Int J Comput Vis (2012) 96:28–45
DOI 10.1007/s11263-011-0442-2

Unsupervised Learning for Graph Matching

Marius Leordeanu · Rahul Sukthankar · Martial Hebert

Received: 5 January 2010 / Accepted: 28 March 2011 / Published online: 14 April 2011

Abstract Graph matching is an essential problem in computer vision that has been successfully applied to 2D and 3D feature matching and object recognition. Despite its importance, little has been published on learning the parameters that control graph matching, even though learning has been shown to be vital for improving the matching rate. In this paper we show how to perform parameter learning in an unsupervised fashion, that is when no correct correspondences between graphs are given during training. Our experiments reveal that unsupervised learning compares favorably to the supervised case, both in terms of efficiency and quality, while avoiding the tedious manual labeling of ground truth correspondences. We verify experimentally that our learning method can improve the performance of several state-of-the art graph matching algorithms. We also show that a similar method can be successfully applied to parameter learning for graphical models and demonstrate its effectiveness empirically.

Keywords Parameter learning · Unsupervised learning · Semi-supervised learning · Feature matching · Graph matching · MAP inference

M. Leordeanu (✉)
Institute of Mathematics of the Romanian Academy,
21 Calea Grivitei, Bucharest, Romania
e-mail: marius.leordeanu@imar.ro

R. Sukthankar
Intel Labs Pittsburgh, Pittsburgh, PA 15213, USA
e-mail: rahuls@cs.cmu.edu

M. Hebert
Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh,
PA 15213, USA
e-mail: hebert@ri.cmu.edu

1 Introduction

While there are many papers on solving the graph matching problem efficiently (Berg et al. 2005; Leordeanu and Hebert 2005; Leordeanu et al. 2009; Cour et al. 2006; Gold and Rangarajan 1996; Schellewald and Schnorr 2005; Carcassoni and Hancock 2002; Umeyama 1988; Zass and Shashua 2008; Duchenne et al. 2009) very few propose a solution for learning the optimal set of parameters for graph matching in the context of computer vision applications (Caetano et al. 2007; Leordeanu and Hebert 2008; Caetano et al. 2009). As shown in previous work, learning the parameters is important for improving the matching performance.

In this paper we show how to efficiently perform unsupervised parameter learning for graph matching. A preliminary version of this work appears in Leordeanu and Hebert (2009). Unsupervised learning for matching is important in practice, since manual labeling of correspondences can be quite time consuming. The same basic algorithm can be used in the supervised or semi-supervised cases with minimal modification, if all or some of the ground truth matches are available. We also show empirically that our learning algorithm is robust to the presence of outliers. Our learning algorithm is inspired from the properties of spectral matching (Leordeanu and Hebert 2005), but it can be successfully used for improving the performance of other state-of-the-art matching algorithms (Sect. 5.3).

In earlier work (Leordeanu and Hebert 2005; Leordeanu et al. 2009), we presented algorithms for finding correspondences between two sets of features mainly based on the second-order relationships between them. The reason why spectral matching (Leordeanu and Hebert 2005) works effi-

ciently is because the pairwise geometric scores favor correct assignments much more than incorrect ones. Accidental assignments are rare, so strong pairwise scores between incorrect assignments are unlikely, while such strong scores between most correct ones are very likely. Of course, this is a qualitative, intuitive explanation based on the assumption that the scores are well designed and meaningful. Therefore, an important issue is how to learn the parameters that control the pairwise scores, which will make the algorithm work at its optimal level in the task of matching a specific object type or shape. The key question is: how can we automatically find the optimal parameters during training, particularly without knowing any ground-truth correspondences in the training set?

One would ideally like to keep the pairwise scores between correct assignments high while lowering as much as possible the pairwise scores between incorrect ones. But how can we quantify this goal, and more importantly, how can we learn these scores automatically? This is the issue that we discuss in this paper, showing that it is possible to learn the pairwise scores in an unsupervised way, that is, without knowing the correct assignments during training. We demonstrate experimentally that we can learn meaningful pairwise scores even in the presence of outliers or in cases when the training set is corrupted with pairs of features for which there is no such set of correct assignments (for example, when trying to find correspondences between a motorbike and a car).

In earlier work (Leordeanu and Hebert 2006), we presented an algorithm for MAP inference for Markov Random Fields, inspired from spectral graph matching (Leordeanu and Hebert 2005). Since graph matching and MAP inference can both be formulated as similar integer quadratic programs, it is not surprising that similar algorithms can address both problems, as shown in (Leordeanu and Hebert 2005; Leordeanu and Hebert 2006; Cour and Shi 2007; Leordeanu et al. 2009). Here we further explore the connection between graph matching and MAP inference by also introducing a method for learning the parameters that optimize the MAP inference problem, inspired from the learning method for graph matching. In the case of graph matching, our learning method is the first to learn the parameters of the higher-order (pairwise) terms in both supervised and unsupervised fashions. Here we present a method for learning the parameters that is not probabilistic and whose only goal is to find the parameters that optimize the performance of the inference algorithm, which is related to Szummer et al. (2008) and Max-Margin Markov Networks Taskar et al. (2004). Moreover, we show that for some problems, such as image denoising, we can learn these parameters in a completely unsupervised manner, similar to our unsupervised learning approach to graph matching.

2 Problem Formulation

The graph matching problem consists of finding the indicator vector $\mathbf{x}^*$ that maximizes a quadratic score function:

$$\mathbf{x}^* = \text{argmax}(\mathbf{x}^T \mathbf{M}\mathbf{x}) \quad \text{s.t. } \mathbf{A}\mathbf{x} = \mathbf{1}, \quad \mathbf{x} \in \{0,1\}^n. \tag{1}$$

Here $\mathbf{x}$ is an indicator vector such that $x_{ia} = 1$ if feature i from one image/object (or graph) is matched to feature a from the other image/object (or graph) and zero otherwise. Usually, $\mathbf{A}\mathbf{x} = \mathbf{1},\ \mathbf{x} \in \{\mathbf{0},\mathbf{1}\}^{\mathbf{n}}$ enforces one-to-one constraints on $\mathbf{x}$ such that one feature from one image can be matched to at most one other feature from the other image. In our work $\mathbf{M}$ is a matrix with positive elements containing the pairwise score functions, such that $M_{ia;jb}$ measures how well the pair of features (i, j) from one image agrees in terms of geometry and appearance (e.g., difference in local appearance descriptors, pairwise distances, angles, etc.) with a pair of candidate matches (a, b) from the other. The local appearance terms of candidate correspondences can be stored on the diagonal of $\mathbf{M}$; in practice we noticed that including them in the pairwise scores $M_{ia;jb}$, and leaving zeros on the diagonal gives better results; $M_{ia;jb}$ is basically a function that is defined by a certain parameter vector $\mathbf{w}$. The type of pairwise scores $M_{ia;jb}$ that we use in our experiments is:

$$M_{ia;jb} = \exp\left(-\mathbf{w}^T \mathbf{g}_{ia;jb}\right), \tag{2}$$

where $\mathbf{w}$ is a vector of weights/parameters (to be learned) and $\mathbf{g}_{ia;jb}$ is a vector (usually containing non-negative errors/deformations) describing the changes in geometry and appearance when matching the pair of features (i, j) to the pair of features (a, b).

The MAP inference problem can have a similar formulation as an integer quadratic program (Leordeanu and Hebert 2006; Cour and Shi 2007; Ravikumar and Lafferty 2006). In this case the matrix $\mathbf{M}$ contains the unary potentials (on the diagonal) and the interaction potentials (on the off diagonal elements) that control the joint probabilities. For MAP inference, the constraints $\mathbf{A}\mathbf{x} = \mathbf{1},\ \mathbf{x} \in \{0,1\}^n$ on the solution are usually many-to-one: many nodes from the graph can have the same label. We present the actual implementation of these potentials in the experiments (see Sect. 6.5). For MAP inference each node i is matched to a possible label/class a. The main difference between the two formulations is that, while in the case of graph matching we usually enforce one-to-one constraints on the indicator solution vector $\mathbf{x}$, for MAP inference many-to-one constraints are imposed.

Parameter learning for both graph matching and MAP inference consists of finding a $\mathbf{w}$ that maximizes the performance (w.r.t. to the ground truth correspondences) of matching, as defined by Eq. 1, over pairs of training images or of classification/labeling in the case of Markov Random Fields.

3 Graph Matching Algorithms

Graph matching (Problem 1) is NP-hard, so efficient algorithms must look for approximate solutions. The problem has received more attention in computer vision after its formulation as a quadratic integer programming problem. Many efficient approximate methods have been proposed recently (Berg et al. 2005; Leordeanu and Hebert 2005; Cour et al. 2006; Gold and Rangarajan 1996; Schellewald and Schnorr 2005; Zass and Shashua 2008; Duchenne et al. 2009; Leordeanu et al. 2009). Here, we briefly review two of our recently published algorithms for graph matching: spectral matching (Leordeanu and Hebert 2005) and integer-projected fixed point (IPFP) (Leordeanu et al. 2009).

3.1 Spectral Matching

Since its publication, our approach to spectral graph matching has been applied successfully in a wide range of computer vision applications such as: discovering texture regularity (Hays et al. 2006), object category recognition (Leordeanu et al. 2007), object discovery (Leordeanu et al. 2005; Parikh and Chen 2007; Parikh and Chen 2007), unsupervised modeling of object categories (Kim et al. 2008; Kim et al. 2008), action recognition in video (Yan et al. 2008), recognizing actions from video (Liu et al. 2009), matching 2D object aspects (Ren 2007), 3D scene acquisition (Huhle et al. 2008), capturing 3D human performance (de Aguiar et al. 2008), and symmetry analysis (Chertok and Keller 2010), among others. Also, spectral matching was the starting point for other matching algorithms, such as spectral matching with affine constraints (SMAC) (Cour et al. 2006), integer projected fixed point (IPFP) (Leordeanu et al. 2009), tensor higher-order matching (Duchenne et al. 2009), and algorithms for MAP inference based on spectral relaxations (Leordeanu and Hebert 2006; Cour and Shi 2007).

Spectral matching optimally solves the following relaxed variant of Problem 1:

$$\mathbf{x}^* = \text{argmax}(\mathbf{x}^T \mathbf{M}\mathbf{x}) \quad \text{s.t. } \mathbf{x}^T\mathbf{x} = \mathbf{1}. \tag{3}$$

The solution to this problem is given by the first eigenvector of $\mathbf{M}$. Since $\mathbf{M}$ has only positive elements, by the Perron-Frobenius Theorem, the eigenvector elements are also positive, which makes the post-processing discretization of the eigenvector easier. This eigenvector also has an intuitive interpretation due to the statistical properties of $\mathbf{M}$. We observe that $\mathbf{M}$ can be interpreted as the adjacency matrix of a graph whose nodes represent candidate assignments and edges $M_{ia;jb}$ represent agreements between these possible assignments. This graph has a particular structure, which helps us understand why using the first eigenvector to find an approximate solution to Problem 1 is a good idea. It contains:

1. A strongly connected cluster formed mainly by the correct assignments that tend to establish agreement links (strong edges) among each other. These agreement links are formed when pairs of assignments agree at the level of pairwise relationships (e.g., geometry) between the features they are putting in correspondence.
2. A lot of incorrect assignments, mostly outside of that cluster or weakly connected to it (through weak edges), which do not form strongly connected clusters due to their small probability of establishing agreement links and random, unstructured way in which they form these links.

These statistical properties motivate the spectral approach to the problem. The eigenvector value corresponding to a given assignment indicates the level of *association* of that assignment with the main cluster. We can employ a variety of discretization procedures in order to find an approximate solution. One idea is to apply the Hungarian method, which efficiently finds the binary solution that obeys the one-to-one mapping constraints and maximizes the dot-product with the eigenvector of $\mathbf{M}$. Another idea is to use the greedy discretization algorithm that we originally proposed in Leordeanu and Hebert (2005): we interpret each element of the principal eigenvector $\mathbf{v}$ of $\mathbf{M}$ as the confidence that the corresponding assignment is correct. We start by choosing the element of maximum confidence as correct, then we remove (zero out in $\mathbf{v}$) all the assignments in conflict (w.r.t. the one-to-one mapping constraints) with the assignment chosen as correct, then we repeat this procedure until all assignments are labeled as either correct or incorrect. The eigenvector relaxation of the graph-matching problem, combined with the greedy discretization procedure, makes spectral matching one of the most efficient algorithms for matching using pairwise constraints.

