Automatic Learning of Edit Costs Based on Interactive and Adaptive Graph Recognition*

Francesc Serratosa, Albert Solé-Ribalta, and Xavier Cortés

Universitat Rovira i Virgili, Departament d'Enginyeria Informàtica i Matemàtiques, Spain
{francesc.serratosa,albert.sole}@urv.cat, xavi1984@gmail.com

Abstract. We propose a new method to automatically obtain edit costs for error-tolerant graph matching based on interactive and adaptive graph recognition. Values of edit costs for deleting and inserting nodes and vertices are crucial to obtain good results in the recognition ratio. Nevertheless, these parameters are difficult to be estimated and they are usually set by a naïve trial and error method. Moreover, we wish to seek these costs such that the system obtains the correct labelling between nodes of the input graph and nodes of the model graph. We consider the labelling imposed by a specialist is the correct one, for this reason, we need to present an interactive and adaptive graph recognition method in which there is a human interaction. Results show that when cost values are automatically found, the quality of labelling increases.

Keywords: Interactive Learning, Adaptive Learning, Graph Edit Distance.

1 Introduction

Graphs refer to a collection of nodes and a collection of edges that connect pairs of nodes. Attributed Graphs are graphs in which some attributes are added on nodes and edges to represent local information or characterisation. Attributed graphs have been widely used in several fields to represent objects composed by local parts and relations between these parts. More precisely, in Pattern Recognition and Computer Vision, attributed graphs have been used to represent structural objects that have to be identified or classified. These graphs can represent 2D or 3D objects, handwritten characters, proteins, fingerprints, and so on. Before using graphs, the pattern recognition process has to extract them from these objects. This is not a trivial task included in the Image Understanding field since the quality of graphs is crucial for the rest of the process. When attributed graphs have been extracted, a process to compare them is needed. This process is called graph matching. Usually, it obtains a similarity value and also a labelling between nodes and arcs of the involved graphs. This labelling between nodes and arcs represents the matching between the local parts that graphs represent.

When we characterise the role that attributed graphs and attributed graph matching do in pattern recognition, we realise that in most of the applications, labelling between

* This research was partially supported by Consolider Ingenio 2010 and by the CICYT project DPI 2010-17112.

X. Jiang, M. Ferrer, and A. Torsello (Eds.): GbRPR 2011, LNCS 6658, pp. 152–163, 2011.

nodes is only partially considered. This is because it is considered in the first stages of pattern recognition process, in which it is desired to find a similarity between graphs, but when this similarity value is obtained, the knowledge of the labelling is not considered any more. Nevertheless, we consider that although the graph (that is, the object that represents) is properly classified or identified, the result of the comparison has not sense if the matching between their local parts is not correct.

In this paper, we present an interactive and adaptive graph recognition model with the aim of increasing the quality of the labelling between the graph to be identified and the reference graphs of the database. To that aim, we have extended the graph recognition model to consider the labelling between nodes proposed by a human specialist. This new knowledge is incorporated into the system and used to modify the weights that tune the similarity function between graphs. Usually, these weights are imposed before executing the learning process in a naïve way. We believe that if the quality of the labelling is increased also should do it the quality of the pattern recognition process. In [1], they considered the problem of partial matchings.

The rest of the paper is organised as follows, in sections 2 and 3, we first introduce concepts related to graph matching and then we define a new space of labellings. In sections 3, 4 and 5, we first present the classical pattern recognition and then we depict the new model of interactive and adaptive graph recognition method. In section 7, we compare the quality of the labellings obtained by the new models respect the classical pattern recognition. Section 8 drops some conclusions.

