number of nodes and edges or the type and number of attributes on nodes and edges.

Table 2. Main features of databases.

		Coil-RAG	*Grec*	*Letter Low*	*Letter Med*	*Letter High*
Number Graphs	**Learn**	2400	286	750	750	750
	Test	1000	528	750	750	750
	Validation	500	286	750	750	750
Number	**classes**	100	22	15	15	15
Number	**attributes**	64	2	2	2	2
	Description	SURF	(x,y)	(x,y)	(x,y)	(x,y)
Avg. num.	**nodes**	3.01	11.4	4.6	4.6	4.6
Avg. num.	**edges**	6	23.8	6.2	6.4	9
Max.	**nodes**	6	24	8	9	9
Max.	**edges**	24	58	12	14	18

The edit functions have been defined as follows: C_{vs} is the Euclidean distance between node attributes. C_{es} is not defined since these databases do not have attributes on the edges. Finally, we impose $C_{vd} = C_{ed} = C_{vi} = C_{ei}$ and they are set as constants.

Table 3 shows the maximum, minimum mean and standard deviation of the node substitution costs in the whole databases. Finally, the last column is half of the maximum value. Given that the insertion and deletion edit functions are constant, to fulfil the triangle inequality on the edit operations, we have to assure that these constants are set, so that $C_{vd} + C_{vi} \geq \max(C_{vs})$. Hence, the minimum value of C_{vd} and C_{vi} would have to be $C_{vd} = C_{vi} = \frac{1}{2} max\ (C_{vs})$.

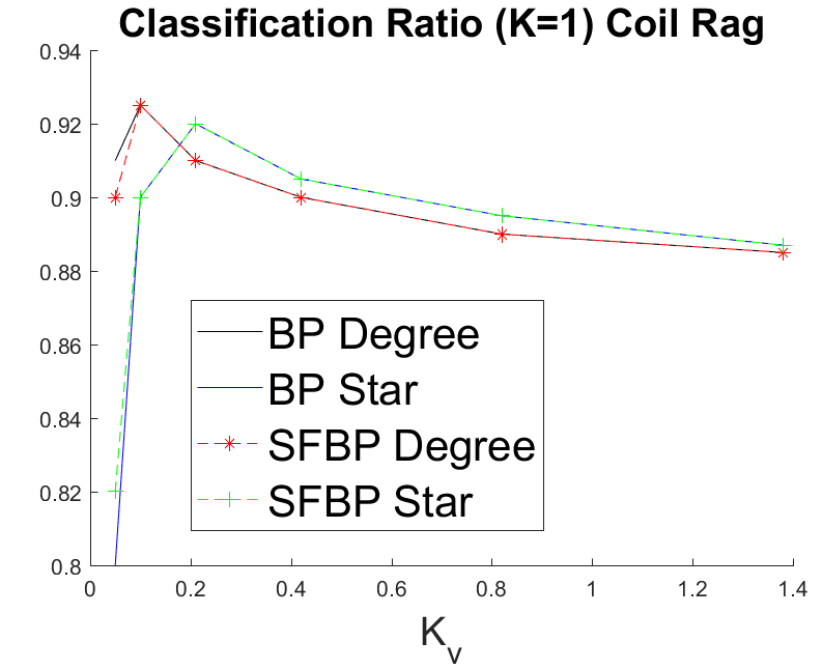
Table 3. Different statistics of the node substitution costs C_{vs}.

