Integer Bayesian Networks

Sebastian Tschiatschek[*], Karin Paul, and Franz Pernkopf[*]

Signal Processing and Speech Communication Laboratory,
Graz University of Technology, Graz, Austria
tschiatschek@tugraz.at,karin.paul@student.tugraz.at,pernkopf@tugraz.at

Abstract. This paper introduces integer Bayesian networks (BNs), i.e. BNs with discrete valued nodes where parameters are stored as integer numbers. These networks allow for efficient implementation in hardware while maintaining a (partial) probabilistic interpretation under scaling. An algorithm for the computation of margin maximizing integer parameters is presented and its efficiency is demonstrated. The resulting parameters have superior classification performance compared to parameters obtained by simple rounding of double-precision parameters, particularly for very low number of bits.

Keywords: Bayesian networks, Bayesian network classifiers, custom precision analysis, parameter learning

1 INTRODUCTION

Bayesian networks (BNs) are probabilistic graphical models used to represent probability distributions. They are widely used for data modeling, e.g. in medicine, bioinformatics, image processing and pattern recognition. Their applications include inference and classification tasks, i.e. BNs are used to answer probabilistic queries on certain random variables.

Inference and classification with BNs are typically performed on computers with high numerical precision, i.e. using double-precision floating-point calculations. However, because of energy and computational constraints, low-power and integrated applications implemented on embedded systems require low complexity algorithms. Such applications are, for example, auditory scene classification in hearing aids and on-satellite computations[(1)]. In these kinds of applications, a trade-off between accuracy and algorithm complexity is essential.

In this paper, we argue that MM BNs, i.e. discriminatively optimized BNs, achieve a good trade-off in this respect and that careful algorithm design for the resource-constrained destination platform is advantageous. This is substantiated in Figure 1 for the *satimage* dataset from the UCI repository [5]. The model

[*] This work was supported by the Austrian Science Fund (project number P25244-N15).

[(1)] Computational capabilities on satellites are still severly limited due to power constraints and restricted availability of hardware satisfying the demanding requirements with respect to radiation tolerance.

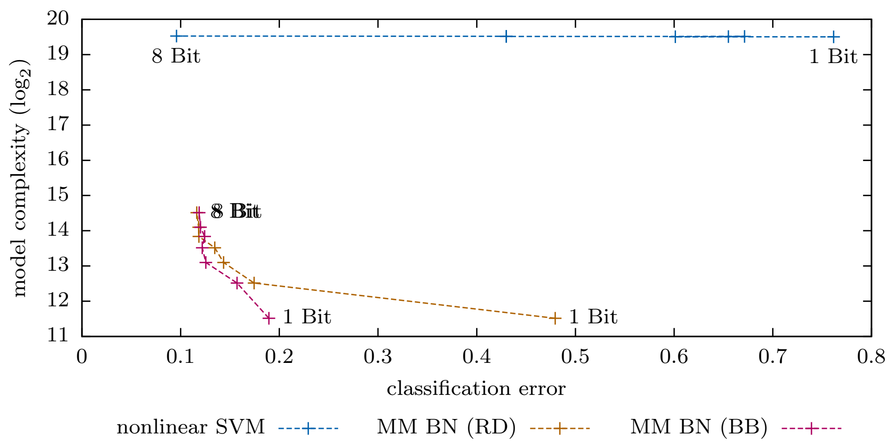


Fig. 1. Model complexities versus achieved classification errors. *Nonlinear SVM* refers to an SVM with radial-basis-function kernel, *MM BN (RD)* refers to an MM BN with parameters obtained by rounding and truncation, and *MM BN (BB)* refers to an MM BN with parameters obtained by the method proposed in this paper.

complexity in terms of bits required to store the classifier parameters versus the achieved classification error for SVMs with radial-basis-function kernels and for MM BNs is shown. In case of MM BNs, the performance of conventionally full-precision optimized and subsequently rounded parameters (*parameter rounding*) and that of parameters optimized for resource constraint environments (*MM (BB)*) is presented — details on the parameters are provided in the forthcoming sections. Note that the model complexity of the SVM is significantly higher than that of MM BNs, while classification performance is only slightly worse. Thus, if the application of interest allows to trade-off (slightly) reduced classification performance for tremendous savings in model complexity, MM BNs are obviously a good choice. If very low complexity models are desired, then MM (branch and bound (BB)) BNs are the best choice.

In this paper, we devise algorithms for efficiently learning such high performance low complexity models. While in [16], the authors already showed that parameters in Bayesian network classifiers (BNCs) can be mapped to the integer domain without considerable loss in classification rate (CR) performance, we take the analysis further: A principled approach for BN (and BNC) parameter learning of margin-maximizing parameters over a discrete search space, i.e. maximum-margin (B&B) parameters, is considered. This includes BNs with fixed-point parameters and (by proper scaling) integer parameters. An algorithm for parameter optimization based on BB techniques is presented. For low bit-widths, the obtained parameters lead to significantly better performance than parameters obtained by rounding double-precision maximum-margin parameters.

Our main contributions can be summarized as follows:

- An efficient algorithm for computing margin maximizing integer parameters. The algorithm is based on the branch-and-bound algorithm and a set of greedy heuristics. This offers a gain in computation time and makes learning tractable.
- Experiments demonstrating that integer BNs with small bit-widths can be widely applied. We especially show that a very low number of integer bits is often sufficient to obtain classification performance close to full-precision MM BNCs and SVMs. This offers considerable advantages when implementing BNs on embedded systems, i.e. data storage and bandwidth requirements are minimized.
- A brief theoretical analysis of BNs with rounded parameters.

