The Most Generative Maximum Margin Bayesian Networks

Robert Peharz* ROBERT.PEHARZ@TUGRAZ.AT
Sebastian Tschiatschek* TSCHIATSCHEK@TUGRAZ.AT
Franz Pernkopf PERNKOPF@TUGRAZ.AT
Signal Processing and Speech Communication Laboratory, Graz University of Technology
*These authors contributed equally to this paper

Abstract

Although discriminative learning in graphical models generally improves classification results, the generative semantics of the model are compromised. In this paper, we introduce a novel approach of hybrid generative-discriminative learning for Bayesian networks. We use an SVM-type large margin formulation for discriminative training, introducing a likelihood-weighted ℓ^1-norm for the SVM-norm-penalization. This simultaneously optimizes the data likelihood and therefore partly maintains the generative character of the model. For many network structures, our method can be formulated as a convex problem, guaranteeing a globally optimal solution. In terms of classification, the resulting models outperform state-of-the art generative and discriminative learning methods for Bayesian networks, and are comparable with linear and kernelized SVMs. Furthermore, the models achieve likelihoods close to the maximum likelihood solution and show robust behavior in classification experiments with missing features.

1. Introduction

In machine learning, there are two primary approaches: generative and discriminative learning. In generative learning, the aim is to estimate an underlying and unknown probability distribution from data. Therefore, generative models represent *probability distributions* and the objective is some form of likelihood. In discriminative learning, the aim is to find a representation of a *function* for mapping features to targets. Here, the objectives are more versatile than in the generative case; dependent on the scenario, one aims to minimize some form of error, or maximize the conditional likelihood, some form of margin or the classification rate. When generative models do not capture the true distribution well, discriminative approaches tend to outperform their generative counterparts.

Bayesian networks (BNs) represent distributions and are therefore well-suited for generative learning. On the other hand, they also represent conditional distributions and classification functions, and can be trained also discriminatively (Friedman et al., 1997; Ng & Jordan, 2001; Wettig et al., 2003; Greiner et al., 2005; Guo et al., 2005; Sha, 2007; Pernkopf et al., 2012). When a BN is trained discriminatively, its generative semantics is abandoned, i.e. its interpretation as joint distribution. The BN is optimized to infer the class value from the features, while other inference tasks are implausible and yield poor results. However, a discriminative BN still represents some spurious marginal feature distribution, which does not fulfill any modeling purpose. Why should we then use a BN, when we are actually interested in the conditional distribution only? One reasonable ramification is to use models which explicitly model conditional distributions, but *not* the marginal feature distribution, such as conditional random fields (Lafferty et al., 2001). The motivation in this paper is different: Even when the conditional distribution obtained by discriminative training is unique, the *representation* as a BN might be not unique. A natural approach is to use this degree of freedom to improve the generative aspect of the model, i.e. to select the representation with highest likelihood. This describes a domain of *likelihood-aware* discriminative models, justifying a generative usage, such as *sampling new examples*, *versatile inference scenarios*, and *consistent treatment of missing features during test time*. A similar philosophy can be found in maximum entropy discrimination (MED) (Jebara, 2001) which combines discriminative estima-

Proceedings of the 30^{th} International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

tion with generative models in a principled way.

In this work, we consider a SVM-type maximum margin approach for BNs (Cortes & Vapnik, 1995; Guo et al., 2005; Pernkopf et al., 2012). We introduce a weighted ℓ^1-norm in the objective, where the weights correspond to the likelihood counts obtained from training data. The motivation for the weighted ℓ^1-norm is *not* that a better classifier is learned; literature provides several alternatives to the classical ℓ^2-norm SVMs (Zhu et al., 2004; Zou & Yuan, 2008) and no general preference can be assessed for any norm. We merely assume that the weighted ℓ^1-norm does typically *not perform worse* than any other norm regularizer. However, we show that for specific network structures the resulting network parameters are automatically normalized, which gives the weighted ℓ^1-norm the additional interpretation as *likelihood-term*. Therefore, we can interpret our model as a *likelihood-aware SVM*. When the SVM-trade-off parameter is zero, the solution of our formulation coincides with maximum likelihood parameters. When the parameter tends towards infinity, the sample-margins are emphasized. Our model is related to hybrid generative-discriminative models (Raina et al., 2003; Bouchard & Triggs, 2004; Bishop & Lasserre, 2007), but there is a substantial difference: Although the objective of our formulation is a trade-off between a likelihood term and a margin term, the objective is *not* a blend of a "generative" and a "discriminative" term. The margin term alone is *not* a discriminative objective, just as a standard SVM without norm-penalization has little discriminative meaning. Rather, the likelihood-term has to be viewed as norm-penalization, while the generative semantics are a desired *side-effect*.

We introduce our notation in Section 2. In Section 3, we present our formulation as convex optimization problem and state Theorems 1 and 2 which guarantee correctly normalized BN parameters, permitting the additional likelihood-interpretation. In Section 4, we propose a projected gradient method which is scalable to large datasets. In Section 5 we report results on benchmark datasets. Section 6 concludes the paper.

2. Background and Notation

Throughout the paper, we assume discrete random variables (RVs), where plain capital letters denote single RVs and capital boldface letters represent sets of RVs. Lower-case plain letters represent states of RVs and lower-case boldface letter represent joint states of variable sets. When $\mathbf{y}$ is a state of $\mathbf{Y}$, and $\mathbf{X} \subseteq \mathbf{Y}$, then $\mathbf{y}(\mathbf{X})$ denotes the corresponding state of $\mathbf{X}$. Furthermore, when $\mathbf{X}$ and $\mathbf{Y}$ are disjoint, then $[\mathbf{x}, \mathbf{y}]$ denotes a state of set $\mathbf{X} \cup \mathbf{Y}$. The set of states which can be assumed by RV X is denoted as $\mathbf{val}(X)$, and similarly we use $\mathbf{val}(\mathbf{X})$ for a set $\mathbf{X}$. For notational ease, we represent (unconditional) distributions as conditional distributions of the form $\mathrm{P}(\mathbf{X}) := \mathrm{P}(\mathbf{X}|\emptyset)$.

