Computing Gaussian Mixture Models with EM using Side-Information

Noam Shental FENOAM@CS.HUJI.AC.IL
Aharon Bar-Hillel AHARONBH@CS.HUJI.AC.IL
Tomer Hertz TOMBOY@CS.HUJI.AC.IL
Daphna Weinshall DAPHNA@CS.HUJI.AC.IL
School of Computer Science and Engineering and the Center for Neural Computation, The Hebrew University of Jerusalem, 91904 Jerusalem, ISRAEL

Abstract

Estimation of Gaussian mixture models is an efficient and popular technique for clustering and density estimation. An EM procedure is widely used to estimate the model parameters. In this paper we show how side information in the form of *equivalence constraints* can be incorporated into this procedure, leading to improved clustering results. *Equivalence constraints* are prior knowledge concerning pairs of data points, indicating if the points arise from the same source (positive constraint) or from different sources (negative constraint). Such constraints can be gathered automatically in some learning problems, and are a natural form of supervision in others. We present a closed form EM procedure for handling positive constraints, and a Generalized EM procedure using a Markov net for the incorporation of negative constraints. Using publicly available data sets we demonstrate that such side information may lead to considerable improvement in clustering tasks, and that our algorithm is preferable to another suggested method using this type of side information.

Keywords: Learning from partial knowledge, semi-supervised learning, Gaussian mixture models, clustering.

1. Introduction

We are used to thinking about learning from labels as supervised learning, and learning without labels as unsupervised learning, where 'supervised' implies a need for human intervention. However, in unsupervised learning we are not limited to using data statistics only. Similarly supervised learning is not limited to using labels. In this work we focus on semi-supervised learning using side-information, which is *not* given as labels. More specifically, we use *equivalence constraints* between pairs of data points, which determine whether each pair was generated by the same source, or by different sources. Such constraints may be acquired without human intervention in a broad class of problems, and are a natural form of supervision in other scenarios. In this paper we show how to incorporate *equivalence constraints* into the EM algorithm (Dempster et al., 1977), in order to compute a generative Gaussian mixture model of the data.

Equivalence constraints are binary functions of pairs of points, indicating whether the two points come from the same source or from two different sources. We denote the first case as "is-equivalent" constraints, and the second as "not-equivalent" constraints. As it turns out, "is-equivalent" constraints can be easily incorporated into EM, while "not-equivalent" constraints require heavy duty inference machinery such as Markov networks. We describe the derivations in Section 2.

The underlying motivation of our approach is the observation that the partition extracted by the EM algorithm may not reflect the actual structure in the data. Thus side information should ideally lead to solutions which are more faithful to the desired results. A simple example demonstrating this point is shown in Fig. 1. Side information can also be used to improve the convergence characteristics of the EM algorithm.

Our work is also motivated by the relative abundance of *equivalence constraints* in real life applications. In a broad family of applications, *equivalence constraints* can be obtained without supervision. Maybe the most important source of unsupervised *equivalence constraints* is temporal continuity in data; for example, in video indexing a sequence of faces obtained from successive frames in roughly the same location are likely to contain the same *unknown* individual. Furthermore, there are several learning applications in which *equivalence constraints* are the natural form of supervision.

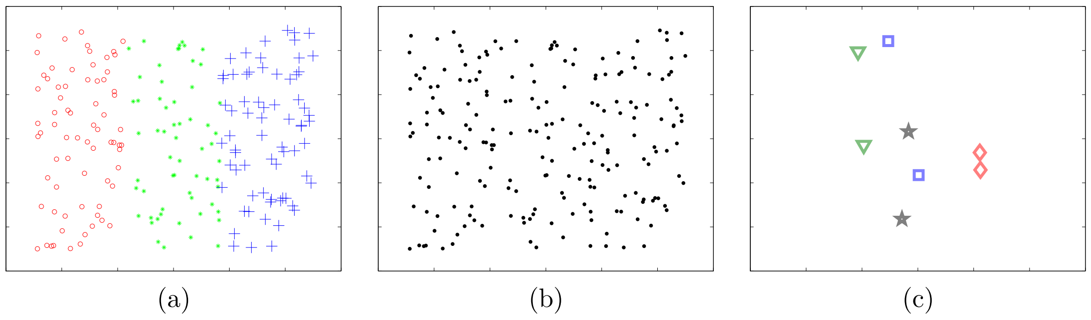

Figure 1. An illustrative example of the importance of *equivalence constraints*: (a) data set with 3 classes; (b) the same data set unlabeled. (c) additional side information in the form of "is-equivalent" constraints: each pair of points (such as the stars and the squares) are known to belong to the same class. The required partition is vertical, which is not evident from the raw data density alone in (b). Hence EM cannot be expected to find the relevant clusters, as the likelihood of finding horizontal clusters is similar to that of finding vertical ones. However, the pairs in (c) change the data probability function, and now vertical partitions are more likely to be found by the EM.