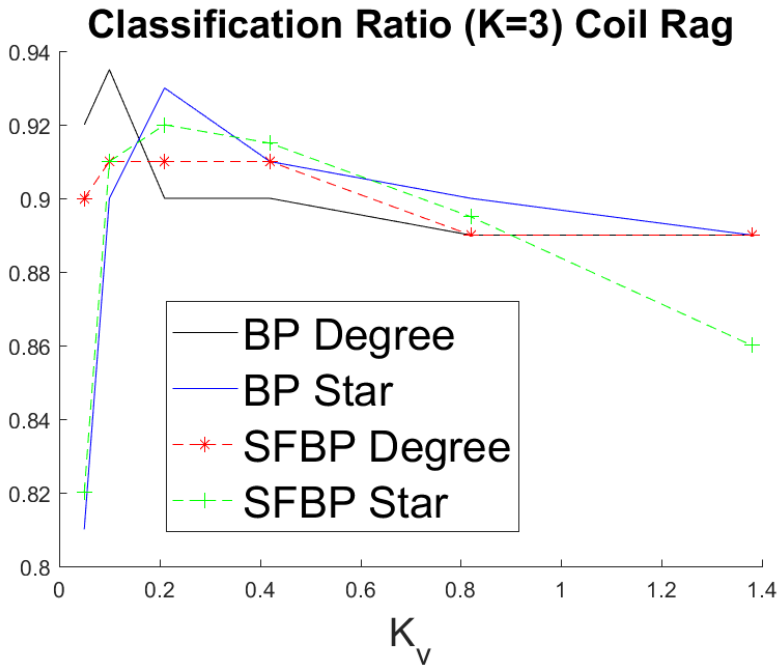
	Max	**Min**	**Mean**	**StdDv**	**½ Max**
Coil Rag	1.38	0	0.82	0.36	0.69
Grec	823.2	0	246.5	128.5	411.6
Letter low	4.09	0	1.61	0.82	2.04
Letter med	4.092	0	1.64	0.82	2.04
Letter high	4.95	0	1.69	0.83	2.46

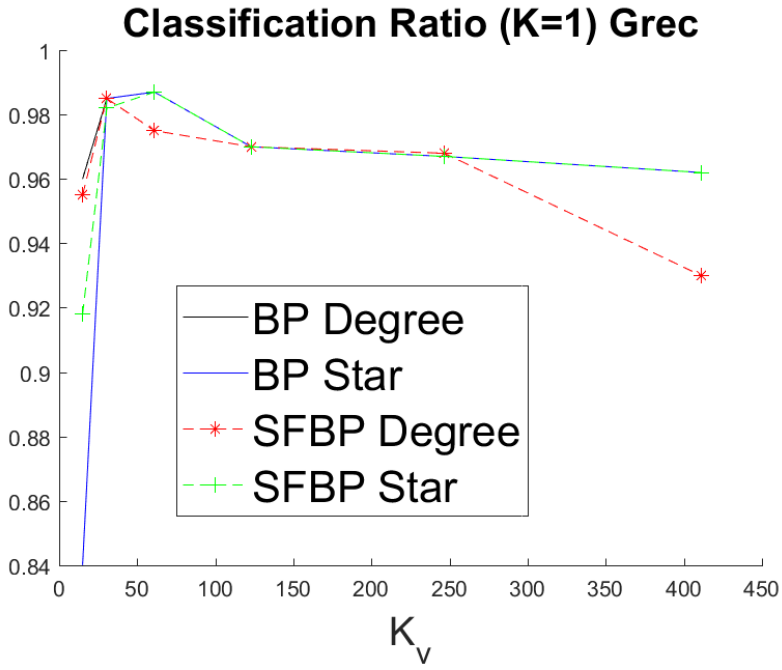
Figure 3 shows the classification ratio of these databases using k-nearest neighbours with k=1 and k=3. Clearly, the plots could be different with other options of k. We have also experimented with k=5 but we do not show these experiments since plots were almost the same as with k=3. Note there are few differences between k=1 and k=3. As examples of graph matching algorithms, we have used the bipartite graph matching, BP, [26] and the Square Fast bipartite graph matching algorithm, SFBP, [28], with two different local structures: degree and star. Moreover, in the Letter databases, we have also used the BP-Beam [31], which is an exact algorithm. In these three databases, the recognition ratio obtained by BP-Beam was almost the same as the ones obtained by BP and SFBP except for the minimum value of K_v. We have not experimented with BP-Beam and Coil Rag and Grec databases due to the runtime was very high.