3.2 Integer-Projected Fixed Point Algorithm

In a more recent paper (Leordeanu et al. 2009) we presented another efficient graph matching algorithm (IPFP) that outperforms most state-of-the-art methods. Since its publication, modified versions of IPFP have been applied to segmentation (Brendel and Todorovic 2010) and higher-order MRFs (Semenovich 2010). An algorithm that shares many of IPFP's properties in a different formulation was recently and independently developed by Zaslavskiy et al. (2009, 2010).

Even though our learning method was inspired by the spectral matching algorithm (Leordeanu and Hebert 2005), it can in fact be used in conjunction with other graph matching algorithms, including IPFP.

IPFP can be used as a stand-alone algorithm, or as a discretization procedure for other graph matching algorithms,

such as spectral matching. Moreover, IPFP can also be used for MAP inference in MRF's and CRF's, formulated as quadratic integer programming problems. It solves efficiently (though not necessarily optimally) a tighter relaxation of Problem 1 that is also known to be NP-hard:

$$\mathbf{x}^* = \text{argmax}(\mathbf{x}^T\mathbf{M}\mathbf{x}) \quad \text{s.t. } \mathbf{A}\mathbf{x} = \mathbf{1},\ \mathbf{x} \geq \mathbf{0}. \tag{4}$$

The only difference between Problem 1 and Problem 4, is that in the latter the solution is allowed to take continuous values. Integer Projected Fixed Point (IPFP) takes as input any initial solution, continuous or discrete, and quickly finds a solution obeying the initial discrete constraints of Problem 1 with a better score (most often significantly better) than the initial one:

1. Initialize $\mathbf{x}^* = \mathbf{x_0}$, $S^* = \mathbf{x}_0^T\mathbf{M}\mathbf{x}_0$, $k = 0$, where $x_i \geq 0$ and $\mathbf{x} \neq \mathbf{0}$;
2. Let $\mathbf{b}_{k+1} = P_d(\mathbf{M}\mathbf{x}_k)$, $C = \mathbf{x}_k^T\mathbf{M}(\mathbf{b}_{k+1} - \mathbf{x}_k)$, $D = (\mathbf{b}_{k+1} - \mathbf{x}_k)^T\mathbf{M}(\mathbf{b}_{k+1} - \mathbf{x}_k)$;
3. If $D \geq 0$, set $\mathbf{x}_{k+1} = \mathbf{b}_{k+1}$. Else let $r = \min(-C/D, 1)$ and set $\mathbf{x}_{k+1} = \mathbf{x}_k + r(\mathbf{b}_{k+1} - \mathbf{x}_k)$;
4. If $\mathbf{b}_{k+1}^T\mathbf{M}\mathbf{b}_{k+1} \geq S^*$ then set $S^* = \mathbf{b}_{k+1}^T\mathbf{M}\mathbf{b}_{k+1}$ and $\mathbf{x}^* = \mathbf{b}_{k+1}$;
5. If $\mathbf{x}_{k+1} = \mathbf{x}_k$, stop and return the solution $\mathbf{x}^*$;
6. Set $k = k + 1$ and return to step 2;

where the $P_d(.)$ in step 2 denotes a projection on to the discrete domain, discussed below.

This algorithm is loosely related to the power method for eigenvectors, also used by spectral matching: at step 2 it replaces the fixed point iteration of the power method $\mathbf{v}_{k+1} = P(\mathbf{M}\mathbf{v}_k)$, where $P(.)$ denotes the projection on the unit sphere, with the analogous update $\mathbf{b}_{k+1} = P_d(\mathbf{M}\mathbf{x}_k)$, in which $P_d(.)$ denotes projection on the one-to-one (for graph matching) or many-to-one (for MAP inference) discrete constraints. Since all possible discrete solutions have the same norm, $P_d(.)$ boils down to finding the discrete vector $\mathbf{b}_{k+1} = \arg\max_{\mathbf{b}}(\mathbf{b}^T\mathbf{M}\mathbf{x}_k)$. For one-to-one constraints, this can be efficiently accomplished using the Hungarian method; for many-to-one constraints, the projection can easily be achieved in linear time.

The intuition behind this algorithm is the following: at every iteration the quadratic score $\mathbf{x}^T\mathbf{M}\mathbf{x}$ can be approximated by the first-order Taylor expansion around the current solution $\mathbf{x}_k$: $\mathbf{x}^T\mathbf{M}\mathbf{x} \approx \mathbf{x}_k^T\mathbf{M}\mathbf{x}_k + 2\mathbf{x}_k^T\mathbf{M}(\mathbf{x} - \mathbf{x}_k)$. This approximation is maximized within the discrete domain of Problem 1, in step 2, where $\mathbf{b}_{k+1}$ is found. From (Leordeanu et al. 2009), Proposition 1 we know that the same discrete $\mathbf{b}_{k+1}$ also maximizes the linear approximation in the continuous domain of Problem 4. The role of $\mathbf{b}_{k+1}$ is to provide a direction of largest possible increase (or ascent) in the first-order approximation, simultaneously within both the continuous and discrete domains. Along this direction, the original quadratic score can be further maximized in the continuous domain of Problem 4 (as long as $\mathbf{b}_{k+1} \neq \mathbf{x}_k$). At step 3 we find the optimal point along this direction, also inside the continuous domain of Problem 4. The hope, also confirmed in practice, is that the algorithm will tend to converge towards discrete solutions that are, or are close to, maxima of Problem 4. For MAP inference problems, as shown in Leordeanu et al. (2009), IPFP always converges to discrete solutions, while for graph matching we observe that it typically converges to discrete solutions (but there is no theoretical guarantee).

IPFP can also be seen as an extension to the popular Iterated Conditional Modes (ICM) algorithm (Besag 1986), having the advantage of updating the solution for all nodes in parallel, while retaining the optimality and convergence properties. It is also related to the Frank-Wolfe method (FW) (Frank and Wolfe 1956), a classical optimization algorithm from 1956 most often used in operations research. The Frank-Wolfe method is applied to convex programming problems with linear constraints. A well-known fact about FW is that it has slow convergence rate around the optimum, which is why in practice it is stopped earlier for obtaining an approximate solution. In contrast, in the case of IPFP (applied to graph matching, which is in general not a convex minimization problem) the local optimum is most often discrete (for MAP it is always discrete). When the solution is discrete the optimum is actually found during the optimization of the linear approximation, when the discrete point is found, so the convergence is immediate. This insight is also demonstrated in our experiments, where IPFP most often converges quickly. Therefore, unlike FW, IPFP finds the solution in very few iterations, which is an important advantage.

4 Theoretical Analysis

Our proposed learning algorithm is motivated by the statistical properties of the matrix $\mathbf{M}$ and of its principal eigenvector $\mathbf{v}$, which is the continuous solution given by the spectral graph matching algorithm (Leordeanu and Hebert 2005). In order to analyze the properties of $\mathbf{M}$ theoretically, we need a few assumptions and approximations. The assumptions we make are intuitive and not necessarily rigorous, but they are validated by our numerous experiments. Each instance of the matching problem is unique so nothing can be said with absolute certainty about $\mathbf{M}$ and its eigenvector $\mathbf{v}$, nor the quality of the solution returned. Therefore, we must be concerned with the average (or expected) properties of $\mathbf{M}$ rather than the infinitely many particular cases. We propose a model for $\mathbf{M}$ (Fig. 1) that we validate through experiments.

For a given matching experiment with its corresponding matrix $\mathbf{M}$, let $p_1 > 0$ be the average value of the second-order scores between correct assignments $E(M_{ia;jb})$ for

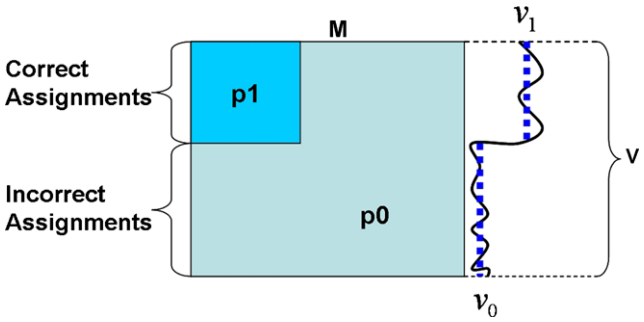

Fig. 1 Pairwise scores (elements of the matching matrix **M**) between correct assignments have a higher expected value p_1 than elements with at least one wrong assignment, with average value p_0. This will be reflected in the eigenvector **v** that will have higher average value v_1 for correct assignments than v_0 for wrong ones

any pair (ia, jb) of correct assignments. Similarly, let $p_0 = E(M_{ia;jb}) \geq 0$ if at least one of the assignments ia and jb is wrong. p_1 should be higher than p_0, since the pairs of correct assignments are expected to agree both in appearance and geometry and have strong second-order scores, while the wrong assignments have such high pairwise scores only accidentally. Intuitively, we expect that the higher p_1 and the lower p_0, the higher the matching rate. We also expect that the performance depends on their ratio $p_r = p_0/p_1$ rather than on their absolute values, since multiplying **M** by a constant does not change the leading eigenvector. Similarly, we define the average eigenvector value $v_1 = E(v_{ia})$ over correct assignments ia, and $v_0 = E(v_{jb})$, over wrong assignments jb. The spectral matching algorithm assumes that correct assignments will correspond to large elements of the eigenvector **v** and the wrong assignments to low values in **v**, so the higher v_1 and the lower v_0 the better the matching rate. As in the case of p_r, if we could minimize during learning the average ratio $v_r = v_0/v_1$ (since the norm of the eigenvector is irrelevant) over all image pairs in a training sequence then we would expect to optimize the overall training matching rate. This model assumes fully-connected graphs, but it can be verified that the results we obtain next are also valid for weakly-connected graphs, as also shown in our experiments.

It is useful to investigate the relationship between v_r and p_r for a given image pair. We know that $\lambda \mathbf{v}_{ia} = \sum_{jb} M_{ia;jb} v_{jb}$. For clarity of presentation, we assume that for each of the n features in the left image there are k candidate correspondences in the right image. We make the following approximation $E(\sum_{jb} M_{ia;jb} v_{jb}) \approx \sum_{jb} E(M_{ia;jb}) E(v_{jb})$, by considering that any v_{jb} is *almost* independent of any particular $M_{ia;jb}$, since **M** is large. The approximation is actually a '$\geq$' inequality, since the correlation is expected to be positive (but very small). For our given matrix **M**, let us call its first eigenvalue λ. It follows that for a correct correspondence ia, $\lambda E(v_{ia}) = \lambda v_1 \approx np_1 v_1 + n(k-1)p_0 v_0$. Similarly, if ia is a wrong correspondence then $\lambda E(v_{ia}) = \lambda v_0 \approx np_0 v_1 + n(k-1)p_0 v_0$. Dividing both equations by $p_1 v_1$ and taking the ratio of the two we obtain:

$$v_r \approx \frac{p_r + (k-1)p_r v_r}{1 + (k-1)p_r v_r}. \tag{5}$$

Solving this quadratic equation for v_r we get:

$$v_r \approx \frac{(k-1)p_r - 1 + \sqrt{(1-(k-1)p_r)^2 + 4(k-1)p_r^2}}{2(k-1)p_r}. \tag{6}$$

Using Eqs. 5 and 6, it can be verified that v_r is a monotonically increasing function of p_r, for $k > 1$. This is in fact not surprising since we expect that the smaller $p_r = p_0/p_1$, the smaller $v_r = v_0/v_1$ and the more binary the eigenvector **v** would be (and closer to the binary ground truth **t**), with the elements of the wrong assignments approaching 0. This approximation turns out to be very accurate in practice, as shown by our experiments in Figs. 3, 4 and 5. Also, the smaller v_r, the higher the expected matching rate, by which we mean the number of correctly matched features divided by the total number of features. For the sake of clarity, during this analysis we assume an equal number of features in both images. We also assume that for each feature in the left image there is one correct match in the right image. However, as we show in the experimental section, our algorithm is robust to the presence of outliers.

One way to minimize v_r is to maximize the correlation between **v** and the ground truth indicator vector **t**, while making sure that one feature from the left images matches one feature in the right image. However, in this paper we want to minimize v_r in an unsupervised fashion, that is without knowing **t** during training. Our proposed solution is to maximize instead the correlation between **v** and its binary version (that is, the binary solution returned by the matching algorithm). How do we know that this procedure will ultimately give a binary version of **v** that is close to the real ground truth? We will investigate this question next.