One such scenario occurs when we wish to enhance a retrieval engine using supervision provided by its users. The users may be asked to help annotate the retrieved set of data points, in what may be viewed as 'generalized relevance feedback'. The categories given by the users have subjective names that may be inconsistent. Therefore, we can only extract *equivalence constraints* from the feedback provided by the users. In another scenario we wish to learn from several teachers who do not know each other and are not able to coordinate among themselves the use of common labels. This can be regarded as a 'distributed learning' scenario. This is the typical scenario when the database contains many classes, which may not have a conventional name, *e.g.*, a large facial image database. As in the former scenario, the information obtained from the teachers is in the form of *equivalence constraints*.

When *equivalence constraints* are obtained in a supervised manner, our method can be viewed as a semi-supervised learning technique. Most of the papers in the field of semi-supervised learning consider the case of partial labels in which a large unlabeled data set is augmented by a much smaller labeled data set (Miller & Uyar, 1997; Szummer & Jaakkola, 2001; Nigam et al., 1998). Specifically, in Miller and Uyar (1997) labels are introduced into an EM formulation of a Gaussian mixture model, leading to improvement of clustering results. Nigam et al. (1998) presents an algorithm for text classification from labeled and unlabeled documents using EM.

There are several papers that use side information in the form of *equivalence constraints*. The *Relevant Component Analysis* algorithm (RCA) presented by Shental et al. (2002) uses *equivalence constraints* to learn a distance metric over the input space. RCA has been shown to enhance clustering performance, and can also be used in our EM derivations. Similar work is presented by Xing et al. (2002). Phillips (1998) represented pairs of data points by their vector difference, and multi-class classification in the original space is mapped to a binary classification in the difference space; the main drawback of this approach is the mapping which is ill-defined.

Wagstaff et al. (2001) introduced *equivalence constraints* into the K-means clustering algorithm. The algorithm is closely related to our work since it allows incorporating both "is-equivalent" and "not-equivalent" constraints. The algorithm partitions the data in the following heuristic manner: In every K-means iteration each point is assigned to the model which is closest to it and complies with the constraints on all of the previously assigned points. If no such model is found the algorithm fails to find a clustering solution.

By introducing the constraints into the EM algorithm we gain significantly better clustering results. One reason may be that the EM of a Gaussian mixture model is preferable to K-means as a clustering algorithm. More importantly, the probabilistic semantics of the EM procedure allows introducing constraints in a principled way, thus overcoming many drawbacks of the heuristic approach. To support this claim, in Section 3 we present comparative results, demonstrating the very significant advantage of our constrained EM algorithm over the constrained K-means algorithm using a number of data sets from the UCI repository and a large database of facial images (Georghiades et al., 2000).

2. Constrained EM: the update rules

A Gaussian mixture model (GMM) is a parametric statistical model which assumes that the data originates from a weighted sum of several Gaussian sources. More formally, a GMM is given by:

$$p(x|\Theta) = \Sigma_{l=1}^{M} \alpha_l p(x|\theta_l),$$

where α_l denotes the weight of each Gaussian, θ_l its respective parameters, and M denotes the number of Gaussian sources in the GMM. EM is a widely used method for estimating the parameter set of the model (Θ) using unlabeled data (Dempster et al., 1977). The algorithm iterates between two steps:

- 'E' step: calculate the expectation of the log-likelihood over all possible assignments of data points to sources.
- 'M' step: maximize the expectation by differentiating w.r.t current parameters.

Equivalence constraints modify the 'E' step in the following way: instead of summing over *all* possible assignments of data points to sources, we sum only over assignments which comply with the given constraints. For example, if points x_i and x_j form an "is-equivalent" constraint, we only consider assignments in which both points are assigned to the *same* Gaussian source. On the other hand, if these points form a "not-equivalent" constraint, we only consider assignments in which each of the points is assigned to a *different* Gaussian source.

However, there is a basic difference between "is-equivalent" (positive) and "not-equivalent" (negative) constraints: While positive constraints are transitive (i.e. a group of pairwise "is-equivalent" constraints can be merged using a transitive closure), negative constraints are not transitive. The outcome of this difference is expressed in the complexity of incorporating each type of constraint into the EM formulation. Therefore, we begin by presenting a formulation for positive constraints (Section 2.1), and then present a different formulation for negative constraints (Section 2.2). We conclude by presenting a unified formulation for both types of constraints (Section 2.3).

2.1. Incorporating positive constraints

Let a *chunklet* denote a small subset of data points that are known to belong to a single unknown class. Chunklets may be obtained by applying the transitive closure to the set of "is-equivalent" constraints.

