Pattern Recognition by Probabilistic Neural Networks
Mixtures of Product Components versus Mixtures of Dependence Trees

Jiří Grim[1] and Pavel Pudil[2]
[1]*Institute of Information Theory and Automation, Czech Academy of Sciences, Prague, Czech Republic*
[2]*Faculty of Management, Prague University of Economics, Jindřichův Hradec, Czech Republic*
http://www.utia.cas.cz/RO and http://www.fm.vse.cz/

Keywords: Probabilistic Neural Networks, Product Mixtures, Mixtures of Dependence Trees, EM Algorithm.

Abstract: We compare two probabilistic approaches to neural networks - the first one based on the mixtures of product components and the second one using the mixtures of dependence-tree distributions. The product mixture models can be efficiently estimated from data by means of EM algorithm and have some practically important properties. However, in some cases the simplicity of product components could appear too restrictive and a natural idea is to use a more complex mixture of dependence-tree distributions. By considering the concept of dependence tree we can explicitly describe the statistical relationships between pairs of variables at the level of individual components and therefore the approximation power of the resulting mixture may essentially increase. Nonetheless, in application to classification of numerals we have found that both models perform comparably and the contribution of the dependence-tree structures decreases in the course of EM iterations. Thus the optimal estimate of the dependence-tree mixture tends to converge to a simple product mixture model. Regardless of computational aspects, the dependence-tree mixtures could help to clarify the role of dendritic branching in the highly selective excitability of neurons.

1 INTRODUCTION

Considering the probabilistic approach to neural networks in the framework of statistical pattern recognition we approximate the unknown class-conditional probability distributions by mixtures of product components (Grim, 1996; Grim, 2007). The basic principle of probabilistic neural networks (PNN) is to view the mixture components as formal neurons. We have shown that the component parameters can be estimated by a sequential strictly modular version of EM algorithm (Grim, 1999b). The estimated mixtures define an information preserving transform which can be used to design multilayer PNN sequentially (Grim, 1996). The independently designed information preserving transforms can be combined both in the horizontal and vertical sense (Grim et al., 2002). A subspace modification of EM algorithm can be used to optimize the structure of incompletely interconnected PNN (Grim et al., 2000). In a series of papers we have analyzed the properties of PNN in application to practical problems of pattern recognition (Grim and Hora, 2008).

The probabilistic neuron can be interpreted from the neurophysiological point of view in terms of the functional properties of biological neurons (Grim, 2007). In particular the explicit formula for the synaptic weight can be viewed as a theoretical counterpart of the well known Hebbian principle of learning (Hebb, 1949). The information preserving transform assumes the activation function of probabilistic neurons in a logarithmic form and, in this way, the product components define the output of a neuron as a weighted sum of synaptic contributions. In addition, the mixtures of product components have some specific advantages, like easily available marginals and conditional distributions, a direct applicability to incomplete data and the possibility of structural optimization of multilayer PNN (Grim, 1986). The concept of PNN is also compatible with the technique of boosting (Grim et al., 2002b) which can be used to model emotional learning.

However, there is also a computational motivation to apply the product mixture model. In the last decades there is an increasing need of estimating multivariate and multimodal probability distributions from large data sets. Such databases are usually produced by information technologies in various areas like medicine, image processing, monitoring systems, communication networks and others. A typical feature of the arising "technical" data is a high dimensionality and a large number of measurements. The unknown underlying probability distributions or density functions are nearly always multimodal and cannot be assumed in a simple parametric form. In this sense, one of the most efficient possibilities is to approximate the unknown multidimensional probability distributions by finite mixtures and, especially, by mixtures of components defined as products of univariate distributions (Grim, 1982; Grim, 1986; Grim et al., 2000; Grim, 2007; Grim and Hora, 2010; Lowd and Domingos, 2005). In the past the approximation potential of product mixtures has been often underestimated probably because of formal similarity with the so-called naive Bayes models which assume the class-conditional independence of variables (Lowd and Domingos, 2005). In connection with product mixtures this term is incorrectly used because there is nothing naive on the assumption of product components of mixtures. In case of discrete variables the product mixtures are universal approximators since any discrete distribution can be expressed as a product mixture (Grim, 2006). Similarly, the Gaussian product mixtures approach the universality of non-parametric Parzen estimates with the increasing number of components.

Nevertheless, despite the advantageous properties of product mixtures, the simplicity of product components may become restrictive in some cases. For this reason it could be advantageous to consider the mixture components in a more specific form. A natural approach is to use dependence-tree distributions (Chow and Liu, 1968) as components. By using the concept of dependence tree we can explicitly describe the statistical relationships between pairs of variables at the level of individual components. Therefore, the approximation "power" of the resulting mixture model should increase. We have shown (Grim, 1984) that mixtures of dependence-tree distributions can be optimized by EM algorithm in full generality.

In the domain of probabilistic neural networks the mixtures of dependence trees could help to explain the role of dendritic branching in biological neurons. It is assumed that easily excitable thin dendritic branches may facilitate the propagation of excitation to neural body and the effect of conditional facilitation could be modeled by the dependence-tree structure.

In this paper we describe first the product mixture model (Sec. 2, Sec. 3). In Sec. 4 we recall the concept of dependence-tree distribution in the framework of finite mixtures. In Sec.5 we discuss different aspects of the two types of mixtures in a computational experiment - in application to recognition of numerals. The results are summarized in the Conclusion.

2 ESTIMATING MIXTURES

Considering distribution mixtures, we approximate the unknown probability distributions by a linear combination of component distributions

$$P(x|w,\Theta) = \sum_{m\in\mathcal{M}} w_m F(x|\theta_m), \tag{1}$$

$$w = (w_1, w_2, \dots, w_M), \quad \theta_m = \{\theta_{m1}, \theta_{m2}, \dots, \theta_{mN}\}.$$

where $x \in X$ are discrete or real data vectors, w is the vector of probabilistic weights, $\mathcal{M} = \{1, \dots, M\}$ is the component index set and $F(x|\theta_m)$ are the component distributions with parameters θ_m.