Let $\mathbf{b}(\mathbf{v})$ be the binary solution obtained from **v**, respecting the one-to-one mapping constraints, as returned by spectral matching for a given pair of images. Let us assume for now that we know how to maximize the correlation $\mathbf{v}^T \mathbf{b}(\mathbf{v})$. We expect that this will lead to minimizing the ratio $v_r^* = E(v_{ia}|b_{ia}(\mathbf{v}) = 0)/E(v_{ia}|b_{ia}(\mathbf{v}) = 1)$. If we let n_m be the number of misclassified assignments, n the number of true correct assignments (same as the number of features, equal in both images) and k the number of candidate assignments for each feature, we can obtain the next two equations: $E(v_{ia}|b_{ia}(\mathbf{v}) = 0) = \frac{n_m v_1 + (n(k-1) - n_m)v_0}{n(k-1)}$ and $E(v_{ia}|b_{ia}(\mathbf{v}) = 1) = \frac{n_m v_0 + (n - n_m)v_1}{n}$. Dividing both by v_1 and taking the ratio of the two we finally obtain:

$$v_r^* = \frac{m/(k-1) + (1 - m/(k-1))v_r}{1 - m + m v_r}, \tag{7}$$

where m is the matching error rate $m = n_m/n$. If we reasonably assume that $v_r < 1$ (eigenvector values higher on average for correct assignments than for wrong ones) and $m < (k-1)/k$ (error rate lower than random) this function of m and v_r has both partial derivatives strictly positive. Since m also increases with v_r, by maximizing $\mathbf{v}^T\mathbf{b}(\mathbf{v})$, we minimize v_r^*, which minimizes both v_r and the true error rate m, so the unsupervised algorithm can be expected to do the right thing. In all of our experiments we obtained values for all p_r, v_r, v_r^* and m that were very close to zero, which is sufficient in practice, even if our gradient-based method (Sect. 5) might not necessarily have found the global minimum.

The model for $\mathbf{M}$ and the equations we obtained in this section are validated experimentally in Sect. 6. By maximizing the correlation between $\mathbf{v}$ and $\mathbf{b}(\mathbf{v})$ over the training sequence we do indeed lower the true misclassification rate m, maximize $\mathbf{v}^T\mathbf{t}$ and also lower p_r, v_r and v_r^*.

5 Algorithms

In this section, we present supervised, semi-supervised and unsupervised learning variants of our approach to graph matching, and detail parameter learning for conditional random fields.

5.1 Supervised Learning for Graph Matching

We want to find the geometric and appearance parameters $\mathbf{w}$ that maximize (in the supervised case) the expected correlation between the principal eigenvector of $\mathbf{M}$ and the ground truth $\mathbf{t}$, which is empirically proportional to the following sum over all training image pairs:

$$J(\mathbf{w}) = \sum_{i=1}^{N} \mathbf{v}^{(i)}(\mathbf{w})^T \mathbf{t}^{(i)}, \tag{8}$$

where $\mathbf{t}^{(i)}$ is the ground truth indicator vector for the ith training image pair. We maximize $J(\mathbf{w})$ by coordinate gradient ascent:

$$w_j^{(k+1)} = w_j^{(k)} + \eta \sum_{i=1}^{N} \mathbf{t}_i^T \frac{\partial \mathbf{v}_i^{(k)}(\mathbf{w})}{\partial w_j}. \tag{9}$$

To simplify notations throughout the rest of this paper, we use $\mathbf{F}'$ to denote the vector or matrix of derivatives of any vector or matrix $\mathbf{F}$ with respect to some element of $\mathbf{w}$. One possible way of taking partial derivatives of an eigenvector of a symmetric matrix (when λ has order 1) is given in Sect. 8.8 of Magnus and Neudecker (1999) and also in Cour et al. (2005) in the context of spectral clustering:

$$\mathbf{v}' = (\lambda\mathbf{I} - \mathbf{M})^{\dagger}(\lambda'\mathbf{I} - \mathbf{M}')\mathbf{v}, \tag{10}$$

where $\mathbf{A}^{\dagger}$ denotes the pseudo-inverse of $\mathbf{A}$ and

$$\lambda' = \frac{\mathbf{v}^T\mathbf{M}'\mathbf{v}}{\mathbf{v}^T\mathbf{v}}. \tag{11}$$

These equations are obtained by using the fact that $\mathbf{M}$ is symmetric and the equalities $\mathbf{v}^T\mathbf{v}' = 0$ and $\mathbf{M}\mathbf{v} = \lambda\mathbf{v}$. However, this method is general and therefore does not take full advantage of the fact that in this case $\mathbf{v}$ is the principal eigenvector of a matrix with large eigengap. $\mathbf{M} - \lambda\mathbf{I}$ is large and also rank deficient so computing its pseudo-inverse is not efficient in practice. Instead, we use the power method to compute the partial derivatives of the approximate principal eigenvector: $\mathbf{v} = \frac{\mathbf{M}^n\mathbf{1}}{\sqrt{(\mathbf{M}^n\mathbf{1})^T(\mathbf{M}^n\mathbf{1})}}$. This is related to Bach and Jordan (2003), but in Bach and Jordan (2003) the method is used for segmentation and as also pointed out by Cour et al. (2005) it could be very unstable in that case, because in segmentation and typical clustering problems the eigengap between the first two eigenvalues is not large.

Here $\mathbf{M}^n\mathbf{1}$ is computed recursively by $\mathbf{M}^{k+1}\mathbf{1} = \mathbf{M}(\mathbf{M}^k\mathbf{1})$. Since the power method is the preferred choice for computing the leading eigenvector, it is justified to use the same approximation for learning. Thus the estimated derivatives are not an approximation, but actually the exact ones, given that $\mathbf{v}$ is itself an approximation based on the power method. Thus, the resulting partial derivatives of $\mathbf{v}$ are computed as follows:

$$\mathbf{v}' = \frac{\|\mathbf{M}^n\mathbf{1}\|^2(\mathbf{M}^n\mathbf{1})' - ((\mathbf{M}^n\mathbf{1})^T(\mathbf{M}^n\mathbf{1})')(\mathbf{M}^n\mathbf{1})}{\|\mathbf{M}^n\mathbf{1}\|^3}. \tag{12}$$

In order to obtain the derivative of $\mathbf{v}$, we first need to compute the derivative of $\mathbf{M}^n\mathbf{1}$, which can be obtained recursively:

$$(\mathbf{M}^n\mathbf{1})' = \mathbf{M}'(\mathbf{M}^{n-1}\mathbf{1}) + \mathbf{M}(\mathbf{M}^{n-1}\mathbf{1})'. \tag{13}$$

Since $\mathbf{M}$ has a large eigengap, as shown in Leordeanu and Hebert (2005), this method is stable and efficient. Figure 2 demonstrates this point empirically. The method is linear in the number of iterations n, but qualitatively insensitive to n, as it works equally well with n as low as 5. These results are averaged over 70 experiments (described later) on 900×900 matrices.

To get a better feeling of the efficiency of our method as compared to Eq. 10, computing Eq. 10 takes 1500 times longer in Matlab (using the function $pinv$) than our method for $n = 10$ on the 900×900 matrices used in our experiments on the House and Hotel datasets. In practice, we manually selected the gradient step size once and used this value in all our experiments.

5.2 Unsupervised and Semi-supervised Learning for Graph Matching

The idea for unsupervised learning (introduced in Sect. 4), is to maximize v_r^* instead of v_r, which could be achieved

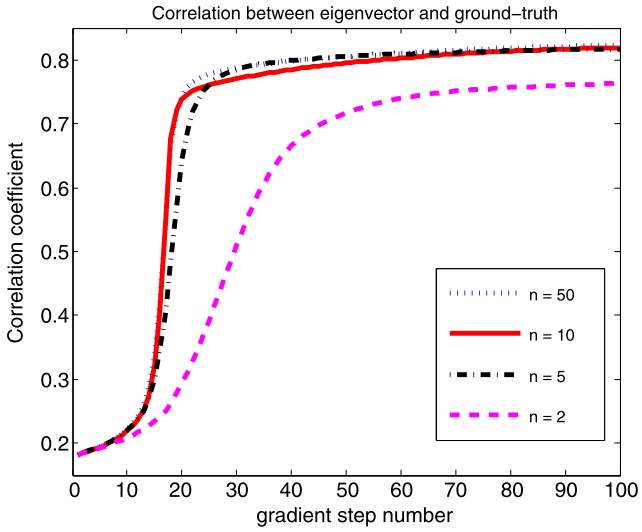

Fig. 2 Experiments on the House sequence. The plots show the normalized correlation between the eigenvector and the ground truth solution for different numbers of recursive iterations n used to compute the approximative derivative of the eigenvector (averages over 70 experiments). Even for n as small as 5 the learning method converges in the same way, returning the same result

through the maximization of the dot-product between the eigenvector and the binary solution obtained from the eigenvector. Thus, during unsupervised training, we maximize the following function:

$$J(\mathbf{w}) = \sum_{i=1}^{N} \mathbf{v}^{(i)}(\mathbf{w})^T \mathbf{b}(\mathbf{v}^{(i)}(\mathbf{w})). \tag{14}$$

The difficulty here is that $\mathbf{b}(\mathbf{v}^{(i)}(\mathbf{w}))$ is not a continuous function and also it may be impossible to express in closed-form, in terms of $\mathbf{w}$, since $\mathbf{b}(\mathbf{v}^{(i)}(\mathbf{w}))$ is the result of an iterative discretization procedure. However, it is important that $\mathbf{b}(\mathbf{v}^{(i)}(\mathbf{w}))$ is piecewise constant and has zero derivatives everywhere except for a finite set of discontinuity points. We can therefore expect that we will evaluate the gradient only at points where $\mathbf{b}$ is constant, and has zero derivatives. Also, at those points, the gradient steps will lower v_r (Eq. 7) because changes in $\mathbf{b}$ (when the gradient updates pass through discontinuity points in $\mathbf{b}$), do not affect v_r. Lowering v_r will increase $\mathbf{v}^T\mathbf{t}$ and also decrease m, so the desired goal will be achieved without having to worry about the discontinuity points of $\mathbf{b}$. This has been verified every time in our experiments. Then, the learning step function becomes:

$$w_j^{(k+1)} = w_j^{(k)} + \eta \sum_{i=1}^{N} \mathbf{b}(\mathbf{v}_i^{(k)}(\mathbf{w}))^T \frac{\partial \mathbf{v}_i^{(k)}(\mathbf{w})}{\partial w_j}. \tag{15}$$

In most practical applications, the user has knowledge of some correct assignments, in which case a semi-supervised approach becomes more appropriate. Our algorithm can easily accommodate such a semi-supervised scenario by naturally combining the supervised and unsupervised learning steps: the discrete solution $\mathbf{b}$ from each step has fixed values for assignments for which the ground-truth information is available, while for the rest of unlabeled assignments we use, as in the unsupervised case, the solution returned by the graph matching algorithm. The ability of easily combining the supervised case with the unsupervised one in a principled manner is another advantage of our proposed method.

5.3 Unsupervised Learning for Other Graph Matching Algorithms

With minimal modification, the unsupervised learning scheme that we proposed can be used for other state-of-the art graph matching algorithms. In Sect. 6.2 we show experimentally that the parameters learned for spectral matching improved the performance of other algorithms. In this section we show that instead of using the binary solutions $\mathbf{b}$ returned by spectral matching during each learning step, we can actually use the binary solutions given by the algorithm for which we want to maximize the performance. This will produce a more efficient learning stage, better suited for that specific graph matching algorithm.

To simplify the notation, we use $\mathbf{M}$ instead of $\mathbf{M}(\mathbf{w})$, which is the more precise notation since all the pairwise scores in the matrix are functions of $\mathbf{w}$. In the same way, let $\mathbf{b}$ denote the binary solution given by some graph matching algorithm, for a given $\mathbf{w}$. For any vector $\mathbf{b}$ with n elements and full rank matrix $\mathbf{M}$ of size $n \times n$, we can write $\mathbf{b}$ as:

$$\mathbf{b} = (\mathbf{b}^T\mathbf{v}_1)\mathbf{v}_1 + (\mathbf{b}^T\mathbf{v}_2)\mathbf{v}_2 + \cdots + (\mathbf{b}^T\mathbf{v}_n)\mathbf{v}_n, \tag{16}$$

where $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ are the eigenvectors of $\mathbf{M}$ ordered in the decreasing order of the magnitudes of their corresponding eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$. Here we can consider each such $\mathbf{M}$ to be full rank due to the presence of random noise in the pairwise scores from that particular matching problem.