In this settings we are given a set of unlabeled data points, and a set of chunklets. In order to write down the likelihood of a given assignment of points to sources, a probabilistic model of how chunklets are obtained must be specified. We consider two such models:

1. Chunklets are sampled i.i.d, with respect to the weight of their corresponding source (points within each chunklet are also sampled i.i.d)
2. Data points are sampled i.i.d, without any knowledge about their class membership, and only afterward chunklets are selected from these points.

The first assumption may be appropriate when chunklets are automatically obtained using temporal continuity. The second sampling assumption is appropriate when *equivalence constraints* are obtained using *distributed learning*. When incorporating these sampling assumptions into the EM formulation for a GMM, different algorithms are obtained: Using the first assumption we obtain closed-form update rules for all of the GMM parameters. When the second sampling assumption is used there is no closed-form solution for the sources' weights. In this section we therefore derive the update rules under the first sampling assumption, and at the end of the section we shortly discuss the second sampling assumption.

More specifically, let $p(x) = \sum_{l=1}^{M} \alpha_l \; p_l(x|\theta_l)$ denote our GMM. Each $p_l(x|\theta_l)$ term is a Gaussian parameterized by $\theta_l = (\mu_l, \Sigma_l)$ with a mixing coefficient α_l. Let $\mathbf{X}$ denote the set of all data points, $\mathbf{X} = \{x_i\}_{i=1}^{N}$. Let $\{X_j\}_{j=1}^{L}$, $L \leq N$ denote the distinct chunklets, where each X_j is a set of points x_i such that $\bigcup_{j=1}^{L} X_j = \{x_i\}_{i=1}^{N}$ (unconstrained data points appear as chunklets of size one). Let $\mathbf{Y} = \{y_i\}_{i=1}^{N}$ denote the source assignment of the respective data-points, and $Y_j = \{y_j^1 \dots y_j^{|X_j|}\}$ denote the source assignment of the chunklet X_j. Finally, let E_Ω denote the event $\{\mathbf{Y}$ complies with the constraints$\}$.

The expectation of the log likelihood is the following:

$$E[log(p(\mathbf{X}, \mathbf{Y}|\Theta^{new}, E_\Omega))|\mathbf{X}\,\Theta^{old}, E_\Omega] \qquad (1)$$

$$= \sum_{\mathbf{Y}} log(p(\mathbf{X}, \mathbf{Y}|\Theta^{new}, E_\Omega)) \cdot p(\mathbf{Y}|\mathbf{X}, \Theta^{old}, E_\Omega)$$

where $\sum_{\mathbf{Y}}$ stands for a summation over all assignments of points to sources: $\sum_{\mathbf{Y}} \equiv \sum_{y_1=1}^{M} \cdots \sum_{y_N=1}^{M}$. In the following discussion we shall also reorder the sum according to chunklets: $\sum_{\mathbf{Y}} \equiv \sum_{Y_1} \cdots \sum_{Y_L}$, where $\sum_{Y_j}$ stands for $\sum_{y_1^j} \cdots \sum_{y_{|X_j|}^j}$.

First, using Bayes rule and the independence of chunklets, we can write

$$p(\mathbf{Y}|\mathbf{X}, \Theta^{old}, E_\Omega) = \frac{p(E_\Omega|\mathbf{Y}, \mathbf{X}, \Theta^{old})\ p(\mathbf{Y}|\mathbf{X}, \Theta^{old})}{\sum_{\mathbf{Y}} p(E_\Omega|\mathbf{Y}, \mathbf{X}, \Theta^{old})\ p(\mathbf{Y}|\mathbf{X}, \Theta^{old})}$$
$$= \frac{\prod_{j=1}^{L} \delta_{Y_j}\ p(Y_j|X_j, \Theta^{old})}{\sum_{Y_1} \cdots \sum_{Y_L} \prod_{j=1}^{L} \delta_{Y_j}\ p(Y_j|X_j, \Theta^{old})} \quad (2)$$

where $\delta_{Y_j} \equiv \delta_{y_1^j,\ldots,y_{|X_j|}^j}$ equals 1 if all the points in chunklet i have the same source, and 0 otherwise.

Next, using chunklet independence and the independence of points within a chunklet we see that

$$p(\mathbf{X}, \mathbf{Y}|\Theta^{new}, E_\Omega) = p(\mathbf{Y}|\Theta^{new}, E_\Omega)\ p(\mathbf{X}|\mathbf{Y}, \Theta^{new}, E_\Omega)$$
$$= \prod_{j=1}^{L} \alpha_{y_j} \prod_{i=1}^{N} p(x_i|y_i, \Theta^{new})$$

Hence the log-likelihood is:

$$log\ p(\mathbf{X}, \mathbf{Y}|\Theta^{new}, E_\Omega) \quad (3)$$
$$= \sum_{j=1}^{L} \sum_{x_i \in X_j} log\ p(x_i|y_i, \Theta^{new}) + \sum_{j=1}^{L} log(\alpha_{y_j})$$