2 Error-Tolerant Graph Matching Based on Edit Operations

One of the most widely used methods for error-tolerant graph matching is the graph edit distance. The basic idea behind the graph edit distance is to define a dissimilarity measure between two graphs by the minimum amount of distortion required to transform one graph into the other [2,3]. To this end, a number of distortion or edit operations ε, consisting of insertion, deletion and substitution of both nodes and edges must be defined. Then, for every pair of graphs (G and G'), there exists a sequence of edit operations, or edit path $path(G, G') = (\varepsilon_1, \ldots, \varepsilon_k)$ (where each ε_i denotes an edit operation) that transforms one graph into the other. In general, several edit paths may exist between two given graphs. This set of edit paths is denoted by $\vartheta(G, G')$. To quantitatively evaluate which is the best edit path, edit cost functions are introduced. The basic idea is to assign a penalty cost C to each edit operation according to the amount of distortion that it introduces in the transformation. The edit distance between two graphs G and G', denoted by $dist_{K_n,K_e}(G, G')$, is defined as the minimum cost of edit path that transforms one graph into the other given parameters K_n (cost of node insertion or deletion) and K_e(cost of edge insertion or deletion) . Several edit paths may obtain the minimum cost. More formally, the edit distance is defined by,

$$dist_{K_n,K_e}(G, G') = \min_{(\varepsilon_1, \ldots, \varepsilon_k) \in \vartheta(G, G')} \left\{ \sum_{i=1}^{k} C(\varepsilon_i) \right\} \quad (1)$$

Usually, edit costs K_n and K_e are estimated in a naïve way or they are learned by trial and error method. The works presented in [4,5,6] are the only ones that aim to automatically estimate these costs. Nevertheless, their method minimises an energy

Table 1. Edit operations, their costs and the relation with the labelling function

Edit operation	Cost	Labelling f
Node deletion (a)	$K_n \in [0, \infty)$	$f^{-1}(a) = null\ element$
Node insertion (a')	$K_n \in [0, \infty)$	$f^{-1}(a') = null\ element$
Node substitution (a,a')	$distance(a, a') \in [0, \infty)$	$f^{-1}(a) = a'$
Edge deletion (b)	$K_e \in [0, \infty)$	$f^{-1}(b) = null\ element$
Edge insertion (b')	$K_e \in [0, \infty)$	$f^{-1}(b') = null\ element$
Edge substitution (b,b')	$distance(b, b') \in [0, \infty)$	$f^{-1}(b) = b'$

related to the recognition ratio in a pattern recognition framework without considering the goodness of the labellings of the involved graphs. Contrarily, in [7], they forced some specific cost. With these costs, they demonstrate the similarity between the edit-distance problem and the maximum common sub-graph problem.

In this paper, we aim to study the impact of K_n and K_e values to the quality of the labellings and present a method to automatically impose them. It seems logical to think that if the labelling between graphs is correct, the recognition ratio of the pattern recognition system has to be the best. First and second columns of Table 1 show the edit operations and edit costs we define throughout this paper. We impose the restriction that insertions and deletions of vertices or arcs have the same cost value. This is done to assure the symmetry property of (1).

Optimal [8] and approximate algorithms [9,10] for the graph edit distance computation have been presented so far, which are out of the scope of this paper. These algorithms obtain the distance value $dist(G, G')$ as well as a labelling f from vertices and arcs of the first graph to vertices and arcs of the second graph. Column 3 of Table 1 shows the labelling related to edit operations. Given any edit path, $path(G, G')$, a labelling $f(G, G')$ can be defined univocally. The cost of this labelling is,

$$Cost_{K_n,K_e}(G, G', f) = \sum_{i=1}^{k} C(\varepsilon_i)\ \ being\ f\ related\ to\ path(G, G') = (\varepsilon_1, \dots, \varepsilon_k) \tag{2}$$

3 Labelling Space Based on K_n and K_e Values

Given an input graph G, a reference graph G' and a labelling f between them, we define the range of values of K_n and K_e that the labelling f has the minimum cost (2). That is, the values of K_n and K_e that f is the labelling that obtains the distance (1) between graphs G and G'. These values (K_n, K_e) hold condition E,

$$E: Cost_{K_n,K_e}(G, G', f) = \min_{\forall f' \in \vartheta(G,G')} \{Cost_{K_n,K_e}(G, G', f')\} \tag{3}$$