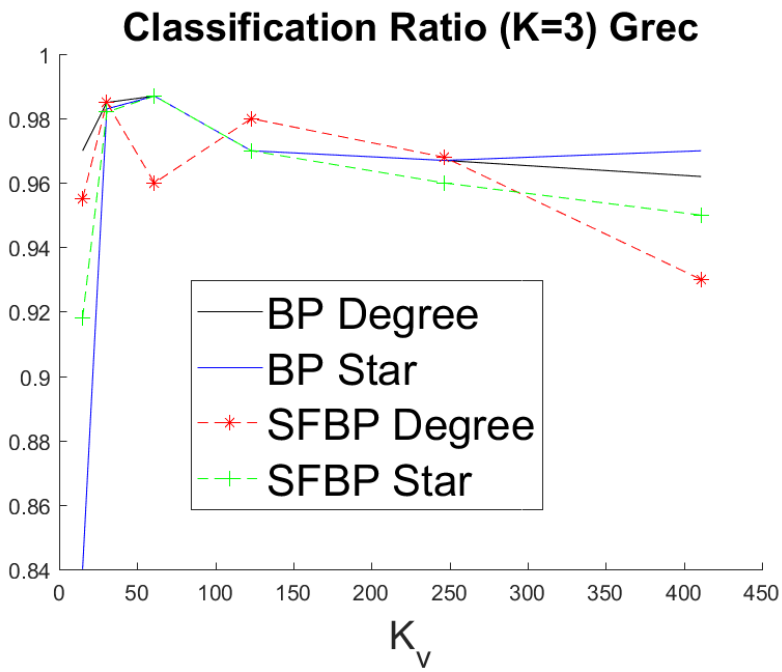
We observe in the five databases that the recognition ratio is maximised at the value of the insertion and deletion cost, which is much smaller than half of the maximum value of the substitution cost. We could consider that the maximum values are highly influenced by the outliers generated by noise on the data and that they are few extreme outliers. In the case of the Letter databases, the noise is a random value that is added on the node position. Thus, the higher the noise, the more chance the database has to have a node with an extreme position. For this reason, the maximum value of the Letter high database is larger than the maximum value of the Letter low database (see Table 3). Nevertheless, the mean is almost the same.

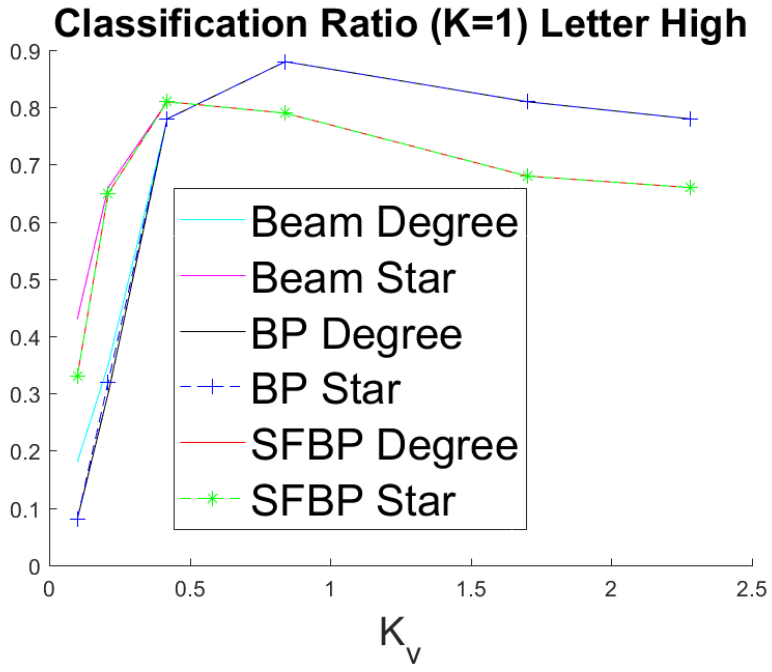
Thus, from the practical experiment, we could deduce that an initial value for tuning the insertion and deletion costs could be a value close to half of the mean of the substitutions, which is much smaller than half of the maximum value.