Finally, we substitute (3) and (2) into (1); after some manipulations, we obtain the following expression:

$$E(LogLikelihood) \quad (4)$$
$$= \sum_{l=1}^{M} \sum_{j=1}^{L} \sum_{x_i \in X_j} log\ p(x_i|l, \Theta^{new}) \cdot p(Y_j = l|X_j, \Theta^{old})$$
$$+ \sum_{l=1}^{M} \sum_{j=1}^{L} log\ \alpha_l \cdot p(Y_j = l|X_j, \Theta^{old})$$

where the chunklet posterior probability is:

$$p(Y_j = l|X_j, \Theta^{old}) = \frac{\alpha_l^{old} \prod_{x_i \in X_j} p(x_i|y_i^j = l, \Theta^{old})}{\sum_{m=1}^{M} \alpha_m^{old} \prod_{x_i \in X_j} p(x_i|y_i^j = m, \Theta^{old})}$$

To find the update rule for each parameter, we differentiate (4) with respect to μ_l, Σ_l and α_l. We get the following rules:

$$\alpha_l^{new} = \frac{1}{L} \sum_{j=1}^{L} p(Y_j = l|X_j, \Theta^{old})$$

$$\mu_l^{new} = \frac{\sum_{j=1}^{L} \bar{X}_j p(Y_j = l|X_j, \Theta^{old})|X_j|}{\sum_{j=1}^{L} p(Y_j = l|X_j, \Theta^{old})|X_j|}$$

$$\Sigma_l^{new} = \frac{\sum_{j=1}^{L} \Sigma_{jl}^{new} p(Y_j = l|X_j, \Theta^{old})|X_j|}{\sum_{j=1}^{L} p(Y_j = l|X_j, \Theta^{old})|X_j|}$$

$$\text{for} \quad \Sigma_{jl}^{new} = \frac{\sum_{x_i \in X_j} (x_i - \mu_l^{new})(x_i - \mu_l^{new})^T}{|X_j|}$$

where $\bar{X}_j$ denotes the sample mean of the points in chunklet j, $|X_j|$ denotes the number of points in chunklet j, and Σ_{jl}^{new} denotes the sample covariance matrix of the jth chunklet of the lth class.

As can be readily seen, the update rules above effectively treat each chunklet as a single data point weighted according to the number of elements in it.

Let us consider briefly the second sampling assumption. It can be shown that Eq. (4) becomes:

$$\sum_{l=1}^{M} \sum_{j=1}^{L} \sum_{x_i \in X_j} log\ p(x_i|l, \Theta^{new}) \cdot p(Y_j = l|X_j, \Theta^{old}) + \quad (5)$$
$$\sum_{l=1}^{M} \sum_{j=1}^{L} |X_j| log\ \alpha_l \cdot p(Y_j = l|X_j, \Theta^{old}) - \sum_{j=1}^{L} log(\sum_{m=1}^{M} \alpha_m^{|X_j|})$$

The difference lies in the last term, which can be interpreted as a "normalization" term. It readily follows that the update rule for μ_l and Σ_l remain the same, but with

$$p(Y_j = l|X_j, \Theta^{old})$$
$$= \frac{(\alpha_l^{old})^{|X_j|} \prod_{x_i \in X_j} p(x_i|y_i^j = l, \Theta^{old})}{\sum_{m=1}^{M} (\alpha_m^{old})^{|X_j|} \prod_{x_i \in X_j} p(x_i|y_i^j = m, \Theta^{old})}$$

The major difficulty lies in the calculation of the sources' weights, α_l. In order to calculate α_l^{new}, we need to differentiate (5) under the constraint $\sum_{l=1}^{M} \alpha_l = 1$. Due to the "normalization" term one cannot derive a closed solution from (5), and as a result we must use a Generalized EM (GEM) scheme where the maximum is obtained numerically.

2.2. Incorporating negative constraints

The probabilistic description of a data set using a GMM attaches to each data point two random variables: an observable and a hidden. The hidden variable of a point describes its source label, while the data point itself is an observed example from the source. Each pair of observable and hidden variables is assumed to be independent of the other pairs. However, negative *equivalence constraints* violate this assumption, as dependencies between the hidden variables are introduced.