This paper is structured as follows: In Section 2 we summarize related work. Section 3 introduces our notation and provides background on BNs, BNCs and parameter learning. In Section 4, integer Bayesian networks (iBNs) are introduced and an efficient algorithm for learning margin maximizing integer parameters is provided. Experimental results are provided in Section 5. We conclude the paper in Section 6 and provide an outlook on future work.

2 RELATED WORK

Literature on BNs with reduced precision parameters is scarce. The only directly related work investigates the effect of parameter quantization in BNCs with focus on comparing the robustness of BNCs with generatively and discriminatively optimized parameters [16]. The authors use bit-width reduced floating point parameters.

Indirectly related work deals with (a) *sensitivity analysis* of Bayesian networks [3, 4], stating essentially that classification using BNCs is insensitive to parameter deviations whenever either of these parameters are not close to zero or the class posteriors differ significantly, (b) credal networks, i.e. generalizations of Bayesian networks that associate a whole set of conditional probability densities (CPDs) with every node in the directed acyclic graph (DAG) [17], allowing for robust classification and supporting imprecisely specified CPDs.

In terms of undirected graphical networks, an interesting work on approximating undirected graphical models using integer parameters has been published recently [15]. The authors propose methods to perform inference and learning entirely in the integer domain. While undirected graphical models are more amenable to an integer approximation, there are domains where directed graphical models are more desirable and describe the probability distributions of interest more naturally, e.g. expert systems in the medical domain.

In terms of parameter learning using a BB scheme, there is related work for integer parameter learning of SVMs in the dual [1]. While some of the ideas presented by the authors are similar, classification with non-linear SVMs is computationally more demanding than classification using BNs(2). Furthermore, when

(2) For classification with non-linear SVMs, the kernel must be evaluated for all support-vectors and a weighted summation must be performed. Classification using BNs with

memory consumption is an issue, non-linear SVMs are disadvantageous because all support-vectors must be stored for classification.

3 BACKGROUND

3.1 Notation and Bayesian Networks

We assume a set of random variables (RVs) $X_0, \ldots, X_L$. These RVs are related by a joint probability distribution $\mathrm{P}^*(\mathbf{X})$, where $\mathbf{X} = (X_0, \ldots, X_L)$ is a random vector. BNs [12, 9] are used to represent such joint probability distributions in a compact and intuitive way. A BN $\mathcal{B} = (\mathcal{G}, \mathcal{P}_\mathcal{G})$ consists of a DAG $\mathcal{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V} = \{X_0, \ldots, X_L\}$ is the set of nodes and $\mathbf{E}$ the set of edges of the graph, and a set of local conditional probability distributions $\mathcal{P}_\mathcal{G} = \{\mathrm{P}(X_0|\mathbf{Pa}(X_0)), \ldots, \mathrm{P}(X_L|\mathbf{Pa}(X_L))\}$. The terms $\mathbf{Pa}(X_0), \ldots, \mathbf{Pa}(X_L)$ denote the set of parents of $X_0, \ldots, X_L$ in $\mathcal{G}$, respectively. Assuming discrete valued nodes, we abbreviate the conditional probability $P(X_i = j|\mathbf{Pa}(X_i) = \mathbf{h})$ as $\theta^i_{j|\mathbf{h}}$ and the corresponding logarithmic probability as $w^i_{j|\mathbf{h}} = \log(\theta^i_{j|\mathbf{h}})$. Without loss of generality, we further assume that $X_i \in \{1, \ldots, |\mathrm{sp}(X_i)|\}$, where $\mathrm{sp}(X_i)$ is the set of possible values of RV X_i. Each node of the graph corresponds to an RV and the edges of the graph determine dependencies between these RVs. A BN induces a joint probability $\mathrm{P}^\mathcal{B}(\mathbf{X})$ according to

$$\mathrm{P}^\mathcal{B}(\mathbf{X}) = \prod_{i=0}^{L} \mathrm{P}(X_i|\mathbf{Pa}(X_i)). \tag{1}$$

To represent $\mathrm{P}^*(\mathbf{X})$ by the BN $\mathcal{B} = (\mathcal{G}, \mathcal{P}_\mathcal{G})$, the graph $\mathcal{G}$ and the conditional probabilities in $\mathcal{P}_\mathcal{G}$ must be selected such that $\mathrm{P}^\mathcal{B}(\mathbf{X})$ *matches* $\mathrm{P}^*(\mathbf{X})$. In typical settings, however, the joint distribution $\mathrm{P}^*(\mathbf{X})$ and its properties are assumed to be unknown and only a limited number of samples drawn from this distribution, i.e. a training set $\mathcal{D}$, is available. This set $\mathcal{D}$ consists of N i.i.d. samples, i.e. $\mathcal{D} = \{\mathbf{x}^{(n)}|1 \leq n \leq N\}$, where $\mathbf{x}^{(n)}$ is the nth training sample and denotes an instantiation of $\mathbf{X}$. From this training set, the graph structure $\mathcal{G}$ of the BN as well as its parameters $\mathcal{P}_\mathcal{G}$ have to be derived. Selecting the graph structure is known as *structure learning* and selecting $\mathcal{P}_\mathcal{G}$ is known as *parameter learning.* The structures considered throughout this paper are fairly simple. In detail, we used naive Bayes (NB) and tree augmented network (TAN) structures [6]. Details on selecting the parameters are provided in Section 3.3.