A Bayesian network (BN) is defined as a tuple $\mathcal{B} = (\mathcal{G}, \mathbf{P})$, containing an acyclic directed graph $\mathcal{G}$ and a set of conditional probability distributions $\mathbf{P}$. The nodes of $\mathcal{G}$ correspond to RVs $\mathbf{X} = \{X_0, \ldots, X_N\}$ and the edges describe direct dependencies among these RVs. For each node in $\mathcal{G}$, the set $\mathbf{P} = \{\mathrm{P}(X_0|\mathbf{Pa}_0), \ldots, \mathrm{P}(X_N|\mathbf{Pa}_N)\}$ contains conditional distributions, where $\mathbf{Pa}_i$ denotes the set of parents of X_i according to $\mathcal{G}$. Similarly, we define $\mathbf{Ch}_i$ as the set of children of X_i. A BN represents the joint distribution $\mathrm{P}^{\mathcal{B}}(\mathbf{X}) = \prod_{i=0}^{N} \mathrm{P}(X_i|\mathbf{Pa}_i)$. For discrete data, a general representation of $\mathbf{P}$ is a collection of conditional probability tables (CPTs) $\boldsymbol{\Theta} = \{\boldsymbol{\theta}^0, \ldots, \boldsymbol{\theta}^N\}$, with $\boldsymbol{\theta}^i = \{\theta^i_{j|\mathbf{h}} | j \in \mathbf{val}(X_i), \mathbf{h} \in \mathbf{val}(\mathbf{Pa}_i)\}$, where $\theta^i_{j|\mathbf{h}} := \mathrm{P}^{\mathcal{B}}(X_i = j|\mathbf{Pa}_i = \mathbf{h})$. The BN distribution can then be written as

$$\mathrm{P}^{\mathcal{B}}(\mathbf{X} = \mathbf{x}) = \prod_{i=0}^{N} \prod_{j \in \mathbf{val}(X_i)} \prod_{\mathbf{h} \in \mathbf{val}(\mathbf{Pa}_i)} {\theta^i_{j|\mathbf{h}}}^{\nu^i_{j|\mathbf{h}}}, \quad (1)$$

where $\nu^i_{j|\mathbf{h}}$ is the indicator function $\mathbb{1}(x_i = j \wedge \mathbf{x}(\mathbf{Pa}_i) = \mathbf{h})$. We represent the BN parameters in the log domain, defined as $\omega^i_{j|\mathbf{h}} := \log \theta^i_{j|\mathbf{h}}$, $\omega^i := \{\omega^i_{j|\mathbf{h}}\}$, and $\boldsymbol{\omega} := \{\omega^i\}$. Often, we will interpret $\boldsymbol{\omega}$ as a vector, whose elements are addressed as $\omega^i_{j|\mathbf{h}}$. We say that $\boldsymbol{\omega}$ is *sub-normalized*, iff

$$\log \sum_{j \in \mathbf{val}(X_i)} \exp(\omega^i_{j|\mathbf{h}}) \leq 0, \quad \forall X_i, \forall \mathbf{h} \in \mathbf{val}(\mathbf{Pa}_i), \quad (2)$$

and $\boldsymbol{\omega}$ is *normalized*, iff (2) holds with equality. A vector $\boldsymbol{\omega}$ is *strictly* sub-normalized, iff it is sub-normalized, but not normalized. In order to represent valid BN parameters, $\boldsymbol{\omega}$ has to be normalized. We define a vector-valued function $\phi(\mathbf{x})$ of the same length as $\boldsymbol{\omega}$, collecting $\nu^i_{j|\mathbf{h}}$, corresponding to the entries $\omega^i_{j|\mathbf{h}}$ in $\boldsymbol{\omega}$. In that way, we can express the log of (1) as $\log \mathrm{P}^{\mathcal{B}}(\mathbf{X} = \mathbf{x}) = \phi(\mathbf{x})^T \boldsymbol{\omega}$.

Assume that we have M i.i.d. samples $\{\mathbf{x}^1, \ldots, \mathbf{x}^M\}$, drawn from an unknown distribution $\mathrm{P}^*(\mathbf{X})$. For a fixed BN structure $\mathcal{G}$, the (smoothed) maximum likelihood (ML) parameters are given as

$$\hat{\omega}^i_{j|\mathbf{h}} = \log\left(\frac{n^i_{j|\mathbf{h}}}{n^i_{\mathbf{h}}}\right), \quad (3)$$

where

$$n^i_{j|\mathbf{h}} = \left(\sum_{m=1}^{M} \nu^{i,m}_{j|\mathbf{h}}\right) + \frac{\alpha}{|\mathbf{val}(X_i)|\,|\mathbf{val}(\mathbf{Pa}_i)|}, \quad (4)$$

$$n^i_{\mathbf{h}} = \sum_{j\in\mathbf{val}(X_i)} n^i_{j|\mathbf{h}}, \text{ and} \quad (5)$$

$$\nu^{i,m}_{j|\mathbf{h}} = \mathbb{1}(x^m_i = j \ \wedge\ \mathbf{x}^m(\mathbf{Pa}_i) = \mathbf{h}). \quad (6)$$

Here, $\alpha \geq 0$ is a smoothing parameter with the interpretation of a virtual sample count, which biases the ML estimates towards a uniform distribution. The normalization by $|\mathbf{val}(X_i)|\,|\mathbf{val}(\mathbf{Pa}_i)|$ achieves that the "virtual samples" are distributed consistently among the CPTs. We say that the likelihood-counts are *consistent*, when for all X_k, $j \in \mathbf{val}(X_k)$, $X_i \in \mathbf{Ch}_k$, and $\mathbf{h} \in \mathbf{val}(\mathbf{Pa}_k \cap \mathbf{Pa}_i)$ it holds that

$$\sum_{\mathbf{h}'\in\mathbf{val}(\mathbf{A})} n^k_{j|[\mathbf{h},\mathbf{h}']} = \sum_{\mathbf{h}''\in\mathbf{val}(\mathbf{B})} \sum_{j'\in\mathbf{val}(X_i)} n^i_{j'|[\mathbf{h},j,\mathbf{h}'']} \quad (7)$$

where $\mathbf{A} = \mathbf{Pa}_k \setminus \mathbf{Pa}_i$ and $\mathbf{B} = \mathbf{Pa}_i \setminus (\mathbf{Pa}_k \cup \{X_k\})$. For $\alpha > 0$, Equation (3) is also the MAP solution using Dirichlet priors according to (Buntine, 1991; Heckerman et al., 1995).

In this paper, we consider classification problems, where w.l.o.g. we assume that the class variable $C = X_0$ and the features are $\mathbf{Z} = \{X_1, \ldots, X_N\}$. Our discussion will concentrate on structures satisfying the following condition, as identified in (Wettig et al., 2003):

Condition 1. *Each child of the class-node has a covering parent.*

We call node Y a *covering parent* of node X iff Y is a parent of X and $\mathbf{Pa}(X) \subseteq \mathbf{Pa}(Y) \cup \{Y\}$. Structures satisfying Condition 1 are denoted as *C1-structures*. The class of these structures is quite rich, containing, amongst others, the naive Bayes (NB) structure, the tree-augmented naive Bayes (TAN) (Friedman et al., 1997), and diagnostic networks (Wettig et al., 2003). C1-structures facilitate discriminative learning, since for each unnormalized parameter vector there exists also a normalized parameter vector, specifying the *same conditional distribution* $\mathrm{P}^{\mathcal{B}}(C|\mathbf{Z})$. Wettig et al. (2003) provided a constructive proof, by proposing Algorithm 1 (shown in the Appendix) for normalizing a set of unnormalized BN parameters, while leaving $\mathrm{P}^{\mathcal{B}}(C|\mathbf{Z})$ unchanged. Condition 1 allows a convex relaxation of our optimization problem, presented in Section 3, i.e. a globally optimal solution can be obtained. However, in principle our methods can also be applied to arbitrary structures, by applying a normalization maintaining parameter transformation such as in (Pernkopf et al., 2012).