Specifically, assume we have a group $\Omega = \{(a_i^1, a_i^2)\}_{i=1}^{P}$ of index pairs corresponding to P pairs of points that are negatively constrained, and define the event $E_\Omega = \{\mathbf{Y} \text{ complies with the constraints}\}$. Now

$$\begin{aligned} p(\mathbf{X},\mathbf{Y}|\Theta,E_\Omega) &= p(\mathbf{X}|\mathbf{Y},\Theta,E_\Omega)\ p(\mathbf{Y}|\Theta,E_\Omega) \\ &= \frac{p(\mathbf{X}|\mathbf{Y},\Theta)\ p(E_\Omega|\mathbf{Y})\ p(\mathbf{Y}|\Theta)}{p(E_\Omega|\Theta)} \end{aligned}$$

Let Z denote the constant $p(E_\Omega|\Theta)$. Assuming sample independence, it follows that $p(\mathbf{X}|\mathbf{Y},\Theta)\cdot p(\mathbf{Y}|\Theta) = \prod_{i=1}^N p(y_i|\Theta)p(x_i|y_i,\Theta)$. By definition $p(E_\Omega|\mathbf{Y}) = 1_{\mathbf{Y}\in E_\Omega}$, hence

$$p(\mathbf{X},\mathbf{Y}|\Theta,E_\Omega) = \frac{1}{Z}1_{\mathbf{Y}\in E_\Omega}\prod_{i=1}^N p(y_i|\Theta)p(x_i|y_i,\Theta) \quad (6)$$

Expanding $1_{\mathbf{Y}\in E_\Omega}$ gives the following expression

$$\begin{aligned} &p(\mathbf{X},\mathbf{Y}|\Theta,E_\Omega) \quad (7) \\ &= \frac{1}{Z}\prod_{(a_i^1,a_i^2)}(1-\delta_{y_{a_i^1},y_{a_i^2}})\prod_{i=1}^N p(y_i|\Theta)p(x_i|y_i,\Theta) \end{aligned}$$

As a product of local components, the distribution in (7) can be readily described using a Markov network. The network nodes are the hidden source variables and the observable data point variables. The potential $p(x_i|y_i,\Theta)$ connects each observable data point, in a Gaussian manner, to a hidden variable corresponding to the label of its source. Each hidden source node holds an initial potential of $p(y_i|\Theta)$ reflecting the prior of the cluster weights. Negative constraints are expressed by edges between hidden variables which prevent them from having the same value. A potential over an edge $(a_i^1 - a_i^2)$ is expressed by $1-\delta_{y_{a_i^1},y_{a_i^2}}$ (see Fig. 2).

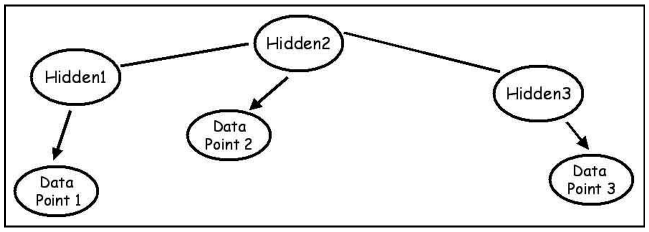

Figure 2. An illustration of the Markov network required for incorporating "not-equivalent" constraints. Data points 1 and 2 have a negative constraint, and so do points 2 and 3.

We derived an EM procedure which maximizes $log(p(\mathbf{X}|\Theta,E_\Omega))$ entailed by this distribution. The update rules for μ_l and Σ_l are still

$$\mu_l^{new} = \frac{\sum_{i=1}^N x_i p(y_i=l|\mathbf{X},\Theta^{old},E_\Omega)}{\sum_{i=1}^N p(y_i=l|\mathbf{X},\Theta^{old},E_\Omega)}$$

$$\Sigma_l^{new} = \frac{\sum_{i=1}^N \widehat{\Sigma_i l} p(y_i=l|\mathbf{X},\Theta^{old},E_\Omega)}{\sum_{i=1}^N p(y_i=l|\mathbf{X},\Theta^{old},E_\Omega)}$$

where $\widehat{\Sigma_i l} = (x_i-\mu_l^{new})(x_i-\mu_l^{new})^T$ denotes the sample covariance matrix. Note, however, that now the vector of probabilities $p(y_i=l|\mathbf{X},\Theta^{old},E_\Omega)$ is inferred using the net.

The update rule of $\alpha_l = p(y_i = l|\Theta_{new},E_\Omega)$ is more intricate, since this parameter appears in the normalization factor Z in the likelihood expression (6):

$$\begin{aligned} Z = p(E_\Omega|\Theta) &= \sum_{\mathbf{Y}} p(\mathbf{Y}|\Theta)p(E_\Omega|\mathbf{Y}) \quad (8) \\ &= \sum_{y_1}\cdots\sum_{y_N}\prod_{i=1}^N \alpha_{y_i}\prod_{(a_i^1,a_i^2)}(1-\delta_{y_{a_i^1},y_{a_i^2}}) \end{aligned}$$

This factor can be calculated using a net which is similar to the one discussed above but lacks the observable nodes. We use such a net to calculate Z and differentiate it w.r.t α_l, after which we perform gradient ascent. Alternatively, we can approximate Z by assuming that the pairs of negatively constrained points are disjoint. Using such an assumption, Z is reduced to the relatively simple expression: $Z = (1-\sum_{i=1}^M \alpha_i^2)^P$. This expression for Z may be easily differentiated, and can be used in the Generalized EM scheme. Although the assumption is not valid in most cases, it is a reasonable approximation in sparse networks, and our empirical tests show that it gives good results.