Let $R_{G,G',f}$ be a region defined by points (K_n, K_e) in which condition E is true,

$$R_{G,G',f} = \{(K_n, K_e) | E\} \tag{4}$$

Table 2. Edit operations, their costs and the relation with the labelling function

Edit operation	Cost R2	Cost R3
Node del. (*a*)	$K_n = 0.3$	$K_n = 0.3$
Node ins. (*a'*)	$K_n = 0.3$	$K_n = 0.3$
Node subs. (*a,a'*)	distance(*a,a'*) = 2D Euclidean	distance(*a,a'*) = 2D Euclidean
Edge del. (*b*)	$K_e = 0.3$	$K_e = 0.5$
Edge ins. (*b'*)	$K_e = 0.3$	$K_e = 0.5$
Edge subs. (*b,b'*)	distance(*b,b'*) = 0	distance(*b,b'*) = 0

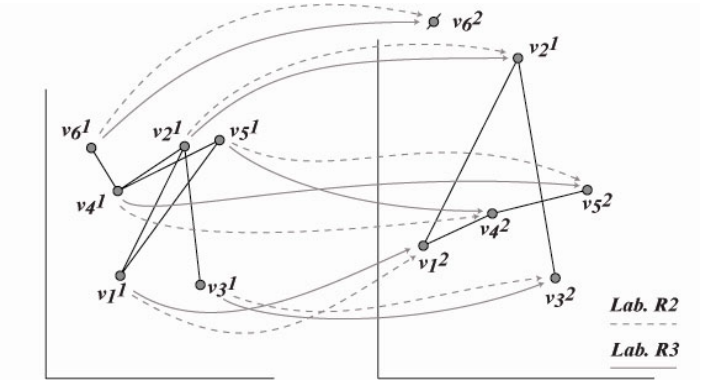

Fig. 1. Two graphs and two different labellings: R2 and R3

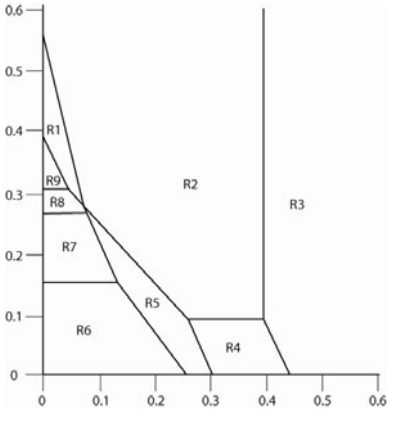

Fig. 2. (a) All possible labellings throughout the labelling space

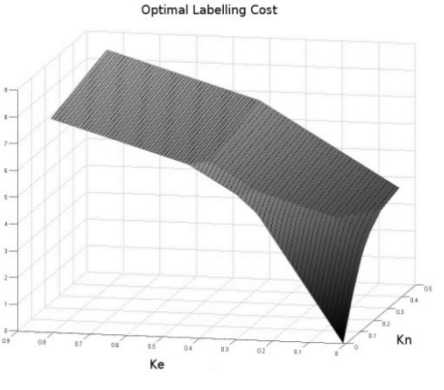

Fig. 2. (b) Edit cost of the optimal labelling given Kn and Ke

Note that $Cost_{K_n,K_e}(G, G_{out}, f')$ takes different values when different values of K_n and K_e are used (2). For this reason, this cost, although it is the minimum one given a point (K_n, K_e) through all labellings f', is not constant throughout region $R_{G,G_{out},f}$.

Figure 1 shows two graphs and two possible labellings between them (called *R2* and *R3*). Table 2 shows the edit costs we have used.

Note that erasing an arc in R3 is more costly than in R2. For this reason R2 erases 3 arcs in the first graph and inserts 1 arc in the second. On the contrary, R3 does not erase any arc and only inserts 1 arc in the second graph. To conclude, R2 gives more importance to the position of the nodes and R3 gives more importance to the structural deformation.