Since the late 1960s the standard way to estimate mixtures is to use the EM algorithm (Hasselblad, 1966; Schlesinger, 1968; Hasselblad, 1969; Day, 1969; Hosmer, 1973; Wolfe, 1970; Dempster et al., 1977; Grim, 1982). Formally, given a finite set $\mathcal{S}$ of independent observations of the underlying N-dimensional random vector

$$\mathcal{S} = \{x^{(1)}, x^{(2)}, \dots\}, \quad x = (x_1, x_2, \dots, x_N) \in X, \tag{2}$$

we maximize the log-likelihood function

$$L(w,\Theta) = \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}} \log\left[\sum_{m\in\mathcal{M}} w_m F(x|\theta_m)\right] \tag{3}$$

by means of the following EM iteration equations ($m \in \mathcal{M}, n \in \mathcal{N}, x \in \mathcal{S}$):

$$q(m|x) = \frac{w_m F(x|\theta_m)}{\sum_{j\in\mathcal{M}} w_j F(x|\theta_j)}, \quad w_m' = \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}} q(m|x), \tag{4}$$

$$Q_m(\theta_m) = \sum_{x\in\mathcal{S}} \frac{q(m|x)}{\sum_{y\in\mathcal{S}} q(m|y)} \log F(x|\theta_m), \tag{5}$$

$$\theta_m' = \arg\max_{\theta_m}\left\{Q_m(\theta_m)\right\}. \tag{6}$$

Here the apostrophe denotes the new parameter values in each iteration. One can easily verify (cf. (Grim, 1982)) that the general iteration scheme (4) - (6) produces nondecreasing sequence of values of the maximized criterion (3). In view of the implicit relation (6) any new application of EM algorithm is reduced to the explicit solution of Eq. (6).

Considering product mixtures, we assume the component distributions $F(x|\theta_m)$ defined by products

$$F(x|\theta_m) = \prod_{n\in\mathcal{N}} f_n(x_n|\theta_{mn}), \quad m \in \mathcal{M} \tag{7}$$

and therefore Eqs. (6) can be specified for variables independently:

$$Q_{mn}(\theta_{mn}) = \sum_{x\in\mathcal{S}} \frac{q(m|x)}{w_m'|\mathcal{S}|} \log f_n(x_n|\theta_{mn}), \tag{8}$$

$$\theta'_{mn} = \arg\max_{\theta_{mn}} \left\{ Q_{mn}(\theta_{mn}) \right\}, \quad n \in \mathcal{N}. \qquad (9)$$

The mixtures of product components have some specific advantages as approximation tools. Recall that any marginal distribution of product mixtures is directly available by omitting superfluous terms in product components. Thus, in case of prediction, we can easily compute arbitrary conditional densities and for the same reason product mixtures can be estimated directly from incomplete data without estimating the missing values (Grim et al., 2010). Product mixtures support a subspace modification for the sake of component-specific feature selection (Grim, 1999; Grim et al., 2006) and can be used for sequential pattern recognition by maximum conditional informativity (Grim, 2014). Moreover, the product components simplify the EM iterations and increase the numerical stability of EM algorithm.

3 MULTIVARIATE BERNOULLI MIXTURES

In case of binary data $x_n \in \{0,1\}$ the product mixture model is known as multivariate Bernoulli mixture based on the univariate distributions

$$f_n(x_n|\theta_{mn}) = (\theta_{mn})^{x_n}(1-\theta_{mn})^{1-x_n}, \quad 0 \le \theta_{mn} \le 1,$$

with the resulting product components

$$F(x|\theta_m) = \prod_{n\in\mathcal{N}} (\theta_{mn})^{x_n}(1-\theta_{mn})^{1-x_n}, \quad m \in \mathcal{M}. \quad (10)$$

The conditional expectation criterion $Q_{mn}(\theta_{mn})$ can be expressed in the form

$$Q_{mn}(\theta_{mn}) = \sum_{\xi\in\mathcal{X}_n} \Big(\sum_{x\in\mathcal{S}} \delta(\xi,x_n) \frac{q(m|x)}{w'_m|\mathcal{S}|} \Big) \log f_n(\xi|\theta_{mn}),$$

and therefore there is a simple solution maximizing the weighted likelihood (9):

$$\theta'_{mn} = \sum_{x\in\mathcal{S}} x_n \frac{q(m|x)}{w'_m|\mathcal{S}|}. \qquad (11)$$

We recall that, as an approximation tool, the multivariate Bernoulli mixtures are not restrictive since, for a sufficiently large number of components, any distribution of a random binary vector can be expressed in the form (1), (10), (cf. (Grim, 2006)).

In case of multivariate Bernoulli mixtures we can easily derive the structural (subspace) modification (Grim, 1986; Grim, 1999) by introducing binary structural parameters $\varphi_{mn} \in \{0,1\}$ in the product components

$$F(x|\theta_m) = \prod_{n\in\mathcal{N}} f_n(x_n|\theta_{mn})^{\varphi_{mn}} f_n(x_n|\theta_{0n})^{1-\varphi_{mn}}. \quad (12)$$

It can be seen that by setting $\varphi_{mn} = 0$ in the formula (12), we can substitute any component-specific univariate distribution $f_n(x_n|\theta_{mn})$ by the respective univariate background distribution $f_n(x_n|\theta_{0n})$. The structural component can be rewritten in the form

$$F(x|\theta_m) = F(x|\theta_0)G(x|\theta_m,\phi_m), \quad m \in \mathcal{M}, \qquad (13)$$

where $F(x|\theta_0)$ is a nonzero "background" probability distribution - usually defined as a fixed product of the unconditional univariate marginals

$$F(x|\theta_0) = \prod_{n\in\mathcal{N}} f_n(x_n|\theta_{0n}), \quad \theta_{0n} = \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}} x_n, \quad n \in \mathcal{N}.$$

In this way we obtain the subspace mixture model

$$P(x|w,\Theta,\Phi) = F(x|\theta_0) \sum_{m\in\mathcal{M}} w_m G(x|\theta_m,\phi_m), \quad (14)$$

where the component functions $G(x|\theta_m,\phi_m)$ include additional binary structural parameters $\varphi_{mn} \in \{0,1\}$:

$$G(x|\theta_m,\phi_m) = \prod_{n\in\mathcal{N}} \left[\frac{f_n(x_n|\theta_{mn})}{f_n(x_n|\theta_{0n})} \right]^{\varphi_{mn}}, \qquad (15)$$

$$\phi_m = (\varphi_1^{(m)}, \ldots, \varphi_{mN}) \in \{0,1\}^N.$$

Consequently, the component functions $G(x|\theta_m,\phi_m)$ may be defined on different subspaces. In other words, each component may "choose" its own optimal subset of informative features. The complexity and "structure" of the finite mixture (14) can be controlled by means of the binary parameters φ_{mn} since the number of parameters is reduced whenever $\varphi_{mn} = 0$. Thus we can estimate product mixtures of high dimensionality while keeping the number of estimated parameters reasonably small.

