Many-to-Many Matching under the l_1 Norm

M. Fatih Demirci and Yusuf Osmanhoğlu

TOBB University of Economics and Technology,
Computer Engineering Department,
Sogutozu Cad. No:43 , 06560 Ankara, Turkey
{mfdemirci,yosmanlioglu}@etu.edu.tr

Abstract. The problem of object recognition can be formulated as matching feature sets of different objects. Segmentation errors and scale difference result in many-to-many matching of feature sets, rather than one-to-one. This paper extends a previous algorithm on many-to-many graph matching. The proposed work represents graphs, which correspond to objects, isometrically in the geometric space under the l_1 norm. Empirical evaluation of the algorithm on a set of recognition trails, including a comparison with the previous approach, demonstrates the efficacy of the overall framework.

Keywords: graph embedding, Earth Mover's Distance, graph matching, object recognition.

1 Introduction

Object matching is one of the core problems in computer vision. Given two objects, the objective of this problem is to establish a correspondence between their features. This objective consists of two steps: extracting features from objects and matching these features. The overall matching process results in a similarity (or, dissimilarity) value, which can be used for classifying an unknown object (query) as an instance of a particular class from a database. In a simple experimental setup, this can be achieved by a linear search, i.e., computing the similarity between the query and each database object and locating the database object with the highest similarity score.

Due to their representational power, graphs have been widely used in several domains such as pattern recognition, computational and molecular biology, linguistics, computer networks, and physics. A graph $G = (V, E)$ is composed of a finite set of vertices (V) and set of connections (edges) (E) between the vertices. Two vertices u and $v \in V$ are adjacent if there exists an edge $e = (u, v) \in E$. In computer vision and pattern recognition, graphs have been used to represent complex structures such as 3D objects, medical images, and fingerprints. In these structures, vertices correspond to regions or features of the object and edges show the relationships between the vertices. When graphs are used to represent the objects, object matching problem is reformulated as that of graph matching.

Fig. 1. The need for many-to-many matching. Although the silhouettes of two camera objects are similar, no one-to-one matching exists between their shock graphs.

The problem of finding the similarity between pairs of objects using their graph representations has been the focus of many researchers in the computer vision and pattern recognition communities for over twenty years. To obtain a good correspondence between two graphs, a lot of work focuses on the search for the best isomorphism between two graphs or subgraphs (e.g., [18,12,11]). Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, the isomorphism of G_1 and G_2 is defined as a bijection between the vertex sets of G_1 and G_2 , $f : V(G_1) \rightarrow V(G_2)$ such that any two vertices u and $v \in G$ are adjacent if and only if $f(u)$ and $f(v)$ are adjacent in G_2 .

While it is still an open question whether the detection of graph isomorphism can be done in polynomial time, the subgraph isomorphism problem is known to be NP-complete [7]. Some researches, such as [10], achieve polynomial time algorithms on the problem of graph isomorphism detection by imposing certain restrictions on graphs. Despite the fact that (sub)graph isomorphism detection algorithms have been successfully applied to various problems in computer vision, due to noise, segmentation and articulation errors, and scale differences no significant isomorphism may exist between graphs of similar objects. The limitations of the exact graph matching is depicted in Figure 1, where the silhouettes of two shapes and their undirected shock graphs [15] are shown at the top and bottom rows, respectively. Although the camera silhouettes are similar, no graph or subgraph isomorphism exist between their graphs. As a result of segmentation errors or scale differences, a feature of a shape may correspond to a group of features in the other. Thus, to encode such feature correspondences in the matching, the graph matching problem is expressed rather as that of inexact (or, error tolerant, error correcting) graph matching. The proposed work is focused on the inexact graph matching, whose goal is to find many-to-many vertex correspondences of two graphs. Heuristic based graph traversing [17], graph editing [5,16], and linear programming [3] are only some examples of inexact graph matching algorithms.

Given two graphs, establishing their many-to-many correspondences cannot be done in linear or polynomial time without imposing a restriction on the graph. In the worst case, a collection of vertices, which are not connected to each other,

may match to a collection of vertices of the other graph. A number of all possible subgraphs of a graph is defined by a Bell number [4]. The objective of this paper is to find a pair of subgraphs, one from each input graph and compute the correspondences between their vertices. Each vertex correspondence contributes to the similarity between the input graphs. The similarity values also show the structural relations between the vertices.

