INTERNATIONAL COMPUTER SCIENCE INSTITUTE

1947 Center Street • Suite 600 • Berkeley, California 94704 • 1-510-642-4274 • FAX 1-510-643-7684

Mixture Models and the EM Algorithm for Object Recognition within Compositional Hierarchies Part 1: Recognition

Joachim Utans
utans@icsi.berkeley.edu

TR-93-004
January 1993

Abstract

We apply the *Expectation Maximization (EM)* algorithm to an assignment problem where in addition to binary assignment variables analog parameters must be estimated. As an example, we use the problem of part labelling in the context of model based object recognition where models are stored in from of a compositional hierarchy. This problem has been formulated previously as a graph matching problem and stated in terms of minimizing an objective function that a recurrent neural network solves [11, 12, 5, 8, 22]. Mjolsness [9, 10] has introduced a *stochastic visual grammar* as a model for this problem; there the matching problem arises from an index renumbering operation via a permutation matrix. The optimization problem w.r.t the match variables is difficult and Mean Field Annealing techniques are used to solve it. Here we propose to model the part labelling problem in terms of a mixture of distributions, each describing the parameters of a part. Under this model, the match variables correspond to the *a posteriori* estimates of the mixture coefficients. The parts in the input image are unlabelled, this problem can be stated as missing data problem and the EM algorithm can be used to recover the labels and estimate parameters. The resulting update equations are identical to the Elastic Net equations; however, the update dynamics differ.

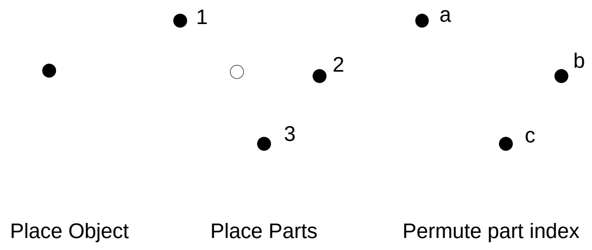

Figure 1: Illustration of the 2-level model for dot configurations.

1 Stochastic Visual Grammars for Object Recognition

We apply the *Expectation Maximization (EM)* algorithm to an assignment problem where in addition to binary assignment variables analog parameters must be estimated and show the close relationship to methods based on the Mean Field approximation.

The general form of this weighted match problem can be formulated as the minimization of an objective function $E(\mathbf{M}, \mathbf{p}) = \sum_{ij} m_{ij} w_{ij}(\mathbf{p})$, where the $\{m_{ij}\}$ are binary match variables and $\{w_{ij}(\mathbf{p})\}$ are weights dependent on parameters $\mathbf{p}$.

As an example, we use the problem of part labelling in the context of model based object recognition where models are stored in from of a compositional hierarchy. This problem has been formulated previously as a graph matching problem and stated in terms of minimizing an objective function that a recurrent neural network solves [11, 12, 5, 8, 22]. Mjolsness [9, 10] has introduced a *stochastic visual grammar* as a forward (generative) model that describes how an object is build up from parts. The description mirrors the representation in form of a compositional hierarchy; at each stage the description becomes more detailed as more parts are added. The stochastic model assigns a probability distribution at each stage of that process. Thus at each level of the hierarchy a more detailed description of parts in terms of their subparts is given by specifying a probability distribution for the coordinates of the subparts. The goal is to derive a joint probability distribution for an instance of an object and its parts as it appears in the scene. This gives the probability of observing such an object prior to the arrival of the data. The problem can then be stated as Bayesian inference problem [9]. The optimization problem w.r.t the match variables is difficult and Mean Field Annealing techniques are used to solve it.

1.1 Example: Stochastic Model for Dot Configurations

The model places the dot-cluster center $\mathbf{x}$ assuming a uniform distribution. While placing the parts (i.e. individual dots) Gaussian distributed noise with mean 0 and variance σ_i^2 is added to the position coordinates d_i to capture the notion of natural variation of the objects shape (see figure 1). The probability for the dot parameters is given by

$$P(\{\mathbf{x}_i\}|\mathbf{x}) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{2n} e^{-\frac{1}{2\sigma^2}\sum_i |\mathbf{x}_i - (\mathbf{x}+\mathbf{d}_i)|^2} \tag{1}$$

with $\{\mathbf{x}_i\}$ denoting the set of parts $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_n\}$, and for simplicity, the noise variance for all parts is assumed equal.