The structural parameters φ_{mn} can be optimized by means of the EM algorithm in full generality (cf. (Grim, 1984; Grim et al., 2000; Grim, 2007)) by maximizing the corresponding likelihood criterion:

$$L = \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}} \log \left[\sum_{m\in\mathcal{M}} w_m F(x|\theta_0) G(x|\theta_m,\phi_m) \right].$$

In the following iteration equations, the apostrophe denotes the new parameter values $(m \in \mathcal{M}, n \in \mathcal{N})$:

$$q(m|x) = \frac{w_m G(x|\theta_m,\phi_m)}{\sum_{j\in\mathcal{M}} w_j G(x|\theta_j,\phi_j)}, \qquad (16)$$

$$w'_m = \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}} q(m|x), \quad \theta'_{mn} = \sum_{x\in\mathcal{S}} x_n \frac{q(m|x)}{w'_m|\mathcal{S}|}, \quad (17)$$

$$\gamma'_{mn} = \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}} q(m|x) \log \frac{f_n(x_n|\theta'_{mn})}{f_n(x_n|\theta_{0n})}. \qquad (18)$$

Assuming a fixed number of component specific parameters λ, we define the optimal subset of nonzero parameters φ'_{mn} by means of the λ highest values $\gamma'_{mn} > 0$. From the computational point of view it is more efficient to specify the structural parameters by simple thresholding

$$\varphi'_{mn} = \begin{cases} 1, & \gamma'_{mn} > \tau \\ 0, & \gamma'_{mn} \le \tau \end{cases}, \quad \left(\tau \approx \frac{\gamma_0}{MN} \sum_{m\in\mathcal{M}} \sum_{n\in\mathcal{N}} \gamma'_{mn}\right)$$

where the threshold τ is derived from the mean value of γ'_{mn} by a coefficient γ_0. The structural criterion γ'_{mn} (cf. (18)) can be rewritten in the form:

$$\gamma'_{mn} = w'_m \sum_{\xi=0}^{1} f_n(\xi|\theta'_{mn}) \log \frac{f_n(\xi|\theta'_{mn})}{f_n(\xi|\theta_{0n})} = \qquad (19)$$

$$= w'_m I(f_n(\cdot|\theta'_{mn})||f_n(\cdot|\theta_{0n})).$$

In other words, the structural criterion γ'_{mn} can be expressed in terms of Kullback-Leibler information divergence $I(f_n(\cdot|\theta'_{mn})||f_n(\cdot|\theta_{0n}))$ (Kullback and Leibler, 1951) between the component-specific distribution $f_n(x_n|\theta'_{mn})$ and the corresponding univariate "background" distribution $f_n(x_n|\theta_{0n})$. Thus, only the most distinct (i.e. specific and informative) distributions $f_n(x_n|\theta'_{mn})$ are included in the components.

It can be verified (Grim, 1999; Grim et al., 2000) that, for a fixed λ, the iteration scheme (16)-(18) guarantees the monotonic property of the EM algorithm. Recently the subspace mixture model has been apparently independently proposed to control the Gaussian mixture model complexity (Markley and Miller, 2010; Bouguila et al., 2004).

The main motivation for the subspace mixture model (14) has been the statistically correct structural optimization of incompletely interconnected probabilistic neural networks (Grim, 1999; Grim, 2007; Grim and Hora, 2008; Grim et al., 2002; Grim et al., 2000). Note that the background probability distribution $F(x|\theta_0)$ can be reduced in the Bayes formula and therefore any decision-making may be confined to just the relevant variables. In particular, considering a finite set of classes $\omega \in \Omega$ with *a priori* probabilities $p(\omega)$ and denoting $\mathcal{M}_\omega$ the respective component index sets, we can express the corresponding class-conditional mixtures in the form:

$$P(x|\omega, w, \Theta, \Phi) = \sum_{m\in\mathcal{M}_\omega} w_m F(x|\theta_0) G(x|\theta_m, \phi_m).$$

In other words, the Bayes decision rule derives from a weighted sum of component functions $G(x|\theta_m, \phi_m)$ which can be defined on different subspaces.

$$\omega^* = d(x) = \arg\max_{\omega\in\Omega}\{p(\omega|x)\} = \qquad (20)$$

$$= \arg\max_{\omega\in\Omega}\{p(\omega) \sum_{m\in\mathcal{M}_\omega} w_m G(x|\theta_m, \phi_m)\}.$$

4 MIXTURES OF DEPENDENCE TREES

As mentioned earlier, the simplicity of product components may appear to be limiting in some cases and a natural way to generalize product mixtures is to use dependence-tree distributions as components (Grim, 1984; Meila and Jordan, 1998, 2001; Meila and Jaakkola, 2000; Kirshner and Smyth, 2007). Of course, marginal distributions of the dependence-tree mixtures are not trivially available anymore and we lose some of the excellent properties of product mixtures, especially the unique possibility of structural optimization of probabilistic neural networks. Nevertheless, in some cases such properties may be unnecessary, while the increased complexity of components could become essential.

The idea of the dependence-tree distribution refers to the known paper (Chow and Liu, 1968) who proposed approximation of multivariate discrete probability distribution $P^*(x)$ by the product distribution

$$P(x|\pi, \beta) = f(x_{i_1}) \prod_{n=2}^{N} f(x_{i_n}|x_{j_n}), \quad j_n \in \{i_1, \ldots, i_{n-1}\}.$$

Here $\pi = (i_1, i_2, \ldots, i_N)$ is a permutation of the index set $\mathcal{N}$ and β is the tree-dependence structure

$$\beta = \{(i_1, -), (i_2, j_2), \ldots, (i_N, j_N)\}, \quad j_n \in \{i_1, .., i_{n-1}\}.$$

In this paper we use a simplified notation of marginal distributions whenever tolerable, e.g.,

$$f(x_n) = f_n(x_n), \quad f(x_n|x_k) = f_{n|k}(x_n|x_k).$$

The above approximation model can be equivalently rewritten in the form

$$P(x|\alpha, \theta) = \left[\prod_{n=1}^{N} f(x_n)\right] \left[\prod_{n=2}^{N} \frac{f(x_n, x_{k_n})}{f(x_n) f(x_{k_n})}\right], \qquad (21)$$

because the first product is permutation-invariant and the second product can always be naturally ordered. Thus, in the last equation, the indices $(k_2, \ldots, k_N)$ briefly describe the ordered edges (n, k_n) of the underlying spanning tree β and we can write

$$P(x|\alpha, \theta) = f(x_1) \prod_{n=2}^{N} f(x_n|x_{k_n}). \qquad (22)$$

Here $\alpha = (k_2, \ldots, k_N)$ describes the dependence structure and $\theta = \{f(x_n, x_{k_n}), n = 2, \ldots, N\}$ stands for the related set of two-dimensional marginals. Note that all univariate marginals can uniquely be derived from the bivariate ones.