3.2 Probabilistic Classification

In probabilistic classifiers, one RV in $X_0, \ldots, X_L$ takes the role of the class variable. Without loss of generality, we assume that X_0 corresponds to this

naive Bayes (NB) or tree augmented network (TAN) structures [6] corresponds to a simple summation of log-probabilities followed by an arg-max operation. Classification using linear SVMs is similar to classification using BNs.

class variable and denote it as C. The remaining RVs $X_1, \ldots, X_L$ represent the attributes/features of the classifier and are collected in the random vector $\widetilde{\mathbf{X}} = [X_1, \ldots, X_L]$. The aim is to induce *good* classifiers provided the training set, i.e. classifiers with high CR. Formally, a classifier h is a mapping

$$\begin{aligned} h \colon \operatorname{sp}(\widetilde{\mathbf{X}}) &\to \operatorname{sp}(C), \\ \widetilde{\mathbf{x}} &\mapsto h(\widetilde{\mathbf{x}}), \end{aligned} \tag{2}$$

where $\widetilde{\mathbf{x}}$ denotes an instantiation of $\widetilde{\mathbf{X}}$, $\operatorname{sp}(\widetilde{\mathbf{X}})$ denotes the set of all assignments of $\widetilde{\mathbf{X}}$ and $\operatorname{sp}(C)$ is the set of classes. The CR of this classifier is

$$\mathrm{CR(h)} := \mathbb{E}_{\mathrm{P}^*(C,\widetilde{\mathbf{X}})}\left[\mathbf{1}\{C = h(\widetilde{\mathbf{X}})\}\right], \tag{3}$$

where $\mathbf{1}\{A\}$ denotes the indicator function and $\mathbb{E}_{\mathrm{P}^*(C,\widetilde{\mathbf{X}})}[\cdot]$ is the expectation operator with respect to the distribution $\mathrm{P}^*(C, \widetilde{\mathbf{X}})$. Typically, the CR cannot be evaluated because $\mathrm{P}^*(C, \widetilde{\mathbf{X}})$ is unknown. It is rather estimated using cross-validation [2]. To determine BNCs, the training set $\mathcal{D}$ is assumed to consist of N i.i.d. labeled samples, i.e. $\mathcal{D} = \{(c^{(n)}, \widetilde{\mathbf{x}}^{(n)}) | 1 \leq n \leq N\}$, where $c^{(n)}$ denotes the instantiation of the RV C and $\widetilde{\mathbf{x}}^{(n)}$ the instantiation of $\widetilde{\mathbf{X}}$ in the nth training sample.

Any probability distribution, hence also any BN $\mathcal{B}$, induces a classifier $h_{\mathrm{P}^{\mathcal{B}}(C,\widetilde{\mathbf{X}})}$ according to

$$\begin{aligned} h_{\mathrm{P}^{\mathcal{B}}(C,\widetilde{\mathbf{X}})} \colon \operatorname{sp}(\widetilde{\mathbf{X}}) &\to \operatorname{sp}(C), \\ \widetilde{\mathbf{x}} &\mapsto \arg\max_{c \in C} \mathrm{P}^{\mathcal{B}}(C = c | \widetilde{\mathbf{X}} = \widetilde{\mathbf{x}}). \end{aligned} \tag{4}$$

In this way, each instantiation $\widetilde{\mathbf{x}}$ of $\widetilde{\mathbf{X}}$ is classified as the maximum a-posteriori (MAP) estimate of C given $\widetilde{\mathbf{x}}$ under $\mathrm{P}^{\mathcal{B}}(C, \widetilde{\mathbf{X}})$.

3.3 Parameter Learning for Bayesian Networks

The parameters of a BN $\mathcal{B}$ can be optimized either generatively or discriminatively [14]. Discriminative parameter learning is suitable for classification tasks, while in generative parameter learning one aims at identifying parameters representing the generative process of the considered data. In this paper, we advocate a hybrid generative-discriminative parameter optimization according to [13]. The objective is the joint maximization of the data likelihood and the margin on the data. Formally, MM parameters $\mathcal{P}_{\mathcal{G}}^{\mathrm{MM}}$ are learned as

$$\begin{aligned} \mathcal{P}_{\mathcal{G}}^{\mathrm{MM}} = \arg\max_{\mathcal{P}_{\mathcal{G}}} \Bigg[&\sum_{n=1}^{N} \log \mathrm{P}^{\mathcal{B}}(\mathbf{x}^{(n)}) \\ &+ \lambda \sum_{n=1}^{N} \min\left(\gamma, \log \mathrm{P}^{\mathcal{B}}(\mathbf{x}^{(n)}) - \max_{c \neq c^{(n)}} \mathrm{P}^{\mathcal{B}}([c, \tilde{\mathbf{x}}^{(n)}])\right)\Bigg], \end{aligned} \tag{5}$$

where $\mathrm{P}^{\mathcal{B}}(\mathbf{X})$ is the joint distribution in (1) induced by the BN $(\mathcal{G}, \mathcal{P}_{\mathcal{G}})$, λ a trade-off parameter between likelihood and margin, i.e. generative and discriminative optimization, and γ the desired margin. The margin of sample n is defined as the difference in log-likelihood of the sample belonging to the correct class to belonging to the most likely competitor class, i.e. $\log \mathrm{P}^{\mathcal{B}}(\mathbf{x}^{(n)}) - \max_{c \neq c^{(n)}} \mathrm{P}^{\mathcal{B}}([c, \tilde{\mathbf{x}}^{(n)}])$. Consequently, a sample is classified correctly iff it has positive margin and incorrectly otherwise. ⟨**Sebastian→Franz:** Reviewer: *In Equation (5), how do you fix the values of λ and γ?*⟩ The parameters λ and γ are typically set using cross-validation. MM parameter learning is considered throughout the paper.