An important step in modeling the visual data the system receives as input is the unordering of parts, i.e. their identity, known up to this point, is lost. For the recognition or inference problem the dots observed in the image are unlabeled and the recovery of the labels is an intrinsic part of the recognition problem. Previous work by Mjolsness [10, 9] models the unordering of parts via a index permutation (renumbering). For example, let the parts in the model description be numbered $\{x_1, x_2, x_3\}$ and the parts observed in the image as $\{x_a, x_b, x_c\}$. Then, x_a could correspond to either x_1 or x_2 or x_3. The renumbering is accomplished via multiplication with a permutation matrix $\mathbf{M}$, a binary matrix for which $M_{ij} \in \{0, 1\}$, $\sum_i M_{ij} = 1$ and $\sum_j M_{ij} = 1$. This matrix permutes the indices of the parts. For the example above, $M_{1a} = 1$ would let x_1 appear in the image as x_a.

First a permutation matrix $\mathbf{M}$ is chosen with probability $P(\mathbf{M})$. The final joint probability distribution becomes [10, 9]

$$P(\mathbf{M}, \{\mathbf{x}_j\}, \mathbf{x}) = C\, P(\mathbf{M})\, e^{-\left(\frac{1}{2\sigma^2}\sum_i \sum_j M_{ij}|\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_i)|^2\right)} \tag{2}$$

where i indexes the unscrambled model parts and j indexes the parts observed in the image (note that the model parts $\{x_i\}$ have been integrated out). The recognition problem is stated in terms of finding the labelling (as represented by $\mathbf{M}$) and the object position $\mathbf{x}$. From Bayes theorem,

$$\begin{aligned} P(\mathbf{x}, \mathbf{M}|\{\mathbf{x}_j\}) &= \frac{P(\{\mathbf{x}_j\}|\mathbf{x}, \mathbf{M})P(\mathbf{x}, \mathbf{M})}{P(\{\mathbf{x}_j\})} \\ &\propto P(\{\mathbf{x}_j\}|\mathbf{x}, \mathbf{M})P(\mathbf{x}, \mathbf{M}) \\ &= P(\{\mathbf{x}_j\}, \mathbf{x}, \mathbf{M}) \end{aligned} \tag{3}$$

and recognition reduces to finding the most probable values for $\mathbf{x}$ and $\mathbf{M}$ given the data:

$$\underset{\mathbf{x},\mathbf{M}}{\operatorname{argmax}}\, P(\{\mathbf{x}_j\}, \mathbf{x}, \mathbf{M}) \tag{4}$$

Solving the inference problem involves finding the MAP estimate and is equivalent to a minimization problem, i.e. it is sufficient to minimize the exponent in equation (2) with respect to the unknown variables $\mathbf{M}$ and $\mathbf{x}$, subject to the constraint that $\mathbf{M}$ is a permutation matrix.

2 Neural Network based on the Mean Field Approximation

The maximization problem (4) has been implemented using a recurrent neural network to solve the optimization problem (see Mjolsness [9, 10]). The most straightforward implementation via a Hopfield network [6, 7] easily gets stuck in local minima of the objective function. Recently, results from statistical physics have been successfully applied to the problem of optimizing objective functions with neural networks [1, 15]. The *Mean Field Approximation* and deterministic simulated annealing allow the design of neural networks that can avoid spurious local minima. The governing probability distribution is assumed Gibbsian. The sum in the partition function Z runs over all possible configurations that are accessible to the system.

In the original formulation, constraints must be are added to the objective functions in form of penalty terms, that discourage certain configurations. For example, the match matrix is must be a permutation matrix for the solution to be interpretable in the context of object recognition.

However, in the neural network implementation as Hopfield match network nodes are represented by individual neurons. Thus, the minimization cannot be carried out just over the space of all permutation matrices, but the ensemble of neurons can represent any arbitrary matrix. Constraints must be added to enforce a solution that corresponds to a permutation matrix.

The Mean Field formulation allows for such constraints to be embedded in the network architecture and thus to be enforced exactly. This can be done by restricting the summation in the partition function to include only those configurations that obey the constraints. Additional penalty terms then are no longer necessary. Unfortunately, it is not possible to express all constraints in that way. The permutation matrix constraints, for example, can only be incorporated in part; either the row or the column constraint. The remaining constraint can be implemented via a penalty term (or is sometimes ignored since solutions that conform to one constraint but not the other are deemed of high cost and therefore most likely avoided by the network).

These methods have been described by Amit *et al* [2, 1], Peterson and Soederberg [14, 15], Simic [18, 17] and Yuille [23].

2.1 Saddle Point Approximation

The Mean Field trick employed here consists of rewriting the sum in the partition function as an integral and evaluating the integral at the saddlepoint, at which point, certain constraint terms can be evaluated [2, 1, 14, 15, 18, 17].