It follows that the quadratic score $\mathbf{b}^T\mathbf{M}\mathbf{b}$ can be written as:

$$\mathbf{b}^T\mathbf{M}\mathbf{b} = \lambda_1(\mathbf{b}^T\mathbf{v}_1)^2 + \cdots + \lambda_n(\mathbf{b}^T\mathbf{v}_n)^2. \tag{17}$$

If we consider that $\mathbf{b}$ has unit norm and that λ_1 is the eigenvalue with largest magnitude (also positive, since $\mathbf{M}$ is symmetric with non-negative elements), we immediately obtain the following inequality:

$$\mathbf{b}^T\mathbf{M}\mathbf{b} \geq (2(\mathbf{b}^T\mathbf{v}_1)^2 - 1)\lambda_1. \tag{18}$$

This inequality is very loose in practice because λ_1 is expected to be much larger than the rest of the eigenvalues. Since $\lambda_1 = \mathbf{v}_1^T\mathbf{M}\mathbf{v}_1$, where $\mathbf{v}_1$ is the principal eigenvector,

and $\mathbf{v}_1^T \mathbf{M} \mathbf{v}_1$ is an upper bound to the optimal score $\mathbf{x}_{\text{opt}}^T \mathbf{M} \mathbf{x}_{\text{opt}}$ that obeys the mapping constraints, we obtain

$$\frac{\mathbf{b}^T \mathbf{M}(\mathbf{w}) \mathbf{b}}{\mathbf{x}_{\text{opt}}(\mathbf{w})^T \mathbf{M}(\mathbf{w}) \mathbf{x}_{\text{opt}}(\mathbf{w})} \geq 2(\mathbf{b}^T \mathbf{v}_1(\mathbf{w}))^2 - 1, \tag{19}$$

where $\mathbf{x}_{\text{opt}}(\mathbf{w})$ is the optimal solution of Eq. 1 for a given $\mathbf{w}$. Therefore, by maximizing $\mathbf{b}(\mathbf{w})^T \mathbf{v}_1(\mathbf{w})$, we maximize this lower bound and expect $\mathbf{b}(\mathbf{w})$ to approach the optimal solution $\mathbf{x}_{\text{opt}}(\mathbf{w})$. This is true for solutions $\mathbf{b}(\mathbf{w})$ returned by any approximate graph matching algorithm. This gives an intuition to why the same unsupervised learning scheme may be applied to other algorithms as well: given a specific graph matching algorithm, maximize the dot-product between its binary solution $\mathbf{b}(\mathbf{w})$ and the principal eigenvector $\mathbf{v}_1(\mathbf{w})$. The learning step becomes:

$$\mathbf{w}_j^{(k+1)} = \mathbf{w}_j^{(k)} + \eta \sum_{i=1}^{N} \mathbf{b}_i (\mathbf{w}^{(k)})^T \frac{\partial \mathbf{v}_i^{(k)}(\mathbf{w})}{\partial \mathbf{w}_j}. \tag{20}$$

5.4 Parameter Learning for Conditional Random Fields

The spectral inference method presented in Leordeanu and Hebert (2006) is based on a fixed point iteration similar to the power method for eigenvectors, which maximizes the quadratic score under the L2 norm constraints $\sum_{b=1}^{L} {v_{ib}^*}^2 = 1$. These constraints require that the subvectors corresponding to the candidate labels for each site i have norm 1:

$$v_{ia}^* = \frac{\sum_{jb} M_{ia;jb} v_{jb}^*}{\sqrt{\sum_{b=1}^{L} {v_{ib}^*}^2}}. \tag{21}$$

This equation looks similar to the eigenvector equation $\mathbf{M}\mathbf{v} = \lambda \mathbf{v}$ were it not for the site-wise normalization instead of the global one which applies to eigenvectors. Starting from a vector with positive elements, the fixed point $\mathbf{v}^*$ of the above equation has positive elements, is unique and it is a global maximum of the quadratic score under the constraints $\sum_a v_{ia}^2 = 1$, due to the fact that $\mathbf{M}$ has non-negative elements (Theorem 5 in Baratchart et al. 1998).

The learning method for the MAP problem, which we propose here, is based on gradient ascent, similar to the one for graph matching, and requires taking the derivatives of $\mathbf{v}$ with respect to the parameters $\mathbf{w}$.

Let $\mathbf{M_i}$ be the non-square submatrix of $\mathbf{M}$ of size nLabels $\times$ nLabels $*$ nSites, corresponding to a particular site i. Also let $\mathbf{v}_i$ be the corresponding sub-vector of $\mathbf{v}$, which is computed by the following iterative procedure. Let n be a particular iteration number:

$$\mathbf{v}_i^{(n+1)} = \frac{\mathbf{M}_i \mathbf{v}_i^{(n)}}{\sqrt{(\mathbf{M}_i \mathbf{v}_i^{(n)})^T (\mathbf{M}_i \mathbf{v}_i^{(n)})}}. \tag{22}$$

Let $\mathbf{h}_i$ be the corresponding sub-vector of an auxiliary vector $\mathbf{h}$ defined at each iteration as follows:

$$\mathbf{h}_i = \frac{\mathbf{M}_i' \mathbf{v}_i^{(n)} + \mathbf{M}_i' (\mathbf{v}_i^{(n)})'}{\sqrt{(\mathbf{M}_i \mathbf{v}_i^{(n)})^T (\mathbf{M}_i \mathbf{v}_i^{(n)})}}. \tag{23}$$

Then the derivatives of $\mathbf{v}_i^{(n+1)}$ with respect to some element of $\mathbf{w}$, at step $n+1$, can be obtained recursively as a function of the derivatives of $\mathbf{v}_i^{(n)}$ at step n, by iterating the following update rule:

$$(\mathbf{v}^{(n+1)})' = \mathbf{h} - (\mathbf{h}^T \mathbf{v}^{(n+1)}) \mathbf{v}^{(n+1)}. \tag{24}$$

This update rule, which can be easily verified, is similar to the one used for computing the derivatives of eigenvectors.

The partial derivatives of the individual elements of $\mathbf{M}$ with respect to the individual elements of $\mathbf{w}$ are computed from the equations that define these pairwise potentials, given in Eq. 27. Of course, other differential functions can also be used to define these potentials.

In both supervised and unsupervised cases, the learning update step is similar to the one used for learning graph matching. Here we present the supervised case. In the case of MAP problems we have noticed that unsupervised learning can be successfully applied only to simpler problems, as shown in the experiments Section 6.5. This is due to the fact that in MAP problems it is easily possible to find parameters that will strongly favor one label and make the solution of the relaxed problem almost perfectly discrete. The supervised learning rule for MAP is

$$w_j^{k+1} = w_j^k + \eta \sum_{i=1}^{N} \mathbf{t}_i^T \frac{\partial \mathbf{v}_i^{(k)}(\mathbf{w})}{\partial w_j}, \tag{25}$$

where $\mathbf{t}_i$ is the ground truth labeling for the ith training image.

6 Experimental Analysis

In the case of graph matching we focus on two objectives. The first one is to validate the theoretical results from Sect. 4, especially Eq. 6, which establishes a relationship between p_r and v_r, and Eq. 7, which connects v_r^* to v_r and the error rate m. Each p_r is empirically estimated from each individual matrix $\mathbf{M}$ over the training sequence, and similarly each v_r^* and v_r from each individual eigenvector. Equation 6 is important because it shows that the more likely the pairwise agreements between correct assignments as compared to pairwise agreements between incorrect ones (as reflected by p_r), the closer the eigenvector $\mathbf{v}$ is to the binary ground truth $\mathbf{t}$ (as reflected by v_r), and, as a direct consequence,

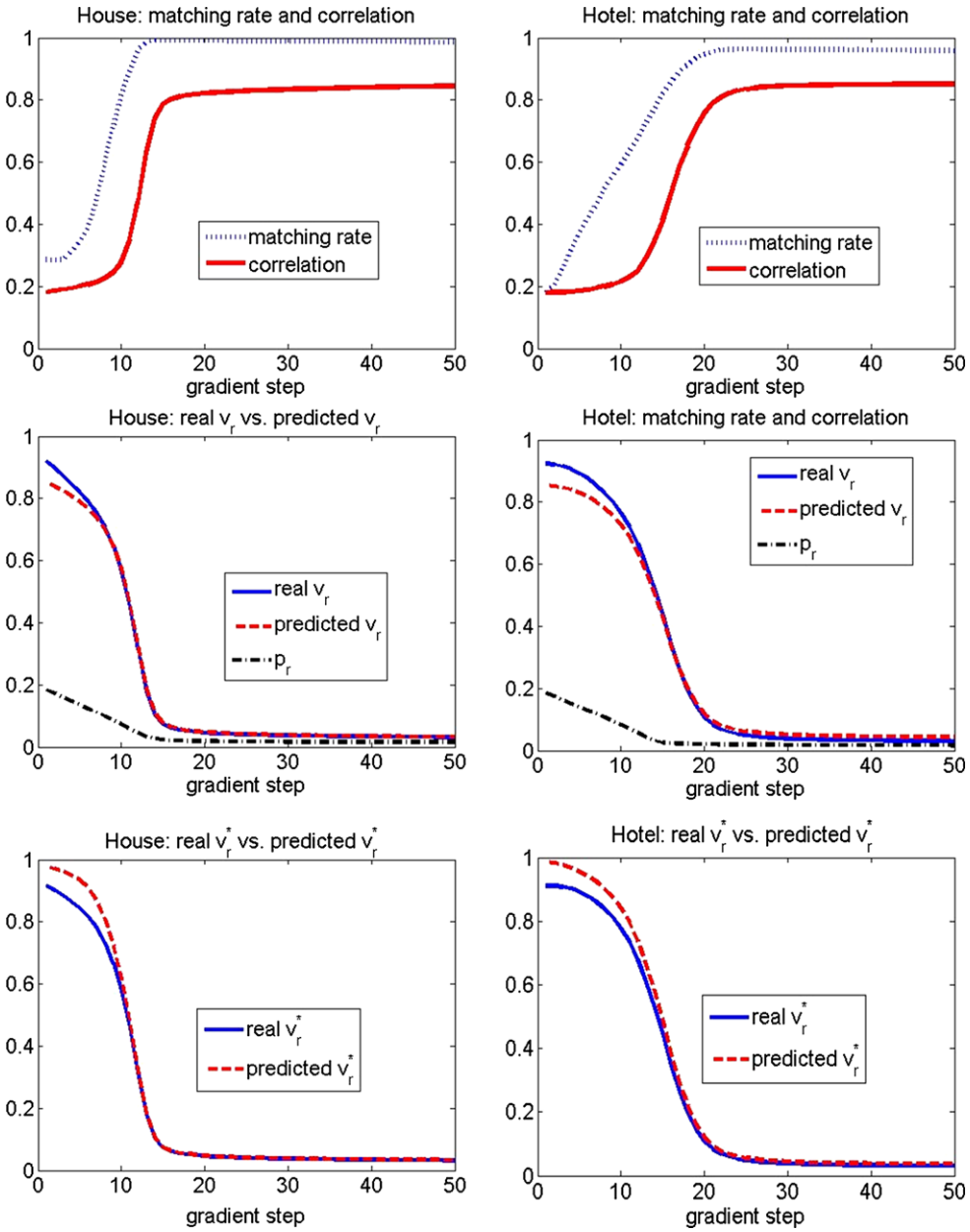

Fig. 3 Unsupervised learning stage. *First row*: matching rate and correlation of eigenvector with the ground truth during training per gradient step. The remaining plots show how *the left hand side* of Eqs. 6 and 7, that is v_r and v_r^*, estimated empirically from the eigenvectors obtained for each image pair, agree with their predicted values (*right hand side* of Eqs. 6 and 7). Results are averages over 70 different experiments

the better the matching performance. This equation also validates our model for the matching matrix $\mathbf{M}$, which is defined by two average values, p_0 and p_1, respectively. Equation 7 is important because it explains why by maximizing the correlation $\mathbf{v}^T\mathbf{b}(\mathbf{v})$ (and implicitly minimizing v_r^*) we in fact minimize v_r and the matching error m. Equation 7 basically shows why the unsupervised algorithm will indeed maximize the performance with respect to the ground truth. We mention that by matching rate/performance we mean the ratio of features that are correctly matched (out of the total number of matched features), while the error rate is 1 minus the matching rate.