In previous work [6], we presented an inexact graph matching that established many-to-many correspondences between vertex sets of two graphs. By transforming graphs into alternative domains, our framework showed that the many-to-many graph matching can be approximately solved in polynomial time. More specifically, the framework represented vertices as d -dimensional points by embedding graphs into geometric space. This embedding ensures that the shortest path distances between the vertices reflect the Euclidean distance between their corresponding points. Using graph embedding techniques, many-to-many graph matching was reformulated as many-to-many point matching. The approach then computed the correspondences between the points using the Earth Mover's Distance (EMD) [14] algorithm. Although the approach provides good retrieval and matching results, it suffers from a major drawback. Namely, the graph embedding under the l_2 norm results in distortion, i.e., the distance between the vertices is not preserved by their corresponding points in the geometric space. In this paper, we overcome this problem through an isometric graph embedding technique. Drawing on an important theorem from the field of graph theory, the graph embedding under l_1 ensures that the distances in the input graphs (trees) are preserved exactly in the geometric space. Experimental evaluation of the proposed paper presents the effectiveness of the approach over the previous work.

The rest of the paper is organized as follows. After providing a brief overview of the previous many-to-many matching algorithm in Section 2, we describe the extension of this algorithm in Section 3. The experiments are presented in Section 4. Finally, we finish the paper with the conclusion in Section 5.

2 Overview of the Many-to-Many Matching Algorithm

The previous work on many-to-many matching is based on representing graphs in geometric spaces. This step is performed by graph embedding techniques. More specifically, the algorithm uses a graph embedding technique to represent each vertex as a point. Here, the objective is to ensure that the distance between any pair of vertices equals to that between their corresponding points. The most important advantage of the embedding used in the previous work comes from the fact that the graphs are embedded into the geometric space of predefined dimensionality.

The embedding approach mentioned above does not work for any type of graphs; it only works for trees. Thus, the many-to-many graph matching approach first represents the graphs as trees and embeds the resulting trees into the d -dimensional geometric space. This two-step transformation reduces the problem of many-to-many graph matching to that of many-to-many point matching,

Fig. 2. Overview of the many-to-many matching algorithm. After representing silhouettes as shock graphs (transition 1), graphs are embedded into geometric spaces of the same dimensionality (transition 2). To compute the matching between the two distributions the Earth Mover's Distance (EMD) algorithm is used (transition 3).

for which the EMD algorithm is used. Note that computing the tree representation of a graph is beyond the scope of this paper. However, the reader is referred to the taxonomy problem [1] in the literature.

The EMD is based on a well-known *transportation problem* [2] and computes the dissimilarity between two distributions. Assume that each element in the first distribution indicates supplies and each element in the second distribution indicates demands at their positions. The EMD then computes the minimum amount of work required to transform one distribution into the other. Formally, let $P = \{(p_1, w_{p1}), \dots, (p_n, w_{pn})\}$ be the first distribution of size n , $Q = \{(q_1, w_{q1}), \dots, (q_m, w_{qm})\}$ be the second distribution of size m , where p_i (or q_i) is the position of the i th element and w_{pi} (or w_{qi}) is the weight, and let d_{ij} denote the ground distance between points p_i and q_j . The objective of this problem is to find a flow matrix $F = [f_{ij}]$ with f_{ij} being the flow between p_i and q_j , which minimizes the overall cost:

$$EMD(P, Q) = \sum_{i=1}^n \sum_{j=1}^m f_{ij} d_{ij}$$

such that $\sum_{i=1}^n f_{ij} \leq w_{pi}$, $\sum_{j=1}^m f_{ij} \leq w_{qj}$, $\sum_{i=1}^n \sum_{j=1}^m f_{ij} = \min(\sum_{i=1}^n w_{pi}, \sum_{j=1}^m w_{qj})$, and $f_{ij} \geq 0$.