The probability distribution is of the form

$$P(\mathbf{M}, \mathbf{x}, \{\mathbf{x}_j\}) = \frac{1}{Z} e^{-\beta E(\mathbf{M}, \mathbf{x}, \{\mathbf{x}_j\})} \tag{5}$$

where $E(\mathbf{M}, \mathbf{x}, \{\mathbf{x}_j\})$ is the exponent from equation (2), $T = 1/\beta$ denotes the computational "temperature" and the partition function is

$$Z = \sum_{\{\mathbf{x}\},\{\mathbf{M}\}} e^{-\beta E(\mathbf{M}, \mathbf{x}, \{\mathbf{x}_j\})} \tag{6}$$

(the summation ranges over all states accessible to the system; strictly speaking, the summation over $\mathbf{x}$ should be replaced by an integral for real-valued parameters).

The effective energy after applying the saddle point approximation is

$$E_{\text{eff}}(\mathbf{M}, \mathbf{U}, \mathbf{x}, \{\mathbf{x}_j\}) \quad = \quad E(\mathbf{M}, \mathbf{x}, \{\mathbf{x}_j\}) - 1/\beta \sum_{ij} u_{ij} m_{ij} + 1/\beta \sum_i \log \sum_j e^{u_{ij}} \tag{7}$$

where u_{ij} are additional variables introduced by the approximation. The saddlepoint can be found by evaluating $\frac{\partial E_{\text{eff}}(\mathbf{M},\mathbf{U},\mathbf{x},\{\mathbf{x}_j\})}{\partial \mathbf{M}} = 0$ and $\frac{\partial E_{\text{eff}}(\mathbf{M},\mathbf{U},\mathbf{x},\{\mathbf{x}_j\})}{\partial \mathbf{U}} = 0$ and the fixed point equations for the u_{ij} and m_{ij} become

$$m_{ij} = \frac{e^{u_{ij}}}{\sum_j e^{u_{ij}}} \; . \tag{8}$$

with

$$u_{ij} \quad = \quad \beta \frac{\partial E(\mathbf{M}, \mathbf{x}, \{\mathbf{x}_j\})}{\partial m_{ij}} = -\beta \frac{1}{2\sigma^2} |\mathbf{x}_j - (\mathbf{x} + \mathbf{d}_i)|^2 \tag{9}$$

and $\mathbf{x}$ is found from

$$\frac{\partial E_{\text{eff}}(\mathbf{M},\mathbf{U},\mathbf{x},\{\mathbf{x}_j\})}{\partial \mathbf{x}} = -\sum_j \sum_i m_{ij} \frac{1}{\sigma^2} |\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_i)| = 0 \tag{10}$$

These equations are used to update $\mathbf{M}$, $\mathbf{U}$ and $\mathbf{x}$ concurrently and deterministic annealing is used by gradually lowering the temperature T.

2.2 Elastic Net

If the network contains both real-valued variables and binary match variables and the problem would be simplified if the match variables could be eliminated entirely (the term "Elastic Net" was introduced by Durbin and Wilshaw [4] to distinguish their formulation of the Traveling Salesman problem from one using explicit assignment variables).

Yuille [23] has shown how to derive the elastic net formulation from the one containing match variables by means of summing over all possible assignments in the partition function (or equivalently computing the marginal distribution). Again, the partition function is modified to restrict the space of possible solutions to the ones satisfying the column (or row) constraint for the permutation matrix. Unfortunately, summing over all permutation matrices does not lead to a simplified expression for Z [23]. Instead the summation is performed over a superset, the set of matrices satisfying only the row constraint ($\sum_i M_{ij} = 1$):

$$\begin{aligned} Z &= \sum_{\{\{m_{ij}\} \mid \sum_i m_{ij}=1\}} \sum_{\mathbf{x},\{\mathbf{x}_j\}} e^{-\beta E(\mathbf{M},\mathbf{x},\mathbf{x}_j)} \\ &= \sum_{\{\{m_{ij}\} \mid \sum_i m_{ij}=1\}} \sum_{\mathbf{x},\{\mathbf{x}_j\}} \left(\prod_i e^{-\beta \frac{1}{2\sigma^2} \sum_j M_{ij} |\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_i)|^2} \right) \\ &= \sum_{\mathbf{x},\{\mathbf{x}_j\}} \left(\prod_i \sum_{\{\{m_{ij}\} \mid \sum_i m_{ij}=1\}} e^{-\beta \frac{1}{2\sigma^2} \sum_j M_{ij} |\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_i)|^2} \right) \\ &= \sum_{\mathbf{x},\{\mathbf{x}_j\}} \left(\prod_i \left(\sum_j e^{-\beta \frac{1}{2\sigma^2} |\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_i)|^2} \right) \right) \\ &= \sum_{\mathbf{x},\{\mathbf{x}_j\}} e^{-\beta E_{\text{eff}}(\mathbf{x},\{\mathbf{x}_j\})} \end{aligned} \tag{11}$$

with

$$E_{eff}(\mathbf{x},\{\mathbf{x}_j\}) = -1/\beta \sum_j \log \left(\sum_i e^{-\beta \frac{1}{2\sigma^2} |\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_i)|^2} \right) \tag{12}$$