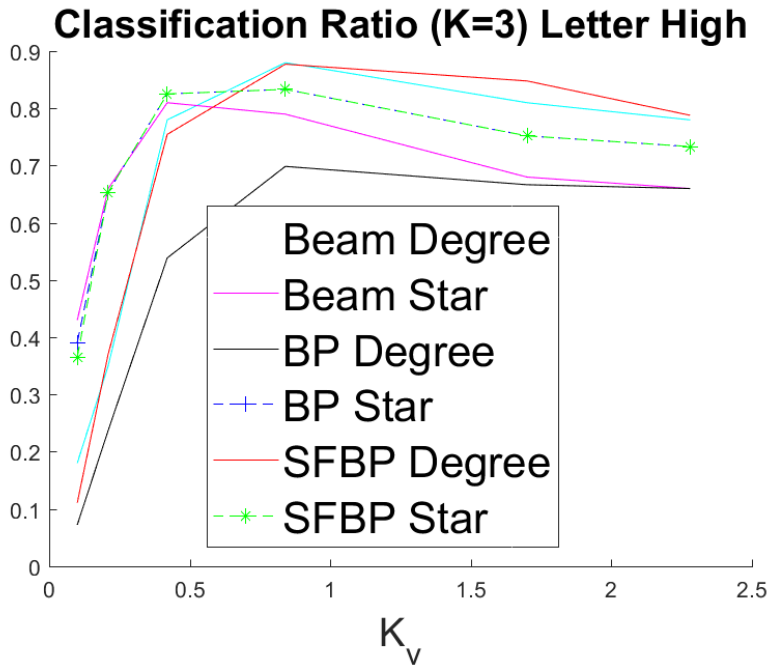

Classification Ratio (K=1) Coil Rag
0.94
0.92
0.9
0.88
0.86
0.84
0.82
0.8
0
0.2
0.4
0.6
0.8
1
1.2
1.4
K_v
BP Degree
BP Star
SFBP Degree
SFBP Star

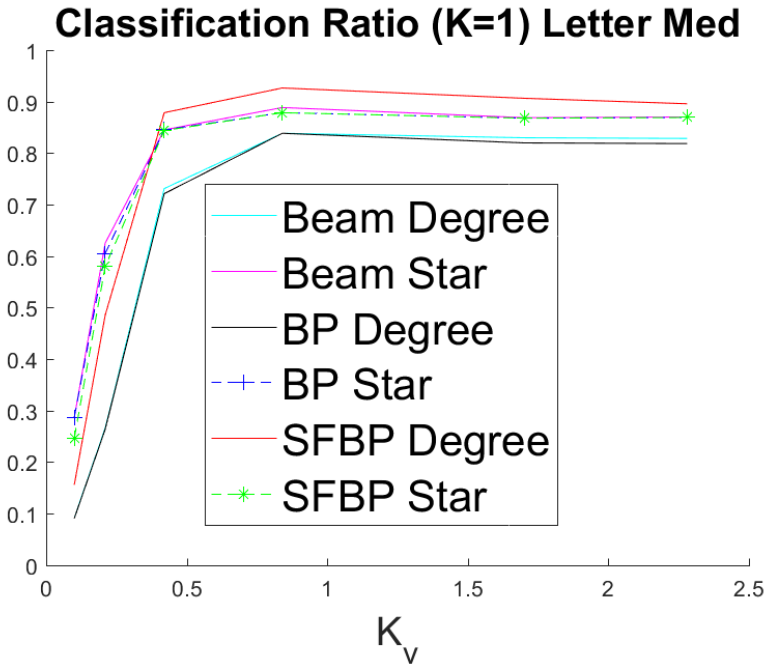

Classification Ratio (K=3) Coil Rag
0.94
0.92
0.9
0.88
0.86
0.84
0.82
0.8
0
0.2
0.4
0.6
0.8
1
1.2
1.4
K_v
BP Degree
BP Star
SFBP Degree
SFBP Star