Figure 2.a shows the labelling space between these two graphs respect K_n (vertical axis) and K_e (horizontal axis). There are only 9 different labellings which is a reduced number since the maximum number of labellings could arise 6! = 720. Finally, figure 2.b shows the minimum cost throughout the labelling space. We realise of four properties: 1. Regions $R_{G,G',f}$ are convex, 2. When costs K_n or K_e increase, it does the cost labelling. 3. Increase of labelling cost is linear throughout a region. 4. Regions located in bigger positions of K_n or K_e are less steeply.

4 Classical Graph Recognition Paradigm

Let G be an input graph and h a hypothesis or output of the graph recognition system, which the system derives from G. Let M be a model used by the graph recognition system to derive this output. M is obtained through a previous or batch training process from a given sequence of pairs composed by a graph and a class (G_i, C_i).

In some applications (for instance, character recognition), the hypothesis h is composed by only the class or cluster obtained by the system $h = \{C\}$. Nevertheless, in some applications, it is desired to know not only the class, but also the graph of the model with the minimum error G_{out} and the labelling f between them (for instance, scene identification). That is, $h = \{C, G_{out}, f\}$.

Figure 3 shows the three main modules of a classical graph recognition system. It is composed by a training process, a recognition processes and a model. The model is generated through a batch training process with a set of pairs (graph, class) and other parameters (such as K_n and K_e in (1)) and it represents the knowledge of the recognition process.

In classical pattern recognition [11] (and specifically, in graph recognition) the output of the system is computed through a function that aims to minimise the number

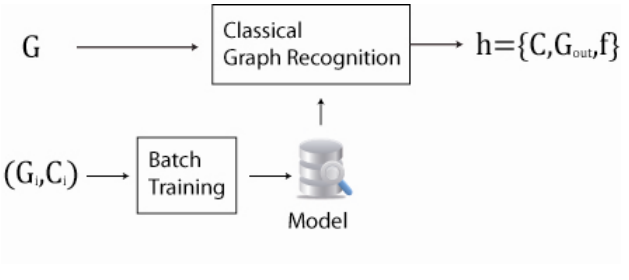

Fig. 3. Scheme of a Classical graph recognition system

of wrong hypothesis, that is, the number of misclassified objects. The best hypothesis is the one that minimises the misclassification of graphs, that is, it is the one which maximises the posterior probability of class C given G, $Pr(C|G)$ and given model M, $P_M(C|G)$.

$$h = \{C, G_{out}, f\} = \underset{\forall C \in M}{max} P_M(C|G) \tag{5}$$

The maximum likelihood approach (5) is the most popular method to estimate the classification not only in the pattern recognition process but also in the training process to train the model M. However, in many cases, it is difficult to directly estimate $P_M(C|G)$ ($P(C|G)$ since now) and it is better to apply the Bayes rule in (5),

$$h = \underset{\forall C \in M}{max} \frac{P(G|C)P(C)}{P(G)} = \underset{\forall C \in M}{max} P(G|C)P(C) \tag{6}$$

The probability $P(C|G)$ is estimated independently for each pair (G, C) and from the available training pairs (G_i, C_i). The probability of each class $P(C)$ depends on the application and, sometimes, it is defined as constant for all classes. Finally, $P(G)$ has been dropped since does not depend on the hypothesis and so, it does not affect the maximisation criteria of (6). With these considerations, we have only to define the probability of having an input graph conditioned to a class. Several methods have been defined, for instance,

$$P(G|C) = \underset{\forall G' \in C}{max}\left\{e^{-dist_{K_n,K_e}(G,G')}\right\} \vee P(G|C) = \underset{\forall G' \in C}{max}\left\{\frac{1}{1+dist_{K_n,K_e}(G,G')}\right\} \tag{7}$$

5 Interactive Graph Recognition Paradigm

Placing human interaction requires adaptation to the way we look at the graph recognition problem [12]. Few research has been done in this field [13] and none related to graphs. One of the main applications has been the semi-automatic transcription of handwritten texts [14, 15].