3. A "Generative" Maximum Margin Formulation

The probabilistic margin δ^m of the m^{th} sample is defined as (Guo et al., 2005; Pernkopf et al., 2012)

$$\delta^m = \frac{\mathrm{P}^{\mathcal{B}}(c^m|\mathbf{z}^m)}{\max\limits_{c\neq c^m} \mathrm{P}^{\mathcal{B}}(c|\mathbf{z}^m)} = \frac{\mathrm{P}^{\mathcal{B}}(c^m,\mathbf{z}^m)}{\max\limits_{c\neq c^m} \mathrm{P}^{\mathcal{B}}(c,\mathbf{z}^m)}. \quad (8)$$

Clearly, when $\delta^m > 1$, then the m^{th} sample is correctly classified, and when $\delta^m < 1$, it is wrongly classified. By defining $\phi_c(\mathbf{x}) := \phi([c, \mathbf{x}(\mathbf{Z})])$, we can express the log of (8) as $\log \delta^m = \min\limits_{c\neq c^m} \left[(\phi_{c^m}(\mathbf{x}^m) - \phi_c(\mathbf{x}^m))^T\, \boldsymbol{\omega}\right]$. When we interpret $\phi_c(\mathbf{x}^m)$ as (class-dependent) feature transformation, we can formulate the following multiclass SVM-type training for BNs (Cortes & Vapnik, 1995; Crammer & Singer, 2001; Guo et al., 2005):

$$\min_{\boldsymbol{\omega},\boldsymbol{\xi}} \quad \|\boldsymbol{\omega}\| + \lambda \sum_{m=1}^{M} \xi_m \quad (9)$$

$$\text{s.t.} \quad (\phi_{c^m}(\mathbf{x}^m) - \phi_c(\mathbf{x}^m))^T\, \boldsymbol{\omega} + \xi_m \geq 1 \quad \forall m, c \neq c^m$$

Here, $\|\boldsymbol{\omega}\|$ denotes some norm, $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_M)$ is a vector of margin slacks, and λ is a trade-off parameter, set by cross validation. We call formulation (9) the *BN-SVM*. In general, a solution of the BN-SVM will not be normalized, i.e. typically $\log \sum_{j'} \exp(\omega^i_{j'|\mathbf{h}}) \neq 0$, for some i, $\mathbf{h}$. However, since we consider C1-structures, we can simply apply Algorithm 1 (see Appendix), and obtain valid BN parameters, with the same class conditional distribution (i.e. the same classifier) as the unnormalized, *optimal* solution.

Although this approach allows to marry SVM-type training with BNs, the following questions naturally rise: Why should we even care about renormalized parameters, corresponding to the same classifier as the solution of (9)? Why should we use a BN at all, when, by training it like an SVM, we abandon any probabilistic interpretation? The answer we give here, is that discriminative training in BNs can be meaningful, when we (partly) maintain a generative interpretation. To this end, we modify (9) and use the following weighted ℓ^1-norm for the BN-SVM norm term: $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}) = \sum_{i,j,\mathbf{h}} |n^i_{j|\mathbf{h}}\, \omega^i_{j|\mathbf{h}}|$. Here, the weights $n^i_{j|\mathbf{h}}$ are the likelihood-counts according to (4), collected in a vector $\mathbf{n}$. Furthermore, we subject the vector $\boldsymbol{\omega}$ to sub-normalization constraints (2). These constraints restrict the parameters to a smooth approximation of the negative orthant, but do not severely restrict the solution space, since an arbitrary constant can be added to a solution vector $\boldsymbol{\omega}$, yielding

the same classifier. However, for the BN-SVM according to (9), we are allowed to arbitrarily assume a function margin of 1, since an optimal solution vector simply scales with this value. By introducing the sub-normalization constraints, this does not hold true any more. Therefore, we introduce a model parameter γ for the function margin, which is set by cross-validation. Since constraints (2) imply $\omega^i_{j|\mathbf{h}} \leq 0$, we can re-write $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}) = -\sum_{i,j,\mathbf{h}} n^i_{j|\mathbf{h}}\, \omega^i_{j|\mathbf{h}} = -\mathbf{n}^T\boldsymbol{\omega}$. Finally, we get the modified convex problem:

$$\begin{aligned}
\min_{\boldsymbol{\omega},\boldsymbol{\xi}} \quad & -\mathbf{n}^T\boldsymbol{\omega} + \lambda \sum_{m=1}^{M} \xi_m && (10)\\
\text{s.t.} \quad & (\phi_{c^m}(\mathbf{x}^m) - \phi_c(\mathbf{x}^m))^T\boldsymbol{\omega} + \xi_m \geq \gamma \quad \forall m, c \neq c^m\\
& \log \sum_{j'} \exp\left(\omega^i_{j'|\mathbf{h}}\right) \leq 0 \qquad \begin{array}{l}\forall\, 0 \leq i \leq N\\ \forall\, \mathbf{h} \in \mathbf{val}(\mathbf{Pa}_i)\end{array}\\
& \xi_m \geq 0 \qquad\qquad \forall m
\end{aligned}$$

Our first interpretation of (10) is that of a special instance of an BN-SVM, with (exotic) weighted ℓ^1-norm term $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega})$ and an arbitrary (but not limiting) sub-normalization constraint on the solution vector. On the other hand, $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}) = -\mathbf{n}^T\boldsymbol{\omega}$ is formally the *negative log-likelihood* of $\boldsymbol{\omega}$. Therefore, although (10) is a *discriminative formulation*, we see that as a *side effect*, it aims to *maximize the data likelihood*. However, there is still a major problem about this generative interpretation: the solution vector $\boldsymbol{\omega}$ might be *strictly* sub-normalized. In this case, $\boldsymbol{\omega}$ does not represent valid BN parameters, and strictly speaking, $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega})$ can not be interpreted as negative log-likelihood. When Algorithm 1 is applied to obtain normalized parameters, the discriminative character is left unchanged. But how does the *generative character* change under Algorithm 1? Fortunately, as shown in Lemma 1, for C1-structures the log-likelihood can *only increase* when Algorithm 1 is applied to sub-normalized parameters. The proofs for Lemma 1 and Theorems 1 and 2 can be found in the Appendix.