2.3. Combining positive and negative constraints

Both kinds of constraints can be combined in a single Markov network by a rather simple extension of the network described in the previous section. Assume we have, in addition to the negative constraints, a set $\{c_i\}$ of chunklets, where each c_i is an index set of points known to share the same label. Now the detailed form of Eq. (7) becomes

$$\begin{aligned} p(\mathbf{X},\mathbf{Y}|\Theta,E_\Omega) &= \frac{1}{Z}\prod_{c_i}\delta_{y_{c_i}}\prod_{(a_i^1,a_i^2)}(1-\delta_{y_{a_i^1},y_{a_i^2}}) \\ &\cdot\prod_{i=1}^N p(y_i|\Theta)p(x_i|y_i,\Theta) \end{aligned}$$

where $\delta_{y_{c_i}}$ is 1 iff all the points in chunklet c_i have the same label. Since the probability is positive only when the hidden variables of the chunklet points have identical values, we can replace those variables with a single variable which states the source label of the chunklet. Doing so for all the data points, we get a smaller set of hidden chunklet variables $\mathbf{H} = \{h_i\}_{i=1}^{L}$. The original negative constraints over $\mathbf{Y}$ are transformed into a set of negative constraints over $\mathbf{H}$, which we denote as $\hat{\Omega} = \{(\hat{a_i^1}, \hat{a_i^2})\}_{i=1}^{P}$. The probability can now be written as

$$p(\mathbf{X}, \mathbf{H}|\Theta, E_{\hat{\Omega}}) = \frac{1}{\hat{Z}} \prod_{(\hat{a}_i^1, \hat{a}_i^2)} (1 - \delta_{h_{\hat{a}_i^1}, h_{\hat{a}_i^2}}) \prod_{i=1}^{L} \alpha_{h_i}^{|h_i|} \cdot \prod_{i=1}^{N} p(x_i | h_{f(i)}, \Theta)$$

where $|h_i|$ denotes the number of data points mapped to h_i.

The distribution above can be expressed by a Markov network similar to the network from the previous section, where every pair of data points related by a positive constraint share a hidden father node (see Fig. 3).

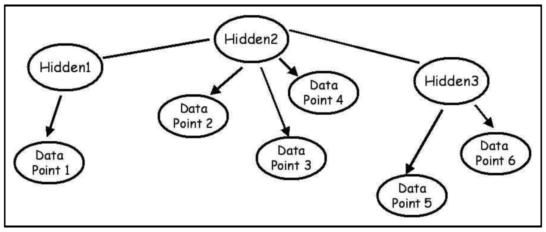

Figure 3. An illustration of the Markov network required for incorporating "not-equivalent" and "is-equivalent" constraints. Data points $2, 3, 4$ are positively constrained, and so are points $5, 6$. Data points $2, 3, 4$ have a negative constraint with point 1 and with points $5, 6$.

The EM procedure derived from this distribution is similar to the one presented above, with some minor changes in the normalizing constant Z. The incorporation of positive constraints in this network is in accordance with the generic sampling assumption described in the end of Section 2.1, as the data points are assumed to be sampled i.i.d before the introduction of the constraints.

The complexity of each EM round is dominated by the inference complexity, which is $O(M^{induced\ width(G)})$ by using the Jtree algorithm (Pearl, 1988). Hence, in practice, the algorithm is limited to $O(N)$ "not-equivalent" constraints. The general case with $O(N^2)$ constraints is NP-hard, as the graph coloring problem can be reduced to it. To achieve scalability to large sets of constraints two approximations are used: the graph is replaced by a spanning tree, and the normalization factor Z is approximated.

3. Experimental results

In order to evaluate the performance of our EM derivations and compare it to the performance of the constrained K-means algorithm presented by Wagstaff et al. (2001), we tested our algorithms using several data sets from the UCI repository. We simulated a 'distributed learning' scenario in order to obtain side information. In this scenario, we obtain *equivalence constraints* using the help of N teachers. Each teacher is given a random selection of K data points from the data set, and is then asked to partition this set of points into equivalence classes. The constraints provided by the teachers are gathered and used as *equivalence constraints.* We compared the performance of the following algorithms:

- K-means algorithm when no side information is used.
- Constrained K-means (Wagstaff et al., 2001), using only positive *equivalence constraints.*
- Constrained K-means (Wagstaff et al., 2001), using both positive and negative *equivalence constraints.*
- EM of a Gaussian mixture model when no side information is used.
- Our closed form EM algorithm when only positive *equivalence constraints* are used.
- Our Markov network EM algorithm, using both positive and negative *equivalence constraints.*