$E_{eff}(\mathbf{x},\{\mathbf{x}_j\})$ must be minimized with respect to $\mathbf{x}$, i.e. the solution for x is found from

$$\frac{\partial E_{eff}(\mathbf{x})}{\partial \mathbf{x}} = -\sum_j \sum_i \frac{1}{\sigma^2} \frac{e^{\left(-\beta \frac{1}{2\sigma^2} |\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_i)|^2\right)}}{\sum_l e^{\left(-\beta \frac{1}{2\sigma^2} |\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_l)|^2\right)}} |\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_i)| = 0 \tag{13}$$

for example via gradient decent.

Note that integrating over the assignment variables $\mathbf{M}$ resulted in an equation for $\mathbf{x}$ similar to the one obtained from the saddle point approximation but with the solution for $\mathbf{M}$ substituted in (compare equations (8)-(10)). There, $\mathbf{M}$ is explicitly represented as a variable and at each intermediate stage, the current values m_{ij} are found by computing u_{ij} and passing it trough the *softmax* transfer function (9). These values are then used to update the position coordinates $\mathbf{x}$. In the Elastic Net formulation, $\mathbf{M}$ is no longer represented as variable but the corresponding expression appears directly in the equation for updating $\mathbf{x}$. In both cases, deterministic annealing is performed by slowly decreasing $T = 1/\beta$ (but only down to 1 to preserve the ratio of possibly different variances for different parts of the object [9]).

3 Mixture Model for Labelling Data

An alternative to the index permutation is to model the unordering step by describing the distribution of parameters of parts in form of a mixture density.

Mixture models have been used in the context for learning (see for example Nowlan [13]). There, the task is to estimate the parameters of the mixture density from data. Ultimately, the goal in the context of the object recognition problem is to learn the model base, its structure and parameters, from data. The mixture formulation is attractive since in its most general form does not assume the data is already labeled. If the structure, i.e. the part hierarchy of the objects in the model data base is predetermined, the use of labeled data simplifies the training procedure considerably. However, even though is seems reasonable that the data is labeled at higher levels in the hierarchy (i.e. the label for an entire object is known), determining the structure of the data base at the part level can be part of the learning algorithm. Such an algorithm should learn from data, which substructures (i.e. "parts") should be allocated for the object to best recognize it. For example, this could mean to find prominent features that are unique to a particular model or features that maximally discriminate between multiple stored models. Then, the training data derived from the image must be unlabeled simply because the part nodes in the model data base do not exist yet.

In this report, the recognition problem is stated in terms of estimating parameters of a mixture distribution and solved using the EM algorithm. The close relationship to the Mean Field methods described above is pointed out. Thus, the recognition and learning problem can be stated in the same framework, but in each case, a different set of variables must be computed.

3.1 A Mixture Density Model for Dot Configurations

The general form of a mixture density $p(x)$ is

$$p(x, \Theta) = \sum_i g_i p_i(x, \theta_i) \tag{14}$$

where the mixture coefficients $g_i > 0$ obey $\sum_i g_i = 1$ and the $p_i(x, \theta_i)$ are themselves density functions with parameters θ_i. The parameters Θ represent the union of all parameters θ_i of the component densities. The coefficients g_i can also be interpreted as the prior probability that an observation x comes from $p_i(x)$.

The probability density for the dot parameters can be viewed as a mixture density. From the joint distribution in equation (1), the probability for generating part i with parameters x_i is given by

$$P_i(\mathbf{x}_i|\theta_i) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{\frac{1}{2\sigma_i^2}|\mathbf{x}_i - (\mathbf{x}+\mathbf{d}_i)|^2} \tag{15}$$

with parameters $\theta_i = (\mathbf{d}_i, \sigma, \mathbf{x})$ and as mixture model

$$P(\mathbf{x}_j|\Theta) = \sum_i g_{ji} \frac{1}{\sqrt{2\pi}\sigma_i} e^{\frac{1}{2\sigma_i^2}|\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_i)|^2} \tag{16}$$

with $g_{ji} \in \{0, 1\}$ denoting which component density was chosen to place the j^{th} dot. The matrix $\mathbf{G} = \{g_{ji}\}$ again is a permutation matrix (since for the dot cluster model, each component density is used exactly once to place one of the dots). In addition to the constrain that the column sum is equal to 1 as required for the mixture coefficients, here, the rows must sum to 1 as well. Thus, the mixture formulation is an alternate way to model the ignorance of the correct part assignment.