Lemma 1. *Let $\mathcal{G}$ be a C1-structure, $\tilde{\boldsymbol{\omega}}$ be a sub-normalized parameter-vector for $\mathcal{G}$, and $\mathbf{n}$ be a non-negative vector of consistent likelihood-counts. Then the likelihood is non-decreasing under Algorithm 1, i.e. when $\boldsymbol{\omega}$ is the output of Algorithm 1 for input $\mathcal{G}$, $\tilde{\boldsymbol{\omega}}$, then $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}) \leq \mathrm{nL}_{\mathbf{n}}(\tilde{\boldsymbol{\omega}})$.*

Using Lemma 1, it is easy to show that (10) always has a normalized solution, as stated in Theorem 1.

Theorem 1. *Let $\mathcal{G}$ be a C1-structure, $\{\mathbf{x}^1, \ldots, \mathbf{x}^M\}$ be an arbitrary data set, and $\mathbf{n}$ be an element-wise non-negative vector of consistent likelihood-counts. Then problem (10) (for $\lambda \geq 0$) always has an optimal solution $\boldsymbol{\omega}$, $\boldsymbol{\xi}$, such that $\boldsymbol{\omega}$ is normalized.*

Furthermore, for positive likelihood-counts (e.g. when $\alpha > 0$ in (4)), the solution is *unique* and *normalized*.

Theorem 2. *Assume the same conditions as in Theorem 1, but where $\mathbf{n}$ is element-wise positive. Then problem (10) has a unique, normalized solution.*

Lemma 1 and Theorems 1 and 2 show that for C1-structures, we can *always interpret* $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega})$ as negative log-likelihood. Due to this generative interpretation, we call formulation (10) the *maximum-likelihood* BN-SVM (ML-BN-SVM). Problem (10) is convex and can be addressed by standard solvers. However, this restricts learning to medium sized data sets. In the following section we describe an optimization method which scales better to large datasets.

4. Optimization for Large-Scale Data

The main limitation in (10) is that we have $M\,(|\mathbf{val}(C)| - 1)$ linear constraints, which restricts application currently to some thousand samples. Therefore, we slightly modify the problem and propose a scalable gradient-based optimization method. First, we eliminate the margin constraints, and substitute the slack variables via the parameters $\boldsymbol{\omega}$, by using a hinge function. In order to obtain a differentiable objective we use the soft-hinge $h_R(\cdot)$, defined as

$$h_R(\zeta) = \begin{cases} 0 & \zeta < \mu \\ \zeta & \zeta > \mu + \frac{R}{\sqrt{2}} \\ R - \sqrt{R^2 - (\zeta - \mu)^2} & \text{otherwise} \end{cases}, \text{ where}$$

R is the radius of a fitted circle-segment, smoothing the discontinuity of the hinge, and $\mu = R\,(1 - \sqrt{2})$. In our experiments we set $R = \min(1, \gamma)$. Now, the slack variables are (approximately) expressed as

$$\xi_m = h_R\left(\operatorname*{smax}_{c \neq c^m}\left[\gamma - (\phi_{c^m}(\mathbf{x}^m) - \phi_c(\mathbf{x}^m))^T\boldsymbol{\omega}\right]\right),$$

where smax is the soft-max, defined as $\operatorname*{smax}_{\zeta_1,\ldots,\zeta_L} = \frac{1}{\eta}\log\sum_{i=1}^{L}\exp(\eta\,\zeta_i)$. When the parameter η tends towards infinity, the soft-max converges to the max. In our experiments we set $\eta = 10$. We obtain the following modified problem

$$\begin{aligned}
\min_{\boldsymbol{\omega}} \quad & -\mathbf{n}^T\boldsymbol{\omega} + && (11)\\
& \lambda \sum_{m=1}^{M} h_R\left(\operatorname*{smax}_{c \neq c^m}\left[\gamma - (\phi_{c^m}(\mathbf{x}^m) - \phi_c(\mathbf{x}^m))^T\boldsymbol{\omega}\right]\right)\\
\text{s.t.} \quad & \log \sum_{j'} \exp\left(\omega^i_{j'|\mathbf{h}}\right) \leq 0 \qquad \begin{array}{l}\forall\, 0 \leq i \leq N\\ \forall\, \mathbf{h} \in \mathbf{val}(\mathbf{Pa}_i)\end{array}
\end{aligned}$$

Problem (11) is convex, with continuous differentiable objective. We use a projected gradient descent method, i.e. $\boldsymbol{\omega}$ is projected onto the set of subnormalized vectors after each gradient step. This can

be done independently for each CPT, i.e. for each combination of $i \in \{0, \ldots, N\}$ and $\mathbf{h} \in \mathbf{val}(\mathbf{Pa}_i)$. Projecting an arbitrary vector $\boldsymbol{\zeta}^* = (\zeta_1^*, \ldots, \zeta_L^*)^T$ onto the set of subnormalized vectors is formulated as $\min_{\boldsymbol{\zeta}} ||\boldsymbol{\zeta} - \boldsymbol{\zeta}^*||_2$, s.t. $\log \sum_{l=1}^{L} \exp(\zeta_l) \leq 0$. This problem has no closed form solution, but can be addressed by the iterative algorithm proposed in (Lin, 2003). This algorithm neatly meets our requirements, since we can use the solution of the previous projected gradient step as initialization, and then perform only some few iterations of the projection algorithm, without needing to iterate until convergence. The proposed projected gradient method scales nicely to large data sets; the evaluation of the objective and its gradient is linear in $(|\mathbf{val}(C)| - 1)\, M\, N$. It is also straightforward to implement parallel and stochastic versions of this method. Further details can be found in the supplementary material.

5. Experiments

In this section we present experiments for illustrative purposes (Sections 5.1 and 5.2) and a comparison on real-world datasets (Section 5.3). We considered 30 datasets from the UCI repository (Frank & Asuncion, 2010), TIMIT (Pernkopf et al., 2012) and USPS data (Hastie et al., 2003). Datasets containing more than 5000 samples were split into training and test set; Otherwise 5-fold cross-validation was used for testing. More details on the datasets can be found in the supplementary material. For discretizing continuous attributes, we used the algorithm described in (Fayyad & Irani, 2003). The smoothing parameter α in (4) was constantly set to 1. Although α can have a great impact on classification (Friedman et al., 1997; Silander et al., 2007), its evaluation is out of the scope of this paper.

5.1. Generative-Discriminative Trade-off

The parameter λ in problem (10) allows to control the trade-off between the generative and discriminative character of the model. Choosing $\lambda = 0$, the ML-BN-SVM parameters coincide with the ML solution. When λ tends towards infinity, a large margin separation of training samples is emphasized. Intermediate choices of λ correspond to a generative/discriminative crossover. To illustrate the effect of parameter λ, we learned ML-BN-SVMs with varying λ, assuming NB structure, using the *car* dataset. The results are shown in Figure 1. With increasing λ, the negative log-likelihood increases, while the sum of slacks decreases. Qualitatively, the classification rate increases correspondingly. Similar behavior can be observed on other datasets.