The dependence-tree mixtures can be optimized by means of EM algorithm in full generality, as shown in the paper (Grim, 1984). Later, the concept of dependence-tree mixtures has been reinvented by (Meila and Jordan, 1998, 2001; Meila and Jaakkola, 2000). Considering binary variables $x_n \in \{0,1\}$ we denote by $P(x|w,\alpha,\Theta)$ a mixture of dependence-tree distributions

$$P(x|w,\alpha,\Theta) = \sum_{m\in\mathcal{M}} w_m F(x|\alpha_m,\theta_m), \quad x \in X,$$

$$F(x|\alpha_m,\theta_m) = f(x_1|m)\prod_{n=2}^{N} f(x_n|x_{k_n},m) \tag{23}$$

with the weight vector w, the two-dimensional marginals $\theta_m = \{f(x_n,x_{k_n}|m), n = 2,\dots,N\}$ and the underlying dependence structures α_m

$$\alpha = \{\alpha_1,\alpha_2,\dots,\alpha_M\}, \quad \Theta = \{\theta_1,\theta_2,\dots,\theta_M\}.$$

The related log-likelihood function can be expressed by the formula

$$L(w,\alpha,\Theta) = \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}}\log\Big[\sum_{m\in\mathcal{M}} w_m F(x|\alpha_m,\theta_m)\Big]. \tag{24}$$

In view of Eq. (6), the EM algorithm reduces the optimization problem to the iterative maximization of the following weighted log-likelihood criteria $Q_m, m \in \mathcal{M}$ with respect to θ_m and α_m:

$$Q_m(\alpha_m,\theta_m) = \sum_{x\in\mathcal{S}}\frac{q(m|x)}{w'_m|\mathcal{S}|}\log F(x|\alpha_m,\theta_m) = \tag{25}$$

$$= \sum_{x\in\mathcal{S}}\frac{q(m|x)}{w'_m|\mathcal{S}|}\Big[\log f(x_1|m) + \sum_{n=2}^{N}\log f(x_n|x_{k_n},m)\Big].$$

By using usual δ-function notation we can write

$$Q_m(\alpha_m,\theta_m) = \sum_{x\in\mathcal{S}}\frac{q(m|x)}{w'_m|\mathcal{S}|}\Big[\sum_{\xi_1=0}^{1}\delta(\xi_1,x_1)\log f(\xi_1|m) +$$

$$+\sum_{n=2}^{N}\sum_{\xi_n=0}^{1}\sum_{\xi_{k_n}=0}^{1}\delta(\xi_n,x_n)\delta(\xi_{k_n},x_{k_n})\log f(\xi_n|\xi_{k_n})\Big] \tag{26}$$

and denoting

$$\hat{f}(\xi_n|m) = \sum_{x\in\mathcal{S}}\frac{q(m|x)}{w'_m|\mathcal{S}|}\delta(\xi_n,x_n), \quad n \in \mathcal{N},$$

$$\hat{f}(\xi_n,\xi_{k_n}|m) = \sum_{x\in\mathcal{S}}\frac{q(m|x)}{w'_m|\mathcal{S}|}\delta(\xi_n,x_n)\delta(\xi_{k_n},x_{k_n}),$$

we obtain:

$$Q_m(\alpha_m,\theta_m) = \sum_{\xi_1=0}^{1}\hat{f}(\xi_1|m)\log f(\xi_1|m) + \tag{27}$$

$$+\sum_{n=2}^{N}\sum_{\xi_{k_n}=0}^{1}\hat{f}(\xi_{k_n}|m)\sum_{\xi_n=0}^{1}\frac{\hat{f}(\xi_n,\xi_{k_n}|m)}{\hat{f}(\xi_{k_n}|m)}\log f(\xi_n|\xi_{k_n},m).$$

Again, for any fixed dependence structure α_m, the last expression is maximized by the two-dimensional marginals $\theta'_m = \{f'(\xi_n,\xi_{k_n}|m), n = 2,\dots,N\}$:

$$f'(\xi_n|m) = \hat{f}(\xi_n|m), \quad f'(\xi_n|\xi_{k_n},m) = \frac{\hat{f}(\xi_n,\xi_{k_n}|m)}{\hat{f}(\xi_{k_n}|m)}. \tag{28}$$

Making substitutions (28) in (27) we can express the weighted log-likelihood criterion $Q_m(\alpha_m,\theta'_m)$ just as a function of the dependence structure α_m:

$$Q_m(\alpha_m,\theta'_m) = \sum_{n=1}^{N}\sum_{\xi_n=0}^{1} f'(\xi_n|m)\log f'(\xi_n|m) +$$

$$+\sum_{n=2}^{N}\sum_{\xi_n=0}^{1}\sum_{\xi_{k_n}=0}^{1} f'(\xi_n,\xi_{k_n}|m)\log\frac{f'(\xi_n,\xi_{k_n}|m)}{f'(\xi_n|m)f'(\xi_{k_n}|m)},$$

where the last expression is the Shannon formula for mutual statistical information between the variables x_n, x_{k_n} (Vajda, 1989). Denoting

$$I(f'_{n|m}, f'_{k_n|m}) = \tag{29}$$

$$= \sum_{\xi_n=0}^{1}\sum_{\xi_{k_n}=0}^{1} f'(\xi_n,\xi_{k_n}|m)\log\frac{f'(\xi_n,\xi_{k_n}|m)}{f'(\xi_n|m)f'(\xi_{k_n}|m)}$$

we can write

$$Q_m(\alpha_m,\theta'_m) = \sum_{n=1}^{N} -H(f'_{n|m}) + \sum_{n=2}^{N} I(f'_{n|m}, f'_{k_n|m}).$$

In the last equation, the sum of entropies is structure-independent and therefore the weighted log-likelihood $Q_m(\alpha_m,\theta'_m)$ is maximized by means of the second sum, in terms of the dependence structure α_m.

The resulting EM iteration equations for mixtures of dependence-tree distributions can be summarized as follows (cf. (Grim, 1984), Eqs. (4.17)-(4.20)):

$$q(m|x) = \frac{w_m F(x|\alpha_m,\theta_m)}{\sum_{j\in\mathcal{M}} w_j F(x|\alpha_j,\theta_j)}, \quad w'_m = \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}} q(m|x), \tag{30}$$

$$f'(\xi_n|m) = \sum_{x\in\mathcal{S}}\frac{q(m|x)}{w'_m|\mathcal{S}|}\delta(\xi_n,x_n), \quad n \in \mathcal{N}, \tag{31}$$

$$f'(\xi_n,\xi_{k_n}|m) = \sum_{x\in\mathcal{S}}\frac{q(m|x)}{w'_m|\mathcal{S}|}\delta(\xi_n,x_n)\delta(\xi_{k_n},x_{k_n}), \tag{32}$$

$$\alpha'_m = \arg\max_{\alpha}\Big\{\sum_{n=2}^{N} I(f'_{n|m}, f'_{k_n|m})\Big\}. \tag{33}$$

The optimal dependence structure α'_m can be found by constructing the maximum-weight spanning tree of the related complete graph with the edge weights $I(f'_{n|m}, f'_{k|m})$ (Chow and Liu, 1968). For this purpose we can use the algorithm of Kruskal (cf. (Kruskal, 1956)) but the algorithm of Prim (Prim, 1957) is computationally more efficient since the ordering of all edge-weights is not necessary (cf. APPENDIX).