4 INTEGER BAYESIAN NETWORKS

In this section, we introduce iBNs, i.e. BNs with integer parameters. Further, we present an algorithm for determining margin maximizing parameters for iBNs.

4.1 Definition

According to (1), the BN $\mathcal{B}$ assigns the probability

$$\mathrm{P}^{\mathcal{B}}(\mathbf{x}) = \prod_{i=0}^{L} \mathrm{P}(X_i = \mathbf{x}_{X_i} | \mathbf{Pa}(X_i) = \mathbf{x}_{\mathbf{Pa}(X_i)}), \tag{6}$$

to an instantiation $\mathbf{x}$ of $\mathbf{X}$, where $\mathbf{x}_{X_k}$ denotes the instantiation of X_k and $\mathbf{x}_{\mathbf{Pa}(X_k)}$ the instantiation of the parents of X_k according to $\mathbf{x}$, respectively. The above equation can be equivalently stated in the logarithmic domain, i.e.

$$\log \mathrm{P}^{\mathcal{B}}(\mathbf{x}) = \sum_{i=0}^{L} \log \mathrm{P}(X_i = \mathbf{x}_{X_i} | \mathbf{Pa}(X_i) = \mathbf{x}_{\mathbf{Pa}(X_i)}). \tag{7}$$

Hence, computing the likelihood of a sample $\mathbf{x}$ corresponds to a summation of log-probabilities. Assuming all log-probabilities are represented using B_I integer bits and B_F fractional bits, they can be written as

$$w^i_{j|\mathbf{h}} = \log \mathrm{P}(X_i = j | \mathbf{Pa}(X_i) = \mathbf{h}) = \sum_{k=-B_F}^{B_I - 1} b^{i,k}_{j|\mathbf{h}} \cdot 2^k, \tag{8}$$

where $b^{i,k}_{j|\mathbf{h}} \in \{0, 1\}$ denotes the k^{th} bit of the binary representation of $w^i_{j|\mathbf{h}}$. Hence, all $w^i_{j|\mathbf{h}}$ are in the set of negative fixed point numbers with B_I integer bits and B_F fractional bits, i.e.

$$w^i_{j|\mathbf{h}} \in -\mathbb{B}^{B_I}_{B_F} = -\left\{ \sum_{k=-B_F}^{B_I - 1} d_k \cdot 2^k : d_k \in \{0, 1\} \right\}. \tag{9}$$

Introducing this in (7) and scaling by 2^{B_F} results in

$$\log \mathrm{P}^{\mathcal{B}}(\mathbf{x}) = \sum_{i=0}^{L} \sum_{k=-B_F}^{B_I-1} b^{i,k}_{\mathbf{x}_{X_i}|\mathbf{x}_{\mathbf{Pa}(X_i)}} \cdot 2^{k+B_F}, \tag{10}$$

i.e. all summands are integer valued. The largest summand is at most $2^{B_I+B_F}-1$. The summation is over $L+1$ log probabilities, i.e. the number of nodes in $\mathcal{B}$. Hence, in total at most

$$\log_2(L+1) + B_I + B_F \tag{11}$$

bits are required to calculate the joint probability. This transformation to the integer domain is advantageous in several aspects: (1) no floating-point rounding errors of any kind are introduced when working purely in the integer domain, (2) computations using integer arithmetic are typically faster and more efficient, (3) the need for a floating point processing unit is eliminated which encourages usage in many embedded systems, and (4) the integer parameters require less memory for storage.

We call BNs parametrized as above iBNs. Note that iBNs could also be formulated considering probabilities instead of log probabilities. Then normalization of the parameters, i.e. probabilities sum up to one, can always be achieved. However, representing the log probabilities has the advantage that a large dynamic range is achieved and that classification essentially resorts to evaluating sums of log probabilities (more generally, max-sum message-passing can be easily performed).

4.2 Learning iBNs

In principle, parameters for iBNs can be determined by first learning BN parameters using full-precision floating-point computations and subsequent rounding (and scaling) to the desired number format — a brief analysis of this approach is provided at the end of this section. However, such parameters are in general not optimal in the sense of the MM criterion (5) and we aim at a more principled approach.

Our approach is based on the branch and bound procedure [10], exploiting convexity of (5) under suitable parametrization. The exact meaning will become clear immediately. Optimization of the MM criterion can be represented as

$$\begin{aligned}
\underset{\mathbf{w}}{\text{maximize}} \quad & \sum_{n=1}^{N} \boldsymbol{\phi}(\mathbf{x}^{(n)})^T \mathbf{w} + \lambda \sum_{n=1}^{N} \min\left(\gamma, \boldsymbol{\phi}(\mathbf{x}^{(n)})^T \mathbf{w} - \max_{c \neq c^{(n)}} \boldsymbol{\phi}([c, \tilde{\mathbf{x}}^{(n)}])^T \mathbf{w}\right) \\
\text{s.t.} \quad & \sum_{j=1}^{|\mathrm{sp}(X_i)|} \exp(w^i_{j|\mathbf{h}}) = 1 \qquad \forall i, \mathbf{h},
\end{aligned} \tag{12}$$

where we exploit that any log probability $\log \mathrm{P}(\mathbf{x})$ can be written as

$$\begin{aligned} \log \mathrm{P}(\mathbf{x}) &= \sum_{i=0}^{L} \sum_{\mathbf{h} \in \mathrm{sp}(\mathbf{Pa}(X_i))} \sum_{j \in \mathrm{sp}(X_i)} w^i_{j|\mathbf{h}} \cdot \mathbf{1}(\mathbf{x}_{X_i} = j, \mathbf{x}_{\mathbf{Pa}(X_i)} = \mathbf{h}) && (13) \\ &= \boldsymbol{\phi}(\mathbf{x})^T \mathbf{w} && (14) \end{aligned}$$

