Pattern Recognition 48 (2015) 1364–1373

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Interactive graph-matching using active query strategies ☆

Francesc Serratosa *, Xavier Cortés

Universitat Rovira i Virgili, Departament d'Enginyeria Informàtica i Matemàtiques, Tarragona, Spain

ARTICLE INFO

Article history:
Received 19 March 2013
Received in revised form
20 September 2013
Accepted 29 October 2014
Available online 11 November 2014

Keywords:
Error-tolerant graph matching
Edit distance
Active query strategies
Interactive node labelling

ABSTRACT

Given two graphs, the aim of graph matching is to find the "best" matching between the nodes of one graph and the nodes of the other graph. Due to distortions of the data and the complexity of the problem, in some applications, completely automatic processes do not return a satisfactory graph matching. We propose a method to perform active and interactive graph matching in which an active module queries one of the nodes of the graphs and the oracle (human or artificial) returns the node of the other graph it has to be mapped with. The interactive algorithm reaches the matching desired by the user in few interactions, since by imposing a node-to-node mapping, other ones are automatically amended. The method uses any graph matching algorithm that iteratively updates a probability matrix between nodes since it only requires access to the probability matrix and to update the costs between nodes and arcs. We present and practically validate different active strategies.

1. Introduction

On one hand, *active learning* is a discipline concerned with the design and development of algorithms that allow computers to evolve behaviours based on examples [1,2]. In this discipline, a learner can take advantage of examples to capture characteristics of interest from the data respect of their class. With the learned characteristics, the learner deducts the class of the new examples. On the other hand, *error-tolerant graph matching* [3] is another discipline that aims to find the best matching between the nodes of both graphs so that the cost of this optimal matching is the minimum among all possible matchings. If we combine the active learning and error-tolerant graph-matching disciplines, we can define a model in which examples and classes in the machine-learning discipline are composed of the set of nodes of one of the graphs and the nodes of the other graphs, respectively. Therefore, what we want to find is the best matching between the nodes of both graphs but with the minimum necessary help of an oracle. Note that we do not perform a learning strategy such as *learning graph matching* [69] or semi-supervised learning [70] since we do not modify the labelling cost function.

Normally, two basic modules compose *pattern recognition systems* [1]. The first one extracts the main features given the raw data. The second one extracts the class of the object or simply obtains the most similar object from a database. In the semi-automatic methods, a specialist usually interacts in the first module and modifies the automatically extracted features. Then, with the updated features, the automatic matching or query process is performed obtaining a result with higher quality. For instance, in the case of AFIS [4], the specialist usually verifies and modifies the extracted minutiae of the fingerprint to be queried. However, it is not so usual to apply any interaction on the second module [5,6].

Fig. 1 shows a classical semi-automatic image-correspondence process in which an intermediate step has been incorporated. We wish to compare input images I^1 and I^2. Both images are represented by some kind of representation that explore the local parts of the image $g^1 = g\left(I^1\right)$ and $g^2 = g\left(I^2\right)$, for instance vectors or attributed graphs. There is a first step in which representations g^1 and g^2 of the images I^1 and I^2 are obtained using methods such as [7,8]. Then, in the semi-automatic methods, there is a second step in which the user edits the local parts of these representations (erase, create or modify their positions or values). We call the user feedback w^1 and w^2. Note that the user not only has access to the obtained representation but also to the original image since it is a valuable knowledge for the human intelligence. The last step obtains the matching between nodes f and a dissimilarity measure or cost C_f in a completely automatic way through methods such as [9–14].

The aim of this paper is to add interactivity to the third step of Fig. 1 and to keep the first and second steps as they are. The interactive part of this method was presented in [15] and the active one in [16]. In this paper we present the whole method and more active strategies. This new interactive method is useful in two types of applications. The first ones are applications where it is crucial to have a perfect match but data is very noisy and it is

☆This research is supported by the Spanish projects DPI2013-42458-P and TIN2013-47245-C2-2-R.