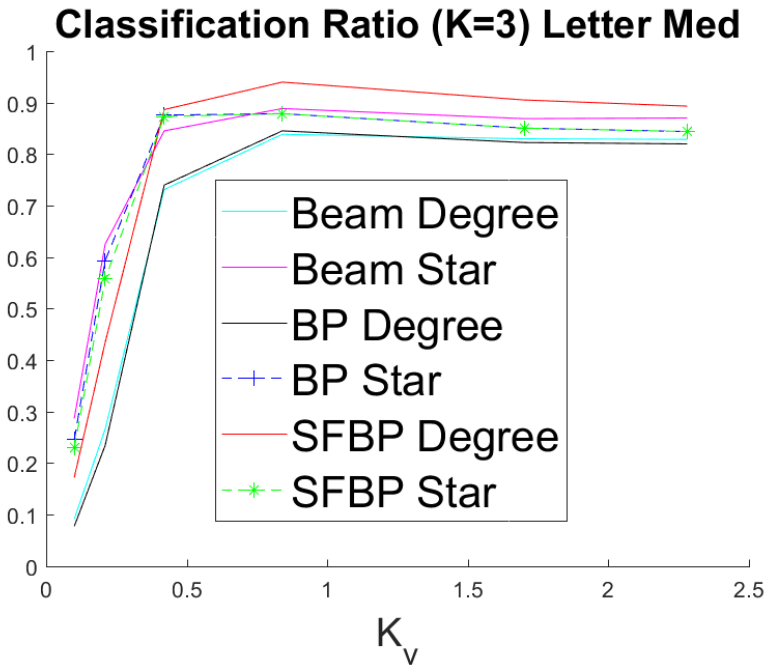

Classification Ratio (K=1) Grec
1
0.98
0.96
0.94
0.92
0.9
0.88
0.86
0.84
0
50
100
150
200
250
300
350
400
450
BP Degree
BP Star
SFBP Degree
SFBP Star
K_v

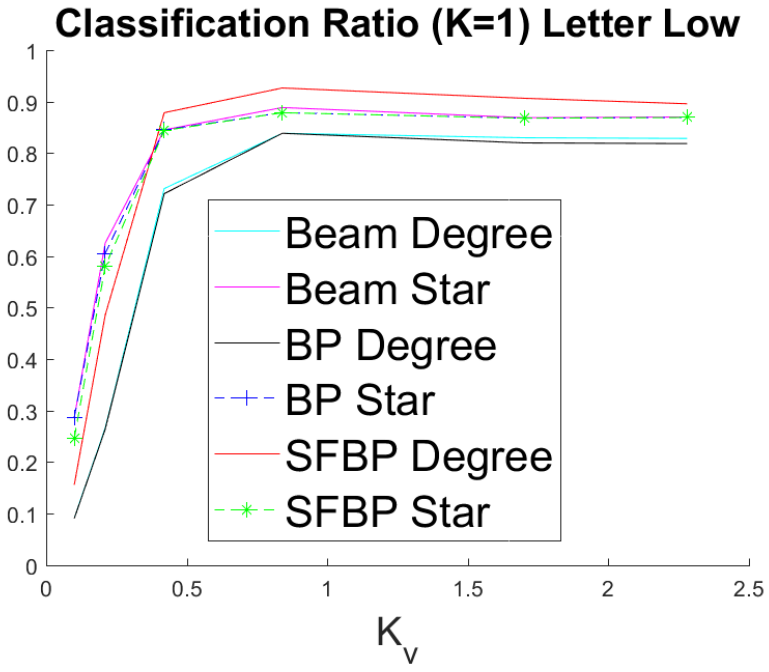

Classification Ratio (K=3) Grec
1
0.98
0.96
0.94
0.92
0.9
0.88
0.86
0.84
0
50
100
150
200
250
300
350
400
450
BP Degree
BP Star
SFBP Degree
SFBP Star
K_v


Classification Ratio (K=1) Letter High
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.5
1
1.5
2
2.5
Beam Degree
Beam Star
BP Degree
BP Star
SFBP Degree
SFBP Star
K_v


Classification Ratio (K=3) Letter High
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.5
1
1.5
2
2.5
Beam Degree
Beam Star
BP Degree
BP Star
SFBP Degree
SFBP Star
K_v


Classification Ratio (K=1) Letter Med
Beam Degree
Beam Star
BP Degree
BP Star
SFBP Degree
SFBP Star
K_v


Classification Ratio (K=3) Letter Med
Beam Degree
Beam Star
BP Degree
BP Star
SFBP Degree
SFBP Star
K_v