by collecting for a given instantiation $\mathbf{x}$ the values of the indicator functions $\mathbf{1}(\mathbf{x}_{X_i} = j, \mathbf{x}_{\mathbf{Pa}(X_i)} = \mathbf{h})$ in vector $\boldsymbol{\phi}(\mathbf{x})$ and the corresponding $w^i_{j|\mathbf{h}}$ in vector $\mathbf{w}$. The above problem in (13) is nonconvex and hard to solve. However, when relaxing normalization constraints to

$$\sum_{j=1}^{|\mathrm{sp}(X_i)|} \exp(w^i_{j|\mathbf{h}}) \leq 1, \tag{15}$$

the problem becomes convex and can hence be solved efficiently. **⟨Sebastian→Franz:** Reviewer: *Sometimes, the relaxation of constraints is not explained. For instance, the authors do not explain the implications of Equation (15). The motivation behind the relaxation is clear, but why can you do that and what does it imply? Another example is Equation (17).*⟩ If all components of $\sum_{n=1}^{N} \boldsymbol{\phi}(\mathbf{x}^{(n)})$ are positive, e.g. when applying Laplace smoothing, then (15) is automatically satisfied with equality by any optimal solution of the relaxed problem, i.e. the original constraints are recovered [13].

For learning integer parameters, we restrict the parameters $\mathbf{w}$ to $-\mathbb{B}^{B_I}_{B_F}$ and further relax the normalization constraints to

$$\sum_{j=1}^{|\mathrm{sp}(X_i)|} \exp(w^i_{j|\mathbf{h}}) \leq 1 + \theta(|\mathrm{sp}(X_i)|, B_I, B_F), \qquad \forall i, \mathbf{h} \tag{16}$$

where $\theta(|\mathrm{sp}(X_i)|, B_I, B_F)$ is an additive constant. This further relaxation of the normalization constraints is necessary, as in general reduced precision parameters do not correspond to correctly normalized parameters. The additive constant is necessary, as for very small bit-widths there are no parameters that are sub-normalized, i.e. $\sum_j \exp(M) > 1$, where $M = -2^{B_I} + 2^{-B_F}$ is the smallest value that can be represented. Therefore, without this constant, our optimization problem would be infeasible. Thus, our final optimization problem is

$$\begin{aligned} \underset{\mathbf{w}}{\text{maximize}} \quad & \sum_{n=1}^{N} \boldsymbol{\phi}(\mathbf{x}^{(n)})^T \mathbf{w} + \lambda \sum_{n=1}^{N} \min\left(\gamma, \boldsymbol{\phi}(\mathbf{x}^{(n)})^T \mathbf{w} - \max_{c \neq c^{(n)}} \boldsymbol{\phi}([c, \tilde{\mathbf{x}}^{(n)}])^T \mathbf{w}\right) \\ & && (17) \\ \text{s.t.} \quad & \sum_{j=1}^{|\mathrm{sp}(X_i)|} \exp(w^i_{j|\mathbf{h}}) \leq 1 + \theta(|\mathrm{sp}(X_i)|, B_I, B_F) \qquad \forall i, \mathbf{h}, \\ & w^i_{j|\mathbf{h}} \in -\mathbb{B}^{B_I}_{B_F} \qquad \forall i, j, \mathbf{h}. \end{aligned}$$

For efficiently finding (global) minimizers of (17), we propose to use a BB algorithm [8] and greedy heuristics for creating candidate solutions and branching orders:

Branch and Bound Algorithm. The optimal iBN parameters have to be searched in a discrete solution space, i.e. $w^i_{j]\mathbf{h}} \in -\mathbb{B}^{B_I}_{B_F}$. For optimization, the BB algorithm is used. BB searches the solution space by creating a tree of subproblems and dynamically adding (*branch*) and discarding (*bound*, also referred to as pruning) branches. The algorithm iteratively solves (17) using upper and lower bounds for the parameters to be found depending on the considered leaf of the search tree. If the determined solution does not fit the required precision for all parameters, the algorithm performs one of the following two options: (1) It either prunes the whole subtree because no global maximizer is to be found (this happens if the best feasible solution found so far has larger objective than the relaxed problem of the current leaf), or (2) it creates two new problems by adding new lower and upper bounds to one of the parameters which does not satisfy the desired precision.

Rounding heuristic. To efficiently apply the BB algorithm, it is important to prune large parts of the search space at an early stage. Therefore, we need to obtain good lower bounds for the objective every time a problem corresponding to a leaf in the search tree has been solved. We try to achieve this using simple rounding heuristics.

Let $\hat{\mathbf{w}}$ correspond to the intermediate solution. Then, the candidate solutions $\mathbf{a}$ and $\mathbf{b}$ are generated as follows:

– *Rounding*: Set

$$\hat{a}^i_{j|\mathbf{h}} = \max\left(M, \left[\frac{\hat{w}^i_{j|\mathbf{h}}}{q}\right]_R q\right), \tag{18}$$

where $[\cdot]_R$ denotes rounding to the closest integer, q is the quantization interval and $M = -2^{B_I} + 2^{-B_F}$ the minimum value that can be represented. Set $\mathbf{a} = \Pi(\hat{\mathbf{a}})$, where Π is a projection-like operator ensuring that $\mathbf{b}$ is feasible for (17).