The paradigm we presented above obtains the minimum graph that minimises (6) independently of the obtained labelling f. Nevertheless, the user aims to obtain a correct labelling f independently of the model parameters imposed in the batch training. Moreover, although the class obtained by the system is considered the correct one by an expert, the hypothesis has to be considered non-correct if the labelling f between the input and output graph is far away from the one considered by an expert. For this reason, the feedback provided by the human is a new labelling g between G and G_{out} (that may be totally or partially equal to f) and then the hypothesis is estimated again.

Without varying the model M, human interaction offers an opportunity to improve the quality of the hypothesis h only using labelling g. Note that, this labelling g between the input graph G and the output graph G_{out}, is the most natural way for a human to interact with the system since it takes direct advantage of the intelligence and general knowledge of the expert. Moreover, it is one of the most complicated tasks to be processed in graph recognition. From an application point of view, f has to be considered as a first approximation of the labelling that helps the specialist to make a final (and possibly better) labelling g.

Since in our new framework, we have two labellings f and g between graphs G and G', it is interesting to have a measure of similarity between both labellings $S_{G,G'}(f,g) \in [0,1]$. We define this measure as follows,

$$S_{G,G'}(f,g) = \frac{1}{|G| + |G'|}\left(\sum_{\forall v \in G} T_1(v) + \sum_{\forall v' \in G'} T_2(v)\right) \in [0,1] \tag{8}$$

Where v and v' are nodes of graphs G and G', respectively. T_1 and T_2 are,

$$T_1(v) = \begin{cases} 1 & if\ f(v) = g(v) \\ 0 & otherwise \end{cases} \quad \text{and} \quad T_2(v') = \begin{cases} 1 & if\ f^{-1}(v') = g^{-1}(v') \\ 0 & otherwise \end{cases} \tag{9}$$

Figure 4 shows the new scheme, in which, it appears the human interaction through labelling g. The batch training generates a model similar to the classical scheme. The interactive recognition process generates a first hypothesis h through the model. Then, it generates the final hypothesis h' using the model and also the human interaction g. Note that, h' can be completely different from h. Not only the graph and labelling can be different but also the class. This is because, with the imposed labelling g, the cost (2) can be higher than the distance (1) between G and another graph of another class.

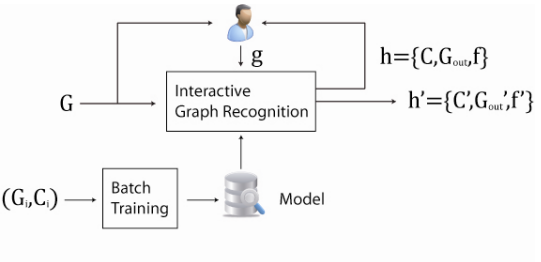

Fig. 4. Scheme of an Interactive graph recognition system

Now, interaction allows adding more conditions to (5) to obtain the new hypothesis,

$$h' = \{C', G'_{out}, f'\} = \underset{\forall C \in M}{max}\ P_M(C|G,g) \tag{10}$$

Applying Bayes rule, we arrive at the following expression,

$$h' = \underset{\forall C \in M}{max}\ \frac{P(G|C,g)P(C|g)P(g)}{P(G,g)} \tag{11}$$

In which probabilities $P(g)$ and $P(G,g)$ do not depend on C and they are dropped off,

$$h' = \underset{\forall C \in M}{max}\ P(G|C,g)P(C|g) \tag{12}$$

Probability $P(C|g)$ represents the probability of a class conditioned to a labelling g between two graphs. We consider in this point that there is no information about the process to extract a graph from an object, commented in the introduction section (for instance, image or social net). Therefore, the probability of a class C has to be defined independent of having a specific labelling g between two graphs and we assume $P(C|g) = P(C)$.