The results that validate our theoretical claims are shown in Figs. 3, 4, 5 and 6 on the House, Hotel, Faces, Cars and Motorbikes experiments, respectively. The details of these experiments are given below.

There are a few relevant results to consider. On all four different experiments the correlation between $\mathbf{v}$ and the ground truth $\mathbf{t}$ increases at every gradient step even though the ground truth is unknown to the learning algorithm. The matching rate improves at the same time and at a similar rate with the correlation, showing that maximizing this correlation also maximizes the final performance. In Fig. 4 we display a representative example of the eigenvector for one pair of faces, as it becomes more and more binary during training. If after the first iteration the eigenvector is almost flat, at the last iteration it is very close to the binary ground truth, with all the correct assignments having larger confi-

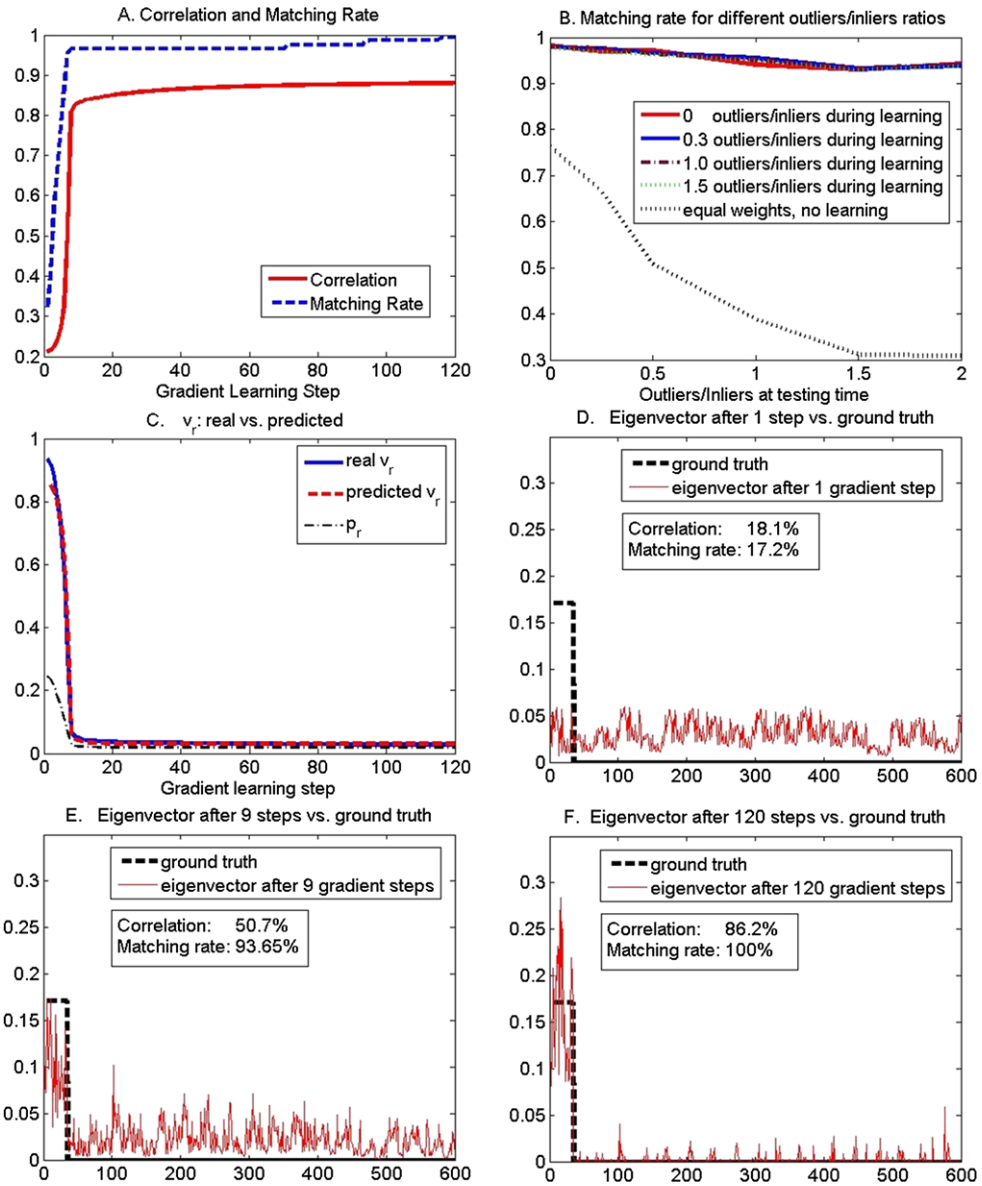

Fig. 4 Results on faces: correlation between eigenvectors and ground truth, and matching rate during training (*top left*), matching rate at testing time, for different outliers/inliers ratios at both learning and test time (*top-right*), verifying Eq. 6 (*middle-left*), example eigenvector for different learning steps. Results in the first three plots are averages over 30 different experiments

dences than any of the wrong ones. Also, on all individual experiments both approximations from Eqs. 6 and 7 become increasingly accurate with each gradient step, from less than 10% accuracy at the first iteration to less than 0.5% error at the last. In all our learning experiments we started from a set of parameters $\mathbf{w}$ that does not favor any assignment ($\mathbf{w} = 0$, which means that before the very first iteration all non-zeros scores in $\mathbf{M}$ are equal to 1). These results motivate both the model proposed for $\mathbf{M}$ (Eq. 6), but also the results (Eq. 7) that support the unsupervised learning scheme.

The second objective of our experiments is to evaluate the matching performance, before and after learning, on new test image pairs. The goal is to show that, at testing time, the matching performance after learning is significantly better than if no learning was done.

6.1 Learning with Unlabeled Correspondences

Matching Rigid Objects under Perspective Transformations We first perform experiments on two tasks that are the same as those in Caetano et al. (2007) and our previous work (Leordeanu and Hebert 2008). We use exactly the same image sequences (House: 110 images and Hotel: 100 images) both for training and testing and the same features, which were manually selected by Caetano et al. For testing we use all the pairs between the remaining images. The pairwise scores $M_{ia;jb}$ are the same as the ones that we previously used in Leordeanu and Hebert (2008), using the Shape-Context descriptor (Belongie et al. 2002) for local appearance, and pairwise distances and angles for the second-order relationships. They measure how well features (i, j) from one image agree in terms of geome-

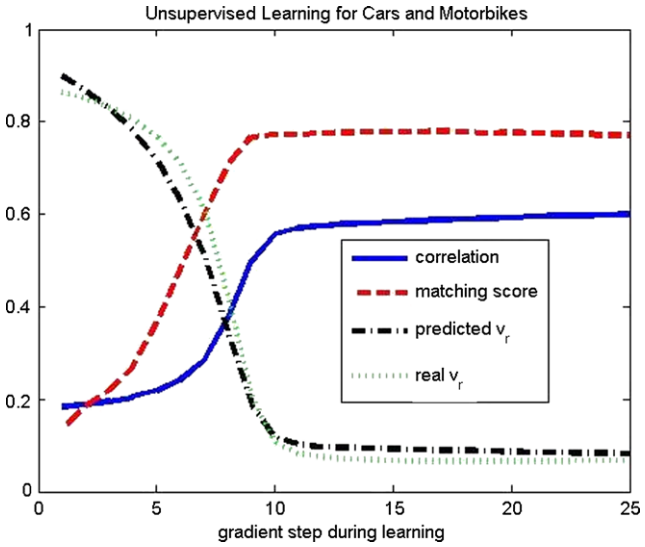

Fig. 5 Correlation and matching rate w.r.t. the ground truth during unsupervised learning for Cars and Motorbikes from Pascal 2007 challenge. Real and predicted v_r decrease as predicted by the model. Results are averaged over 30 different experiments

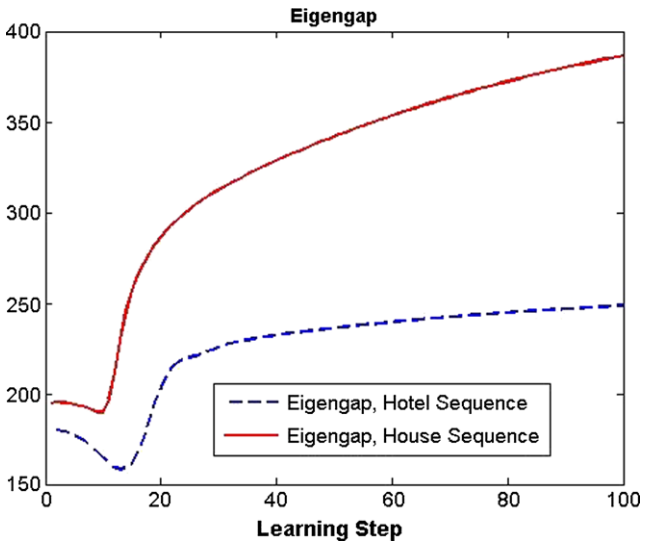

Fig. 6 During unsupervised learning, the normalized eigengap (eigengap divided by the mean value in **M**) starts increasing after a few iterations, indicating that the leading eigenvector becomes more and more stable. Results are on the House and Hotel datasets averaged over 70 random experiments

try and appearance with their candidate correspondences (a, b). More explicitly, the pairwise scores have the form $M_{ia;jb} = \exp(-\mathbf{w}^T \mathbf{g}_{ia;jb})$, where $\mathbf{g}_{ia;jb} = [|s_i - s_a|, |s_j - s_b|, \frac{|d_{ij}-d_{ab}|}{|d_{ij}+d_{ab}|}, |\alpha_{ij} - \alpha_{ab}|]$. Here, s_a denotes the shape context of features a; d_{ij} is the distance between features (i, j); and α_{ij} is the angle between the horizontal axis and the vector $\overrightarrow{ij}$. Learning consists of finding the vector of parameters $\mathbf{w}$ that maximizes the matching performance on the training sequence.

Table 1 Matching performance on the hotel and house datasets at testing time. In our experiments we used only 5 training images from the 'House' sequence, while for Caetano et al. (2007), we report upper bounds of their published results using both 5 and 106 training images. Notation: 'S' and 'U' denote 'supervised' and 'unsupervised', respectively

Dataset	Ours: S(5)	Ours: U(5)	(Caetano et al. 2007): S(5)	(Caetano et al. 2007): S(106)
House	99.8%	99.8%	< 84%	< 96%
Hotel	94.8%	94.8%	< 87%	< 90%

As in both (Caetano et al. 2007) and (Leordeanu and Hebert 2008), we first obtain a Delaunay triangulation and allow non-zero pairwise scores $M_{ia;jb}$ if and only if both (i, j) and (a, b) are connected in their corresponding triangulation. Our previous method (Leordeanu and Hebert 2008) is supervised and based on a global optimization scheme that is more likely to find the true global optimum than the unsupervised gradient based method proposed in this paper. Therefore, it is significant to note that the proposed unsupervised learning method matches our previous results, while significantly outperforming Caetano et al. (2007) (Table 1).

We point out that the main reason for the significant difference in performance between ours and Caetano et al. (2007) has to do with the fact that (Caetano et al. 2007) puts less emphasis on the second-order geometry in the pairwise scores $M_{ia;jb}$, by using only information from the 0–1 Delaunay triangulation and no information about pairwise distances and angles. On the contrary, we emphasize the importance of second-order relationships, since in our experiments, even when we leave out completely the shape-context descriptors and use only the pairwise geometric information, the performance of our method does not degrade. Of course, it is also important to stress that, while the method by Caetano et al. (2007) learns the parameters in a supervised way, ours is the first to do so in an unsupervised fashion.