– *Gradient Guided Rounding*: Let $\mathbf{g}$ be the gradient of the objective at $\hat{\mathbf{w}}$. Then,

$$\hat{b}^i_{j|\mathbf{h}} = \begin{cases} \left\lceil \frac{\hat{w}^i_{j|\mathbf{h}}}{q} \right\rceil q & \text{if } g^i_{j|\mathbf{h}} > 0, \text{and} \\ \max\left(M, \left\lfloor \frac{\hat{w}^i_{j|\mathbf{h}}}{q} \right\rfloor q\right) & \text{if } g^i_{j|\mathbf{h}} \leq 0, \end{cases} \tag{19}$$

where $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ denote the floor and ceil function, respectively. Set $\mathbf{b} = \Pi(\hat{\mathbf{b}})$, where Π is as above.

Branching Heuristics. After solving one of the subproblems of the search tree, we check the obtained solution $\hat{\mathbf{w}}$ for optimality. If the solution is not optimal, we branch on the entry $\hat{w}^i_{j|\mathbf{h}}$ that has the largest deviation from the desired precision, i.e.

$$(i', j', \mathbf{h}') = \arg\max_{i,j,\mathbf{h}} \left| \hat{w}^i_{j|\mathbf{h}} - \left[\frac{\hat{w}^i_{j|\mathbf{h}}}{q} \right]_R q \right|. \tag{20}$$

Subproblems are processed in order of their upper-bounds, i.e. the subproblem of the search tree that has the heighest upper-bound is processed next.

4.3 Approximate integer Bayesian network classifiers (iBNCs)

In this section, we provide a short analysis of the effect of rounding log parameters to their closest fixed point representation. This reveals interesting insights into why classification performance of BNCs with rounded parameters is better than one might anticipate. ⟨**Sebastian→Franz:** Reviewer: *Section 4.3 is devoted to the theoretical study of iBN-RD. However, the authors claim that iBN-BB provides the best approximation. This is confusing to me. Why the theoretical study has not been done for iBN-BB?* Was hältst du von folgendem Vorschlag um das klarer zu stellen?⟩ Performing a similar analysis for iBNCs is much more difficult because the objective for learning margin maximizing parameters does not decompose as a product of conditional probabilities.

We start by analyzing the Kullback-Leibler (KL)-divergence introduced by rounding, i.e. the KL-divergence between an *optimal* distribution, e.g. the original full-precision distribution, and its *approximation* obtained by rounding of the log-probabilities. Clearly, the approximate distribution is not necessarily properly normalized. Therefore, we compare the KL-divergence of the optimal distribution and the renormalized approximate distribution. This yields the following lemma:

Lemma 1 (KL-divergence). *Let $\mathbf{w}^i_{\cdot|\mathbf{h}}$ be a vector of normalized log probabilities (optimal distribution), i.e. $\sum_j \exp(w^i_{j|\mathbf{h}}) = 1$, and let $\widetilde{\mathbf{w}}^i_{\cdot|\mathbf{h}}$ (approximate distribution) be such that*

$$\widetilde{w}^i_{j|\mathbf{h}} = \left[\frac{w^i_{j|\mathbf{h}}}{q} \right]_R q, \tag{21}$$

where $q = 2^{-B_F}$ is the quantization interval. Then the KL-divergence between the optimal and the renormalized approximate distribution is bounded by q, i.e.

$$\mathcal{D}(\mathbf{w}^i_{\cdot|\mathbf{h}} || \alpha + \widetilde{\mathbf{w}}^i_{\cdot|\mathbf{h}}) \leq q, \tag{22}$$

where $\alpha = (\sum_j \exp(\widetilde{w}^i_{j|\mathbf{h}}))^{-1}$ ensures proper normalization of $\widetilde{\mathbf{w}}$, i.e.

$$\sum_j \exp(\alpha + \widetilde{w}^i_{j|\mathbf{h}}) = 1. \tag{23}$$

Proof. We calculate

$$\begin{aligned}
\mathcal{D}(w^i_{\cdot|\mathbf{h}}||\widetilde{w}^i_{\cdot|\mathbf{h}}) &= \sum_j \exp(w^i_{j|\mathbf{h}}) \log \frac{\exp(w^i_{j|\mathbf{h}})}{\alpha \exp(\widetilde{w}^i_{j|\mathbf{h}})} && (24)\\
&= \sum_j \exp(w^i_{j|\mathbf{h}})(w^i_{j|\mathbf{h}} - \widetilde{w}^i_{j|\mathbf{h}}) - \log\alpha && (25)\\
&\overset{(a)}{\leq} \sum_i \exp(w^i_{j|\mathbf{h}})\frac{q}{2} - \log\alpha && (26)\\
&= \frac{q}{2} - \log\alpha, && (27)
\end{aligned}$$

where (a) is because $\widetilde{\mathbf{w}}^i_{\cdot|\mathbf{h}}$ is derived from $\mathbf{w}^i_{\cdot|\mathbf{h}}$ by rounding the parameters. It remains to upper bound $-\log\alpha$. Straightforward calculation yields

$$\begin{aligned}
-\log\alpha &= \log \sum_j \exp(\widetilde{w}^i_{j|\mathbf{h}}) && (28)\\
&\leq \log \sum_j \exp(w^i_{j|\mathbf{h}} + \frac{q}{2}) && (29)\\
&= \frac{q}{2}. && (30)
\end{aligned}$$

Hence,

$$\mathcal{D}(\mathbf{w}^i_{\cdot|\mathbf{h}}||\alpha + \widetilde{\mathbf{w}}^i_{\cdot|\mathbf{h}}) \leq q. \tag{31}$$

This bound is tight. Assuming that sufficient integer bits are used so that no log-probabilities have to be truncated, $q = 2^{-B_F}$. Hence, the KL-divergence decays rapidly with increasing B_F.