Under this model, the last two step in the stochastic grammar are combined into one. Instead of placing labeled dots and then permuting their indices, a matrix of binary mixture coefficient g_{ji} is chosen (similar to choosing the permutation matrix $\mathbf{M}$). Then, for each dot j to be placed in the image, the position x_j is determined from equation (16). Each dot is placed independently as before, thus the joint distribution for the position of all image dots is

$$P(\{\mathbf{x}_j\}, \mathbf{x}, \mathbf{G}) \quad = \quad P(\mathbf{G}) \prod_j \sum_i g_{ji} \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^2 e^{-\frac{1}{2\sigma^2}|\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_i)|^2} \tag{17}$$

(P($\mathbf{G}$) is again assumed uniform over the space of permutation matrices and thus assumed a constant it what follows).

For the recognition problem only a single exemplar is available, but each exemplar consists of n datapoints corresponding to the parts of the object. Thus, learning the parameters θ_i and $\mathbf{G}$ is not meaningful. However, the n component distributions share parameters at higher levels in the compositional hierarchy and these can be estimated. The parameters of the mixture density are assumed known and the task is to compute the parameters of the object in the scene and label its parts. For the dot cluster model, the single parameter is the cluster position $\mathbf{x}$. The part labels can be interpreted as the *a posteriori* estimate of the mixture coefficients (the estimate computed after the data $\{\mathbf{x}_j\}$ has been observed).

3.2 EM Algorithm

The observed image data consists of unlabeled dots $\{x_j\} \quad j = 1 \ldots n$. Here, the lack of labels is treated as a missing data problem. Instead of observing the dot position and its label, only the position is given. Thus, it is not known which density function describes the parameters of a particular dot. In addition to the observed dot positions $\{x_j\}$ unobservable indicator variables $\mathbf{m}_j$ denoting the labels are introduced with

$$m_{ji} = \begin{cases} 1 & \text{if } x_j \text{ belongs to the } i^{th} \text{ component density } p_i(\cdot) \\ 0 & \text{otherwise} \end{cases} \quad . \tag{18}$$

The complete data can be defined as pairs $\{(x_j, \mathbf{m}_j)\}$. (Note that the same symbol m that was used earlier to denote the elements of the permutation matrix $\mathbf{M}$ now represents the indicator variables. In the context of the recognition problem both quantities represent the solution to the labelling problem; however, the probabilistic interpretation is different in each case.)

An algorithm for estimating the parameters of a mixture distribution from unlabeled data is the *Expectation Maximization (EM)* algorithm. The EM algorithm has been described in detail by Demster *et al.* [3] and Redner and Walker [16] (see also Nowlan [13]). The EM algorithm attempts to compute the most likely values for the parameters of the component density functions and the mixture coefficients from the observable data alone. The algorithm estimates the statistics of the unobservable indicator variables and uses these estimates to improve the parameter estimates. If the data were labeled, the EM algorithm would just compute the maximum likelihood estimates for the unknown quantities. For unlabeled data, for each iteration, in a first step (the E-step) the expected value for the indicator variables are determined from the data given that the current values for the parameters of the density functions. Then, in a second step (the M-step) new maximum likelihood estimates for the parameters are computed but instead of using the true coefficients, the estimates of the indicator variables from the E step are used. This amounts to weighting each component density with the current estimate of the corresponding indicator variable. Note that for the recognition problem, only the indicator variables denoting the part labels and the estimate for $\mathbf{x}$ are of interest (it would not be meaningful to estimate the mixture coefficients and the offsets $\mathbf{d}_i$ from a single exemplar).

For the mixture density from equation (16), the log-likelihood for the complete data pairs can be written as [3, 13]:

$$\log P(\mathbf{x}, \mathbf{M}|\Theta) = \sum_j \sum_i m_{ji} \left(\log P_i(\mathbf{x}_j|\theta_i) + \log P(m_{ji}|\Theta)\right) \tag{19}$$

where

$$P_i(\mathbf{x}_j|\theta_i) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}|\mathbf{x}_j - (\mathbf{x}+\mathbf{d}_i)|^2} \tag{20}$$

and

$$P(m_{ji}|\Theta) = 1/n \tag{21}$$

denotes the prior density for the indicator variables before the data arrives, and is assumed uniform for the recognition problem. (Note that this ignores the requirement that for the problem considered here $\mathbf{M}$ must be a permutation matrix, i.e. that individual rows cannot be chosen independently. This constraint could be included by constructing a prior (Gibbs) distribution from the corresponding penalty term.)