Table 1: Recognition of numerals from the NIST SD19 database by mixtures with different number of product components. In the third row the number of parameters denotes the total number of component specific parameters $\theta_n^{(m)}$.

Experiment No.	I	II	III	IV	V	VI	VII	VIII	IX
Components	10	40	100	299	858	1288	1370	1459	1571
Parameters	10240	38758	89973	290442	696537	1131246	1247156	1274099	1462373
Classif. error in %	**11.93**	**4.81**	**4.28**	**2.93**	**2.40**	**1.95**	**1.91**	**1.86**	**1.84**

Table 2: Recognition of numerals from the NIST SD19 database by mixtures with different number of dependence trees. The dependence-tree mixtures achieve only slightly better recognition accuracy with comparable number of parameters.

Experiment No.	I	II	III	IV	V	VI	VII	VIII	IX
Components	10	40	80	100	150	200	300	400	500
Parameters	20480	81920	163840	204800	307200	409600	614400	819200	1024000
Classif. error in %	**6.69**	**4.13**	**2.86**	**2.64**	**2.53**	**2.22**	**2.13**	**1.97**	**2.01**

5 RECOGNITION OF NUMERALS

In recent years we have repeatedly applied multivariate Bernoulli mixtures to recognition of hand-written numerals from the NIST benchmark database, with the aim to verify different decision-making aspects of probabilistic neural networks (cf. (Grim, 2007; Grim and Hora, 2008)). In this paper we use the same data to compare performance of the product (Bernoulli) mixtures and mixtures of dependence trees. We assume that the underlying 45 binary (two class) subproblems may reveal even very subtle differences between the classifiers. Moreover, the relatively stable graphical structure of numerals should be advantageous from the point of view of dependence-tree mixtures.

The considered NIST Special Database 19 (SD19) contains about 400000 handwritten numerals in binary raster representation (about 40000 for each numeral). We normalized all digit patterns to a 32×32 binary raster to obtain 1024-dimensional binary data vectors. In order to guarantee the same statistical properties of the training- and test data sets, we have used the odd samples of each class for training and the even samples for testing. Also, to increase the variability of the binary patterns, we extended both the training- and test data sets four times by making three differently rotated variants of each pattern (by -4, -2 and +2 degrees). Thus we have obtained for each class 80 000 data vectors both for training and testing.

In order to make the classification test we estimated for all ten numerals the class-conditional distributions by using Bernoulli mixtures in the subspace modification (14) and also by using dependence-tree mixtures. Recall that we need 2048 parameters to define each component of the dependence-tree distribution (23). The marginal probabilities of dependence-tree components displayed in raster arrangement (cf. Fig.1) correspond to the typical variants of the training numerals. Simultaneously, the figure shows the corresponding maximum-weight spanning tree α_m. Note that the superimposed optimal dependence structure naturally "reveals" how the numerals have been written because the "successive" raster points are strongly correlated.

For the sake of comparison we used the best solutions obtained in a series of experiments - both for the product mixtures and for the dependence-tree mixtures. The independent test patterns were classified by means of Bayes decision function (20). Each test numeral was classified by using mean Bayes probabilities obtained with the four differently rotated variants. Table 1 shows the classification error as a function of model complexity. Number of parameters in the third row denotes the total number of component-specific parameters $\theta_n^{(m)}$ (for which $\phi_n^{(m)} = 1$). Similar to Table 1 we can see in Table 2 the classification error as a function of model complexity, now represented by different numbers of dependence-tree components.

The detailed classification results for the best solutions are described by the error matrix in Table 3 (ten class-conditional mixtures with the total number of M=1571 product components including 1462373 parameters) and Table 4 (ten mixtures with total number of M=400 dependence tree components including 819200 parameters). As it can be seen the global recognition accuracy (right lower corner) is comparable in both cases. Note that in both tables the detailed

Table 3: Classification error matrix obtained by means of multivariate Bernoulli mixtures (the total number of components M=1571, number of parameters: 1462373). The last column contains the percentage of false negative decisions. The last row contains the total frequencies of false positive rates in percent of the respective class test patterns with the global error rate in bold.

CLASS	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	false n.
0	19950	8	43	19	39	32	36	0	38	17	1.1 %
1	2	22162	30	4	35	7	18	56	32	6	0.9 %
2	32	37	19742	43	30	9	8	29	90	16	1.5 %
3	20	17	62	20021	4	137	2	28	210	55	2.6 %
4	11	6	19	1	19170	11	31	51	30	247	2.1 %
5	25	11	9	154	4	17925	39	6	96	34	2.1 %
6	63	10	17	6	23	140	19652	1	54	3	1.6 %
7	7	12	73	10	73	4	0	20497	22	249	2.1 %
8	22	25	53	97	30	100	11	11	19369	72	2.1 %
9	15	13	25	62	114	22	3	146	93	19274	2.5 %
false p.	0.9%	0.7%	2.7%	2.0%	1.7%	2.3%	0.7%	1.6%	3.3%	3.5%	**1.84%**

Table 4: Classification error matrix obtained by means of dependence-tree mixtures (number of components M=400, number of parameters: 819200). The last column contains percentage of false negative decisions. The last row contains false positive rates in percent of the respective class test patterns with the global error rate in bold.

CLASS	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	false n.
0	19979	11	62	21	18	26	25	2	28	10	1.0 %
1	5	21981	78	13	74	1	20	155	21	4	1.7 %
2	22	15	19777	72	26	5	6	35	72	6	1.3 %
3	20	10	66	20169	1	120	1	20	122	27	1.9 %
4	12	16	13	4	19245	1	13	52	44	177	1.7 %
5	25	5	15	157	8	17874	45	9	129	36	2.3 %
6	100	19	38	25	43	90	19575	1	75	3	2.0 %
7	17	33	108	24	71	0	0	20367	28	299	2.8 %
8	18	30	47	167	27	55	22	17	19337	70	2.3 %
9	12	20	62	74	89	33	3	144	134	19196	2.9 %
false p.	1.4%	0.7%	2.4%	2.7%	1.8%	1.8%	0.7%	1.6%	3.1%	3.2%	**1.97%**