When using only a finite number of bits for the integer part, log-probabilities may be discarded. Still, a bound on the KL-divergence can be derived:

Lemma 2 (KL-divergence). *Let $\mathbf{w}^i_{\cdot|\mathbf{h}}$ be a vector of normalized log probabilities (optimal distribution), and let $\widetilde{\mathbf{w}}^i_{\cdot|\mathbf{h}}$ (approximate distribution) be such that*

$$\widetilde{w}^i_{j|\mathbf{h}} = \max\left(M, \left[\frac{w^i_{j|\mathbf{h}}}{q}\right]_R q\right), \tag{32}$$

where q is the quantization interval and M the minimal representable log-probability. Then the KL-divergence between the optimal and the renormalized approximate distribution is bounded by q, i.e.

$$\mathcal{D}(\mathbf{w}^i_{\cdot|\mathbf{h}}||\alpha + \widetilde{\mathbf{w}}^i_{\cdot|\mathbf{h}}) \leq q + \mathit{length}(w)\exp(-q/2 + M), \tag{33}$$

where $\alpha = (\sum_j \exp(\widetilde{w}^i_{j|\mathbf{h}}))^{-1}$ ensures proper normalization of $\widetilde{\mathbf{w}}$.

Typically, $M = -2^{B_I} + 2^{-B_F}$. Hence, also in this case the bound decays rapidly with an increasing number of bits. One can further observe a dependency on the size of individual conditional probability tables (CPTs).

Both, Lemma 1 and 2, guarantee that simply rounding the log-probabilities of an optimal distribution does yield a good approximation in terms of KL-divergence. Therefore, it is not surprising that BNCs with parameters obtained by rounding achieve good performance. Furthermore, this justifies the usage of rounding as a heuristic for obtaining good candidate solutions in the BB algorithm.

5 EXPERIMENTAL RESULTS

In the following, we present classification experiments. In particular, we use BNCs with parameters determined as follows:

- *branch and bound* (BB): These integer parameters are obtained using the branch and bound algorithm presented in Section 4.2.
- *rounded* (RD): Integer parameters are obtained by rounding double-precision log parameters. If necessary, parameters are clipped to the considered number of integer bits.
- *double precision* (DP): Double precision parameters are obtained by solving (12) using methods proposed in [13].

5.1 Classification Experiments

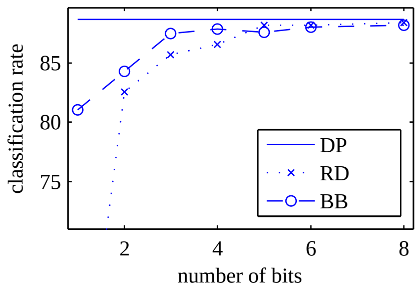
We consider classification experiments for four real world datasets:

- **USPS** [7]. This dataset contains 11000 uniformly distributed handwritten digit images from zip codes of mail envelopes, of which 8000 are used for training and 3000 for testing. Each digit is represented as a 16×16 grayscale image, where each pixel is considered as feature.
- **MNIST** [11]. The MNIST dataset contains a training set of 60000 size-normalized and centered images of handwritten digits of size 16×16, accompanied by a test set of 10000 samples.
- **satimage/letter** [5]. From the UCI repository, we considered the *satimage* and the *letter* dataset.
  The satimage dataset consists of multi-spectral satellite images. Given a 3×3 multi-spectral pixel image patch, the task is to classify the central pixel as either red soil, cotton crop, grey soil, damp grey soil, soil with vegetation stubble, mixture class (all types present), or very damp grey soil. In total there are 6435 samples with 36 attributes. Performance is evaluated using 5-fold cross-valdiation.
  The letter dataset consists of 20000 samples, where two third of the data are used for training and one third for testing. Each sample is a character from the English alphabet and described by 16 numerical attributes, i.e. statistical moments and edge counts. The task is, based on these attributes, to classify each character as the represented English letter.

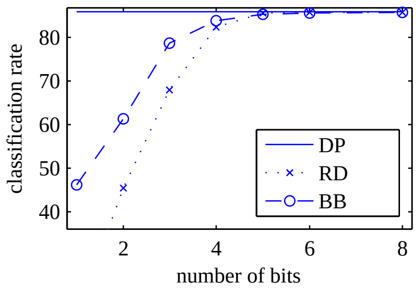
On these datasets, we compare the CR performance of BNCs and iBNCs with BB, RD, and DP parameters [3].

For RD parameters and a specific number of bits $B = B_I + B_F$, we determine the splitting into integer bits B_I and fractional bits B_F such that the classification rate on the training data is maximized. The same splitting is used for learning BB parameters with B bits. The hyper-parameters λ and γ in (17) are set using 5-fold cross-validation. For learning BB parameters with B bits, we allowed for up to five hours CPU time on a 3 GHz personal computer. If the parameter learning did not finish within this time, the best solution found so far was returned, cf. Section 4.2.

The observed CRs are shown in Figures 2, 3 and 4, for satimage, letter, USPS, and MNIST data using BNCs with NB structures, respectively. In case of USPS data, also CR performance for BNCs with TAN structures is shown. Only 5 to 6 integer bits for RD parameters are necessary to achieve CRs close to DP CRs. BNCs with BB parameters achieve better CRs than BNCs with RD parameters. Especially for low number of bits, BNCs with BB parameters are significantly better in terms of CR performance. This suggests that parameter learning under precision constraints is advantageous over full-precision parameter learning followed by subsequent rounding.