Next we investigate the performance at learning and testing stages of the unsupervised learning method vs. its supervised variant (when the ground truth assignments are known). We perform 70 different experiments using both datasets, by randomly choosing 10 training images (and using all image pairs from the training set) and leaving all pairs of the rest of images for testing. As expected, we observe that the unsupervised method learns somewhat slower on average than the supervised one, but the parameters they learn are almost identical. In Fig. 7 we plot the average correlation (between the eigenvectors and ground truth) and matching rate at each gradient step for all training pairs and all experiments vs. each gradient step, for both the supervised and unsupervised cases. It is interesting that while the unsupervised version tends to converge slower, after several itera-

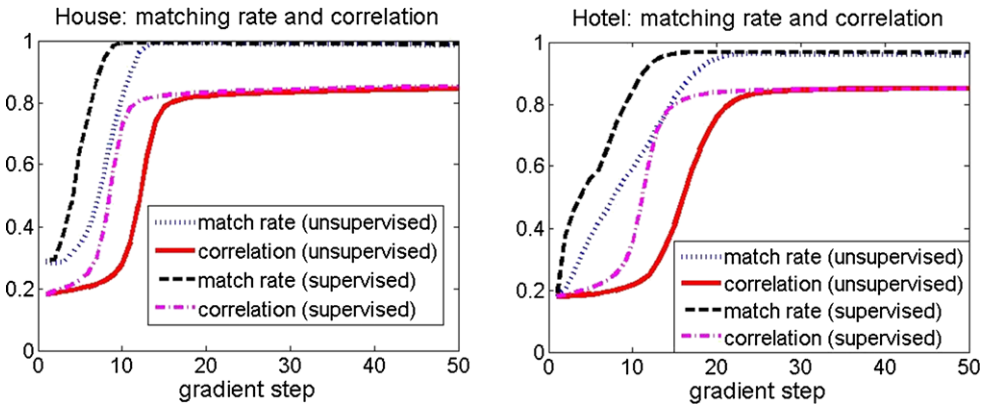

Fig. 7 Supervised vs. unsupervised learning: Average match rate and correlation between the eigenvector and the ground truth over all training image pairs, over 70 different experiments using 10 randomly-chosen training images from the House and Hotel sequences, respectively. Notice how unsupervised learning converges to the same correlation and matching rate as the supervised one

Table 2 Comparison of average matching performance at testing time on the house and hotel datasets for 70 different experiments (10 training images, the rest used for testing). We compare the case of unsupervised learning vs. no learning. *First column*: unsupervised learning; *Second*: no learning, equal default weights **w**

Datasests	Unsupervised learning	No learning
House+Hotel	99.14%	93.24%

tions their performances (and also parameters) converge to the same values. During testing the two methods performed identically in terms of matching performance (average percentage of correctly matched features over all 70 experiments). As compared to the same matching algorithm without learned parameters the two algorithms performed clearly better (Table 2). Without learning, the default parameters (elements of **w**) were chosen to be all equal.

Matching Deformable 2D Shapes with Outliers The third dataset used for evaluation consists of 30 random image pairs selected from Caltech-4 Faces dataset. The experiments on this dataset are different from the previous ones for two reasons: the images contain not only faces but also background clutter, and, the faces belong to different people, both women and men, with different facial expressions, so there are significant non-rigid deformations between the faces that have to be matched. The features we used are oriented points sampled along contours extracted in the image in a similar fashion as in our previous work (Leordeanu et al. 2007) (Fig. 8). The orientation of each point is the normal vector at that point to the contour where the point was sampled. The points on the faces that have to be matched (the inliers) were selected manually, while the outliers (features in the background) were selected randomly, while making sure that each outlier is not too close (15 pixels) to any other point. For each pair of faces we manually selected the ground truth (the correct matches) for the inliers only. The pairwise scores contain only geometric information about

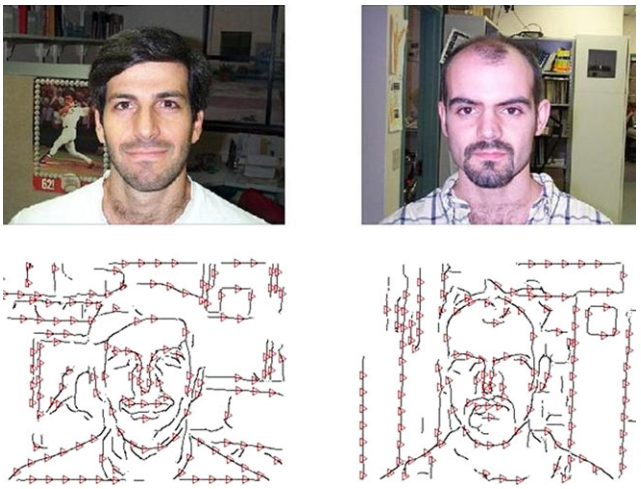

Fig. 8 *Top row*: a pair of faces from Caltech-4 dataset used in our experiments. *Bottom row*: the contours extracted and the points selected as features

pairwise distances and angles:

$$M_{ia;jb} = e^{-\mathbf{w}^T \mathbf{g}_{ia;jb}}, \tag{26}$$

where **w** is a vector of 7 parameters (that have to be learned) and $\mathbf{g}_{ia;jb} = [|d_{ij} - d_{ab}|/d_{ij}, |\theta_i - \theta_a|, |\theta_j - \theta_b|, |\sigma_{ij} - \sigma_{ab}|, |\sigma_{ji} - \sigma_{ba}|, |\alpha_{ij} - \alpha_{ab}|, |\beta_{ij} - \beta_{ab}|]$. Here d_{ij} is the distance between the features (i, j), θ_i is the angle between the normal of feature i and the horizontal axis, σ_{ij} is the angle between the normal at point i and the vector $\overrightarrow{ij}$, α_{ij} is the angle between $\overrightarrow{ij}$ and the horizontal axis and β_{ij} is the angle between the normals of i and j.

We performed 30 random experiments by randomly picking 10 pairs for training and leaving the rest 20 for testing. The results shown in Fig. 4 are averages over the 30 experiments. The top-left plot shows how, as in the previous experiments, both the correlation $\mathbf{v}^T\mathbf{t}$ and the matching performance during training improves with every learning step. During training and testing we used different percentages of outliers to evaluate the robustness of the method (top-right

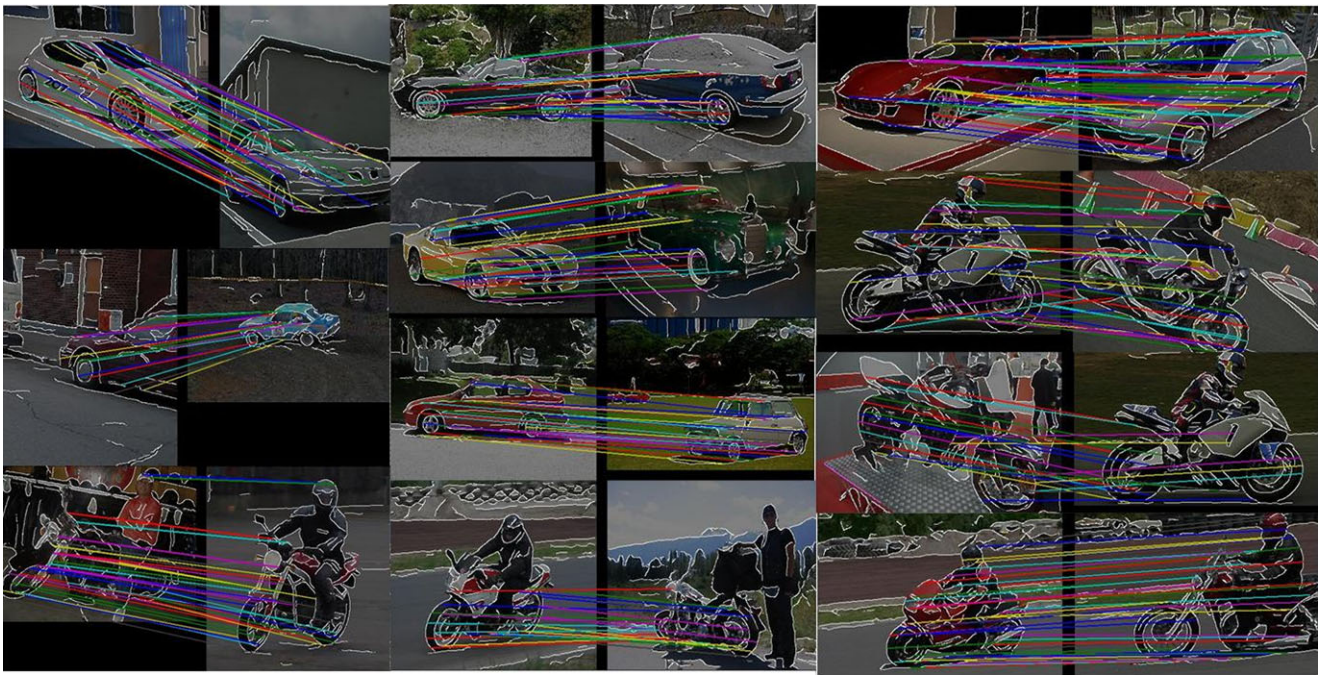

Fig. 9 (Color online) Matching results on image pairs from Pascal 2007 challenge. Notice the significant differences in shape, view-point and scale. Best viewed in color

plot). The learning method is robust to outliers, since the matching performance during testing does not depend on the percentage of outliers introduced during training (the percentage of outliers is always the same in the left and the right images), but only on the percentage of outliers present at testing time. Without learning (the dotted black plot), when the default parameters chosen are all equal, the performance is much worse and degrades faster as the percentage of outliers at testing time increases. This suggests that learning not only increases the matching rate, but it also makes it more robust to the presence of outliers.

6.2 Learning with Unlabeled Object Classes and Correspondences

In our previous experiments every pair of training images contained the same object/category, so a set of inliers exists for each such pair. Next, we evaluated the algorithm on a more difficult task: the training set is corrupted such that half of the image pairs contain different object categories. In this experiment we used cars and motorbikes from Pascal 2007, a much more difficult dataset (see Fig. 9). For each class we selected 30 pairs of images and for each pair between 30 to 60 ground truth correspondences. The features and the pair-wise scores were of the same type as in the experiments on faces: points and their normals selected from pieces of contours. In Fig. 9 we show some representative results after learning, with matching rates over 80%; contours are overlaid in white. During each training experiment we randomly picked 5 pairs containing cars, 5 containing motorbikes and 10 discordant pairs: one containing a car and the other one a motorbike (a total of 20 pairs for each learning experiment). For testing we used the remaining pairs of images, such that each pair contains the same object class. The learning algorithm had no knowledge of which pairs are discordant, what classes they contain and which are the ground truth correspondences. As can be seen in Fig. 5, at each gradient step both the matching rate and the correlation of the eigenvector w.r.t. the ground truth increases (monitored only for pairs containing the same category). The proposed model is again verified as shown by the plots of the real and ideal v_r that are almost identical. Not only that the learning algorithm was not significantly influenced by the presence of discordant pairs but it was also able to find a single set of parameters that matched well both cars and motorbikes. Learning and testing results are averaged over 30 experiments.

Using the testing image pairs of cars and motorbikes, we used several graph matching algorithms (for a more extensive discussion and comparison see Sect. 6.4): spectral matching (SM) using the row/column procedure from Zass and Shashua (2008) during post-processing of the eigenvector, with probabilistic matching (PM) using pair-wise constraints from Zass and Shashua (2008), and the well-known graduated assignment algorithm from Gold and Rangarajan (1996) (GA). The same parameters and pair-wise scores were used by all algorithms, learned as described above. When no outliers were allowed all algorithms had similar matching rates (above 75%) with learning moderately im-

Table 3 Comparison of matching rates for 3 graph matching algorithms before and after unsupervised learning on Cars and Motorbikes from Pascal07 database, with all outliers from the right image allowed and no outliers in the left image. When no outliers were allowed all algorithms had a matching rate of over 75%, with learning moderately improving the performance

Dataset	SM	PM	GA
Cars: no learning	26.3%	20.9%	**31.9**%
Cars: with learning	**62.2**%	34.2%	47.5%
Motorbikes: no learning	29.5%	26.1%	**34.2**%
Motorbikes: with learning	**52.7**%	41.3%	45.9%

proving the performance. When outliers were introduced in the right image (in the same fashion as in the experiments on Faces) the performance improvement after learning was much more significant for all algorithms, with spectral matching benefiting the most (Table 3).