The number of constrained points was determined by the number of teachers N and the size of the subset K that they were each given. By controlling the product NK we modified the amount of side information provided to the different algorithms. Specifically, we experimented with two conditions: using "little" side information (approximately 15% of the data points are constrained) and using "much" side information (approximately 30% of the points are constrained).[1] All algorithms were given initial conditions that did not take into account the available *equivalence constraints.* The clustering obtained was evaluated using a combined measure of precision P and recall R scores. Specifically, the score we used was: $f_{\frac{1}{2}} = \frac{2PR}{R+P}$

The results over several UCI data sets are presented in Fig. 4. Several effects can clearly be seen:

[1]on two of the datasets presented we used more side-information, since the amounts described above showed little or no improvement.

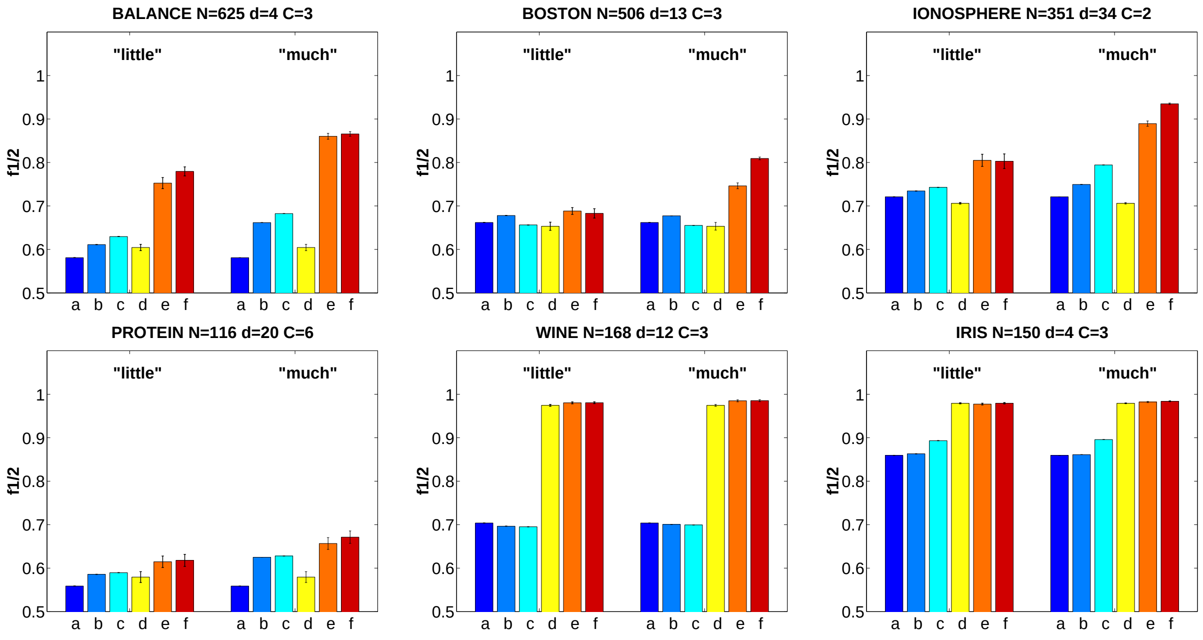

Figure 4. Combined precision and recall scores ($f_{\frac{1}{2}}$) of several clustering algorithms over 5 data sets from the UCI repository. Results are presented for the following algorithms: (a) K-means, (b) constrained K-means using only positive constraints, (c) constrained K-means using both positive and negative constraints, (d) regular EM, (e) EM using positive constraints, and (f) EM using both positive and negative constraints. Results are shown twice, using 15% of the data points in constraints (left bars) and 30% of the points constrained (right bars). The results were averaged over 100 realizations of constraints. Also shown are the names of the data sets used and some of their parameters: N - the size of the data set; C - the number of classes; d - the dimensionality of the data.

- As can be expected, the performance of the EM algorithms is generally better than the performance of the K-means algorithms. Specifically, our constrained EM outperforms the constrained K-means algorithm on all databases, and shows substantial improvements over the baseline EM as well.

- Introducing side information in the form of *equivalence constraints* clearly improves the results of both K-means and the EM algorithms. As the amount of side-information increases, the algorithms which make use of it tend to improve.

- Most of the improvement can be attributed to the positive constraints, and can be achieved using our closed form EM version. In most cases adding the negative constraints contributes a small but significant improvement over results obtained when using only positive constraints.

It should be noted that most of the UCI data sets considered so far contain two or three classes. Thus in the 'distributed learning' setting a relatively large fraction of the constraints gathered were positive. In a more realistic situation, when the number of classes is large, we are likely to gather much more negative constraints than positive constraints. This is an important point in light of the results in Fig. 4, where the major boost in performance was due to the use of positive constraints.