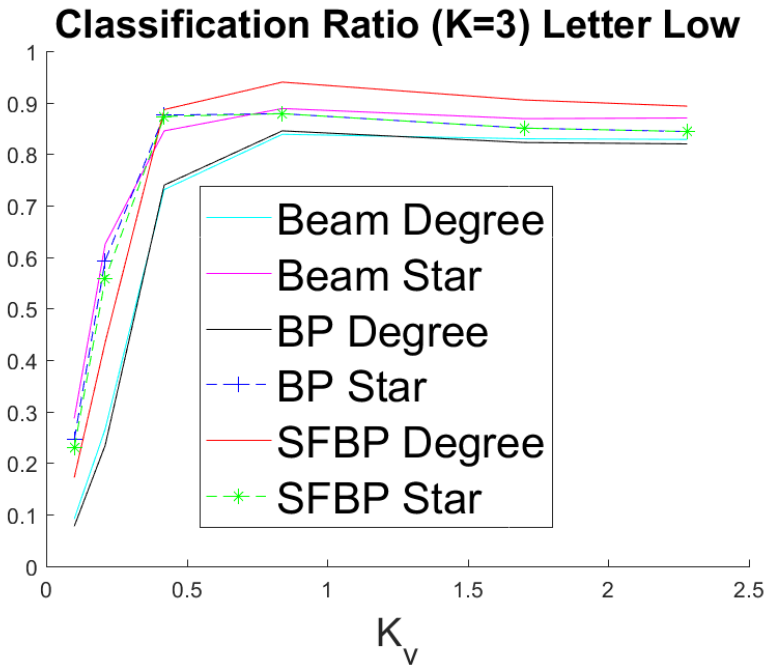


Figure 3: Recognition ratio of Coil Rag, Grec, Letter low, Letter med and letter high given different algorithms

To conclude the experimental section, we would like to comment on the results published in [43], [44], [45]. In these papers, a methodology to automatically deduce the edit functions is presented. We have two remarks on the experimental section of these papers that concern on the aim of this paper. First, their automatically obtained edit functions do not fulfil the triangle inequality. Second, their recognition ratio is again maximised at the point where the triangle inequality is not fulfilled on the edit functions.

6. Conclusions

From a theoretical point of view, we observe that:

a) The graph edit distance is required to be defined as a metric in some high-level algorithms such as object classification or database exploration.

b) The triangle inequality of the edit functions has been considered to be necessary for the graph edit distance to be a metric.

c) Current graph matching algorithms compute the graph edit distance (or the sub-optimal graph edit distance) through deducing the optimal graph correspondence instead of the optimal edit path.

From a practical point of view, we observe that:

a) Practical experimentation shows that the highest recognition ratios appear when this restriction is not fulfilled.

b) The learning algorithms tend to return cost functions that do not fulfil the triangle inequality.

In this paper, we have shown that, the triangle inequality restriction does not need to be applied on the edit functions if the graph edit distance is deduced through an edit path, which was the original definition of this distance. Nevertheless, it does need to be applied in the case that the graph edit distance is deduced through a graph correspondence, which is the most currently used definition.

Therefore, in the case that the graph matching algorithm explores the space of all the edit paths, we can tune the edit functions at the point where the recognition ratio is maximised, without the concern of returning a non-metric definition of the graph edit distance. On the other hand, in the case that the graph edit distance is deduced through a graph correspondence, we have to tune the edit functions considering the triangle inequality restriction.

Moreover, the papers that claim that they define an algorithm to compute the exact graph edit distance should add that this is only this case if the edit functions are defined so that they hold the distance restrictions. This is because, these algorithms explore the graph bijection space instead of the edit path space.

This new finding could encourage research on graph matching in two directions. From a theoretical point of view, discovering new graph matching algorithms that explore the edit path space instead of the graph bijection space. From a practical point of view, using the edit functions automatically learned, even if these functions do not fulfil the triangle inequality.

Acknowledgments

We acknowledge Professor Horse Bunke and Professor Alberto Sanfeliu for their contribution on the graph matching field. This research is supported by the Spanish

projects TIN2016-77836-C2-1-R and ColRobTransp MINECO DPI2016-78957-R AEI/FEDER EU; and also the European projects AEROARMS (H2020-ICT-2014-1-644271) and NanoInformaTIX (H2020-NMBP-TO-IND-2018-2020-814426).