6.3 Learning in the Context of Recognition

Next we investigate how unsupervised learning for graph matching can improve object recognition. Here we are not interested in developing a recognition algorithm, but only on demonstrating the improvement in recognition after learning for matching, while using a simple nearest neighbor object classification algorithm. We believe that better matching should better cluster together images showing objects of the same class, while separating more images of objects from different classes. The nearest-neighbor algorithm is perfectly suited for evaluating this intuition. For this experiment, we used the cropped training images (using the bounding boxes provided) from the Pascal '05 database (see Fig. 10). We split the cropped images randomly in equal training and testing sets. On the training set we learn the matching parameters on pairs containing objects from the same category. The features used and the pairwise scores are the same as in the experiments on faces, except that this time we used fully connected models (about 100–200 features per image). At testing time, for each test image, we return the class of the training image that returned the highest matching score $\mathbf{x}^T\mathbf{M}\mathbf{x}$ (Eq. 1). We perform this classification task both with and without learning (with default equal $\mathbf{w}$ weights). The results (Table 4) suggest that unsupervised learning for graph matching can be used effectively in an object recognition system. In Fig. 11 we see some results during learning. In this case we monitor only the eigengap and $\mathbf{b}(\mathbf{v})^T\mathbf{v}$ because we did not have the ground truth available.

6.4 Learning for Different Graph Matching Algorithms

We perform unsupervised learning for five state of the art graph matching algorithms on the Cars and Motorbikes data

Fig. 10 Cropped images (using bounding boxes) from the Pascal '05 training set, used in our classification experiments. The images in this dataset, even though they are cropped using bounding boxes, are more difficult than in the previous experiments, since the objects of the same category have sometimes very different shapes, undergo significant change in viewpoint, and contain background clutter

Table 4 Comparison of 4-class (bikes, cars, motorbikes and people) classification performance at testing time on the task from Sect. 6.3. Unsupervised learning for graph matching significantly reduces the classification error rate by more than 2-fold

With learning	No learning
80.8%	57.4%

from Pascal '07 challenge, the same as that used in Sect. 6.2. We use the same features (oriented points selected from pieces of contours) as described in Sect. 6.2. The algorithms are: spectral matching (Leordeanu and Hebert 2005) (SM), spectral matching with affine constraints (Cour et al. 2006) (SMAC), graduated assignment (Gold and Rangarajan 1996) (GA), probabilistic graph matching (Zass and Shashua 2008) (PM) and our Integer Projected Fixed Point Algorithm (Leordeanu et al. 2009) (IPFP).

For each algorithm, we perform 30 different learning and testing experiments for each class and we average the results. For each experiment we randomly pick 10 pairs of images for learning (with outliers) and leave the remaining 20 for testing (with and without outliers). During training we add outliers to one image in every pair, such that the ratio of outliers to inliers is 0.5. The other image from the pair contains no outliers. We introduce this moderate amount of outliers during training in order to test the robustness of the unsupervised learning method in real-world experiments, where, especially in the unsupervised case, it is time consuming to enforce an equal number of features in both images in every pair. During testing we have two cases: we had no outliers in both images in the first case, and allowed all outliers possible in only one image in the second case. The number of outliers introduced was significant, the ratio of outliers to inliers ranging from 1.4 to 8.2 for the Cars class (average of 3.7), and from 1.8 to 10.5 for

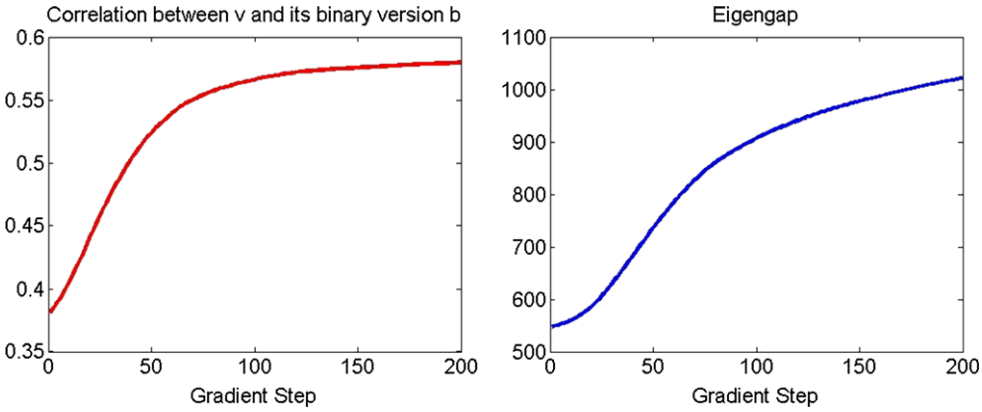

Fig. 11 Pascal '05, learning stage: (*left plot*) the average correlation (Eq. 14) increases with each gradient step, which validates the gradient update rule; (*right plot*) the normalized eigengap increases, indicating that the eigenvector is more and more stable and robust to noise as it becomes more binary

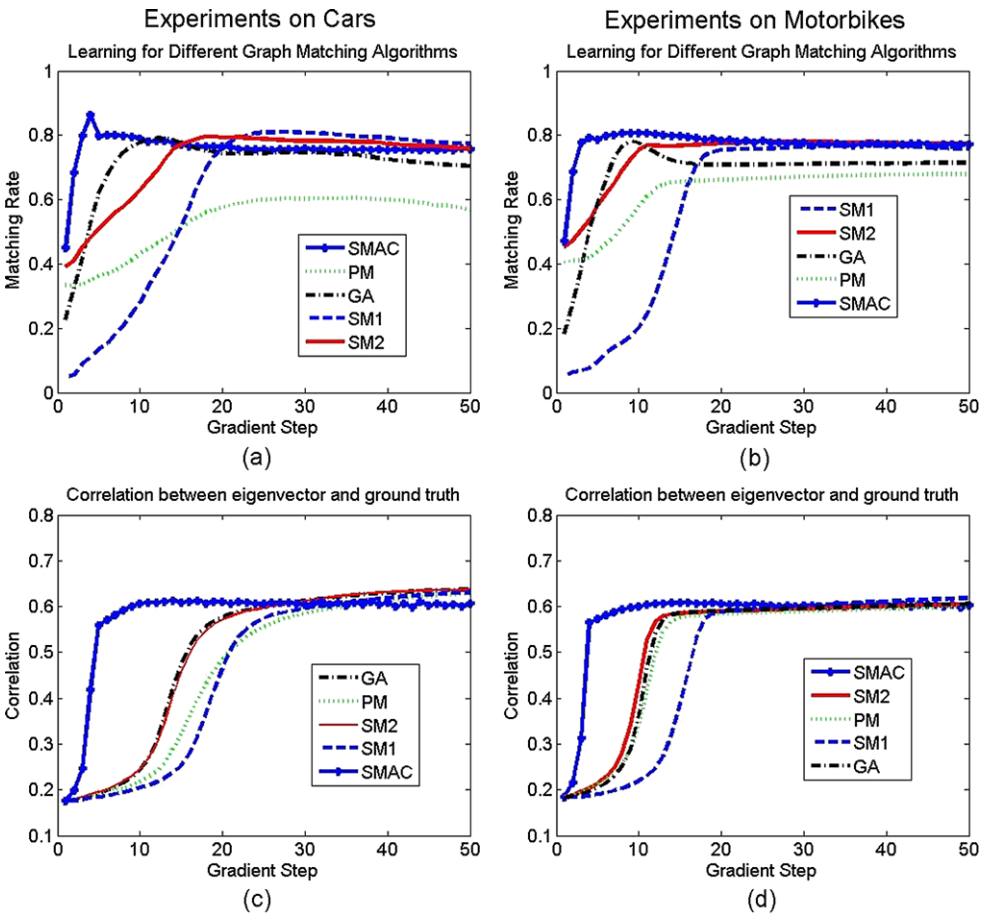

Fig. 12 *First row*: the average matching rate with respect to the ground truth, during training for each algorithm at each gradient step. *Second row*: average correlation between the principal eigenvector and the ground truth during learning for each algorithm at each gradient step. SMAC converges faster then the other algorithms. The final parameters learned by all algorithms are similar

the Motorbikes class (average of 5.3). As in all our other tests, by an inlier we mean a feature for which there exists a correspondence in the other image, according to the ground truth, whereas an outlier has no such correct correspondence. The inliers were manually picked, the same as the ones used in Sect. 6.2, whereas the outliers were chosen randomly on pieces of contours such that no outlier is closer than 15 pixels to any other feature selected.

In Fig. 12, we display the behavior of each algorithm during learning: average matching rate and average correlation of the eigenvector with the ground truth at each learning step. There are several important aspects to notice: the correlation between the eigenvector and the ground truth increases with every gradient step for all algorithms, SMAC converging much faster than the others. This is reflected also in the matching rate, that increases much faster for SMAC. All algorithms benefit from learning, as all matching rates improve significantly after several iterations. The vector of parameters $\mathbf{w}$ was initialized to zero and the final $\mathbf{w}$'s learned are similar for all the algorithms. GA and SMAC

Table 5 Comparison of matching rates at testing time for different graph matching algorithms before (NL) and after (WL) unsupervised learning on Cars and Motorbikes from Pascal '07 database, with outliers: all outliers allowed in the right image, no outliers in the left image. The algorithm used during testing was the same as the one used for learning. Results are averages over 30 different experiments. The same parameters were used by all algorithms for the case of no learning with and without outliers: all elements of **w** being equal

Dataset	IPFP	SM	SMAC	GA	PM
Cars (NL)	**50.9**%	26.3%	39.1%	31.9%	20.9%
Cars (WL)	**73.1**%	61.6%	64.8%	46.6%	33.6%
Motorbikes (NL)	32.9%	29.7%	**39.2**%	34.2%	26.1%
Motorbikes (WL)	**55.7**%	54.8%	52.4%	46.8%	42.2%

have a rapid improvement in the first 10 steps, followed by a small decline and a plateau for the remaining iterations. This might suggest that for GA and SMAC learning should be performed only for a few iterations. For the other three algorithms learning constantly improves the matching rate during training.

In Table 5, we show the test results of all algorithms with and without learning, for both datasets, when outliers are introduced. Without learning all algorithms use a parameter vector $\mathbf{w} = [0.2, 0.2, 0.2, 0.2, 0.2]$ on both datasets. In our experiments, the more outliers we introduce during testing the more beneficial learning becomes. This is also in agreement with our experiments on faces (Fig. 4). Table 5 shows the results with and without learning in the presence of a significant number of outliers (no outliers in one image and all possible outliers in the other image, as explained previously). It is evident that learning significantly improves the performance of all algorithms. The results shown in this section strongly suggest that our unsupervised learning scheme can significantly improve the performance of other algorithms on difficult data (such as the Cars and Motorbikes from Pascal '07) in the presence of a large number of outliers.

6.5 Parameter Learning for Conditional Random Fields

In order to compare our method to previous work on CRFs, we have followed exactly the experiments of Kumar on image denoising, following the implementation details and test data provided in Kumar (2005). The task is to obtain denoised images from corrupted binary 64×64 images. We used the same four images and the same noise models. For the easier task the noise model is Gaussian with mean $\mu = 0$ and standard deviation $\sigma = 0.3$ added to the 0–1 binary images. For the more difficult task we used as in Kumar (2005), for each class, a different mixture of two Gaussians with equal mixing weights yielding a bimodal noise. The model parameters (mean, std) for the two Gaussians

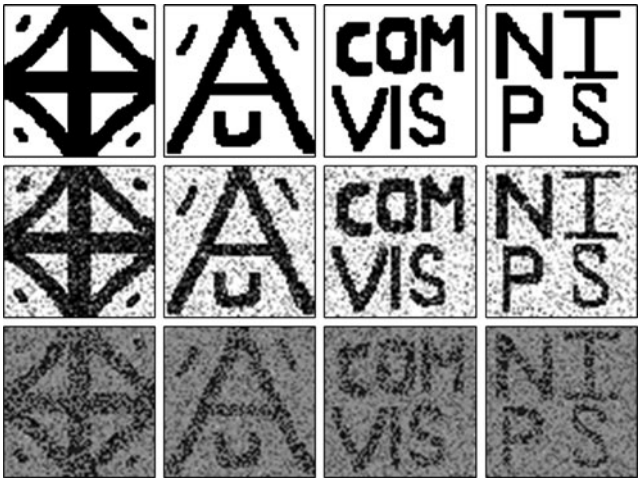

Fig. 13 *First row*: original binary images (left one used for training, next three for testing). *Second row*: images corrupted with unimodal noise. *Third row*: images corrupted with bimodal noise

Table 6 Comparisons with Kumar (2005) on the same experiments. In Kumar (2005), 50 noisy versions of the first image are used for training. We used only 5 noisy versions of the first image are used for training. For testing both approaches use 50 noisy versions of the remaining three images. Note that the unsupervised learning matches the performance of the supervised one. The inference method used in Kumar (2005) is graph cuts and the learning methods are maximum pseudo-likelihood (PL) and maximum penalized pseudo-likelihood (PPL)

Algorithm	L2QP	IPFP	(Kumar 2005): PPL	(Kumar 2005): PL
Unimodal (sup.)	0.75%	**0.73**%	2.3%	3.82%
Unimodal (unsup.)	0.85%	**0.69**%	NA	NA
Bimodal (sup.)	7.15%	15.94%	**6.21**%	17.69%

were [(0.08, 0.03), (0.46, 0.03)] for the foreground class and [(0.55, 0.02), (0.42, 0.10)] for the background class. The original images together with examples of their noisy versions are shown in Fig. 13.