* Corresponding author.
E-mail addresses: francesc.serratosa@urv.cat (F. Serratosa), xavier.cortes@urv.cat (X. Cortés).

http://dx.doi.org/10.1016/j.patcog.2014.10.033
0031-3203/

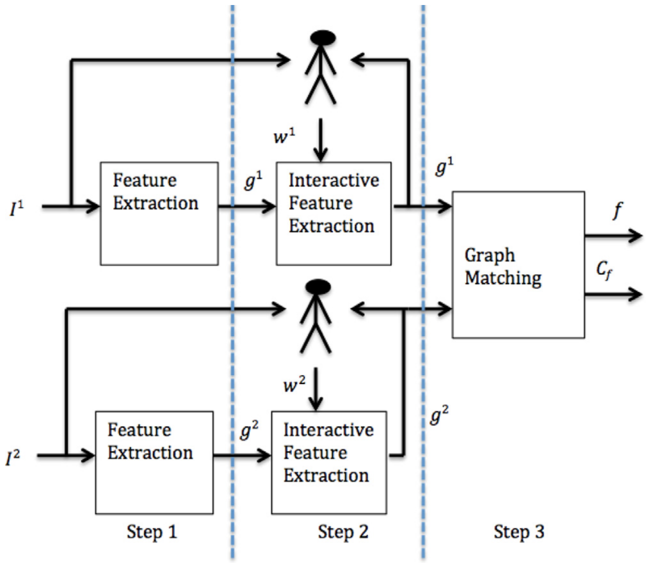

Fig. 1. Classical Image Correspondence Process with human interaction in the local parts extraction and based on structural pattern recognition.

difficult to extract the local parts of objects even though the number of these local parts may not be large. For instance, in medical applications, in which graphs are extracted from images. Other applications are the ones in which graphs have a large cardinality. Then, graph-matching algorithms have to be very greedy and they are unlikely to obtain a satisfactory match. Conversely, this method is not useful in applications where an unclassified graph has to be compared with a large number of graphs in a database. For instance, fingerprint identification. The human interaction in each graph comparison would increase the run time considerably. In this type of application, it is usual to interact in the first step of the pattern recognition process as described in Fig. 1. The extracted graph is corrected in the first step of the recognition process and graphs of the database are corrected in the enrolment process.

The rest of the paper is organised as follows. In the following section, we present the basic notation and summarise the well-known methods used. In Section 3, we present the active and interactive method, in other words, we show how to add interactivity to the third step of the image correspondence process (Fig. 1). In Section 3.3, we show the algorithm to compute the active and interactive graph matching. Finally, in Section 4 we show the practical evaluation and we conclude the paper in Section 5.

2. Notation and methods

In this section, we summarise the basic notation and concepts of the two disciplines we have combined in our model: Graph matching and Active learning.

2.1. Graph matching

Let g^1 and g^2 be two attributed graphs. We suppose that g^1 and g^2 have the same number of nodes n since they have been enlarged enough to incorporate null nodes. We define nodes in g^1 and g^2 as $v_i^1 \in \Sigma_v^1$ and $v_a^2 \in \Sigma_v^2$ and we define arcs as $e_{ij}^1 \in \Sigma_e^1$ and $e_{ab}^2 \in \Sigma_e^2$, $\forall i,j,a,b \in \{1,\ldots,n\}$. Moreover, let f be a bijective labelling between nodes of both graphs. The cost of matching graphs g^1 and g^2, given this isomorphism f, is represented by

$$C_f(g^1,g^2) = \sum_{v_i^1 \in \Sigma_v^1} c_v(v_i^1, v_a^2) + \sum_{e_{ij}^1 \in \Sigma_e^1} c_e\left(e_{ij}^1, e_{ab}^2\right) \tag{1}$$

where $f(v_i^1) = v_a^2$ and $f\left(v_j^1\right) = v_b^2$. That is, the cost is defined as the addition of the pairwise costs of matching nodes and arcs [3]. These local costs can be represented through two matrices $C_v \in \mathbb{R}^{+2}, C_v[v_i^1, v_a^2] = c_v(v_i^1, v_a^2)$ and $C_e \in \mathbb{R}^{+4}$, $C_e\left[v_i^1, v_a^2, v_j^1, v_b^2\right] = c_e\left(e_{ij}^1, e_{ab}^2\right)$ and their definition depends on the application. Usual examples are the Euclidean distance, when attributes have the position of the node in the image or the distance between local features such as Harris corners [7], SIFTs [8] and others [17].