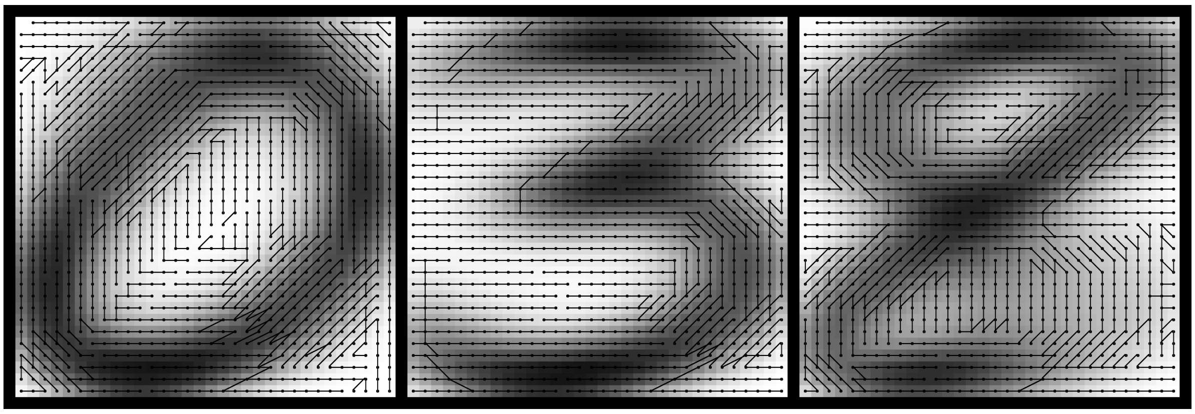
Figure 1: Mixture of dependence trees for binary data - examples of marginal component probabilities in raster arrangement. The superimposed optimal dependence structure (defined by maximum-weight spanning tree) reflects the way the respective numerals have been written.

frequencies of false negative and false positive decisions are also comparable.

Roughly speaking, the dependence-tree mixtures achieve only slightly better recognition accuracy with

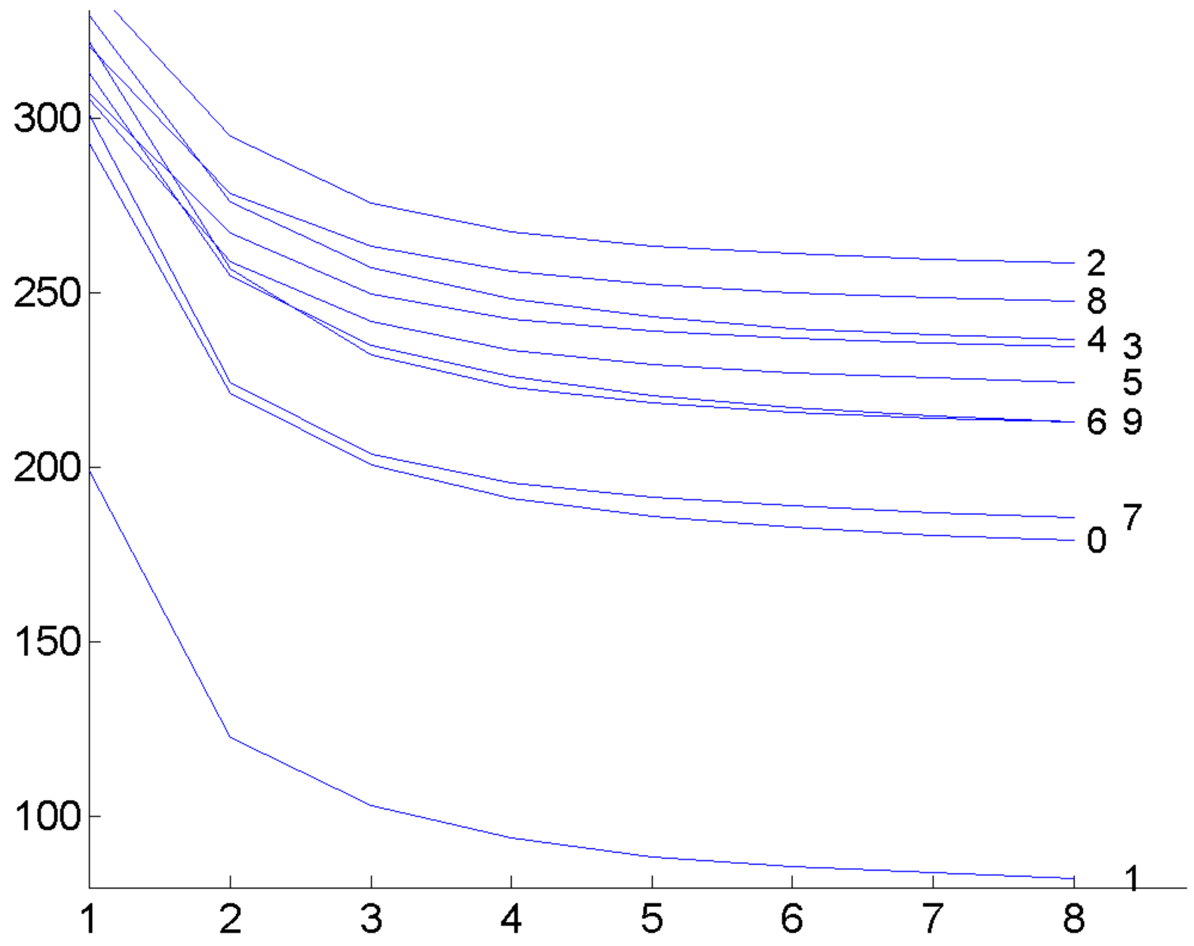

Figure 2: The decreasing information contribution of the dependence structure to the estimated dependence-tree mixtures (the first eight iterations of the ten estimated class-conditional distributions). The EM algorithm tends to suppress the information contribution of the dependence structures to the optimal estimate.

a comparable number of parameters, but the most complex model (M=500) already seems to overfit. Expectedly, the dependence tree mixtures needed much less components for the best performance but they have stronger tendency to overfitting. The best recognition accuracy in Table 1 (cf. col. IX) well illustrates the power of the subspace product mixtures.

The most surprising result of the numerical experiments is the decreasing importance of the component dependence structure during the EM estimation process. We have noticed that the cumulative weight of all dependence trees expressed by the weighted sum

$$\Sigma' = \sum_{m\in\mathcal{M}} w'_m \sum_{n=2}^{N} I(f'_{n|m}, f'_{k_n|m})\} \qquad (34)$$

is decreasing in the course of EM iterations (cf. Fig.2). In other words the optimal estimate of the dependence tree mixture tends to suppress the information contribution of the dependence structures in components, i.e. the component dependence trees tend to degenerate to simple products. Nevertheless, this observation is probably typical only for mixtures having a large number of components since a single product component is clearly more restrictive than a single dependence tree.