The EMD is formulated as a linear programming problem and its solution provides the many-to-many point correspondences. Applying these correspondences back to the original graphs, many-to-many graph matching is obtained. It should be noted that the solution also presents the distance between the point sets. The distance is then used as a dissimilarity score between the original graphs. Thus, given a query and database, the database graph, which most resembles the query can be retrieved by computing the dissimilarity score between the query and each database graph. In the proposed work, the weights of the points are derived from the vertices. In shock graphs, the weight of a vertex represents the maximal bi-tangent circle centered at the vertex. This graph format

is outlined in Section 4. The overview of the many-to-many matching algorithm is given in Figure 2.

3 Embedding Graphs into the l_1 Norm and Many-to-Many Point Matching

The matching algorithm outlined in the previous section suffers from a major drawback. Namely, after representing graphs as trees, embedding the resulting trees into the l_2 norm introduces distortion. As a result, the distance between vertex pairs is not equal to that between their corresponding points in the geometric space. The following theorem, which appeared in [9] specifies this distortion.

Theorem 1. *For a tree T with $l(T)$ leaves, there exists an embedding $\phi : V(T) \rightarrow l_2^d$ with distortion:*

$$D(\phi) \leq O(l^{1/(d-1)})(\min\{\log l(T), d\})^{1/2} \quad (1)$$

where d is the dimension of the geometric space.

As stated in [6], the graph embedding approach introduces on average 17% of distortion in a 100 dimensional space. Consequently, the graphs are not represented exactly in the geometric space and the many-to-many matching between the points may result in wrong correspondences.

This paper overcomes this problem by embedding graphs into l_1 . It is well known that any tree can be embedded isometrically in the l_1 norm [8]. Using this fact, we ensure that no distortion is introduced when moving from the graph domain to the geometric domain. In a d -dimensional geometric space, the distance, U_1 , between two points $X = \{x_1, x_2, \dots, x_d\}$ and $Y = \{y_1, y_2, \dots, y_d\}$ can be computed under the l_1 norm as follows :

$$U_1 = \sum_{k=1}^d |x_k - y_k|. \quad (2)$$

For each vertex in the graph, the coordinate of its corresponding point in the geometric space is computed using the concept of caterpillar decomposition, which captures the topological structure of the graph. Caterpillar decomposition is the collection of the edge-disjoint (sub)root-leaf paths. This concept is described in a sample tree shown in Figure 3. The three paths between a and c , a and g , a and m are called level 1 paths and represent first three paths in caterpillar decomposition. If we remove these three level 1 paths from the tree, we are left with the 3 edge-disjoint paths. These are the paths between e and f , i and j , k and l , called level 2 paths, which represent the other three paths in caterpillar decomposition. If removing the level 2 paths had left additional connected components, the process would be repeated until all the edges in the tree had been removed. The union of the paths is called the caterpillar decomposition, denoted by $\mathfrak{P}$. The total number of paths in $\mathfrak{P}$ specifies the dimensionality of the geometric space into which the graph is embedded.

Fig. 3. Edge-disjoint paths extracted from caterpillar decomposition of a tree

To compute the coordinate of vertex v in the geometric space, we first find the unique path $P(v)$ between v and the root r . Assume that the first segment of $P(v)$ of weight l_1 follows some path $P^1 \in \mathfrak{P}$, the second segment of weight l_2 follows a path $P^2 \in \mathfrak{P}$, and the last segment of weight l_a follows a path $P^a \in \mathfrak{P}$. Let the sequences $\langle P^1, \dots, P^a \rangle$ and $\langle l_1, \dots, l_a \rangle$ be the decomposition sequence and the weight sequence of $P(v)$, respectively. Since each path in $\mathfrak{P}$ corresponds to a coordinate axis, the following process is used to find the coordinate of v . If the decomposition sequence of $P(v)$ consists of a path $P^i \in \mathfrak{P}$, its corresponding coordinate will have a value of l_i as defined in the weight sequence. Otherwise, the corresponding coordinate will have 0. It is easy to see that the embedding obtained through this procedure is isometric under the l_1 norm. To illustrate this procedure, we turn back to Figure 3 in which the tree is embedded into a 6-dimensional space. For example, to compute the coordinates of vertex f , observe that the path between a and f consists of one level 1 path (between a and e) of weight 1.5 and one level 2 path (between e and f) of weight 1.5. Since these paths correspond to the 2nd and 4th paths in caterpillar decomposition of the tree, only 2nd and 4th coordinate of the point representing f will be non-zero. Thus, the coordinates of vertex f in the geometric space are $(0, 1.5, 0, 1.5, 0, 0)$. Although caterpillar decomposition $\mathfrak{P}$ is not unique, the resulting embeddings are all isometric under l_1 . However, to be consistent in our embedding procedure, the order in which the paths in caterpillar decomposition are selected is done by their weights.