The conditioned probability of the input graph G, $P(G|C)$, is defined depending on the data (7), but now, we add another conditional, which is the specialist labelling g,

$$P(G|C,g) = e^{-Cost_{K_n,K_e}(G,G',g)} \quad \text{or} \quad P(G|C,g) = \frac{1}{1+Cost_{K_n,K_e}(G,G',g)} \tag{13}$$

And the probability of class $P(C)$ is defined to be constant if there is no information. In some examples, it depends on the information extracted from the learning process.

6 Adaptive Graph Learning

In the previous section the model M has been assumed to be fixed. But now, the human interaction offers another opportunity to improve the system not only modifying the current hypothesis but modifying the model M when new information is available. We present a methodology to update parameters K_n and K_e of equation (1). These parameters are crucial for the quality of the recognition but they are very difficult to be estimated since they are data dependent. We propose a model that each time the specialist imposes a labelling g between an input graph G and an output graph G_{out}, these parameters are updated.

The adaptive-graph learning scheme is composed of three modules (figure 5). The training and recognition module are similar to the interactive scheme. The adaptive-learning module updates values of K_n and K_e in model M.

This scheme is implemented in an iterative algorithm that each time a new graph is introduced into the system to be identified, the following steps are carried out:

- Given input G, the graph recognition module outputs $h = \{C, G_{out}, f\}$
- Specialist analyses G and h and introduces labelling g into the system (g is the same than f if h is an ideal prediction)
- The interactive recognition module outputs h'. (C, G_{out} and f might be different to C', G'_{out} and f')
- The adaptive-learning module inputs h', g and G and modifies values of K_n and K_e using intersection of regions $R_{G,G'_{out},g}$.

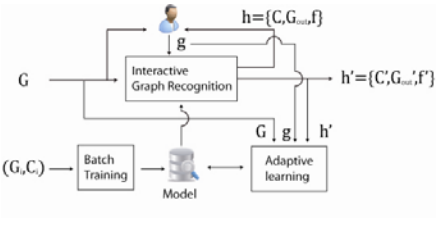

Fig. 5. Scheme of an Adaptive graph recognition system

7 Practical Evaluation

To evaluate and compare the quality of output labelling f using the Interactive Learning scheme (IL) with respect to the Classical Learning (CL) scheme, we propose to use the similarity $S(f,g)$ (8) between labelling f and the labelling imposed by the specialist g.

The evaluation measure we used is the Area under the Learning Curve (ALC). Our Learning curve plots in each point the addition of similarities obtained by processing the whole test set normalised by the cardinality of the test set. That is,

$$Learning\ Curve = \frac{1}{|TestSet|} \sum_{\forall G \in TestSet} S_{G,G_{out}}(f,g) \in [0,1] \tag{15}$$

Moreover, we obtain a global measure of the quality of the learning process when it is considered to be finished. It is independent of the cardinality of the test and reference set and also the cardinality of the graphs,

$$GlobalScore = \frac{ALC - A_{rand}}{A_{max} - A_{rand}} \tag{16}$$

Values A_{max} and A_{rand} are obtained through two baseline learning curves. The first one is the ideal learning curve that at first point goes up to the maximum value that equals 1. The second one is the "lazy" learning curve that follows a straight line. It is obtained by making random predictions. If $m = |test\ set|$ and n is the average cardinality of the graphs in the test set, the lazy learning curve can be approximated as $\frac{1}{(n!)^m}$.

We have used the Tarragona-graph database [16]. It is a database of graphs that represents handwritten letters and it has the peculiarity that nodes of graphs of the same class have been manually labelled. Nodes represent junctions or ending points of strokes and non-directional edges are strokes. Attributes on the nodes are the 2D position and edges do not have attributes. Figure 6 shows some examples of character A. Nodes have been manually labelled and so the i^{th} node of each graph represents the i^{th} ending-point or junction. Some nodes are missing in some characters.