6 CONCLUSIONS

We compare the computational properties of mixtures of product components and mixtures of dependence trees in application to recognition of numerals from the NIST Special Database 19. The underlying classification problem involves separation of 45 pairs of classes and therefore the related classification errors should reveal even small differences between the two considered classifiers. For the sake of comparison we have used for each of the considered mixture models the best solution obtained in series of experiments.

The detailed description of the classification performance (cf. Table 3, Table 4) shows that the recognition accuracy of both models is comparable. It appears that, in our case, the dependence structure of components does not improve the approximation power of the product mixture essentially and, moreover, the information contribution of the dependence structure decreases in the course of EM iterations as shown in Fig. 2. Thus, the optimal estimate of the dependence tree mixture tends to approach a simple product mixture model. However, this observation is probably related to the large number of components only.

We assume that the dependence tree distribution is advantageous if we try to fit a small number of components to a complex data set. However, in case of a large number of multidimensional components the component functions are almost non-overlapping (Grim and Hora, 2010), the structural parameters tend to fit to small compact subsets of data and the structurally modified form of the components is less important. We can summarize the properties of dependence tree mixtures as follows:

In Case of a Large Number of Components:

- intuitively, the large number of components is the main source of the resulting approximation power
- dependence structure of components does not improve the approximation power of product mixtures essentially
- the total information contribution of the component dependence structures decreases in the course of EM iterations
- the optimal estimate of the dependence tree mixture tends to approach a simple product mixture model

In Case of a Small Number of Components:

- a single dependence tree component is capable to describe the statistical relations between pairs variables
- consequently, the approximation power of a single dependence tree component is much higher than that of a product component
- information contribution of the dependence structure can increase in the course of EM iterations
- dependence structure of components can essentially improve the approximation quality

In this sense, the computational properties of dependence tree mixtures provide an additional argument to prefer the product mixture models in case of large multidimensional data sets.

From the point of view of neural networks and regardless of the computational aspects, the concept of dependence tree distribution could help to clarify the role of dendritic ramification in the highly selective excitability of neurons. The output of formal neurons is usually defined by thresholding the activation sum of weighted synaptic inputs. Unlike this formal summation model assuming statistically independent inputs the thin dendritic branches of biological neurons may be depolarized by weak signals which can facilitate the conditional activation of the neuron as a whole. In other words, it is assumed that easily excitable thin dendritic branches may facilitate the propagation of excitation to neural body. The effect of facilitation could be modeled by the conditional distributions of the dependence-tree components.

ACKNOWLEDGEMENTS

This work was supported by the Czech Science Foundation Projects No. 14-02652S and P403/12/1557.

REFERENCES

Boruvka, O. (1926). On a minimal problem, *Transaction of the Moravian Society for Natural Sciences* (in czech), No. 3.

Bouguila, N., Ziou, D. and Vaillancourt, J. (2004). Unsupervised learning of a finite mixture model based on the Dirichlet distribution and its application. *IEEE Trans. on Image Processing,* Vol. 13, No. 11, pp. 1533-1543.

Chow, C. and Liu, C. (1968). Approximating discrete probability distributions with dependence trees, *IEEE Trans. on Information Theory,* Vol. IT-14, No.3, pp. 462- 467.

Dempster, A.P., Laird, N.M. and Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Statist. Soc., B,* Vol. 39, pp. l-38.

Day, N.E. (1969). Estimating the components of a mixture of normal distributions, *Biometrika,* Vol. 56, pp. 463-474.

Grim, J. (1982). On numerical evaluation of maximum - likelihood estimates for finite mixtures of distributions, *Kybernetika,* Vol.l8, No.3, pp.173-190. *http://dml.cz/dmlcz/124132*

Grim, J. (1984). On structural approximating multivariate discrete probability distributions, *Kybernetika,* Vol. 20, No. 1, pp. 1-17. *http://dml.cz/dmlcz/125676*

Grim, J. (1986). Multivariate statistical pattern recognition with nonreduced dimensionality, *Kybernetika,* Vol. 22, No. 2, pp. 142-157. *http://dml.cz/dmlcz/125022*

Grim, J. (1996). Design of multilayer neural networks by information preserving transforms. In *Third European Congress on Systems Science.* (Eds. Pessa E., Penna

M. P., Montesanto A.). (Edizioni Kappa, Roma 1996) 977–982.

Grim, J. (1999). Information approach to structural optimization of probabilistic neural networks, In *Proc. 4th System Science European Congress*, Eds. Ferrer, L. et al., Valencia: Soc. Espanola de Sistemas Generales, pp. 527-540.

Grim, J. (1999b). A sequential modification of EM algorithm, In *Studies in Classification, Data Analysis and Knowledge Organization*, Eds. Gaul W., Locarek-Junge H., Springer 1999, pp. 163 - 170.

Grim, J. (2006). EM cluster analysis for categorical data, In *Structural, Syntactic and Statistical Pattern Recognition.* Eds. Yeung D. Y., Kwok J. T., Fred A., Springer: Berlin, LNCS 4109, pp. 640-648.

Grim, J. (2007). Neuromorphic features of probabilistic neural networks. *Kybernetika*, Vol. 43, No. 5, pp.697-712. *http://dml.cz/dmlcz/135807*

Grim, J. (2014). Sequential pattern recognition by maximum conditional informativity, *Pattern Recognition Letters*, Vol. 45C, pp. 39-45. *http:// dx.doi.org/10.1016/j.patrec.2014.02.024*

Grim, J., Haindl, M., Somol, P. and P. Pudil (2006). A subspace approach to texture modelling by using Gaussian mixtures, In *Proceedings of the 18th IAPR International Conference on Pattern Recognition ICPR 2006*, Eds. B. Haralick, T.K. Ho, Los Alamitos, IEEE Computer Society, pp. 235-238.

Grim, J. and Hora, J. (2008). Iterative principles of recognition in probabilistic neural networks, *Neural Networks*. Vol. 21, No. 6, pp. 838-846.

Grim, J. and Hora, J. (2009). Recognition of Properties by Probabilistic Neural Networks, In *Artificial Neural Networks - ICANN 2009*, Springer: Berlin, LNCS 5769, pp. 165-174.

Grim, J. and Hora, J. (2010). Computational Properties of Probabilistic Neural Networks, In *Artificial Neural Networks - ICANN 2010 Part II*, Springer: Berlin, LNCS 5164, pp. 52-61.

Grim, J., Hora, J., Boček P., Somol, P. and Pudil, P. (2010). Statistical Model of the 2001 Czech Census for Interactive Presentation, *Journal of Official Statistics*. Vol. 26, No. 4, pp. 673694. *http://ro.utia.cas.cz/dem.html*

Grim, J., Kittler, J., Pudil, P. and Somol, P. (2002). Multiple classifier fusion in probabilistic neural networks, *Pattern Analysis and Applications*, Vol. 5, No. 7, pp. 221-233.