The algorithm consists of two steps. During the E-step, estimates of the indicator variables are computed given the observed data and the current parameters. Here, these estimates are the conditional probabilities that observation x_j belongs to the i^{th} component density. Thus (via Bayes theorem)

$$\langle m_{ji} \rangle \quad = \quad p(m_{ji}|\mathbf{x}_j, \Theta)$$

$$\begin{aligned} &= \frac{g_{ji} p_i(\mathbf{x}_j|\theta_i)}{\sum_l g_{jl} p_l(\mathbf{x}_j|\theta_l)} \\ &= \frac{p_i(\mathbf{x}_j|\theta_i)}{\sum_l p_l(\mathbf{x}_j|\theta_l)} \end{aligned} \tag{22}$$

Note that due to the normalization, the sum constraint for mixture coefficients is always satisfied.

The M-step consists of computing the most likely parameter values given the estimates of the indicator variables. Each of the component distribution contributes weighted by the estimated mixture coefficient for each data point.

Here, we are only interested in the estimate for the cluster position $\hat{\mathbf{x}}$ from a single image:

$$\sum_i \sum_j \langle m_{ij} \rangle \frac{\partial \log P_i(\mathbf{x}_j|\theta_i)}{\partial \hat{\mathbf{x}}} = 0 \tag{23}$$

and after substituting equations (22) and (20)

$$-\sum_i \sum_j \frac{1}{\sigma^2} \frac{e^{(-1/2\sigma^2)|\mathbf{x}_j - (\hat{\mathbf{x}}^{t-1} + \mathbf{d}_i)|^2}}{\sum_k e^{(-1/2\sigma^2)|\mathbf{x}_j - (\hat{\mathbf{x}}^{t-1} + \mathbf{d}_k)|^2}} \left| \mathbf{x}_j - (\hat{\mathbf{x}}^t + \mathbf{d}_i) \right| = 0 \tag{24}$$

Consecutive iterations of the algorithm proceed as follows. The current parameter estimate ($\hat{\mathbf{x}}^{t-1}$ from the previous M-step) is used to compute $\langle m_{ji}^t \rangle$ at iteration t. Then the solution of (24), $\hat{\mathbf{x}}^t$ is found. The algorithm alternates between the E and the M-step until the final solution is found.

The solution for the indicator variables m_{ji} is not necessarily binary. In the context of a recognition problem, an annealing temperature can be introduced (similar to the Mean Field Annealing methods) to force a binary solution using

$$\lim_{\beta \to \infty} \frac{e^{\beta x_j}}{\sum_i e^{\beta x_i}} = \begin{cases} 1 & \text{if } x_j > x_i \ \forall \, i \neq j \\ 0 & \text{otherwise} \end{cases} \quad . \tag{25}$$

3.3 Comparison to Mean Field Methods

Note that the form of the EM equations is identical to the Elastic Net equation (13). This is not surprising since the Mean Field Approximation replaces the binary assignment coefficients with their mean value and thus performs an operation equivalent to the E step of the EM algorithm for binary valued assignment coefficients.

Note however, that the probabilistic interpretation of the assignment variables m_{ij} differs. In the original formulation of the stochastic grammar, no probabilistic formulation for the elements of the permutation matrix $\mathbf{M}$ exists; they represent a deterministic renumbering operation. They are interpreted as random variables only trough the Mean Field method which regards them as *Markov Random Field (MRF)* corresponding to the assumption that the governing distribution is Gibbsian. In particular, each element m_{ij} is considered a independent random variable, no notion of a conditional density is introduced and consequently, the sum constraints are expressed as additional terms in the distribution derived from the corresponding penalty terms.

The mixture model includes the assignment variables in the form of mixture coefficients in the model. Also, the model distinguishes between the prior density as specified for the generative model and the posteriori estimates computed for the recognition problem. This becomes important in the context of learning where the EM algorithm still would calculate the posteriori estimates (if the data is unlabelled) but in this case these would just be auxiliary variables used in learning the mixture coefficients throughout the compositional hierarchy. By regarding the assignment coefficients as being probabilities throughout the calculation, they obey the sum constraint without the need to introduce penalty terms (however, the row sum constraint is not automatically satisfied since it corresponds to a correlation of mixture coefficients for different parts x_j and thus is not included in the model described here).

As a maximum likelihood technique, the EM algorithm does not include the notion of a computational temperature T. It can still beneficial to perform deterministic annealing since the optimization problem is multi-model and only local convergence is guaranteed. Also, as mentioned above, in the context of the recognition problem annealing can be used to force a binary solution. It is known that the model's coefficients are binary (as in the example used here), one can regard this as an alternative to explicitly including a penalty term to that effect.