There are 26 different characters and 11 elements for each character. From each character, we have used the 4 first elements as the test set and the other 7 elements as the reference set.

We have performed 4 experiments using CL and other 4 experiments using IL. The difference between these four experiments is the initial values of K_n and K_e. Note that, in CL, these values are constant through the learning process but in IL they are not constant. Figure 7 shows the average learning curves of the 8 experiments.

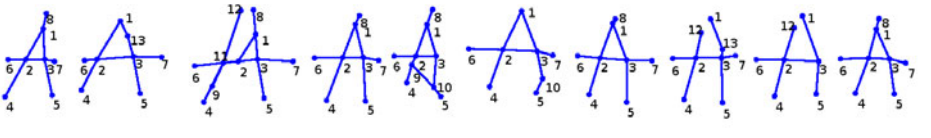

Fig. 6. Character A of Tarragona-graph database

We have carried out the following process to plot each of the 8 curves:

```
Model empty
For i = 1 to 7 do
    For C = 1 to 26 do
        G^c_i = Select_New_Graph_from_Reference_Set_belonging_to_Class_C
        Include G^c_i into de model
    If IL recompute K_n and K_e
    S_aux = 0
    For j = 1 to 4 do
        For C = 1 to 26 do
            G^c_j = Select_New_Graph_from_Test_Set_belonging_to_Class_C
                (G_out,f) = 1-Nearest_Neighbour_belonging_to_Class_C (G_j)
                g = Ideal_Labelling (G_j,G_out)
                S_aux = S_aux +  Similarity (g,f)
    Plot (S_aux/(4x26),i)
```

Figure 8 shows the evolution of values of K_n and K_e when new graphs are added into the IL system. The table on the left shows the initial values. Each of the 4 curves has 7 points. The model has 1 graph per class in the first point, 2 graphs per class in the second point and so on. Note that the four curves move to the same point, which has the be considered the optimal one, and it is independent of the initial values.

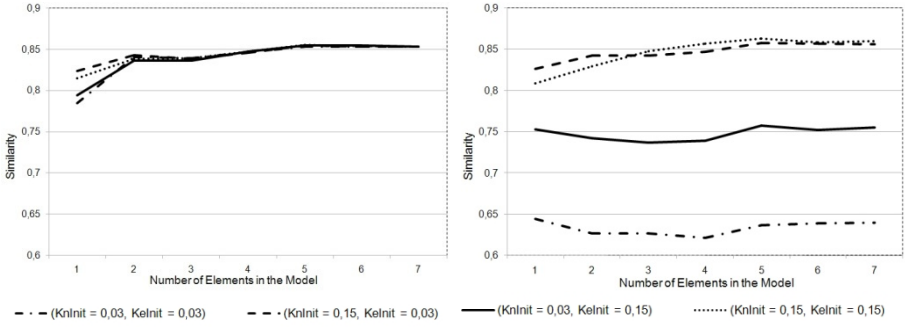

Fig. 7. Average learning curves respect the number of graphs in the model and given 4 different initial K_n and K_e values. Left plot: CL and right plot: IL.

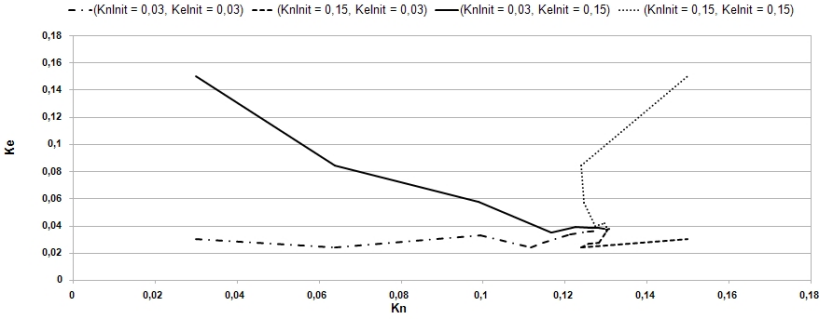

Fig. 8. Evolution of K_n and K_e. in the 4 IL experiments

Table 3. Global score of CL and IL depending on the initial K_n and K_e values

K_n	0.03	0.15	0.03	0.15
K_e	0.03	0.03	0.15	0.15
CL	0.632	0.848	0.747	0.848
IL	0.841	0.846	0.840	0.844