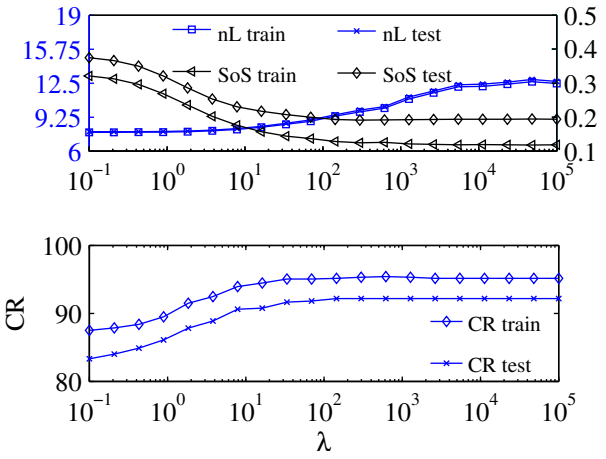

Figure 1. Influence of parameter λ using the *car* dataset. (top) Negative log-likelihood (nL) and sum of slacks (SoS), normalized by M. (bottom) Classification rate (CR).

5.2. Classification with Missing Features

Although the ML-BN-SVM is primarily trained for classification, its generative character justifies other inference tasks, e.g. marginalizing out missing features. The assumption is that the *more generative* the model is, the more robust the classifier is against missing data. To this end, we conducted an experiment with missing features in test data, using the *vehicle* dataset. We trained ML-BN-SVMs for different values of λ, cross-validating γ. In the test set, we varied the number of missing features, selected uniformly at random. For classification, missing features were marginalized out using junction-tree message passing. Classification results are shown in Figure 2, where results are averaged over 100 independent runs. While the purely generative model has the worst performance when no features are missing, its classification rate is almost constant until about 40% of missing features, and degrades slowly over the whole range of missing features. In contrast, models that are *more discriminative* (i.e. larger λ) show a better performance when all features are used, but their classification rates degrade rapidly with increasing percentage of missing features. This effect can be controlled; for $\lambda = 1$ and using all available features, the classification rate is almost as good as for classifiers trained with larger values of λ. Furthermore, the results are better than for the purely generative classifier for almost the whole range of missing features.

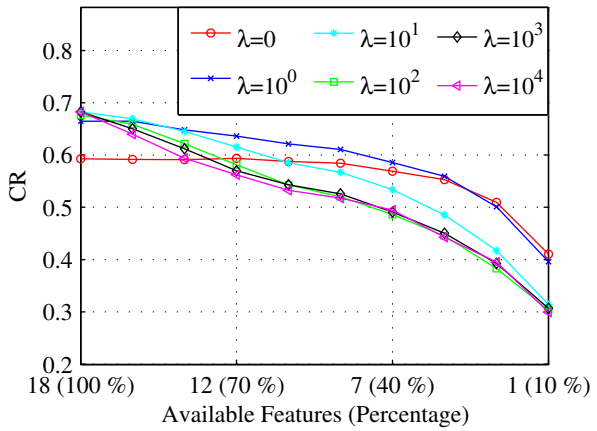

Figure 2. Classification rates (CR) for the *vehicle* dataset for varying numbers of missing features and varying λ.

5.3. Benchmark Classification Results

We compared ML-BN-SVMs with ML, maximum conditional likelihood (MCL) and maximum margin (MM) parameters using the algorithm proposed in (Pernkopf et al., 2012). In order to enable a fair comparison, MM was executed without early stopping. Experiments *with* early stopping are provided in the supplementary material. Furthermore, we compared with linear SVMs and SVMs equipped with Gaussian kernels (Chang & Lin, 2011). For ML-BN-SVMs we validated the ranges $\gamma \in \{0.1, 1, 5, 10, 15, 20\}$, and $\lambda \in \{0, 2^{-2}, 2^{-1}, \dots 2^{10}\}$. For MM, we used 10 values for κ and λ, uniformly spaced in the intervals $[0.01, 0.5]$ and $[0.01, 1]$, respectively (see (Pernkopf et al., 2012) for details). For SVMs we validated the trade-off parameter $\lambda \in \{2^{-2}, 2^{-1}, \dots 2^{10}\}$ and, for kernelized SVMs, the kernel width $\sigma \in \{2^{-5}, 2^{-4}, \dots, 2^{5}\}$. For the classifiers based on BNs, we used NB and maximum-likelihood TAN structures (Friedman et al., 1997). Classification results for the compared methods are shown in Table 1. Due to limited space, we omit the results for NB, averaged results for TIMIT, and show only a subset of the datasets. Further results are provided in the supplementary material. We see that ML-BN-SVM parameters clearly outperform both ML and MCL parameters. Furthermore, ML-BN-SVM performs better than MM in 17 out of 27 datasets. ML-BN-SVM also compares well to linear SVMs. We observe a slight preference for kernelized SVMs, which can be attributed to the kernel trick, and its implicit high dimensional feature transform. However, generally we see that the ML-BN-SVM delivers satisfying classification results.

To demonstrate the *generative character* of the ML-BN-SVM, we compare the likelihoods of the trained BN models. In Figure 3 we plot the likelihood (normalized by the sample size) of ML parameters against the likelihood of MCL, MM, and ML-BN-SVM parameters, respectively. The results for NB and TAN are combined. For cross-validated results, each fold is used as individual dataset, i.e. one dot in the scatter plot. Since ML parameters maximize the likelihood, no points on the left hand side of the 45°-line are possible. We observe that the scatter plot for ML-BN-SVM is clearly more concentrated in the vicinity of the 45°-line than for MCL and MM parameters, constituting the generative character of the ML-BN-SVM. A similar result is achieved for the likelihood on the *test* sets. Averaged over all datasets, the ML-BN-SVM achieved a likelihood of 91.09% relative to maximum likelihood (89.84% on the test sets); on the other hand, MCL training achieved on average a likelihood of 67.23% (61.47% on the test sets) and MM 39.99% (39.10% on the test set), relative to ML.

Furthermore, we performed missing feature experiments on the UCI datasets. We randomly removed features from the test sets, were we varied the percentage of missing features between 0 and 90%. Classifiers based on BNs treated missing features by marginalization. For the SVM (here we only considered the Gaussian kernel), K-nearest-neighbor (K-NN) imputation (with $K = 5$) was used to replace missing values. For all BN-classifiers, TAN structures were used. We also provide results for logistic regression (LR), using K-NN imputation. The result, averaged over all UCI datasets, are shown in Figure 4. As expected, the ML solution shows the most robust behavior against missing features, and for a percentage larger 60%, it performs best of all compared methods. However, ML-BN-SVMs perform better than ML in the case of no or little missing features, and are almost as robust against missing features as the ML solution. The purely discriminative BN parameters, MCL and MM, show a quick drop-off in performance when the percentage of missing features is increased. For large portions of missing features ($> 60\%$) also SVMs perform poorly compared to ML and ML-BN-SVM. This experiments indicates that ML-BN-SVMs are favorable in conditions where many features might be missing, and where the percentage of missing features varies strongly.