The important difference between the EM algorithm and the Mean Field networks is that while in the later, the calculation of parameters estimates and assignment coefficients proceed concurrently, the EM algorithm alternates between the E-step and the M-step. That is, given the correct estimate of the parameters, new estimates for the assignment coefficients are computed using (22). Then, holding these estimates constant, new estimates for the parameters of the density functions are computed. Switching back to the E-step, the estimates for the assignment coefficients are not updated starting from their old value, but computed anew based on the new parameters estimates from the previous M-step. Thus, while the form of the Mean Field equations is identical to the EM equations, each iteration of the EM algorithm involves completely solving the optimization problem of estimating $\mathbf{x}$. While the Mean Field methods update $\mathbf{M}$ and $\mathbf{x}$ concurrently, with the EM algorithm, new estimates for $\mathbf{M}$ are computed only after the values for the parameters $\mathbf{x}$ have converged at each intermediate stage.

The optimization problem with respect to the assignment variables is difficult due to many local minima but the optimization with respect to the parameters is quadratic (for the Gaussian distributions used here) and thus much easier to solve. By solving this easier problem at each stage and then computing new estimates for the assignment coefficients, the EM algorithm can compute these based on a more accurate estimates of the parameters.

An approximation to the behavior of the EM algorithm would be to update the assignment variables more slowly by using a larger time constant for their dynamics. Since the optimization problem for the parameters is easier, presumably a solution can be found faster than for the assignment coefficients. Assuming some continuity in the space of solutions, by updating the assignment coefficients more slowly than the parameters, the later can track the solution. On the other hand, the assignment coefficients avoid local minima by allowing the parameters to stay close to their optimal value.

In the context of the mixture density model and the EM algorithm, the Bootstrap method described by Utans and Gindi [20, 19, 21] can be interpreted as the first (or first few steps) of the EM algorithm. There, it was proposed to improve convergence in hierarchical matching networks by initializing the parameter estimates from a coarse scale version of the input image. In the case of a single object, this was shown to be equivalent to computing parameter estimates via the Mean Field equations in the limit for large temperatures. For high temperatures, the assignment coefficients are equal to $1/n$, i.e. all assignments are weighted equally. This corresponds to using the prior

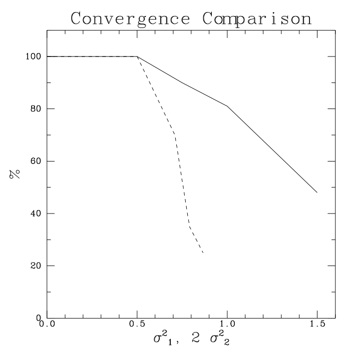

Figure 2: Performance Results comparing the EM algorithm (solid line) to a Hopfield type match network. The performance of the EM algorithm is comparable to the that of the Elastic Net for this problem. The success rate indicates the rate at which the network converged to the correct solutions. σ_1^2 denotes the noise variance at the intermediate level of the model and σ_2^2 the noise variance at the lowest level. Only one set of 10 experiments (for each variance value tested) was used for the graph but in all simulations performed the EM algorithm proved to converge more robustly.

probabilities for the assignment coefficients as the values to use in the first M-step to compute initial parameter estimates. Then, using these estimates, the assignment coefficients are updated (corresponding to the second E-step). However, both are updated concurrently after initialization.

As a final comment, while the mixture formulation appears more consistent by including the assignment coefficients in the probabilistic framework from the beginning, methods based on the Mean Field approximation can be more flexible for solving constrained optimization problems in general. In the general case, where the constraints do not directly correspond to a sum constraint, a formulation in terms of conditional densities will not preserve that constraint. The Mean Field methods, by treating the binary variables as independent random variables and only later incorporate constraints be means of modifying the partition function in the Gibbs distribution, can potentially deal with other constrains as well. Also, since the way the partition function is modified is at the will of the network designer, more that one way can exist to incorporate a given constraint, allowing a choice that best maps the constraint onto a neural network architecture (i.e. choosing the functional form that is optimized most easily).

3.4 Simulation Results

The EM algorithm has been implemented for a 3-level hierarchical model for dot configurations (see [9]). In an intermediate step, dot cluster centers $\mathbf{x}_c$ are placed, the final dot position are taken relative to these centers. The parameters to be estimated are $\mathbf{x}$ and the $\{\mathbf{x}_c\}$. Figure 2 compares the performance of the algorithm to a Hopfield match network as the noise variance is increased (σ_1^2 denotes the noise variance at the intermediate level of the model and σ_2^2 the noise variance at the lowest level). For the simulation, $\sigma_2^2 = 2\sigma_1^2$; the noise variance was identical for all parts.