Unlike Kumar (2005), which uses 50 randomly-generated noisy versions of the first image for training, we used only 5 such images. For the simpler task we also performed completely unsupervised learning (Fig. 14) getting almost identical results (Table 6). Our results were significantly better for the simpler noise model, while matching the results from Kumar (2005) for the more difficult noise model. Also note that our learning method is easier to implement and improves the performance of IPFP, not just L2QP (our spectral MAP inference algorithm from Leordeanu and Hebert (2006) for which it was originally designed). The pairwise potentials we used are:

$$M_{ia;jb} = \sigma(\mathbf{w}^T[t_a;\ t_a I_i;\ t_b;\ t_b I_j;\ t_a t_b |I_i - I_j|]), \qquad (27)$$

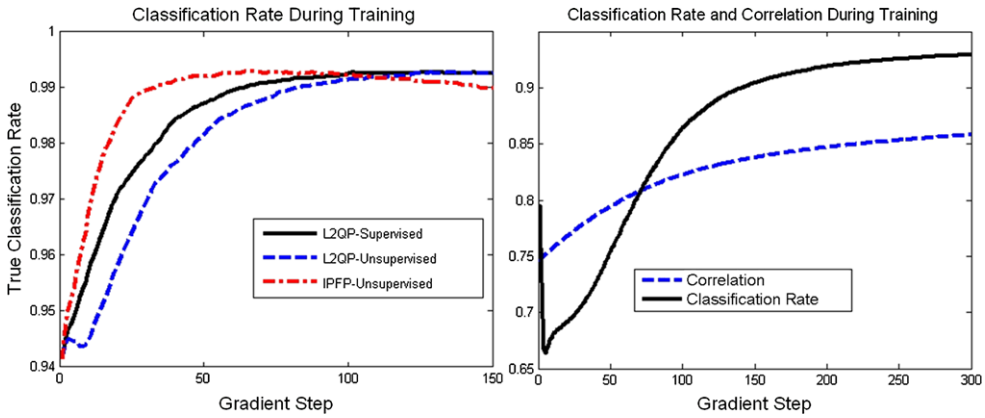

Fig. 14 *Left plot*: supervised and unsupervised learning when unimodal noise is used. *Right plot*: supervised learning when bimodal noise is used. Results are averages over 5 training images for each learning gradient step

where I_i is the value of pixel i in the image and $|I_i - I_j|$ is the absolute difference in image pixel values between connected sites i and j. Following (Kumar 2005), we used 4-connected lattices. We also experimented with 8-connected neighborhoods with no significant difference in performance.

The parameter values for all of our learning experiments were:

- Initial,
  $\mathbf{w} = [0.5;\ -1;\ 0.5;\ -1;\ -0.5;\ 1]$;
- Unsupervised, unimodal noise,
  $\mathbf{w} = [1.27;\ -2.55;\ 1.27;\ -2.55;\ -2.50;\ 0.47]$;
- Supervised, unimodal noise,
  $\mathbf{w} = [1.27;\ -2.55;\ 1.26;\ -2.55;\ -2.63;\ 0.26]$;
- Supervised, bimodal noise,
  $\mathbf{w} = [1.98;\ -5.24;\ 1.98;\ -5.24;\ -2.99;\ 0.22]$.

Our learning method avoids the computational bottlenecks of most probabilistic approaches such as maximum likelihood and pseudo-likelihood, which need the estimation of the normalization function Z. The main reason for unsupervised learning to not do as well for MAP problems as for graph matching is the different structure of the matrix $\mathbf{M}$. In the case of graph matching this matrix contains a single strong cluster formed mainly by the correct assignments, while in the case of MAP problems, the matrix could contain several such clusters corresponding to completely different labelings. The idea of accidental alignment is not applicable to most MAP problems, thus the learning algorithm could converge to several parameter vectors that would binarize the continuous solution, in which case supervised learning is required. Moreover, even in the case of supervised learning, training is sensitive to initialization in the case of MAP problems, a fact also observed by other researchers.

7 Conclusions

We present an efficient way of performing both supervised and unsupervised learning for graph matching in the context of computer vision applications. We show that the performance of our unsupervised learning algorithm is comparable with that of the supervised case. The proposed algorithm significantly improves the performance of several state-of-the-art graph matching algorithms, which makes it widely applicable. We also present a new approach to learning for MAP problems and demonstrate that for some problems, such as image denoising, it is possible to perform successful learning in a completely unsupervised manner. As future work we plan to extend the learning algorithms presented here to the case of matching and MAP inference beyond pairwise to higher-order constraints.

Acknowledgements This work was supported in part by the National Science Foundation under Grant IIS0713406; by an Intel Graduate Fellowship; by the 21st Century Frontier R&D Program of the Korean Ministry of Commerce, Industry, and Energy; and by CNCSIS-UEFISCSU, project number PN II-RU-RC-2/2009. Any opinions, findings, and conclusions or recommendations expressed in this material are solely those of the authors.

References

Bach, F. R., & Jordan, M. I. (2003). Learning spectral clustering. In *Neural Information Processing Systems*.

Baratchart, L., Berthod, M., & Pottier, L. (1998). Optimization of positive generalized polynomials under l^p constraints. *Journal of Convex Analysis*, *5*(2), 353–379.

Belongie, S., Malik, J., & Puzicha, J. (2002). Shape matching and object recognition using shape context. *Pattern Analysis and. IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(4), 509–522.

Berg, A., Berg, T., & Malik, J. (2005). Shape matching and object recognition using low distortion correspondences. In *International Conference on Computer Vision and Pattern Recognition*.

Besag, J. (1986). On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society*, *48*(5), 259–302.

Brendel, W., & Todorovic, S. (2010). Segmentation as maximum-weight independent set. In *Neural Information Processing Systems*.

Caetano, T., Cheng, L., Le, Q. V., & Smola, A. J. (2007). Learning graph matching. In *International Conference on Computer Vision*.

Caetano, T., McAuley, J. J., Cheng, L., Le, Q. V., & Smola, A. J. (2009). Learning graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(6), 1048–1058.

Carcassoni, M., & Hancock, E. R. (2002). Alignment using spectral clusters. In *British Machine Vision Conference*.

Chertok, M., & Keller, Y. (2010). Spectral symmetry analysis. *Pattern Analysis and. IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(7), 1227–1238.

Cour, T., & Shi, J. (2007). Solving Markov random fields with spectral relaxation. In *International Conference on Artificial Intelligence and Statistics*.

Cour, T., Shi, J., & Gogin, N. (2005). Learning spectral graph segmentation. In *International Conference on Artificial Intelligence and Statistics*.

Cour, T., Srinivasan, P., & Shi, J. (2006). Balanced graph matching. In *Neural Information Processing Systems*.

de Aguiar, E., Stoll, C., Theobalt, C., Ahmed, N., Seidel, H.-P., & Thrun, S. (2008). Performance capture from sparse multi-view video. In *SIGGRAPH*.

Duchenne, O., Bach, F., Kweon, I., & Ponce, J. (2009). A tensor-based algorithm for high-order graph matching. In *International Conference on Computer Vision and Pattern Recognition*.

Frank, M., & Wolfe, P. (1956). An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, *3*(1–2), 95–110.

Gold, S., & Rangarajan, A. (1996). A graduated assignment algorithm for graph matching. *Pattern Analysis and. IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(4), 377–388.

Hays, J., Leordeanu, M., Efros, A., & Liu, Y. (2006). Discovering texture regularity as a higher-order correspondence problem. In *European Conference on Computer Vision*.

Huhle, B., Jenke, P., & Straßer, W. (2008). On-the-fly scene acquisition with a handy multi-sensor system. *International Journal of Intelligent Systems Technologies and Applications*, *5*(3–4), 255–263.

Kim, G., Faloutsos, C., & Hebert, M. (2008). Unsupervised modeling and recognition of object categories with combination of visual contents and geometric similarity links. In *ACM International Conference on Multimedia Information Retrieval*.

Kim, G., Faloutsos, C., & Hebert, M. (2008). Unsupervised modeling of object categories using link analysis techniques. In *International Conference on Computer Vision and Pattern Recognition*.

Kumar, S. Models for learning spatial interactions in natural images for context-based classification. PhD thesis, The Robotics Institute, Carnegie Mellon University, 2005.

Leordeanu, M., Collins, R., & Hebert, M. (2005). Unsupervised learning of object features from video sequences. In *International Conference on Computer Vision and Pattern Recognition*.

Leordeanu, M., & Hebert, M. (2005). A spectral technique for correspondence problems using pairwise constraints. In *International Conference on Computer Vision*.

Leordeanu, M., & Hebert, M. (2006). Efficient MAP approximation for dense energy functions. In *International Conference on Machine Learning*.

Leordeanu, M., & Hebert, M. (2008). Smoothing-based optimization. In *International Conference on Computer Vision and Pattern Recognition*.

Leordeanu, M., & Hebert, M. (2009). Unsupervised learning for graph matching. In *International Conference on Computer Vision and Pattern Recognition*.

Leordeanu, M., Hebert, M., & Sukthankar, R. (2007). Beyond local appearance: Category recognition from pairwise interactions of simple features. In *International Conference on Computer Vision and Pattern Recognition*.

Leordeanu, M., Hebert, M., & Sukthankar, R. (2009). An integer projected fixed point method for graph matching and MAP inference. In *Neural Information Processing Systems*.

Liu, J., Luo, J., & Shah, M. (2009). Recognizing realistic actions from videos "in the wild. In *International Conference on Computer Vision and Pattern Recognition*.

Magnus, J., & Neudecker, H. (1999). *Matrix Differential Calculus with Applications in Statistics and Econometrics*. New York: Wiley.

Parikh, D., & Chen, T. (2007). Unsupervised identification of multiple objects of interest from multiple images: dISCOVER. In *Asian Conference on Computer Vision*.

Parikh, D., & Chen, T. (2007). Unsupervised learning of hierarchical semantics of objects (hSOs). In *International Conference on Computer Vision and Pattern Recognition*.

Ravikumar, P., & Lafferty, J. (2006). Quadratic programming relaxations for metric labeling and Markov random field MAP estimation. In *International Conference on Machine Learning*.

Ren, X. (2007). Learning and matching line aspects for articulated objects. In *International Conference on Computer Vision and Pattern Recognition*.

Schellewald, C., & Schnorr, C. (2005). Probabilistic subgraph matching based on convex relaxation. In *International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition*.

Semenovich, D. (2010). Tensor power method for efficient MAP inference in higher-order MRFs. In *International Conference on Pattern Recognition*.

Szummer, M., Kohli, P., & Hoiem, D. (2008). Learning CRFs using graph cuts. In *European Conference on Computer Vision*.

Taskar, B., Guestrin, C., & Koller, D. (2004). Max-margin Markov networks. In *Neural Information Processing Systems*.

Umeyama, S. (1988). An eigen decomposition approach to weighted graph matching problems. In *Pattern Analysis and Machine Intelligence*.

Yan, P., Khan, S. M., & Shah, M. (2008). Learning 4D action feature models for arbitrary view action recognition. In *International Conference on Computer Vision and Pattern Recognition*.

Zaslavskiy, M. Graph matching and its application in computer vision and bioinformatics. PhD thesis, l'Ecole nationale superieure des mines de Paris, 2010.

Zaslavskiy, M., Bach, F., & Vert, J.-P. (2009). A path following algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(12), 2227–2242.

Zass, R., & Shashua, A. (2008). Probabilistic graph and hypergraph matching. In *International Conference on Computer Vision and Pattern Recognition*.