6. Conclusion

A BN distribution is a log-linear model, enabling SVM-type training for BNs (Guo et al., 2005; Pernkopf et al., 2012), which we call BN-SVM. For a

Table 1. Mean classification rates with 95% confidence intervals for UCI datasets, TIMIT and USPS data.

dataset	ML	MCL	MM	ML-BN-SVM	Linear SVM	SVM
abalone	57.70 ± 1.58	57.92 ± 1.65	57.78 ± 0.96	58.69 ± 1.86	58.42 ± 1.77	59.29 ± 1.40
adult	85.70 ± 0.66	86.65 ± 0.64	86.54 ± 0.65	86.76 ± 0.64	86.86 ± 0.64	86.87 ± 0.64
australian	81.67 ± 2.66	81.97 ± 3.70	85.49 ± 3.40	84.76 ± 3.78	85.78 ± 1.69	86.80 ± 2.34
breast	95.56 ± 2.06	95.56 ± 1.45	96.59 ± 0.50	96.00 ± 2.31	96.15 ± 1.51	97.19 ± 0.41
car	94.24 ± 1.50	98.08 ± 0.75	97.79 ± 0.79	98.08 ± 1.07	93.84 ± 0.65	99.65 ± 0.30
chess	92.19 ± 1.62	97.65 ± 0.81	97.43 ± 0.79	97.99 ± 0.92	97.02 ± 0.82	99.50 ± 0.25
cleve	79.43 ± 6.34	77.74 ± 7.53	79.09 ± 7.56	80.79 ± 7.58	83.57 ± 5.29	82.19 ± 6.37
crx	84.04 ± 4.64	80.32 ± 5.20	83.89 ± 5.89	84.20 ± 4.56	85.75 ± 3.20	85.75 ± 2.65
diabetes	74.35 ± 4.23	74.22 ± 5.50	73.31 ± 5.71	74.35 ± 5.42	73.96 ± 4.46	74.48 ± 4.65
flare	81.57 ± 1.27	81.48 ± 1.91	84.45 ± 0.28	83.30 ± 1.06	84.45 ± 0.28	84.45 ± 0.28
german	71.90 ± 1.83	69.50 ± 3.54	73.20 ± 4.01	72.60 ± 2.89	76.10 ± 1.11	75.80 ± 2.80
glass	72.68 ± 5.29	68.55 ± 4.03	71.71 ± 10.88	72.61 ± 6.35	71.61 ± 5.50	73.24 ± 5.33
heart	80.74 ± 10.36	77.04 ± 10.61	77.41 ± 9.81	81.48 ± 9.34	84.81 ± 4.11	81.85 ± 9.40
hepatitis	86.17 ± 10.00	86.08 ± 11.48	86.08 ± 3.38	86.17 ± 6.31	87.42 ± 10.89	88.67 ± 6.37
letter	86.21 ± 0.84	87.65 ± 0.80	89.58 ± 0.74	88.57 ± 0.77	90.07 ± 0.73	94.07 ± 0.58
lymphography	80.77 ± 7.36	75.38 ± 10.86	80.66 ± 11.11	76.92 ± 10.54	83.57 ± 10.44	86.48 ± 9.99
nursery	92.96 ± 0.77	98.31 ± 0.40	98.84 ± 0.33	98.68 ± 0.35	93.31 ± 0.76	100.00 ± 0.04
satimage	85.79 ± 1.92	81.52 ± 0.95	86.82 ± 2.66	86.98 ± 1.30	88.36 ± 1.58	90.59 ± 1.59
segment	94.89 ± 1.02	94.37 ± 1.57	96.02 ± 1.21	95.76 ± 0.62	96.19 ± 0.73	96.84 ± 1.17
shuttle	99.88 ± 0.05	99.84 ± 0.06	99.91 ± 0.05	99.92 ± 0.04	99.96 ± 0.03	99.96 ± 0.03
soybean-large	91.88 ± 1.28	82.66 ± 4.59	90.77 ± 2.16	91.87 ± 2.26	91.15 ± 3.72	93.54 ± 1.19
spambase	92.97 ± 0.85	92.99 ± 1.10	93.62 ± 0.80	94.03 ± 0.84	94.27 ± 0.72	95.04 ± 0.37
TIMIT	86.85 ± 6.86	84.43 ± 6.45	86.76 ± 5.33	88.74 ± 6.21	89.29 ± 5.90	89.46 ± 6.13
USPS	91.20 ± 0.93	90.46 ± 0.97	95.98 ± 0.65	95.98 ± 0.65	95.82 ± 0.66	91.80 ± 0.90
vehicle	70.60 ± 2.00	69.64 ± 3.69	69.04 ± 4.30	69.88 ± 2.41	70.12 ± 1.26	69.76 ± 2.43
vote	94.37 ± 2.62	94.15 ± 2.04	96.01 ± 2.45	95.31 ± 2.74	94.85 ± 2.20	95.54 ± 3.18
waveform-21	82.36 ± 0.71	80.55 ± 1.00	82.86 ± 0.51	83.48 ± 0.56	84.78 ± 1.77	85.16 ± 1.29

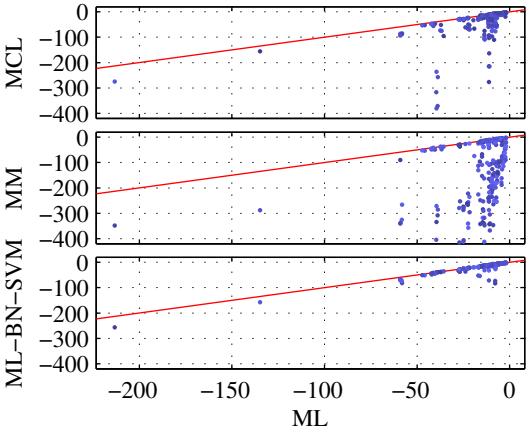

Figure 3. Likelihood scatter plot over all data sets. The train likelihoods (normalized by the sample size) of ML parameters are plotted against the train likelihoods of MCL (top), MM (center), and ML-BN-SVM (bottom).

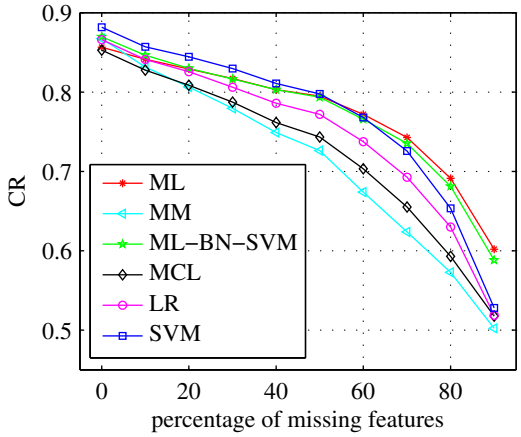

Figure 4. Classification results, averaged over UCI datasets, with varying percentage of missing features.

large class of network structures (Wettig et al., 2003), one can always obtain correctly normalized parameters, i.e. a formally valid BN. In this paper, we proposed the *maximum-likelihood* BN-SVM, where during discriminative training the log-likelihood of the model is maximized as a desired side-effect, partly maintaining a generative interpretation. In experiments we showed that in terms of classification our models outperform standard generative and discriminative learning methods for BNs (i.e. ML, MCL and MM), compete with linear SVMs, and are in range with kernelized SVMs. Furthermore, our models achieve likelihoods close to the ML solutions. We demonstrated the benefit of the generative character in missing feature experiments. In future work, we will extend the ML-BN-SVM to treat missing data during *learning*. In the BN framework, this naturally includes *learning with missing features* and *semi-supervised learning*.