The performance of the EM algorithm is comparable to the that of the Elastic Net for this problem. The network computed the correct solution more reliably for large noise variances. In such cases the performance of the Hopfield match network deteriorates rapidly.

Acknowledgements

This work continues research done at Yale University with G. Gindi and E. Mjolsness. Discussions with S. Omohundro and A. Rangarajan where helpful in focusing on the EM algorithm.

References

[1] Daniel J. Amit. *Modeling Brain Function.* Cambridge University Press, 1989.

[2] Daniel J. Amit, Hanoch Gutfreund, and H. Sompolinsky. Spin-glass models of neural networks. *Physical Review A*, 32(2):1007–1018, August 1985.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statist. Soc. B*, 39:1–39, 1977.

[4] R. Durbin and D. Willshaw. An analog approach to the travelling salesman problem using an elastic net method. *Nature*, 326:689–691, 1987.

[5] G. Gindi, E. Mjolsness, and P. Anandan. Neural networks for model based recognition. In *Neural Networks: Concepts, Applications and Implementations*, pages 144–173. Prentice–Hall, 1991.

[6] J. J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences USA*, vol. 81:3088–3092, May 1984.

[7] J. J. Hopfield and D. W. Tank. 'Neural' computation of decisions in optimization problems. *Biological Cybernetics*, vol. 52:141–152, 1985.

[8] Christoph von der Malsburg. Pattern recognition by labeled graph matching. *Neural Networks*, 1:141–148, 1988.

[9] E. Mjolsness. Bayesian inference on visual grammars by neural nets that optimize. Technical Report YALEU–DCS–TR–854, Yale University, Dept. of Computer Science, 1991.

[10] E. Mjolsness. Visual grammars and their neural nets. In R.P. Lippmann J.E. Moody, S.J. Hanson, editor, *Advances in Neural Information Processing Systems 4*. Morgan Kaufmann Publishers, San Mateo, CA, 1992.

[11] Eric Mjolsness, Gene Gindi, and P. Anandan. Optimization in model matching and perceptual organization: A first look. Research report yaleu/dcs/rr-634, Yale University, Department of Computer Science, 1988.

[12] Eric Mjolsness, Gene R. Gindi, and P. Anandan. Optimization in model matching and perceptual organization. *Neural Computation*, vol. 1, no. 2, 1989.

[13] Steven J. Nowlan. *Soft Competitive Adaptation: Neural Network Learning Algorithms based on Fitting Statistical Mixtures.* PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, 1991.

[14] Carsten Peterson and James R. Anderson. Neural networks and NP–complete optimization problems: A performance study on the graph bisection problem. TR MCC-EI-287-87, Microelectronics and Computer Technology Corporation, 3500 West Balcones Drive, Austin, TX 78759, 1987.

[15] Carsten Peterson and Bo Soderberg. A new method for mapping optimization problems onto neural networks. *International Journal of Neural Systems*, 1(1):3–22, 1989.

[16] Richard A. Redner and Homer F. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239, April 1984.

[17] Petar D. Simic. "Constrained nets" for graph matching and other quadratic assignment problems. TR CALT–68–1672, California Institute of Technology, Pasadena, CA 91125, June 1990.

[18] Petar D. Simic. Statistical mechanics as the underlying theory of "elastic net" and "neural" optimization. *Network*, 1:89–103, 1990.

[19] J. Utans and G. R. Gindi. A neural network approach to object recognition and image partitioning within a resolution hierarchy. In *SPIE Intelligent Information Systems OR'92*, Orlando, FL, April 20–24 1992. SPIE.

[20] J. Utans and G. R. Gindi. A neural network performs context guided segmentation of shapes via bayesian inference. In *presented at Neural Networks for Computing*, Snowbird, Utah, April 7–10 1992.

[21] Joachim Utans and Gene R. Gindi. Improving convergence in hierarchical matching networks for object recognition. In Stephen J. Hanson, Jack Cowan, and Lee Giles, editors, *Advances in Neural Information Processing 5*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[22] Joachim Utans, Gene R. Gindi, Eric Mjolsness, and P. Anandan. Neural networks for object recognition within compositional hierarchies: Initial experiments. Technical report 8903, Yale University, Center for Systems Science, Department Electrical Engineering, 1989.

[23] A. L. Yuille. Generalized deformable models, statistical physics, and matching problems. *Neural Computation*, 2(2):1–24, 1990.

[24] Alan Yuille, Paul Stolorz, and Joachim Utans. Statistical physics, mixtures of distributions and the EM algorithm. *to appear*, 1993.