Grim, J., Pudil, P. and Somol, P. (2000). Recognition of handwritten numerals by structural probabilistic neural networks, In *Proceedings of the Second ICSC Symposium on Neural Computation*, Berlin, 2000. (Bothe H., Rojas R. eds.). ICSC, Wetaskiwin, pp. 528-534.

Grim, J., Pudil, P. and Somol, P. (2002b). Boosting in probabilistic neural networks, In *Proceedings of the 16th International Conference on Pattern Recognition*, (Kasturi R., Laurendeau D., Suen C. eds.). IEEE Computer Society, Los Alamitos, pp. 136–139.

Grim, J., Somol, P., Haindl, M. and Daneš, J. (2009). Computer-Aided Evaluation of Screening Mammograms Based on Local Texture Models, *IEEE Trans. on Image Processing*, Vol. 18, No. 4, pp. 765-773.

Hasselblad, V. (1966). Estimation of prameters for a mixture of normal distributions, *Technometrics*, Vol. 8, pp. 431-444.

Hasselblad, V. (1969). Estimation of finite mixtures of distributions from the exponential family, *Journal of Amer. Statist. Assoc.*, Vol. 58, pp. 1459-1471.

Hebb, D.O. (1949). *The Organization of Behavior: A Neuropsychological Theory*, (New York: Wiley 1949).

Hosmer Jr, D.W. (1973). A comparison of iterative maximum likelihood estimates of the parameters of a mixture of two normal distributions under three different types of sample, *Biometrics*, pp. 761-770.

Kirshner, S. and Smyth, P. (2007). Infinite mixtures of trees, In *Proceedings of the 24th International Conference on Machine Learning (ICML'07)*, Ed. Zoubin Ghahramani, ACM, New York, USA, pp. 417-423.

Kruskal, J.B. (1956). On the shortest spanning sub-tree of a graph, *Proc. Amer. Math. Soc.*, No. 7, pp. 48-50.

Kullback, S. and Leibler, R.A. (1951). On Information and Sufficiency, *The Annals of Mathematical Statistics*, Vol. 22, No. 1, pp. 79-86.

Lowd, D. and Domingos, P. (2005). Naive Bayes models for probability estimation, In *Proceedings of the 22nd international conference on machine learning*, ACM 2005, pp. 529-536.

Markley, S.C. and Miller, D.J. (2010). Joint parsimonious modeling and model order selection for multivariate Gaussian mixtures, *IEEE Journal of Selected Topics in Signal Processing*, Vol. 4, No. 3, pp. 548-559.

Meila, M. and Jordan, M.I. (1998). Estimating dependency structure as a hidden variable, In *Proceedings of the 1997 Conference on advances in neural information processing systems 10*, pp. 584-590.

Meila, M. and Jaakkola T. (2000). Tractable Bayesian Learning of Tree Belief Networks, In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pp. 380-388.

Meila, M. and Jordan, M.I. (2001). Learning with mixtures of trees, *Journal of Machine Learning Research*, Vol. 1, No. 9, pp. 1-48.

Prim, R.C. (1957). Shortest connection networks and some generalizations, *Bell System Tech. J.*, Vol. 36 , pp. 1389-1401.

Schlesinger, M.I. (1968). Relation between learning and self learning in pattern recognition, (in Russian), *Kibernetika*, (Kiev), No. 2, pp. 81-88.

Vajda, I. *Theory of statistical inference and information*, Kluwer Academic Publishers (Dordrecht and Boston), 1989.

Wolfe, J.H. (1970). Pattern clustering by multivariate mixture analysis, *Multivariate Behavioral Research*, Vol. 5, pp. 329-350.

APPENDIX

Maximum-weight Spanning Tree

The algorithm of Boruvka-Kruskal (cf. (Kruskal, 1956), (Chow and Liu, 1968)) assumes ordering of all $N(N-1)/2$ edge weights in descending order. The maximum-weight spanning tree is then constructed sequentially, starting with the first two (heaviest) edges. The next edges are added sequentially in descending order if they do not form a cycle with the previously chosen edges. Multiple solutions are possible if several edge weights are equal, but they are ignored as having the same maximum weight.

The algorithm of Prim (Prim, 1957) does not need any ordering of edge weights. We start from any variable by choosing the neighbor with the maximum edge weight. This first edge of the maximum-weight spanning tree is then sequentially extended by adding the maximum-weight neighbors of the currently chosen subtree. Again, any ties may be decided arbitrarily since we are not interested in multiple solutions.

Both Kruskal and Prim refer to an “obscure Czech paper” of Otakar Boruvka from the year 1926 giving an alternative construction of the minimum-weight spanning tree and the corresponding proof of uniqueness (Boruvka, 1926). The algorithm of Prim can be summarized as follows (in C-pseudo-code).

It can be seen that in case of dependence-tree mixtures with many components the application of the algorithm of Kruskal (cf. (Kruskal, 1956)) may become prohibitive in high-dimensional spaces because the repeated ordering of the edge-weights for all components is time-consuming (cf. (Meila and Jordan, 1998, 2001; Meila and Jaakkola, 2000)).

```
//**********************************************
//  Maximum-weight spanning tree construction
//**********************************************
//
//  NN........ number of nodes, N=1,2,...,NN
//  T[N]...... characteristic function of the
//             defined part of spanning tree
//  E[N][K]... positive weight of the edge <N,K>
//  A[K]...... index of the heaviest neighbor
//             of node K in the defined subtree
//  GE[K]..... greatest edge weight between the
//             node K and the defined subtree
//  K0........ index of the most heavy neighbor
//             of the defined part of tree
//  SUM....... total weight of the spanning tree
//  spanning tree: {<2,A[2]>,...,<NN,A[NN]>}
//**********************************************
    for(N=1; N<=NN; N++)   // initial values
    {
       GE[N]=-1;  T[N]=0;  A[N]=0;
    }  //  end of N-loop
    N0=1; T[N0]=1; K0=0;
    //****************************************
    for(I=2; I<=NN; I++)  // spanning tree loop
    {
       FMAX=-1E0;
       for(N=2; N<=NN; N++)
       if(T[N]<1)
         {
            F=E[N0][N];
            if(F>GE[N]) {GE[N]=F; A[N]=N0;
         }
       else F=GE[N];
       if(F>FMAX)  {FMAX=F; K0=N;}
    }  //  end of N-loop
       N0=K0; T[N0]=1;
       SUM+=FMAX;
    }  // end of I-loop
//**********************************************
//  end of spanning tree construction
//**********************************************
```