It is important to note that embeddings produced by the above algorithm can be of different dimensions. Therefore, in order to match the two embeddings, we must first perform a registration step, whose objective is to project the two distributions into the same normed space. In general, this step can be done in two different ways. The first one is to project the distributions onto the first K right singular vectors of their covariance matrices. This technique is based on Principal Component Analysis and retains the maximum information about the original vectors among all projections onto subspaces of dimension K . Although this technique equalizes the dimensions of the two distributions while

Fig. 4. Sample silhouettes from the dataset

loosing minimal information, it still introduces distortion in the geometric space. Since our objective is to perform a distortion-free embedding, this method is not used in the proposed paper. Given two distributions of different dimensions, the second technique equalizes their dimensions simply by padding zeros to lower dimensional distribution. Despite the fact that this increases the complexity of the EMD algorithm, we use the second technique in the paper. As indicated in the following paragraph, a more efficient implementation of this technique is employed in our framework.

Having isometrically embedded the graphs into the same dimension, we can now proceed with finding the matching between the points. As mentioned before, this step is performed by the EMD algorithm. To generate the many-to-many matching efficiently under l_1 , we use the EMD- L_1 approach presented in [13], which has a simplified structure and better time complexity than the original EMD formulation. In addition, the authors formally proved that the EMD- L_1 is equivalent to the original EMD with l_1 ground distance without approximation.

3.1 Complexity

Since the proposed algorithm consists of several components, we first state the complexity of each component. Computing the tree for a given graph takes $O(|V|^2)$ (see [1] for details). Performing an isometric embedding of trees into the geometric space under l_1 can be done in linear time using depth first search. The EMD is formulated as a linear programming problem and can be solved using a network flow algorithm in $O(|V|^3)$. Finally, mapping the EMD solution back to the graph is $O(|V|)$. Thus, the overall complexity of the proposed approach is bounded by $O(|V|^3)$. Although the complexity of the previous work is not improved theoretically, the experiments presented in the next section demonstrate the effectiveness of the framework on a 2D recognition task.

4 Experiments

In this section, we perform the experimental evaluation of the proposed method and its comparison with the previous work [6]. In our dataset, there are 1620 silhouettes of 9 objects, with 180 views for each. Example silhouettes are shown in Figure 4. Each silhouette is represented by an undirected shock tree, whose nodes represent shocks [15] and whose edges connect adjacent shock points.

Fig. 5. The result of applying the proposed algorithm to the graphs in Figure 1. Many-to-many correspondences have been colored the same. The portion shown in an ellipse in the first graph represents unmatched vertices.

Each shock point p on the skeleton is associated with a 3-dimensional vector $v(p) = (x, y, r)$, where (x, y) are the Euclidean coordinates of the point and r is the radius of the maximal bi-tangent circle centered at the point. Each shock point becomes a node in the shock graph. Each pair of shock points is connected by an edge whose weight reflects the Euclidean distance between them. The graph is converted into a tree by computing its minimum spanning tree. Thus, tree nodes correspond to shock points, and tree edges connect nearby shock points. We choose the root of the tree to be the node that minimizes the sum of the tree-based shortest path distances to all other nodes.

As an illustration of our approach, we first use the example shown in Figure 1, where the need for many-to-many matching is observed. The result of applying our method to these graphs is given in Figure 5 in which many-to-many correspondences are colored the same. The portion shown in an ellipse in the first graph represents unmatched vertices.

To provide a more comprehensive evaluation of our framework, we conducted the following experiment. We first use each database graph as a query (with replacement) to the remaining database. Then, we compute the distance between each query and each of the remaining database graphs using our proposed algorithm. Ideally, given a query view of an object, the matching algorithm should return a neighboring view of the same object. We classify this as a correct pose estimation. According to the results, our framework and the previous work obtain 93.1% and 90.8% pose estimation rates, respectively. In a second experiment, we measure the object recognition rates for both algorithm. Namely, if the algorithm returns a database graph, which belongs to the same object as the query regardless of whether it is a neighboring view, we classify this as a correct object recognition. While our framework achieves 97.5%, the previous work results in 95.3% object recognition rates. The results clearly demonstrate the improved performance offered by the isometric embedding technique.