There are several error-tolerant graph-matching algorithms that return the best isomorphism f between two graphs: Probabilistic relaxation [18], Graduated-Assignment [9], Expectation-Maximisation [10] or Bipartite Graph Matching [19]. In fact, the input of these algorithms can be matrices C_v and C_e instead of graphs g^1 and g^2 since matrices capture all the differences between graphs and the minimisation cost is defined through these matrices (1). Considering that the involved graphs have a degree of disturbance and also the exponential complexity of the problem, these algorithms do not return exactly the isomorphism f but a probability matrix related to it (except [19] which directly returns the matrix labelling given that it is not a stochastic algorithm). We represent this matrix by P where each cell contains $P[v_i^1, v_a^2] = Prob(f(v_i^1) = v_a^2)$. Thus, given the probability matrix P, it is necessary to derive the final labelling f by a discretization process. There are several techniques to perform this discretization, e.g. [20]. Fig. 2 represents the probabilistic graph-matching paradigm.

In general, if we want to solve the error-tolerant graph-matching problem based on probabilities [9,10] or [18], given two graphs g^1 and g^2, the general objective function to optimize corresponds to the quadratic assignment problem objective function,

$$C_P(g^1,g^2) = \sum_{v_i^1 \in \Sigma_v^1} \sum_{v_a^2 \in \Sigma_v^2} \sum_{\substack{v_j^1 \in \Sigma_v^1 \\ v_j^1 \neq v_i^1}} \sum_{\substack{v_b^2 \in \Sigma_v^2 \\ v_b^2 \neq v_a^2}} P[v_i^1, v_a^2] P\left[v_j^1, v_b^2\right] C_e\left[v_i^1, v_a^2, v_j^1, v_b^2\right] + \sum_{v_i^1 \in \Sigma_v^1} \sum_{v_a^2 \in \Sigma_v^2} P[v_i^1, v_a^2] C_v[v_i^1, v_a^2] \tag{2}$$

where P is restricted to

$$\sum_{v_i^1 \in \Sigma_v^1} P[v_i^1, v_a^2] = 1,\ \forall v_a^2 \in \Sigma_v^2 \textit{ and } \sum_{v_a^2 \in \Sigma_v^2} P[v_i^1, v_a^2] = 1,\ \forall v_i^1 \in \Sigma_v^1 \tag{3}$$

Some methods use a Gradient Ascent technique [21] or a similar technique to get a local maximum of C_P where $P[v_i^1, v_a^2]$, $\forall v_i^1 \in \Sigma_v^1$ and $\forall v_a^2 \in \Sigma_v^2$, are the set of variables of the function. This technique takes steps proportional to the magnitude of the positive gradient with the aim of approaching to a local maximum of function C_P. The magnitude of the gradient of C_P respect variable $P[v_i^1, v_a^2]$ is

$$M(v_i^1, v_a^2) = \frac{dC_P(g^1,g^2)}{dP[v_i^1, v_a^2]} = \sum_{\substack{v_j^1 \in \Sigma_v^1 \\ v_j^1 \neq v_i^1}} \sum_{\substack{v_b^2 \in \Sigma_v^2 \\ v_b^2 \neq v_a^2}} P\left[v_j^1, v_b^2\right] C_e\left[v_i^1, v_a^2, v_j^1, v_b^2\right] + C_v[v_i^1, v_a^2] \tag{4}$$

In Section 3, we show how the feedback of the human is used to update matrices C_v and C_e. Besides, we present different

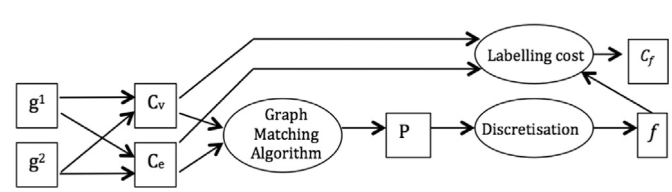

Fig. 2. Probabilistic graph matching framework (step 3 of Fig. 1).