Finally, Table 3 shows the Global scores (16) of the learning curves shown in figure 7. Global scores of CL curves that the initial points are far from the final point in figure 8 have lower values. Conversely, there is a slight difference between CL and IL global scores when the initial points are close to the optimal point.

8 Conclusions and Future Work

We have presented an interactive and adaptive graph recognition method. It is based on the idea that if the quality of the labelling between graphs is increased, it also does the quality of the recognition or identification process. The feedback of the human specialist is introduced into the system as the ideal labelling between graphs and it is used to tone the weights of the edit costs in the similarity function between graphs. These weights are application dependent and difficult to be manually toned. Practical evaluation shows an example of obtaining K_n and K_e values. We conclude the final values are independent of the initial ones and also there is an increase of the quality of the labellings when these costs are automatically obtained.

References

1. Caetano, T.S., McAuley, J.J., Cheng, L., Le, Q.V., Smola, A.J.: Learning Graph Matching. IEEE Trans. Pattern Anal. Mach. Intell. 31(6), 1048–1058 (2009)
2. Sanfeliu, A., Fu, K.: A distance measure between attributed relational graphs for pattern recognition. IEEE Trans. on Sys., Man and Cybern. 13, 353–362 (1983)
3. Bunke, H., Allerman, G.: Inexact graph matching for structural pattern recognition. Pattern Recognition Letters 1(4), 245–253 (1983)
4. Neuhaus, M., Bunke, H.: A Probabilistic Approach to Learning Costs for Graph Edit Distance. ICPR (3), 389–393 (2004)
5. Neuhaus, M., Bunke, H.: Self-organizing maps for learning the edit costs in graph matching. IEEE Trans. on Sys., Man, and Cybernetics, Part B 35(3), 503–514 (2005)
6. Neuhaus, M., Bunke, H.: Automatic learning of cost functions for graph edit distance. Inf. Sci. 177(1), 239–247 (2007)
7. Bunke, H.: On a relation between graph edit distance and maximum common subgraph. Pattern Recognition Letters 18(8), 689–694 (1997)
8. Lladós, J., Martí, E., Villanueva, J.J.: Symbol Recognition by Error-Tolerant Subgraph Matching between Region Adjacency Graphs. IEEE Trans. Pattern Anal. Mach. Intell. 23(10), 1137–1143 (2001)
9. Gold, S., Rangarajan, A.: A Graduated Assignment Algorithm for Graph Matching. IEEE TPAMI 18(4), 377–388 (1996)
10. Christmas, W.J., Kittler, J., Petrou, M.: Structural matching in computer vision using probabilistic relaxation. IEEE TPAMI 17(8), 749–764 (1995)

11. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. J. Wiley, Chichester (1973)
12. Canny, J.: The future of human-computer interaction. Queue,ACM 4(6), 24–32 (2006)
13. Vidal, E., Rodríguez, L., Casacuberta, F., García-Varea, I.: Interactive Pattern Recognition. In: MLMI, pp. 60–71 (2007)
14. Toselli, A.H., Romero, V., Pastor, M., Vidal, E.: Multimodal interactive transcription of text images. Pattern Recognition 43(5), 1814–1825 (2010)
15. Casacuberta, F., Civera, J., Cubel, E., Lagarda, A.L., Lapalme, G., Macklovitch, E., Vidal, E.: Human interaction for high-quality machine translation. Commun. ACM 52(10), 135–138 (2009)
16. http://deim.urv.cat/~francesc.serratosa/Tarragona_Graph_Database