Acknowledgments

This work was supported by the Austrian Science Fund (project number **P22488-N23**).

Algorithm 1 (Wettig et al., 2003)

Input: $\mathcal{G}$, unnormalized parameters $\tilde{\boldsymbol{\omega}}$
Output: Normalized parameters $\boldsymbol{\omega}$, with $\mathrm{P}^{\mathcal{B}}(C|\mathbf{X};\boldsymbol{\omega}) = \mathrm{P}^{\mathcal{B}}(C|\mathbf{X};\tilde{\boldsymbol{\omega}})$

1: $\boldsymbol{\omega} \leftarrow \tilde{\boldsymbol{\omega}}$
2: Find a topological order $(\pi_0, \ldots, \pi_N)$, i.e. any edge $X_{\pi_i} \to X_{\pi_j}$ is *not* contained in $\mathcal{G}$, $\forall\, 0 \le i < j \le N$.
3: **for** $i = 0 \ldots N$ **do**
4: **for** $\mathbf{h} \in \mathbf{val}(\mathbf{Pa}_{\pi_i})$ **do**
5: $\beta \leftarrow \log \sum_{j'} \exp\left(\omega^{\pi_i}_{j'|\mathbf{h}}\right)$
6: $\forall j \in \mathbf{val}(X_{\pi_i})$: $\omega^{\pi_i}_{j|\mathbf{h}} \leftarrow \omega^{\pi_i}_{j|\mathbf{h}} - \beta$
7: **if** X_{π_i} is a class-child **then**
8: Let X_{k_i} be a covering parent of X_{π_i}
9: $\mathbf{I} \leftarrow \mathbf{Pa}_{k_i} \cap \mathbf{Pa}_{\pi_i}$
10: $\mathbf{A} \leftarrow \mathbf{Pa}_{k_i} \setminus \mathbf{Pa}_{\pi_i}$
11: **for** $\mathbf{a} \in \mathbf{val}(\mathbf{A})$ **do**
12: $\omega^{k_i}_{\mathbf{h}(X_{k_i})|[\mathbf{h}(\mathbf{I}),\mathbf{a}]} \leftarrow \omega^{k_i}_{\mathbf{h}(X_{k_i})|[\mathbf{h}(\mathbf{I}),\mathbf{a}]} + \beta$
13: **end for**
14: **end if**
15: **end for**
16: **end for**

Appendix

Proof of **Lemma 1**. First note that in Algorithm 1, $\boldsymbol{\omega}$ always remains sub-normalized: If $\boldsymbol{\omega}$ is sub-normalized, then $\beta \le 0$ in step 5. In step 6 a CPT becomes normalized, and in step 12, β is added to some CPT entry, which again yields a sub-normalized CPT. By induction, $\boldsymbol{\omega}$ remains sub-normalized and $\beta \le 0$.

Algorithm 1 iterates over all X_{π_i} and all $\mathbf{h} \in \mathbf{val}(\mathbf{Pa}_{\pi_i})$, modifying $\boldsymbol{\omega}$. Let $\boldsymbol{\omega}'$ be the vector before some modification, and $\boldsymbol{\omega}''$ the vector afterwards. We show, that $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}'') \le \mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}')$, and therefore $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}) \le \mathrm{nL}_{\mathbf{n}}(\tilde{\boldsymbol{\omega}})$.
First, $\boldsymbol{\omega}$ is modified in step 6, where ${\omega''}^{\pi_i}_{j|\mathbf{h}} = {\omega'}^{\pi_i}_{j|\mathbf{h}} - \beta$, $\forall j \in \mathbf{val}(X_{\pi_i})$. By nonnegativity of $\mathbf{n}$ and $\beta \le 0$ we have

$$\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}'') - \mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}') = -\mathbf{n}^T\,\boldsymbol{\omega}'' + \mathbf{n}^T\,\boldsymbol{\omega}' = \beta \sum_j n^{\pi_i}_{j|\mathbf{h}} \le 0 \tag{12}$$

Therefore, when X_{π_i} is *not* a class-child, $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}'') \le \mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}')$. When X_{π_i} *is* a class-child, we additionally have in step 12

$${\omega''}^{k_i}_{\mathbf{h}(X_{k_i})|[\mathbf{h}(\mathbf{I}),\mathbf{a}]} = {\omega'}^{k_i}_{\mathbf{h}(X_{k_i})|[\mathbf{h}(\mathbf{I}),\mathbf{a}]} + \beta \quad \forall \mathbf{a} \in \mathbf{val}(\mathbf{A}), \tag{13}$$

where X_{k_i} is a covering parent of X_{π_i}, $\mathbf{I}$ are their common parents, and $\mathbf{A}$ are the extra parents of X_{k_i}. Since $\mathcal{G}$ is a C1-structure, it holds that $\mathbf{Pa}_{\pi_i} \setminus (\mathbf{Pa}_{k_i} \cup \{X_{k_i}\}) = \emptyset$. Therefore, since $\mathbf{n}$ are consistent likelihood-counts (cf. (7)), we have that $\sum_{\mathbf{a}\in\mathbf{val}(\mathbf{A})} n^{k_i}_{\mathbf{h}(X_{k_i})|[\mathbf{h}(\mathbf{I}),\mathbf{a}]} = \sum_{j'\in\mathbf{val}(X_{\pi_i})} n^{\pi_i}_{j'|\mathbf{h}}$, and thus

$$\begin{aligned}\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}'') - \mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}') &= -\mathbf{n}^T\,\boldsymbol{\omega}'' + \mathbf{n}^T\,\boldsymbol{\omega}' \\ &= \beta \sum_{j'} n^{\pi_i}_{j'|\mathbf{h}} - \beta \sum_{\mathbf{a}} n^{k_i}_{\mathbf{h}(X_{k_i})|[\mathbf{h}(\mathbf{I}),\mathbf{a}]} = 0.\end{aligned} \tag{14}$$

We see that $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}'') \le \mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}')$, and by induction $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}) \le \mathrm{nL}_{\mathbf{n}}(\tilde{\boldsymbol{\omega}})$. □