(a) satimage

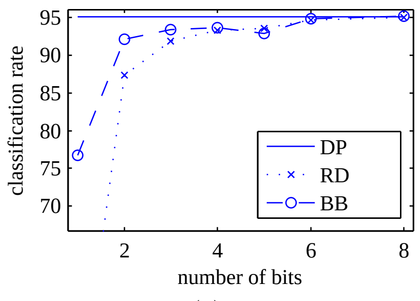


(b) letter

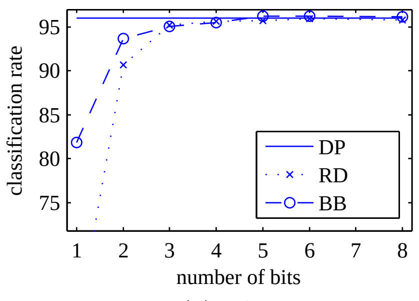
Fig. 2. CRs for *satimage* and *letter* data of BNCs and iBNCs with NB and TAN structures.

One important aspect of integer/reduced precision parameters, is their lower memory usage. This is exemplarily shown for USPS and MNIST data and NB and TAN structures in Table 1. The reduction in storage requirements by a factor of ~ 10 can positively influence the memory access when implementing iBNCs on embedded hardware.

[3] As mentioned in Section 4, up to $\log_2(L+1) + B_I + B_F$ bits are necessary for classification using BNCs with reduced precision parameters. In the presented experiments, we assume that these additional bits are available, i.e. summation does not cause overflows.

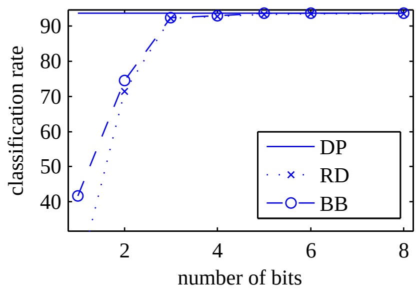


(a) NB



(b) TAN

Fig. 3. CRs for USPS data of BNCs and iBNCs with NB and TAN structures.



(a) MNIST

Fig. 4. CRs of iBNCs using NB structure and double precision (DP), rounded integer (RD) and integer parameters (BB) obtained through branch and bound.

Table 1. Memory usage for parameter storage using different datasets and structures in double and minimum optimal integer precision.

Dataset	Structure	# Parameters	# bits	Storage [kB]	
				double	integer
USPS	NB	8650	6	67.6	6.3
	TAN	32970	6	257.6	24.1
MNIST	NB	25800	3	201.6	9.4

6 CONCLUSIONS AND FUTURE WORK

In this paper we considered BNs with discrete valued nodes and parameters represented by integer numbers. We presented an efficient algorithm for computing margin maximizing integer parameters, where subproblems are convex and can be solved using subgradient methods.

In experiments, we showed that a low number of bits is sufficient to achieve good performance in classification scenarios. Furthermore, we showed that parameter learning under precision constraints is advantageous over full-precision parameter learning followed by subsequent rounding to the desired precision. The presented results aid in understanding the implications of implementing BNCs on embedded hardware and can greatly reduce the storage requirements and thus the time required for memory access.

Future work aims at a sample implementation of iBNs on embedded hardware for speed comparison. Furthermore, we want to derive methods for parameter learning using integer/reduced precision computations only. **⟨Sebastian→Franz:** Review: *(c) The method doesn't consider any innovation in structure learning. Is it possible to extend the margin maximization principle to the structural learning task?*⟩ Another interesting direction for future work is to incorporate reduced precision constraints into the task of structure learning, e.g. learning BN structures such that rounding of parameters degrades classification performance as little as possible.

References

1. Anguita, D., Ghio, A., Pischiutta, S., Ridella, S.: A support vector machine with integer parameters. Neurocomputing 72(1-3), 480–489 (2008)
2. Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer (2007)
3. Chan, H., Darwiche, A.: When do numbers really matter? Articial Intelligence Research 17(1), 265–287 (2002)
4. Chan, H., Darwiche, A.: Sensitivity analysis in Bayesian networks: From single to multiple parameters. In: Uncertainty in Artificial Intelligence (UAI). pp. 67–75 (2004)
5. Frank, A., Asuncion, A.: UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences (2010), `http://archive.ics.uci.edu/ml`
6. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning pp. 131–163 (1997)
7. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer (Aug 2003)
8. Horst, R., Tuy, H.: Global Optimization: Deterministic Approaches. Springer-Verlag (1996)
9. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press (2009)
10. Land, A.H., Doig, A.G.: An automatic method of solving discrete programming problems. Econometrica 28(3), pp. 497–520 (1960), `http://www.jstor.org/stable/1910129`

11. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)
12. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1988)
13. Peharz, R., Tschiatschek, S., Pernkopf, F.: The most generative maximum margin bayesian networks. In: Dasgupta, S., Mcallester, D. (eds.) Proceedings of the 30th International Conference on Machine Learning (ICML-13). vol. 28, pp. 235–243. JMLR Workshop and Conference Proceedings (May 2013), `http://jmlr.org/proceedings/papers/v28/peharz13.pdf`
14. Pernkopf, F., Wohlmayr, M., Tschiatschek, S.: Maximum margin Bayesian network classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 34(3), 521–531 (2012)
15. Piatkowski, N., Sangkyun, L., Morik, K.: The integer approximation of undirected graphical models. In: 3rd International Conference on Pattern Recognition Applications and Methods (ICPRAM) (2014)
16. Tschiatschek, S., Reinprecht, P., Mücke, M., Pernkopf, F.: Bayesian network classifiers with reduced precision parameters. In: European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD). pp. 74–89 (2012)
17. Zaffalon, M.: A credal approach to naive classification. In: ISIPTA (1999)