One important point to consider is that many of the objects are symmetric, and if a query neighbor has an identical view elsewhere on the object, that view might be chosen and scored as an error. Thus, the pose estimation rates, in these

experiments, should be considered as worst-case. In addition, by improving the sampling resolutions of the viewing sphere, we expect that both pose estimation and object recognition rates increase.

5 Conclusions

Matching object features many-to-many is a critical process for object recognition and classification. One-to-one matching algorithms cannot handle segmentation/articulation errors or scale difference, which may exist between features of similar objects. This paper computes the dissimilarity between object pairs represented as graphs by computing the many-to-many matching between their vertices. The proposed framework embeds the input graphs isometrically into a geometric space under the l_1 norm. Unlike the previous algorithm, graph embedding used in the proposed method is distortion-free. After computing the EMD between the distributions, desired many-to-many matchings are obtained. Experimental evaluation of the framework represents more effective results over the previous work. Performing a more comprehensive experimental test using a larger dataset, studying different isometric embedding techniques under various norms, and employing other distribution-based matching algorithms in our framework are our future plans.

References

1. Agarwala, R., Bafna, V., Farach, M., Paterson, M., Thorup, M.: On the approximability of numerical taxonomy (fitting distances by tree metrics. *SIAM Journal on Computing* 28(2), 1073–1085 (1999)
2. Ahuja, R.K., Magnanti, T.L., Orlin, J.B.: *Network Flows: Theory, Algorithms, and Applications*, pp. 4–7. Prentice Hall, Englewood Cliffs (1993)
3. Almohamad, H.A., Duffuaa, S.O.: A linear programming approach for the weighted graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15(5), 522–525 (1993)
4. Bell, E.T.: Exponential numbers. *American Mathematics Monthly* 41, 411–419 (1934)
5. Bunke, H.: Error correcting graph matching: On the influence of the underlying cost function. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21(9), 917–922 (1999)
6. Demirci, F., Shokoufandeh, A., Keselman, Y., Bretzner, L., Dickinson, S.: Object recognition as many-to-many feature matching. *International Journal of Computer Vision* 69(2), 203–222 (2006)
7. Garey, M.R., Johnson, D.S.: *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co, New York (1979)
8. Gupta, A.: Embedding tree metrics into low dimensional euclidean spaces. In: *STOC 1999: Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pp. 694–700. ACM, New York (1999)
9. Gupta, A.: Embedding tree metrics into low-dimensional euclidean spaces. *Discrete & Computational Geometry* 24(1), 105–116 (2000)

10. Hoffmann, C.M.: Group-theoretic algorithms and graph isomorphism. Springer, Berlin (1982)
11. Horaud, R., Skordas, T.: Structural matching for stereo vision. In: Ninth International Conference on Pattern Recognition, Rome, Italy, pp. 439–445 (1988)
12. Lee, S.W., Kim, J.H.: Attributed stroke graph matching for seal imprint verification. *Pattern Recognition Letters* 9, 137–145 (1989)
13. Ling, H., Okada, K.: An efficient earth mover's distance algorithm for robust histogram comparison. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(5), 840–853 (2007)
14. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision* 40(2), 99–121 (2000)
15. Siddiqi, K., Shokoufandeh, A., Dickinson, S., Zucker, S.: Shock graphs and shape matching. *International Journal of Computer Vision* 35(1), 13–32 (1999)
16. Wang, Y.K., Fan, K.C., Horng, J.T.: Genetic-based search for error-correcting graph isomorphism. *IEEE Transactions on Systems, Man, and Cybernetics* 27 (1997)
17. Williams, M.L., Wilson, R.C., Hancock, E.R.: Deterministic search for relational graph matching. *Pattern Recognition* 32(7), 1255–1271 (1999)
18. Wong, E.K.: Model matching in robot vision by subgraph isomorphism. *Pattern Recognition* 25(3), 287–303 (1992)