Proof of **Theorem 1**. Let $\boldsymbol{\omega}^*, \boldsymbol{\xi}^*$ be an optimal solution of (10). When we apply Algorithm 1 to $\boldsymbol{\omega}^*$, obtaining the normalized $\boldsymbol{\omega}$ as output, we see that $\boldsymbol{\omega}$, $\boldsymbol{\xi}$, with $\boldsymbol{\xi} = \boldsymbol{\xi}^*$, is feasible, since the class-conditional distribution is invariant under Algorithm 1. Furthermore, since $\boldsymbol{\omega}^*$ is sub-normalized, we have by Lemma 1 that $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}) \le \mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}^*)$. Therefore, $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}) + C\sum_m \xi_m \le \mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}^*) + C\sum_m \xi^*_m$, which implies that $\boldsymbol{\omega}$, $\boldsymbol{\xi}$ is optimal. □

Proof of **Theorem 2**. We first prove by contradiction, that under the conditions of Theorem 2, all solutions are normalized. Assume that $\boldsymbol{\omega}^*, \boldsymbol{\xi}^*$ are optimal for (10), where for some X_{π_i} and $\mathbf{h} \in \mathbf{val}(\mathbf{Pa}_{\pi_i})$, the corresponding CPT in $\boldsymbol{\omega}^*$ is *strictly sub-normalized*. Let $\boldsymbol{\omega}$ be the output of Algorithm 1 for input $\mathcal{G}$, $\boldsymbol{\omega}^*$. Let $\boldsymbol{\omega}'$ be the vector before the strictly sub-normalized CPT is processed, and $\boldsymbol{\omega}''$ be the vector afterwards.
When X_{π_i} is *not* a class child, then the negative log-likelihood is *strictly* decreased in step 6, i.e. $\mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}'') < \mathrm{nL}_{\mathbf{n}}(\boldsymbol{\omega}')$. Since the negative log-likelihood is never increased afterwards, $\boldsymbol{\omega}^*$, $\boldsymbol{\xi}^*$ can not be optimal.
When X_{π_i} *is* a class child, this decrease of negative log-likelihood is compensated in step 12 (cf. (14)). However, at the same time, some entries of some CPTs of the covering parent are *strictly* decreased, i.e. they become *strictly* sub-normalized. Due to the topological ordering, these CPTs are processed at a later step. By induction, some CPTs of the *class node* become strictly sub-normalized, since the class node has to be the covering parent for *some* class child. Finally, when the CPTs of the class node are normalized, the negative log-likelihood is strictly decreased, which contradicts that $\boldsymbol{\omega}^*, \boldsymbol{\xi}^*$ are optimal.
Now we show that the solution is unique. Assume two optimal solutions $\boldsymbol{\omega}^*$, $\boldsymbol{\xi}^*$ and $\boldsymbol{\omega}^{*\prime}$, $\boldsymbol{\xi}^{*\prime}$, $\boldsymbol{\omega}^* \ne \boldsymbol{\omega}^{*\prime}$. Since (10) is a convex problem, the convex combination $\boldsymbol{\omega} = 0.5\,\boldsymbol{\omega}^* + 0.5\,\boldsymbol{\omega}^{*\prime}$, $\boldsymbol{\xi} = 0.5\,\boldsymbol{\xi}^* + 0.5\,\boldsymbol{\xi}^{*\prime}$ is also optimal. Since all solutions are normalized, $\log\sum_j \exp(\omega^*{}^i_{j|\mathbf{h}}) = \log\sum_j \exp(\omega^{*\prime}{}^i_{j|\mathbf{h}}) = 0$, $\forall i, \mathbf{h}$. However, since $\log\sum\exp$ is a *strictly* convex function, $\boldsymbol{\omega}$ is *strictly* sub-normalized, which contradicts that $\boldsymbol{\omega}$, $\boldsymbol{\xi}$ is optimal. □

References

Bishop, C. M. and Lasserre, J. Generative or discriminative? Getting the best of both worlds. *Bayesian Statistics*, 8:3–24, 2007.

Bouchard, G. and Triggs, B. The trade-off between generative and discriminative classifiers. In *COMPSTAT*, pp. 721–728, 2004.

Buntine, W. Theory refinement on bayesian networks. In *Uncertainty in Artificial Intelligence (UAI)*, pp. 52–60, 1991.

Chang, C.-C. and Lin, C.-J. LIBSVM: A library for support vector machines. *ACM TIST*, 2:27:1–27:27, 2011. URL http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Cortes, C. and Vapnik, V. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

Crammer, K. and Singer, Y. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2: 265–292, 2001.

Fayyad, M. U. and Irani, B. K. Multi-interval discretization of continuous-valued attributes for classification learning. *IJCAI*, pp. 1022–1029, 2003.

Frank, A. and Asuncion, A. UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences, 2010. URL http://archive.ics.uci.edu/ml.

Friedman, N., Geiger, D., and Goldszmidt, M. Bayesian network classifiers. *Machine Learning*, (29):131–163, 1997.

Greiner, R., Su, X., Shen, B., and Zhou, W. Structural extension to logistic regression: Discriminative parameter learning of belief net classifiers. *Machine Learning*, 59(3):297–322, June 2005.

Guo, Y., Wilkinson, D., and Schuurmans, D. Maximum margin Bayesian networks. In *Uncertainty in Artificial Intelligence (UAI)*, pp. 233–242, 2005.

Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, August 2003.

Heckerman, D., Geiger, D., and Chickering, D. M. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

Jebara, T. *Discriminative, Generative and Imitative learning*. PhD thesis, MIT, 2001.

Lafferty, J., McCallum, A., and Pereira, F. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)*, pp. 282–289, 2001.

Lin, A. A class of methods for projection on a convex set. *Advanced Modeling and Optimization (AMO)*, 5(3), 2003.

Ng, A. Y. and Jordan, M. I. On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

Pernkopf, F., Wohlmayr, M., and Tschiatschek, S. Maximum margin Bayesian network classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):521–531, 2012.

Raina, R., Shen, Y., Ng, A., and McCallum, A. Classification with hybrid generative/discriminative models. In *Advances in Neural Information Processing Systems (NIPS)*, 2003.

Sha, F. *Large margin training of acoustic models for speech recognition*. PhD thesis, University of Pennsylvania, 2007.

Silander, T., Kontkanen, P., and Myllymäki, P. On sensitivity of the MAP bayesian network structure to the equivalent sample size parameter. In *Proceedings of UAI*, pp. 360–367, 2007.

Wettig, H., Grünwald, P., Roos, T., Myllymaki, P., and Tirri, H. When discriminative learning of Bayesian network parameters is easy. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, pp. 491–496, 2003.

Zhu, J., Rosset, S., Hastie, T., and Tibshirani, R. 1-norm support vector machines. *Advances in Neural Information Processing Systems*, 16:49–56, 2004.

Zou, H. and Yuan, M. The f_∞-norm support vector machine. *Statistica Sinia*, 18:379–398, 2008.