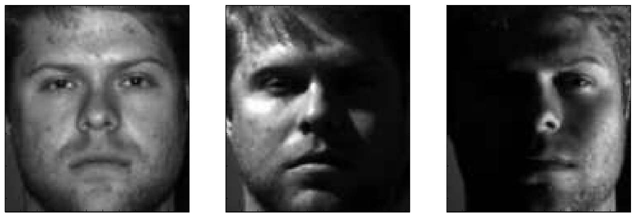

Figure 5. Three images of the same person from the YaleB data set.

In order to test the case of a data set with many classes we conducted the same experiment using a subset of the yaleB facial image dataset (Georghiades et al., 2000) which contains a total of 640 images, including 64 frontal pose images of 10 different subjects. In this database the variability between images of the same person is due mainly to different lighting conditions.

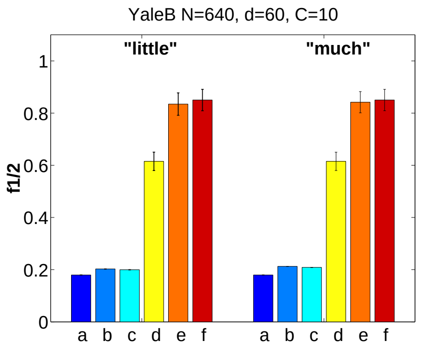

Figure 6. combined precision and recall scores of several clustering algorithms over the YaleB face data set. The results are presented using the same format as in Fig. 4. Percentage of data in constraints was 50% (left bars) and 75% (right bars). Results presented are averaged over more than 1000 realizations of constraints. It should be noted that when using 75% of the data in constraints the constrained K-means algorithm failed to converge in more than half of its runs.

We automatically centered all the images using optical flow. Images were then converted to vectors, and each image was represented using the first 60 PCA coefficients. The task was to cluster the facial images belonging to these 10 subjects.

Some example images from the data set are shown in Fig. 5. Due to the random selection of images given to each of the N teachers, most of the constraints obtained were negative constraints. Our results over this data set are summarized in Fig. 6. As can be seen, even though there was only a small number of positive constraints, our algorithms substantially outperformed the regular EM algorithm.

4. Discussion and Concluding Remarks

We have presented several variants of the EM algorithm which incorporate *equivalence constraints*. When using only positive constraints, we provided an efficient closed form solution for the update rules, and demonstrated that using positive constraints can significantly boost clustering performance. When negative constraints are added, the computational cost increases, a Markov network is used as an inference tool, and we must defer to approximations of the source weights update rules. Still, using negative constraints also boosts performance over results obtained by our positive constraints EM variant.

We have explored several other techniques of learning from *equivalence constraints* (Hertz et al., 2002; Shental et al., 2002). Specifically we presented the RCA algorithm which uses positive *equivalence constraints* to compute a Mahalanobis metric. We have shown, that RCA improves the performance of metric based clustering algorithms. In most cases when RCA is used in conjunction with the EM algorithms presented here, it further boosts their performance.

References

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *JRSSB*, *39*, 1–38.

Georghiades, A., Belhumeur, P., & Kriegman, D. (2000). From few to many: Generative models for recognition under variable pose and illumination. *IEEE international Conference on Automatic Face and Gesture Recognition*, 277–284.

Hertz, T., Shental, N., Bar-hillel, A., & Weinshall, D. (2002). Enhancing image and video retrieval: Learning via equivalence constraints. *http://www.cs.huji.ac.il/ daphna/*.

Miller, D., & Uyar, S. (1997). A mixture of experts classifier with learning based on both labelled and unlabelled data. *NIPS 9* (pp. 571–578). MIT Press.

Nigam, K., McCallum, A., Thrun, S., & Mitchell, T. (1998). Learning to classify text from labeled and unlabeled documents. *Proceedings of AAAI-98* (pp. 792–799). Madison, US: AAAI Press, Menlo Park, US.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference.* Morgan Kaufmann Publishers, Inc.

Phillips, P. (1998). Support vector machines applied to face recognition. *NIPS 11* (p. 803ff). MIT Press.

Shental, N., Hertz, T., Weinshall, D., & Pavel, M. (2002). Adjustment learning and relevant component analysis. *Computer Vision - ECCV 2002* (p. 776ff).

Szummer, M., & Jaakkola, T. (2001). Partially labeled classification with markov random walks. *NIPS*. The MIT Press.

Wagstaff, K., Cardie, C., Rogers, S., & Schroedl, S. (2001). Constrained K-means clustering with background knowledge. *Proc. 18th International Conf. on Machine Learning* (pp. 577–584). Morgan Kaufmann, San Francisco, CA.

Xing, E., Ng, A., Jordan, M., & Russell, S. (2002). Distance metric learnign with application to clustering with side-information. *Advances in Neural Information Processing Systems*. The MIT Press.