References

1. H. Bunke and G. Allermann, Inexact graph matching for structural pattern recognition, Pattern Recognition Letters, 1(4), pp: 245–253, 1983.
2. A. Sanfeliu and K.S. Fu, A Distance measure between attributed relational graphs for pattern recognition. IEEE transactions on systems, man, and cybernetics, 13(3), pp: 353-362, 1983.
3. A. Sanfeliu, R. Alquézar, J. Andrade, J. Climent, F. Serratosa and J. Vergés, "Graph-based Representations and Techniques for Image Processing and Image Analysis", Pattern Recognition 35 (3), pp: 639-650, 2002.
4. D. Conte, P. Foggia, C. Sansone and M. Vento, Thirty Years of Graph Matching In Pattern Recognition, International Journal of Pattern Recognition and Artificial Intelligence 18 (3), pp: 265-298, 2004.
5. M. Vento, A long trip in the charming world of graphs for Pattern Recognition, Pattern Recognition, 48, pp: 291-301, 2015.
6. L. Livi and A. Rizzi, The graph matching problem, Pattern Analysis and Applications 16 (3), pp: 253-283, 2013.
7. P. Foggia, G. Percannella and M. Vento, Graph matching and learning in Pattern Recognition in the last 10 years, International Journal of Pattern Recognition and Artificial Intelligence, 28 (1), pp: 1450001 [40 pages], 2014.

8. A. Solé, F. Serratosa and A. Sanfeliu, On the Graph Edit Distance cost: Properties and Applications, International Journal of Pattern Recognition and Artificial Intelligence 26 (5), pp: 1260004 [21 pages], 2012.
9. F. Serratosa and X. Cortés, Graph Edit Distance: moving from global to local structure to solve the graph-matching problem, Pattern Recognition Letters, 65, pp: 204-210, 2015.
10. X. Gao, B. Xiao, D. Tao and X. Li, A survey of graph edit distance. Pattern Analysis and applications, 13 (1), pp: 113-129, 2010.
11. K. Riesen, Structural Pattern Recognition with Graph Edit Distance. Approximation Algorithms and Applications, Springer, 2015.
12. F. Serratosa, X. Cortés and A. Solé-Ribalta, Component Retrieval based on a Database of Graphs for Hand-Written Electronic-Scheme Digitalisation, Expert Systems with Applications, 40, pp: 2493-2502, 2013.
13. A. Sanfeliu, F. Serratosa & R. Alquézar, Second-Order Random Graphs for modelling sets of Attributed Graphs and their application to object learning and recognition, International Journal of Pattern Recognition and Artificial Intelligence 18 (3), pp: 375-396, 2004.
14. F. Serratosa, R. Alquézar & A. Sanfeliu, Function-Described Graphs for modelling objects represented by attributed graphs, Pattern Recognition 36 (3), pp: 781-798, 2003
15. F. Serratosa, R. Alquézar & A. Sanfeliu, Synthesis of function-described graphs and clustering of attributed graphs, International Journal of Pattern Recognition and Artificial Intelligence 16 (6), pp: 621-655, 2002.
16. Lin Han, Richard C. Wilson, Edwin R. Hancock: Generative Graph Prototypes from Information Theory. IEEE Trans. Pattern Anal. Mach. Intell. 37(10): 2013-2027 (2015).

17. Richard C. Wilson, Edwin R. Hancock, Elzbieta Pekalska, Robert P. W. Duin: Spherical and Hyperbolic Embeddings of Data. IEEE Trans. Pattern Anal. Mach. Intell. 36(11): 2255-2269 (2014)
18. Xiao Bai, Edwin R. Hancock, Richard C. Wilson: Graph characteristics from the heat kernel trace. Pattern Recognition 42(11): 2589-2606 (2009)
19. Xiao Bai, Edwin R. Hancock, Richard C. Wilson: Geometric characterization and clustering of graphs using heat kernel embeddings. Image Vision Comput. 28(6): 1003-1021 (2010)
20. R. Wilson and E. Hancock, Structural matching by discrete relaxation, IEEE Trans. Pattern Anal. Mach. Intell. 19, pp: 634–648, 1997.
21. R. Myers, R. Wilson and E. Hancock, Bayesian graph edit distance, IEEE Trans. Pattern Anal. Mach. Intell. 22, pp: 628–635, 2000.
22. A. Cross, R. Wilson and E. Hancock, Inexact graph matching using genetic search, Pattern Recognition, 30, pp: 953–970, 1997.
23. D. Justice and A. Hero, A binary linear programming formulation of the graph edit distance, IEEE Trans. Pattern Anal. Mach. Intell. 28, pp: 1200–1214, 2006.
24. N. Rebagliati, A. Solé, M. Pelillo and F. Serratosa, Computing the Graph Edit Distance Using Dominant Sets, International Conference on Pattern Recognition, pp: 1080-1083, 2012.
25. A. Fischer, C. Y. Suen, V. Frinken, K. Riesen and H. Bunke, Approximation of graph edit distance based on Hausdorff matching, Pattern Recognition 48(2), pp: 331-343, 2015.
26. K. Riesen and H. Bunke, Approximate graph edit distance computation by means of bipartite graph matching, Image Vision Computing 27(7), pp: 950-959, 2009.
27. F. Serratosa, Fast Computation of bipartite Graph Matching, Pattern Recognition Letters 45, pp: 244 - 250, 2014.

28. F. Serratosa, Speeding up Fast bipartite Graph Matching trough a new cost matrix, International Journal of Pattern Recognition and Artificial Intelligence, 29 (2), pp: 1550010 [17 pages], 2015.
29. F. Serratosa, Computation of Graph Edit Distance: Reasoning about Optimality and Speed-up, Image and Vision Computing, 40, pp: 38-48, 2015.
30. K. Riesen, H. Bunke and A. Fischer, Improving Approximate Graph Edit Distance using Genetic Algorithms, Syntactic and Structural Pattern Recognition, pp: 63-72, 2014.
31. M. Ferrer, F. Serratosa and K. Riesen, Improving bipartite Graph Matching by Assessing the Assignment Confidence, Pattern Recognition Letters, 65, pp: 29-36, 2015.
32. P. Santacruz & F. Serratosa, Error-tolerant graph matching in linear computational cost using an initial small partial matching, Pattern Recognition Letters, published online 2018.
33. M. Leordeanu, M. Hebert, R. Sukthankar, An integer projected fixed point method for graph matching and map inference, in: Advances in Neural Information Processing Systems, 22, 2009, pp. 1114–1122.
34. Z.-Y. Liu, H. Qiao, GNCCP–Graduated nonconvexity and concavity procedure, IEEE Trans. Pattern Anal. Mach. Intell. 36 (6) (2014) 1258–1267.
35. F. Zhou, F. De la Torre, Factorized graph matching, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 127–134.
36. A. V. Arkhangel'skii and L. S. Pontryagin, General Topology I: Basic Concepts and Constructions Dimension Theory, Encyclopaedia of Mathematical Sciences, Springer, 1990.
37. On-line string matching algorithms: Survey and experimental results, P.D. Michailidis, K.G. Margaritis, On-line string matching algorithms: Survey and

experimental results, International Journal of Computer Mathematics 76(4), pp: 411-434, 2001.

38. M.A. Tahraoui, K. Pinel-Sauvagnat, C. Laitang, M. Boughanem, H. Kheddouci and L. Ning, A survey on tree matching and XML retrieval, Computer Science Review 8, pp. 1-23, 2013.
39. S. V. Rice, H. Bunke and T. A. Nartker, Classes of Cost Functions for String Edit Distance, Algorithmica 18, pp: 271-280, 1997.
40. M. Neuhaus and H. Bunke. Self-organizing Maps for Learning the Edit Costs in Graph Matching, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 35(3), pp: 503-514, 2005.
41. K. Riesen and H. Bunke, IAM graph database repository for graph based pattern recognition and machine learning, Structural Syntactic and Statistical Pattern Recognition, Springer, pp: 287 - 297, 2008.
42. C. Moreno, X. Cortés and F. Serratosa, A Graph Repository for Learning Error-Tolerant Graph Matching, Syntactic and Structural Pattern Recognition, SSPR2016, LNCS 10029, pp: 519-529, 2016.
43. X. Cortés and F. Serratosa, Learning Graph Matching Substitution Weights based on the Ground Truth Node Correspondence, International Journal of Pattern Recognition and Artificial Intelligence, 30 (2), pp: 1650005 [22 pages], 2016.
44. X. Cortés and F. Serratosa, Learning Graph-Matching Edit-Costs based on the Optimality of the Oracle's Node Correspondences, Pattern Recognition Letters, 56, pp: 22 - 29, 2015.
45. S. Algabli & F. Serratosa, Embedding the node-to-node mappings to learn the Graph edit distance parameters, Pattern Recognition Letters, 112, pp: 353-360, 2018.